Volume 23 Number 1

ACTA CYBERNETICA

Szeged, 2017

ACTA CYBERNETICA

Information for authors. Acta Cybernetica publishes only original papers in the field of Computer Science. Manuscripts must be written in good English. Contributions are accepted for review with the understanding that the same work has not been published elsewhere. Papers previously published in conference proceedings, digests, preprints are eligible for consideration provided that the author informs the Editor at the time of submission and that the papers have undergone substantial revision. If authors have used their own previously published material as a basis for a new submission, they are required to cite the previous work(s) and very clearly indicate how the new submission offers substantively novel or different contributions beyond those of the previously published work(s). Each submission is peer-reviewed by at least two referees. The length of the review process depends on many factors such as the availability of an Editor and the time it takes to locate qualified reviewers. Usually, a review process takes 6 months to be completed. There are no page charges. An electronic version of the puplished paper is provided for the authors in PDF format.

Manuscript Formatting Requirements. All submissions must include a title page with the following elements:

- title of the paper
- author name(s) and affiliation
- name, address and email of the corresponding author
- An abstract clearly stating the nature and significance of the paper. Abstracts must not include mathematical expressions or bibliographic references.

References should appear in a separate bibliography at the end of the paper, with items in alphabetical order referred to by numerals in square brackets. Please prepare your submission as one single PostScript or PDF file including all elements of the manuscript (title page, main text, illustrations, bibliography, etc.). Manuscripts must be submitted by email as a single attachment to either the most competent Editor, the Managing Editor, or the Editor-in-Chief. In addition, your email has to contain the information appearing on the title page as plain ASCII text. When your paper is accepted for publication, you will be asked to send the complete electronic version of your manuscript to the Managing Editor. For technical reasons we can only accept files in LaTeX format.

Subscription Information. Acta Cybernetica is published by the Institute of Informatics, University of Szeged, Hungary. Each volume consists of four issues, two issues are published in a calendar year. Subscription rates for one issue are as follows: 5000 Ft within Hungary, €40 outside Hungary. Special rates for distributors and bulk orders are available upon request from the publisher. Printed issues are delivered by surface mail in Europe, and by air mail to overseas countries. Claims for missing issues are accepted within six months from the publication date. Please address all requests to:

Acta Cybernetica, Institute of Informatics, University of Szeged
P.O. Box 652, H-6701 Szeged, Hungary
Tel: +36 62 546 396, Fax: +36 62 546 397, Email: `acta@inf.u-szeged.hu`

Web access. The above information along with the contents of past issues are available at the Acta Cybernetica homepage `https://www.inf.u-szeged.hu/en/kutatas/acta-cybernetica` .

Zoltan Kato
Department of Image Processing
and Computer Graphics
Szeged, Hungary
kato@inf.u-szeged.hu

Alice Kelemenová
Institute of Computer Science
Silesian University at Opava
Opava, Czech Republic
Alica.Kelemenova@fpf.slu.cz

László Lovász
Department of Computer Science
Eötvös Loránd University
Budapest, Hungary
lovasz@cs.elte.hu

Gheorghe Păun
Institute of Mathematics of the
Romanian Academy
Bucharest, Romania
George.Paun@imar.ro

Arto Salomaa
Department of Mathematics
University of Turku
Turku, Finland
asalomaa@utu.fi

László Varga
Department of Software Technology
and Methodology
Eötvös Loránd University
Budapest, Hungary
varga@ludens.elte.hu

Heiko Vogler
Department of Computer Science
Dresden University of Technology
Dresden, Germany
Heiko.Vogler@tu-dresden.de

Gerhard J. Woeginger
Department of Mathematics and
Computer Science
Eindhoven University of Technology
Eindhoven, The Netherlands
gwoegi@win.tue.nl

Preface

The present volume is a collection of papers dedicated to the memory of Professor Zoltán Ésik who passed away unexpectedly in 2016. These papers were written by his friends, colleagues, and former students. Zoltán is coauthor of some of the papers which were unpublished and some cases unfinished at the time of his sudden death. It is likely that we are publishing his very last paper which he was able to finish completely. Zoltán's scientific vita and the list of his publications are also included. I wish to express my most sincere gratitude to everyone who accepted the invitation to contribute to this volume. Without their work it would have been impossible to publish this collection of papers.

March, 2017

Zoltán Fülöp
Editor

Professor Zoltán Ésik
(1951–2016)

In Memoriam Zoltán Ésik

We got the news via e-mail that our colleague Zoltán Ésik, professor at the University of Szeged, passed away in a hotel in Reykjavik on 25th May 2016. He had plans for future research, teaching, and advising students, therefore the news was so unexpected and inconceivable that at first we did not believe it, and we did not dare to say it out loud. Just few days before we had talked to him, and few hours before we had exchanged emails about work and plans to be accomplished. But the news sadly proved to be true.

Zoltán's special talent for mathematics emerged as early as his school years. As a student of the specialised mathematics programme of the High School of the University of Szeged he achieved notable results at the competition of the Mathematical Journal for High Schools. This competition is more than 120 years old in Hungary and it serves as a platform for high school students to show their skills and talents. Several famous Hungarian mathematicians took their first steps in this competition toward research. It came naturally for Zoltán to continue his studies in mathematics at the University of Szeged, where he earned his degree in 1974. After graduation he got a position at the university and he remained a distinguished and highly respected member of our institute. As a young faculty member, he had long hair, wore blue jeans, and it looked like he was still a student. He was happy to stop for a chat with students. Many of them realized only that he was a teacher when he appeared in front of them in the class room and started his lectures. Even at such a young age, he introduced students to many subjects that became fundamental later on. Those who knew him well, praised his talent, his hard work and zest for science.

His professional career followed an unbroken path. He earned the title doctor of university in 1979, and became the candidate of science (CSc, an academic degree that used to be awarded in former Eastern Bloc countries, in Hungary by the Hungarian Academy of Sciences) in 1985. Then he earned the title of doctor of science (DSc) in 1996, which is also awarded by the Academy and is still the most prestigious academic rank in Hungary today. He was appointed as full professor in 1997, and in 2003 he became the head of the Department of Foundations of Computer Science of the University of Szeged.

Only some highlights of his work can be mentioned within the scope of this article. I only list a few of his numerous results and might even miss the most significant ones. I know that he would not mind the short summary of his achievements, since he never liked praise.

As a university teacher he made special efforts to introduce modern subjects taught at western universities into the curriculum, to bring the newest results and research topics to the University of Szeged. This was crucial in the era of the iron curtain and before the internet age. He travelled a lot to learn about new ideas, to broaden his own knowledge, which he later transferred to his students. He liked visiting foreign institutes, and it is a cruel twist of fate, that he had died during such a trip. In Szeged he was the first to teach category theory, theory of computability and complexity, logic in computer science, connections of automata and logics, fixed point theory and its applications to students of computer science. He devised the educational profile and the subjects taught at the Department of Foundations of Computer Science. He wrote four university textbooks, supervised winning papers at Conferences of Association of Students in Science (a Hungarian specialty), and successfully defended doctoral theses. He was instrumental to start the professional career of several colleagues, the author of this article being one of them. I wrote my university thesis under his supervision in 1979, although he was only four years older than me. A number of his former students are now leading experts or teachers at universities, not only in Hungary but also abroad.

He obtained his first results in the field of structural properties of finite automata, and later in the theory of tree automata and tree transducers. He was the first to prove the often cited result, in his thesis for doctor of university, that the equivalence problem of deterministic tree transducers is decidable. In the mid 1980's, probably due to the influence of Stephen Bloom, he turned his attention towards iteration theories. His field of research later became even more complex, and extended towards algebraic methods in computer science, category theory, theory of fixed points, modern logic in computer science, order theory, and semiring theory. He achieved thoughtful and essential results in all these fields. In the second half of his career he worked together and obtained important results with Werner Kuich in the theory of semirings and weighted automata. He had a special talent to recognize that in the seemingly different fields of computer science the same principles are in effect and used. One of his most important observations was that most fixed point models commonly used in computer science share the same equational properties.

His unparalleled talent, diligence, and his love for work resulted in countless publications. He worked together with more than 40 co-authors. The number of his scientific articles is above 230, and he edited nearly 30 collections of scientific papers. He is the author of the monographs Iteration Theories (with Stephen Bloom) and Modern Automata Theory (with Werner Kuich) which are regarded as fundamental handbooks all over the world. The latter was also translated into Russian. He gave talks at more than 50 universities, and was invited speaker at more than 30 conferences and workshops. His most significant conference activity was probably his talk at the conference of Mathematical Foundations of Computer Science in Milan in 2015. This was the 40th MFCS Conference, and Zoltán was the 'anniversary invited speaker' of the event. These numbers already alone are very impressive, but they are even more compelling when we recall the precision, the high standard, and the elegance that characterised each of his articles and talks.

Organising and participating in conferences were particularly important for him. He was a member of the steering committee of the conferences FCT, CAI, AFL, and FICS, and of the program committee of nearly 60 international conferences. It was specifically due to his scientific reputation and prestige that many renowned international conferences, like FCT, CSL, and MFCS could be organised in Hungary, most of them in Szeged. He played an essential role in moving the quality of the traditional Hungarian conference series Automata and Formal Languages gradually closer to European standards.

He was a member of the editorial board of a number of prestigious computer science journals, being the only Hungarian member of the editorial board of Theoretical Computer Science, a leading journal in the field. For three terms he was a member of the Council of the European Association for Theoretical Computer Science and for one term of the European Association for Computer Science Logic. He represented the Hungarian computer science community in Section TC1 of the International Federation for Information Processing. As an acknowledgement of his work, he was an elected member of The Academy of Europe in 2010 and earned the title Fellow of the EATCS in 2016, given only for the greatest scientists.

He received numerous Hungarian awards as well, such as the Kató Rényi Research Award and the Gyula Farkas Research Award of the János Bolyai Mathematical Society for outstanding mathematicians, the Award for Excellence, and in 2005 the title of Master Teacher. He also earned Humboldt and Fulbright research scholarships, and the Széchenyi Professorial Award in Hungary.

Zoltán Ésik was, without doubt, one of the most prominent figures in theoretical computer science. His results contributed significantly to the development of the foundations of this discipline. His achievements and efforts were crucial to the establishment of the Institute of Informatics of the University of Szeged an internationally acknowledged research institute, and putting it on the map of science. His death is a serious loss to the Institute, to the University of Szeged, and to the community of theoretical computer science.

Dear Professor Ésik, dear Zoli, your memory will stay with us forever. Farewell to you and may you rest in peace!

Zoltán Fülöp

Szeged, December 2016

Acta Cybernetica **23** (2017) 9–41.

Complexity of Right-Ideal, Prefix-Closed, and Prefix-Free Regular Languages*

Janusz A. Brzozowski[a] and Corwin Sinnamon[a]

Abstract

A language L over an alphabet Σ is prefix-convex if, for any words $x, y, z \in \Sigma^*$, whenever x and xyz are in L, then so is xy. Prefix-convex languages include right-ideal, prefix-closed, and prefix-free languages as special cases. We examine complexity properties of these special prefix-convex languages. In particular, we study the quotient/state complexity of boolean operations, product (concatenation), star, and reversal, the size of the syntactic semigroup, and the quotient complexity of atoms. For binary operations we use arguments with different alphabets when appropriate; this leads to higher tight upper bounds than those obtained with equal alphabets. We exhibit right-ideal, prefix-closed, and prefix-free languages that meet the complexity bounds for all the measures listed above.

Keywords: atoms, complexity of operations, prefix-closed, prefix-convex, prefix-free, quotient complexity, regular languages, right ideals, state complexity, syntactic semigroup, unrestricted alphabets

I have known Zoltán Ésik for about 30 years. We met at many scientific conferences, seven of them in Hungary. In 2000 I invited Zoltán to spend a month in Waterloo so that we could work on a problem in algebra with which I was struggling. I thought that the problem was purely of theoretical interest, but it turned out that the algebra we discovered was applicable to the detection of hazards in logic circuits. Zoltán also helped me with two other algebraic problems; he was always ready to give advice, and was modest about taking credit for his contributions. As the years went by we became good friends. In June 2015 my wife and I had the honour of celebrating his 64th birthday at our house. His passing was a great shock to me and I greatly miss his friendship.

Janusz Brzozowski

*This work was supported by the Natural Sciences and Engineering Research Council of Canada grant No. OGP0000871.

[a]David R. Cheriton School of Computer Science, University of Waterloo,Waterloo, ON, Canada N2L 3G1, E-mail: `brzozo@uwaterloo.ca` `sinncore@gmail.com`

DOI: 10.14232/actacyb.23.1.2017.3

1 Motivation

For words w, x, y over an alphabet Σ, if $w = xy$, then x is a *prefix* of w. A language $L \subseteq \Sigma^*$ is *prefix-convex* [1, 28] if, whenever x and xyz are in L, then xy is also in L. The class of prefix-convex languages includes three well-known subclasses: right-ideal, prefix-closed, and prefix-free languages; we study complexity properties of these languages.

A language L is a *right ideal* if it is non-empty and satisfies the equation $L = L\Sigma^*$. Right ideals play a role in pattern matching: If one is searching for all words beginning with words in some language L in a given text (a word over Σ^*), then one is looking for words in $L\Sigma^*$. Right ideals also constitute a basic concept in semigroup theory.

A language L is *prefix-closed* if, whenever w is in L and x is a prefix of w, then x is also in L. The complement of every right ideal is a prefix-closed language. The set of allowed input sequences to any digital system is a prefix-closed language.

A language L is *prefix-free* if no word in L is a prefix of another word in L. Prefix-free languages (other than $\{\varepsilon\}$, where ε is the empty word) are prefix codes. They play an important role in coding theory, and have many applications [3].

The *alphabet of a regular language* L is Σ (or L *is a language over* Σ) if $L \subseteq \Sigma^*$ and every letter of Σ appears in a word of L. The *(left) quotient* of L by a word $w \in \Sigma^*$ is $w^{-1}L = \{x \mid wx \in L\}$. A language is regular if and only if it has a finite number of distinct quotients. So the number of quotients of L is a natural measure of complexity for L; it is called the *quotient complexity* [4] of L and is denoted it by $\kappa(L)$. An equivalent concept is the *state complexity* [29] of L, which is the number of states in a complete minimal deterministic finite automaton (DFA) with alphabet Σ recognizing L.

If L_n is a regular language of quotient complexity n, and $\circ$ is a unary operation, then the *quotient/state complexity of* $\circ$ is the maximal value of $\kappa(L_n^\circ)$, expressed as a function of n, as L_n ranges over all regular languages of complexity n. If L'_m and L_n are regular languages of quotient complexities m and n respectively, and $\circ$ is a binary operation, then the *quotient/state complexity of* $\circ$ is the maximal value of $\kappa(L'_m \circ L_n)$, expressed as a function of m and n, as L'_m and L_n range over all regular languages of complexities m and n, respectively. The quotient/state complexity of an operation gives a worst-case lower bound on the time and space complexities of the operation, and has been studied extensively [4, 5, 29]; we refer to quotient/state complexity simply as *complexity*.

In all the past literature on binary operations it has always been assumed that the alphabets of the two operands are restricted to be the same. However, it has been shown recently [6, 14] that this is an unnecessary restriction: larger complexity bounds can be reached in some cases if the alphabets differ. In the present paper we examine both *restricted complexity* of binary operations, where the alphabets must be the same, and *unrestricted complexity*, where they may differ.

To find the complexity of a unary operation one first finds an upper bound on this complexity, and then exhibits languages that meet this bound. Since we require a language L_n for each $n \geq k$, we need a sequence $(L_k, L_{k+1}, \dots)$; here k is usually a small integer because the bound may not hold for a few small values of n. We call such a sequence a *stream* of languages. Usually the languages in a stream have the same basic structure and differ only in the parameter n. For example, $((a^n)^* \mid n \geq 2)$ is a stream. For a binary operation we require two streams.

While the complexity of languages is a useful measure, it is not entirely satisfactory. Two languages may have the same complexity n but the syntactic semigroup [26] of one may have $n-1$ elements, while that of the other has n^n elements [18]. For this reason, the size of the syntactic semigroup of a language – which is the same as the size of the transition semigroup of a minimal DFA accepting the language [26] – has been added as another complexity measure. Secondly, *star-free* languages meet the complexity bounds of regular languages for all operations except reversal, which only reaches the bound $2^n - 1$ instead of 2^n [13]. While regular languages are the smallest class containing the finite languages and closed under boolean operations, product and star, star-free languages are the smallest class containing the finite languages and closed only under boolean operations and product. In view of the results in [13], quotient/state complexity does not distinguish between these two classes.

The complexities of the atoms of a regular language have been proposed as an additional measure [5]. Atoms are defined by the following left congruence: two words x and y are equivalent if $ux \in L$ if and only if $uy \in L$ for all $u \in \Sigma^*$. Thus x and y are equivalent if $x \in u^{-1}L$ if and only if $y \in u^{-1}L$. An equivalence class of this relation is an *atom* of L [17, 21]. Thus an atom is a non-empty intersection of complemented and uncomplemented quotients of L. If $K_0, \dots, K_{n-1}$ are the quotients of L, and $S \subseteq Q_n = \{0, \dots, n-1\}$, then atom A_S is the intersection of quotients with subscripts in S and complemented quotients with subscripts in $Q_n \setminus S$. For more information about atoms see [16, 17, 21].

There exists a stream $(L_3, L_4, \dots)$ of regular languages $L_n(a, b, c)$ that meets the restricted complexity bounds for all boolean operations, product (concatenation), star, and reversal, and also has the largest syntactic semigroup and most complex atoms [5]. This stream modified by the addition of an input d that performs the identity transformation also meets the unrestricted bounds for product and boolean operations [6, 14]; such a stream is called *most complex.* Most complex streams are useful when one designs a system dealing with regular languages and finite automata. If one would like to know the maximal sizes of automata the system can handle, one can use the one most complex stream to test all the operations.

2 Contributions

We first present a most complex regular language stream similar to that of [5], but one that is better suited for prefix-convex languages. We then exhibit most complex language streams for right-ideal, prefix-closed, and prefix-free languages. More specifically, our contributions are as follows:

1. We generalize the concept of permutational dialect defined in [5, 9] by allowing letters of an alphabet to be mapped to letters from a different alphabet.

2. For regular languages we prove that there exists a most complex language stream $(L_n(a,b,c) \mid n \ge 3)$. The following results are new:
   - $L'_m(a,b)L_n(a,-,b)$ and $L'_m(a,b)L_n(a,c,b)$ meet the known bounds $(m-1)2^n + 2^{n-1}$ and $m2^n + 2^{n-1}$ for restricted and unrestricted products, respectively.
   - For the unrestricted case the following hold:
     - $L'_m(a,b,c) \circ L_n(b,a,d)$ meets the known bound $(m+1)(n+1)$ when $\circ \in \{\cup, \oplus\}$, where $\oplus$ is symmetric difference.
     - $L'_m(a,b,c) \setminus L_n(b,a)$ meets the known bound $mn + m$.
3. For right-ideal languages we prove that there exists a most complex language stream $(L_n(a,b,c,d) \mid n \ge 4)$. The following results are new:
   - $L'_m(a,-,c,d)L_n(a,-,c,d)$ meets the known bound $m + 2^{n-2}$ for restricted product, and $L'_m(a,-,c,d)L_n(b,-,c,d)$ meets the bound $m + 2^{n-1} + 2^{n-2} + 1$ for unrestricted product.
   - For the restricted case the known bounds mn if $\circ \in \{\cap, \oplus\}$, $mn-(m-1)$ if $\circ = \setminus$, and $mn-(m+n-2)$ if $\circ = \cup$ are all met by $L'_m(a,-,-,d) \circ L_n(-,-,d,a)$.
   - For the unrestricted case the bounds are the same as for regular languages and they are met by $L'_m(a,-,c,d) \circ L_n(b,-,d,a)$ if $\circ \in \{\cup, \oplus\}$, $L'_m(a,-,c,d) \setminus L_n(-,-,d,a)$, and $L'_m(a,-,-,d) \cap L_n(-,-,d,a)$.
4. For prefix-closed languages we prove that there exists a most complex language stream $(L_n(a,b,c,d) \mid n \ge 4)$. Here restricted and unrestricted cases coincide. The following results are new:
   - $L'_m(a,b,c,d)L_n(a,d,b,c)$ meets the known bound $(m+1)2^{n-2}$.
   - The known bounds mn if $\circ \in \{\cup, \oplus\}$, $mn-(m-1)$ if $\circ = \setminus$, and $mn-(m+n-2)$ if $\circ = \cup$ are met by $L'_m(a,b,-,d) \circ L_n(b,a,-,d)$.
5. For prefix-free languages we prove that there exists a most complex language stream $(L_n(a,b,c,d,e_0,\dots,e_{n-3}) \mid n \ge 4)$; restricted and unrestricted cases coincide. The following results are new:
   - At least $n+2$ inputs are required for a most complex prefix-free witness.
   - At least $n+1$ inputs are necessary to reach the known bound n^{n-2} for the size of the syntactic semigroup.
   - We derive upper bounds for the complexity of atoms of prefix-free languages, and prove that the atoms of the language $L_n(a,b,c,-,e_0)$ meet these bounds.
   - $L'_m(a,b,c,d)L_n(a,d,b,c)$ meets the known bound $(m+1)2^{n-2}$.
   - The known bounds $mn-2$ if $\circ \in \{\cup, \oplus\}$, $mn-(m+2n-4)$ if $\circ = \setminus$, and $mn-2(m+n-3)$ if $\circ = \cap$ are met by $L'_m(a,b,-,-,e_0,e_{m-3}) \circ L_n(b,a,-,-,e_0,e_{m-3})$.

3 Finite Automata, Transformations, Semigroups

A *deterministic finite automaton (DFA)* is a quintuple $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite non-empty set of *states*, Σ is a finite non-empty *alphabet*, $\delta\colon Q \times \Sigma \to Q$ is the *transition function*, $q_0 \in Q$ is the *initial* state, and $F \subseteq Q$ is the set of *final* states. We extend δ to a function $\delta\colon Q \times \Sigma^* \to Q$ as usual. A DFA $\mathcal{D}$ *accepts* a word $w \in \Sigma^*$ if $\delta(q_0, w) \in F$. The language accepted by $\mathcal{D}$ is denoted by $L(\mathcal{D})$. If q is a state of $\mathcal{D}$, then the language L^q of q is the language accepted by the DFA $(Q, \Sigma, \delta, q, F)$. A state is *empty* or *dead* or *a sink* if its language is empty. Two states p and q of $\mathcal{D}$ are *equivalent* if $L^p = L^q$; otherwise they are *distinguishable.* A state q is *reachable* if there exists $w \in \Sigma^*$ such that $\delta(q_0, w) = q$. A DFA is *minimal* if all of its states are reachable and no two states are equivalent. Usually DFAs are used to establish upper bounds on the complexity of operations and also as witnesses that meet these bounds.

A *nondeterministic finite automaton (NFA)* is a quintuple $\mathcal{D} = (Q, \Sigma, \delta, I, F)$, where Q, Σ and F are as in a DFA, $\delta\colon Q \times \Sigma \to 2^Q$, and $I \subseteq Q$ is the *set of initial states.* An *ε-NFA* is an NFA in which transitions under the empty word ε are also permitted.

Without loss of generality we use $Q_n = \{0, \dots, n-1\}$ as the set of states of every DFA with n states. A *transformation* of Q_n is a mapping $t\colon Q_n \to Q_n$. The *image* of $q \in Q_n$ under t is denoted by qt. In any DFA, each letter $a \in \Sigma$ induces a transformation δ_a of the set Q_n defined by $q\delta_a = \delta(q, a)$; we denote this by $a\colon \delta_a$. By a slight abuse of notation we use the letter a to denote the transformation it induces; thus we write qa instead of $q\delta_a$. We extend the notation to sets of states: if $P \subseteq Q_n$, then $Pa = \{pa \mid p \in P\}$. We also write $P \xrightarrow{a} Pa$ to mean that the image of P under a is Pa. Let $\mathcal{T}_{Q_n}$ be the set of all n^n transformations of Q_n; then $\mathcal{T}_{Q_n}$ is a monoid under composition.

For $k \geq 2$, a transformation (permutation) t of a set $P = \{q_0, q_1, \dots, q_{k-1}\} \subseteq Q_n$ is a *k-cycle* if $q_0t = q_1, q_1t = q_2, \dots, q_{k-2}t = q_{k-1}, q_{k-1}t = q_0$. This k-cycle is denoted by the transformation $(q_0, q_1, \dots, q_{k-1})$ of Q_n, which acts as the identity on the states outside the cycle. A 2-cycle (q_0, q_1) is called a *transposition.* A transformation that sends all the states of P to q and acts as the identity on the remaining states is denoted by $(P \to q)$. If $P = \{p\}$ we write $(p \to q)$ for $(\{p\} \to q)$. The identity transformation is denoted by $\mathbb{1}$. The notation $(_i^j\ q \to q+1)$ denotes a transformation that sends q to $q+1$ for $i \leq q \leq j$ and is the identity for the remaining states, and $(_i^j\ q \to q-1)$ is defined similarly.

Let $\mathcal{D} = (Q_n, \Sigma, \delta, q_0, F)$ be a DFA. For each word $w \in \Sigma^*$, the transition function induces a transformation δ_w of Q_n by w: for all $q \in Q_n$, $q\delta_w = \delta(q, w)$. The set $T_{\mathcal{D}}$ of all such transformations by non-empty words forms a semigroup of transformations called the *transition semigroup* of $\mathcal{D}$ [26]. We can use a set $\{\delta_a \mid a \in \Sigma\}$ of transformations to define δ, and so the DFA $\mathcal{D}$.

The *Myhill congruence* [25] $\approx_L$ of a language $L \subseteq \Sigma^*$ is defined on Σ^+ as follows:

$$\text{For } x, y \in \Sigma^+,\ x \approx_L y \text{ if and only if } wxz \in L \Leftrightarrow wyz \in L \text{ for all } w, z \in \Sigma^*.$$

This congruence is also known as the *syntactic congruence* of L. The quotient set

$\Sigma^+/{\approx_L}$ of equivalence classes of the relation $\approx_L$ is a semigroup called the *syntactic semigroup* of L. If $\mathcal{D}$ is a minimal DFA of L, then $T_{\mathcal{D}}$ is isomorphic to the syntactic semigroup T_L of L [26], and we represent elements of T_L by transformations in $T_{\mathcal{D}}$. The size of the syntactic semigroup has been used as a measure of complexity for regular languages [5, 18, 20, 24].

Recall that binary operations require two language streams to determine the complexity of the operation. Sometimes the same stream can be used for both operands, and it has been shown in [5, 6] that for all common binary operations on regular languages the second stream can be a "dialect" of the first, that is, it can "differ only slightly" from the first and all the bounds can still be met. Let $\Sigma = \{a_1, \dots, a_k\}$ be an alphabet ordered as shown; if $L \subseteq \Sigma^*$, we denote it by $L(a_1, \dots, a_k)$ to stress its dependence on Σ. A *dialect* of L is a related language obtained by replacing or deleting letters of Σ in the words of L. More precisely, for an alphabet Σ' and a partial map $\pi\colon \Sigma \mapsto \Sigma'$, we obtain a dialect of L by replacing each letter $a \in \Sigma$ by $\pi(a)$ in every word of L, or deleting the word entirely if $\pi(a)$ is undefined. We write $L(\pi(a_1), \dots, \pi(a_k))$ to denote the dialect of $L(a_1, \dots, a_k)$ given by π, and we denote undefined values of π by "$-$". For example, if $L(a, b, c) = \{a, ab, ac\}$ then its dialect $L(b, -, d)$ is the language $\{b, bd\}$. Undefined values for letters at the end of the alphabet are omitted; thus, for example, if $\Sigma = \{a, b, c, d, e\}$, $\pi(a) = b$, $\pi(b) = a$, $\pi(c) = c$ and $\pi(d) = \pi(e) = -$, we write $L(b, a, c)$ for $L(b, a, c, -, -)$.

The language stream that meets all the complexity bounds is referred to as the *master* language stream. Every master language stream we present here uses the smallest possible alphabet sufficient to meet all the bounds. Individual bounds are frequently met by dialects on reduced alphabets, and we prefer to use the smallest alphabet possible for each bound. For binary operations, we try to minimize the size of the combined alphabet of the two dialects.

As each letter induces a transformation on the states of a DFA (or equivalently, the quotients of a language) we count the number of distinct transformations induced by letters of the alphabet. In any language this number is at most the size of the alphabet, but there may be multiple letters which induce the same transformation; this does not occur in this paper as no language has a repeated transformation. For binary operations on two dialects of the same master language, we count the number of distinct transformations of the master language present in either dialect. For example, suppose $L(a, b, c, -)$ and $L(a, -, b, c)$ are two dialects of a language $L(a, b, c, d)$, which we assume has four distinct transformations. Each dialect has three letters and three distinct transformations, and between them they have three letters and four distinct transformations.

Although a given complexity bound may be met by many dialects of the master language, we favour dialects, or pairs of dialects, that use small alphabets and few distinct transformations. In many cases the dialects we present are minimal in these respects, though we do not always prove this.

4 A Most Complex Regular Stream

We now define a DFA stream that we use as a basic component. It is similar to the stream defined in [5] for the case of equal alphabets, except that there the transformation induced by c is $(n-1 \to 0)$. It is also similar to the DFA of [6], except that there the transformation induced by c is $(n-1 \to 0)$ and an additional input d inducing the identity transformation is used.

Definition 1. *For $n \geq 3$, let $\mathcal{D}_n = \mathcal{D}_n(a,b,c) = (Q_n, \Sigma, \delta_n, 0, \{n-1\})$, where $\Sigma = \{a,b,c\}$, and δ_n is defined by the transformations $a\colon (0,\dots,n-1)$, $b\colon (0,1)$, and $c\colon (1 \to 0)$. Let $L_n = L_n(a,b,c)$ be the language accepted by $\mathcal{D}_n$. The structure of $\mathcal{D}_n(a,b,c)$ is shown in Figure 1.*

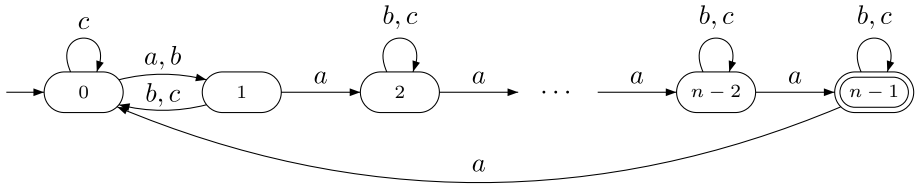


Figure 1: Minimal DFA of a most complex regular language.

Theorem 1 (Most Complex Regular Languages)**.** *For each $n \geq 3$, the DFA of Definition 1 is minimal and its language $L_n(a,b,c)$ has complexity n. The stream $(L_m(a,b,c) \mid m \geq 3)$ with some dialect streams is most complex in the class of regular languages. In particular, it meets all the complexity bounds below. At least three letters are required in any witness meeting all these bounds and a total of four distinct letters is required for any two witnesses for unrestricted union and symmetric difference. In several cases the bounds can be met with a smaller alphabet as shown below.*

1. *The syntactic semigroup of $L_n(a,b,c)$ has cardinality n^n.*
2. *Each quotient of $L_n(a)$ has complexity n.*
3. *The reverse of $L_n(a,b,c)$ has complexity 2^n, and $L_n(a,b,c)$ has 2^n atoms.*
4. *Each atom A_S of $L_n(a,b,c)$ has maximal complexity:*

$$\kappa(A_S) = \begin{cases} 2^n - 1, & \text{if } S \in \{\emptyset, Q_n\}; \\ 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n}{x}\binom{n-x}{y}, & \text{if } \emptyset \subsetneq S \subsetneq Q_n. \end{cases}$$

5. *The star of $L_n(a,b)$ has complexity $2^{n-1} + 2^{n-2}$.*
6. *a) Restricted Complexity:*
   The product $L'_m(a,b)L_n(a,-,b)$ has complexity $m2^n - 2^{n-1}$.

b) Unrestricted Complexity:
The product $L'_m(a,b)L_n(a,c,b)$ has complexity $m2^n + 2^{n-1}$.

7. *a) Restricted Complexity:*
The complexity of $L'_m(a,b) \circ L_n(b,a)$ is mn for $\circ \in \{\cup, \oplus, \setminus, \cap\}$.

b) Unrestricted Complexity:
The complexity of union and symmetric difference is $mn + m + n + 1$ and this bound is met by $L'_m(a,b,c)$ and $L_n(b,a,d)$, that of difference is $mn+m$ and this bound is met by $L'_m(a,b,c)$ and $L_n(b,a)$, and that of intersection is mn and this bound is met by $L'_m(a,b)$ and $L_n(b,a)$. A total of four letters is required to meet the bounds for union and symmetric difference.

Proof. Clearly $L_n(a)$ has complexity n as the DFA of Definition 1 is minimal.

1. **Syntactic Semigroup** The transformations $a\colon (0,\dots,n-1)$, $b\colon (0,1)$, and $c\colon (n-1 \to 0)$ were used in [5]. It is well known that these transformations as well as a, b, and $c\colon (1 \to 0)$ generate the semigroup of all transformations of Q_n.

2. **Quotients** Obvious.

3. **Reversal** This follows from a theorem in [27] which states that if the transition semigroup has n^n elements, then the complexity of reversal is 2^n. Also, it was shown in [17] that the number of atoms is the same as the complexity of the reverse.

4. **Atoms** Proved in [7, Theorem 3].

5. **Star** Proved in [5].

6. **Product** Let $\mathcal{D}' = (Q'_m, \Sigma', \delta', 0', F')$ and $\mathcal{D} = (Q_n, \Sigma, \delta, 0, F)$ be minimal DFAs of languages L' and L, respectively. We use the standard construction of the ε-NFA $\mathcal{N}$ for the product $L'L$: the final states of $\mathcal{D}'$ becomes non-final, and an ε-transition is added from each state of F' to the initial state 0 of $\mathcal{D}$.

   The subset construction on this NFA yields sets $\{p'\} \cup S$ where $p' \in Q'_m \setminus F'$ and $S \subseteq Q_n$ and sets $\{p', 0\} \cup S$ where $p' \in F'$ and $S \subseteq Q_n \setminus \{0\}$, as well as sets $S \subseteq Q_n$ which can only be reached by letters in $\Sigma \setminus \Sigma'$. Hence the restricted complexity of $L'L$ is bounded by $(m-|F'|)2^n + |F'|2^{n-1} \le m2^n - 2^{n-1}$, and the unrestricted complexity of $L'L$ is bounded by $(m-|F'|)2^n + |F'|2^{n-1} + 2^n \le m2^n + 2^{n-1}$.

   Restricted Complexity: Consider $L'_m(a,b)$ and $L_n(a,-,b)$ of Definition 1; we show that their product meets the upper bound for restricted complexity. As before, we construct an NFA recognizing $L'_m(a,b)L_n(a,-,b)$ and then apply the subset construction to obtain a DFA. Figure 2 shows the NFA for the

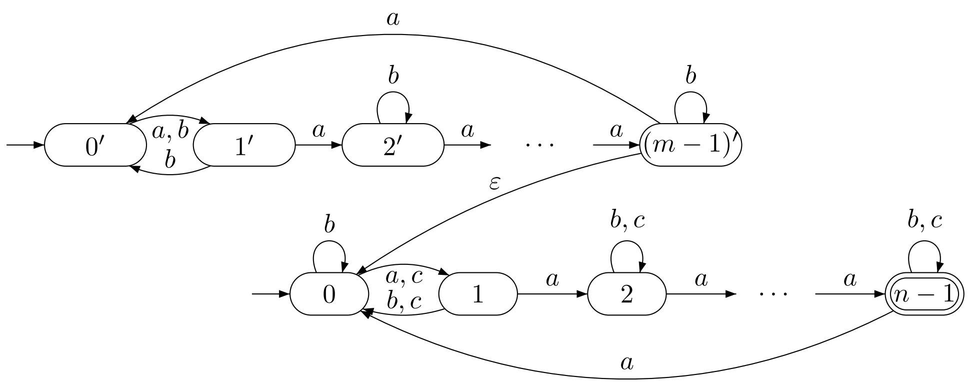


Figure 2: An NFA for the product of $L'_m(a,b)$ and $L_n(a,c,b)$. The NFA for the product of $L'_m(a,b)$ and $L_n(a,-,b)$ is the same except c is omitted.

unrestricted product $L'_m(a,b)L_n(a,c,b)$; the product $L'_m(a,b)L_n(a,-,b)$ is the same except c is omitted.

The initial state is $\{0'\}$ and each state $\{p'\}$ for $0 \le p \le m-2$ is reached by a^p. Consider $\{0'\} \cup S$, where $S = \{q_1, q_2, \dots, q_k\}$ with $0 \le q_1 < q_2 < \dots < q_k \le n-1$. If $q_1 \ge 1$ then $\{(m-2)', q_2 - q_1 - 1, \dots, q_k - q_1 - 1\} \xrightarrow{a^2} \{0', 1, q_2 - q_1 + 1, \dots, q_k - q_1 + 1\} \xrightarrow{(ab)^{q_1 - 1}} \{0'\} \cup S$. If $q_1 = 0$ and $k \ge 2$, then $\{(m-2)', n-2, q_3 - q_2 - 1, \dots, q_k - q_2 - 1\} \xrightarrow{a^2} \{0', 0, 1, q_3 - q_2 + 1, \dots, q_k - q_2 + 1\} \xrightarrow{(ab)^{q_2 - 1}} \{0'\} \cup S$. State $\{0', 0\}$ is reached by $a^m b^2$. Hence for any non-empty $S \subseteq Q_n$, state $\{0'\} \cup S$ is reachable from $\{(m-2)'\} \cup T$ for some $T \subseteq Q_n$ of size $|S| - 1$. We reach $\{p'\} \cup S$ from $\{0'\} \cup (S - p)$ by a^p, where $S - p$ denotes $\{q - p \mid q \in S\}$ taken mod n. By induction, $\{p'\} \cup S$ is always reachable and thus all $m2^n - 2^{n-1}$ states are reachable.

We check that all states are pairwise distinguishable.

a) Any two sets which differ by $q \in Q_n$ are distinguished by a^{n-1-q}.
b) States $\{p'_1\}$ and $\{p'_2\}$ with $p_1 < p_2$ are distinguished by $a^{m-1-p_2} a^{n-1}$.
c) States $\{0', 0\}$ and $\{p', 0\}$ are distinguished by $(ab)^{m-2-p} a a^{n-1}$ if $p' \ne (m-1)'$; otherwise apply ab to simplify to this case.
d) States $\{p'_1, 0\}$ and $\{p'_2, 0\}$, $p_1 < p_2$, reduce to Case (c) by $(ab^2)^{m-p_2}$.
e) States $\{p'_1\} \cup S$ and $\{p'_2\} \cup S$, where $S \ne \emptyset$ and $p_1 < p_2$, reduce to Case (d) by $(ab)^n$ since $S \xrightarrow{(ab)^n} \{0\}$ and $(ab)^n$ permutes Q'_m.

We can distinguish any pair of states; so the complexity of $L'_m(a,b)L_n(a,-,b)$ is $m2^n - 2^{n-1}$ for all $m, n \ge 3$.

Unrestricted Complexity: The NFA for the product of $L'_m(a,b)L_n(a,c,b)$ is illustrated in Figure 2. The NFA is the same as the restricted case except it has the additional transformation $c\colon (0,1)(Q'_m \to \emptyset)$. Hence the subset construction yields the $m2^n - 2^{n-1}$ sets of the restricted case, as well as all sets $S \subseteq Q_n$ since S is reachable from $\{0'\} \cup S$ by c^2. We check that these sets are distinguishable from all previously reached sets.

a) Any two sets which differ by $q \in Q_n$ are distinguished by a^{n-1-q}.
b) State $\{p'\}$ is distinguishable from $\emptyset$ by $a^{m-1-p}a^{n-1}$.
c) States $\{0', 0\}$ and $\{0\}$ are distinguished by $a^{m-1}a^{n-1}$ if $m-1$ is not a multiple of n, and by $ba^{m-2}a^{n-1}$ otherwise.
d) States $\{p', 0\}$ and $\{0\}$ reduce to Case (c) by $(ab^2)^{m-p}$.
e) States $\{p'\} \cup S$ and S, where $S \neq \emptyset$, reduce to Case (d) by $(ab)^n$ since $S \xrightarrow{(ab)^n} \{0\}$ and $(ab)^n$ permutes Q'_m.

Hence $L'_m(a,b)L_n(a,c,b)$ has complexity $m2^n + 2^{n-1}$.

7. Boolean Operations

Restricted Complexity: All operations have complexity at most mn [4]. Applying the standard construction for boolean operations we consider the direct product of $\mathcal{D}'_m(a,b)$ and $\mathcal{D}_n(b,a)$ which has states $Q'_m \times Q_n$; the final states vary depending on the operation. By [2, Theorem 1] and computation for the cases $(m,n) \in \{(3,4),(4,3),(4,4)\}$, the states of $Q'_m \times Q_n$ are reachable and pairwise distinguishable for each operation $\circ \in \{\cup, \oplus, \setminus, \cap\}$; hence each operation has complexity mn.

Note that two letters are required to meet these bounds: To a contradiction suppose a single letter ℓ is sufficient to reach $Q'_m \times Q_n$ in the direct product, where $m, n \geq 2$ are not coprime. Letter ℓ must induce an m-element permutation on Q'_m; otherwise there is an unreachable state in Q'_m or the sequence $0', 0'\ell, 0'\ell^2, \dots, 0'\ell^k, \dots$ never returns to $0'$. Similarly ℓ must induce an n-cycle in Q_n. Hence ℓ has order $\mathrm{lcm}(mn)$ in the direct product; however, it must have order mn if the bound is to be reached, and this occurs only when m and n are coprime.

Unrestricted Complexity: The upper bounds on the unrestricted complexity of boolean operations are derived in [6]. To compute $L'_m(a,b,c) \circ L_n(b,a,d)$, where $\circ$ is a boolean operation, add an empty state $\emptyset'$ to $\mathcal{D}'_m(a,b,c)$, and send all the transitions from any state of Q'_m under d to $\emptyset'$. Similarly, add an empty state $\emptyset$ to $\mathcal{D}_n(b,a,d)$ together with appropriate transitions; now the alphabets of the resulting DFAs are the same. We consider the direct product of $\mathcal{D}'_{m,\emptyset'}$ and $\mathcal{D}_{n,\emptyset}$ which has states $\{(p',q) \mid p' \in Q'_m \cup \{\emptyset'\}, q \in Q_n \cup \{\emptyset\}\}$. A DFA recognizing $L'_m(a,b,c) \cup L_n(b,a,d)$ is shown in Figure 3 for $m=3$ and $n=4$.

As in the restricted case all the states of $Q'_m \times Q_n$ are reachable by words in $\{a,b\}^*$. The remaining states in $C = \{(p',\emptyset) \mid p' \in Q'_m \cup \{\emptyset'\}\}$ and

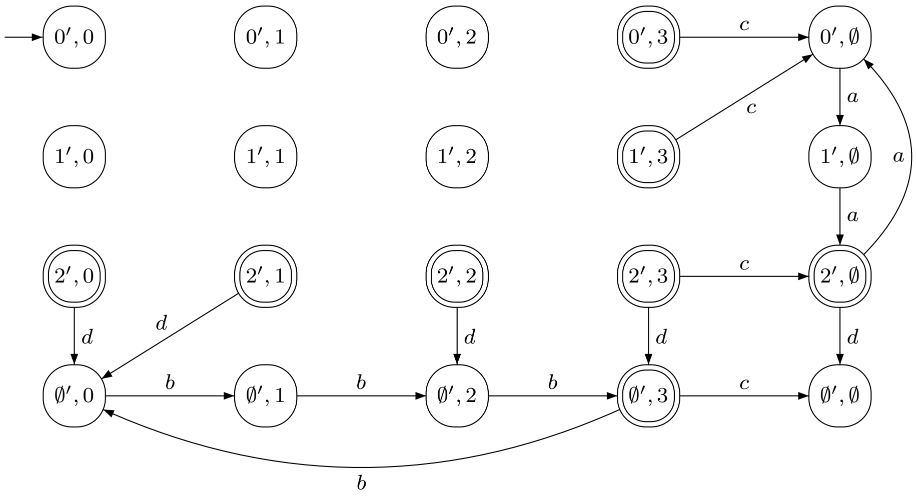


Figure 3: Direct product for union of $\mathcal{D}'_3(a,b,c)$ and $\mathcal{D}_4(b,a,d)$ shown partially.

$R = \{(\emptyset', q) \mid q \in Q_n \cup \{\emptyset\}\}$ are reachable using c and d in addition to a and b as shown in Figure 3. Hence all $(m+1)(n+1)$ states are reachable.

For union and symmetric difference, the states of C are pairwise distinguishable by words in a^* and they are distinguished from all other states by words in b^*d. Similarly the states of R are distinguishable from each other and all other states; hence all $mn+m+n+1$ states are distinguishable.

For difference, the final states are $((m-1)', q)$ for $q \neq n-1$. The states of R are all empty, and they are only reachable by d. As the words of $L'_m(a,b,c) \setminus L_n(b,a,d)$ do not contain d, the alphabet is $\{a,b,c\}$; hence we can omit d and delete the states of R, and be left with a DFA recognizing the same language. We check that the remaining $mn+m$ states are pairwise distinguishable. Any states $(p'_1, \emptyset)$ and (p'_2, q) where $p'_1 \neq p'_2$ and $q \in Q_n \cup \{\emptyset\}$ are distinguished by words in a^*. State $(p', \emptyset)$ is distinguished from (p', q) by some $w \in \{a,b\}^*$ that maps (p', q) to $((m-1)', n-1)$, since w must send $(p', \emptyset)$ to the final state $((m-1)', \emptyset)$; such a word exists because a and b induce permutations on the direct product, and so every state in $Q'_m \times Q_n$ is reachable from every other.

For intersection the only final state is $((m-1)', n-1)$. The alphabet of $L'_m(a,b,c) \cap L_n(b,a,d)$ is $\{a,b\}$; hence we can omit c and d and delete the states of $R \cup C$, and be left with a DFA recognizing the same language. The remaining mn states are pairwise distinguishable as in the restricted case.

Note that a total of four letters between the alphabets Σ' of $\mathcal{D}'_m$ and Σ of $\mathcal{D}_n$ is required for union and symmetric difference. As in the restricted case, two letters in $\Sigma' \cap \Sigma$ are required to reach the states of $Q'_m \times Q_n$ for general values of m and n. Letters in both alphabets cannot be used to reach states $(p', \emptyset)$ and $(\emptyset', q)$ as the empty states in each coordinate are only reached by letters outside the corresponding alphabet. Thus two additional letters are required, one in $\Sigma' \setminus \Sigma$ and one in $\Sigma \setminus \Sigma'$. Hence each alphabet must contain at least three letters, and $\Sigma' \cup \Sigma$ must contain at least four. In contrast, the bound for difference is met by $L'_m(a,b,c)$ and $L_n(b,a)$, and the bound for intersection is met by $L'_m(a,b)$ and $L_n(b,a)$.

Since all the claims have been verified, the theorem holds. □

5 Right Ideals

The results in this section are based on [8, 9, 18]; however, the stream below is different from that of [18], where c: $(n-2 \to 0)$ and d: $(n-2 \to n-1)$.

Definition 2. *For $n \geq 4$, let $\mathcal{D}_n = \mathcal{D}_n(a,b,c,d) = (Q_n, \Sigma, \delta_n, 0, \{n-1\})$, where $\Sigma = \{a,b,c,d\}$ and δ_n is defined by the transformations a: $(0,\dots,n-2)$, b: $(0,1)$, c: $(1 \to 0)$, and d: $({}_0^{n-2} q \to q+1)$. Let $L_n = L_n(a,b,c,d)$ be the language accepted by $\mathcal{D}_n$. For the structure of $\mathcal{D}_n(a,b,c,d)$ see Figure 4.*

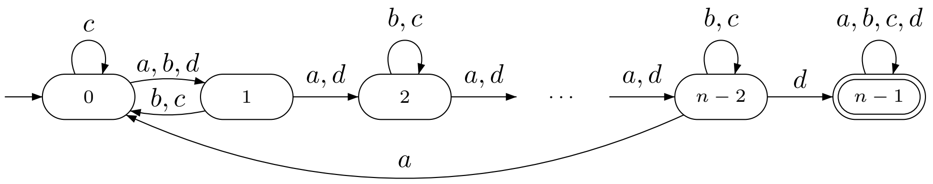


Figure 4: Minimal DFA of a most complex right ideal.

Theorem 2 (Most Complex Right Ideals)**.** *For each $n \geq 4$, the DFA of Definition 2 is minimal and $L_n(a,b,c,d)$ is a right ideal of complexity n. The stream $(L_n(a,b,c,d) \mid n \geq 4)$ with some dialect streams is most complex in the class of right ideals. In particular, it meets all the bounds below. At least four letters are required to meet these bounds.*

1. *The syntactic semigroup of $L_n(a,b,c,d)$ has cardinality n^{n-1}. There is only one maximal transition semigroup of a minimal DFA accepting a right ideal, since it consists of all the transformations of Q_n that fix $n-1$. At least four letters are needed for this bound.*

2. *The quotients of $L_n(a,-,-,d)$ have complexity n, except that $\kappa(\{a,d\}^*) = 1$.*

3. *The reverse of $L_n(a,-,-,d)$ has complexity 2^{n-1}, and $L_n(a,-,-,d)$ has 2^{n-1} atoms.*
4. *Each atom A_S of $L_n(a,b,c,d)$ has maximal complexity:*
$$\kappa(A_S) = \begin{cases} 2^{n-1}, & \text{if } S = Q_n; \\ 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n-1}{x-1}\binom{n-x}{y}, & \text{if } \emptyset \subsetneq S \subsetneq Q_n. \end{cases}$$
5. *The star of $L_n(a,-,-,d)$ has complexity $n+1$.*
6. *a) Restricted Complexity:*
   The product $L'_m(a,-,c,d)L_n(a,-,c,d)$ has complexity $m + 2^{n-2}$.

   b) Unrestricted Complexity:
   The product $L'_m(a,-,c,d)L_n(b,-,c,d)$ has complexity $m+2^{n-1}+2^{n-2}+1$. At least three letters for each language and four letters in total are required to meet this bound.
7. *a) Restricted Complexity:*
   The complexity of $\circ$ is mn if $\circ \in \{\cap, \oplus\}$, $mn-(m-1)$ if $\circ = \setminus$, and $mn-(m+n-2)$ if $\circ = \cup$, and these bounds are met by $L'_m(a,-,-,d) \circ L_n(-,-,d,a)$. At least two letters are required to meet these bounds.

   b) Unrestricted Complexity:
   The complexity of $L'_m(a,-,c,d) \circ L_n(b,-,d,a)$ is the same as for arbitrary regular languages: $mn+m+n+1$ if $\circ \in \{\cup, \oplus\}$, $mn+m$ if $\circ = \setminus$, and mn if $\circ = \cap$. At least three letters in each language and four letters in total are required to meet the bounds for intersection and symmetric difference. The bound for difference is also met by $L'_m(a,-,c,d) \setminus L_n(-,-,d,a)$ and the bound for intersection is met by $L'_m(a,-,-,d) \cap L_n(-,-,d,a)$.

Proof. DFA $\mathcal{D}_n(-,-,-,d)$ is minimal because the shortest word in d^* accepted by state q is d^{n-1-q}, and $L_n(a,b,c,d)$ is a right ideal because it has only one final state and that state accepts Σ^*.

1. **Semigroup** The transformations induced by a, b, and c generate all transformations of Q_{n-1}. Also, since the transformation induced by da^{n-2} is $(n-2 \to n-1)$, the transition semigroup of $\mathcal{D}_n(a,b,c,d)$ contains the one in [18], which is maximal for right ideals. Hence the syntactic semigroup of $L_n(a,b,c,d)$ has size n^{n-1} as well. The fact that at least four letters are needed was proved in [15].
2. **Quotients** If the initial state of $\mathcal{D}_n(a,-,-,d)$ is changed to q with $0 \le q \le n-2$, the new DFA accepts a quotient of L_n and is still minimal; hence the complexity of that quotient is n.
3. **Reversal** It was proved in [10] that the reverse has complexity 2^{n-1}, and in [17] that the number of atoms is the same as the complexity of the reverse.

4. **Atoms** The proof in [8] applies since the DFA has all the transformations that fix $n-1$.

5. **Star** If L_n is a right ideal, then $L_n^* = L_n \cup \{\varepsilon\}$. If we add a new initial state $0'$ to the DFA of Definition 2 with the same transitions as those from 0 and make $0'$ final, the new DFA accepts L_n^* and is minimal for $0'$ does not accept a, and so is not equivalent to $n-1$.

6. **Product** Let $\mathcal{D}' = (Q'_m, \Sigma', \delta', 0', \{(m-1)'\})$ and $\mathcal{D} = (Q_n, \Sigma, \delta, 0, \{n-1\})$ be minimal DFAs of L' and L, respectively, where L' and L are right ideals. We use the standard construction of the NFA for the product $L'L$: the final state $(m-1)'$ of $\mathcal{D}'$ becomes non-final, and an ε-transition is added from that state to the initial state 0 of $\mathcal{D}$. We bound the complexity of the product by counting the reachable states in the subset construction on this NFA. The $m-1$ non-final states $\{p'\}$ of $\mathcal{D}'$ may be reachable, as well as $\{(m-1)', 0\}$. From $\{(m-1)', 0\}$ we may reach all 2^{n-2} subsets of Q_n which contain 0 but not $n-1$, and 2^{n-2} states that contain both 0 and $n-1$; however, the latter 2^{n-2} states all accept Σ^* and are therefore equivalent. So far, we have $m-1+2^{n-2}+1 = m+2^{n-2}$ states; these are the only reachable sets if the witnesses are restricted to the same alphabet.

   For the unrestricted case, suppose that $\ell' \in \Sigma' \setminus \Sigma$ and $\ell \in \Sigma \setminus \Sigma'$. By applying ℓ to $\{(m-1)', 0\} \cup S$, $S \subseteq Q_n \setminus \{0\}$, we may reach all $2^n - 1$ non-empty subsets of Q_n, and then by applying ℓ' we reach the empty subset. However, the 2^{n-1} subsets of Q_n that contain $n-1$ all accept Σ^*. Hence there are at most $2^{n-1}+1$ additional sets, for a total of $m+2^{n-2}+2^{n-1}+1$ reachable sets.

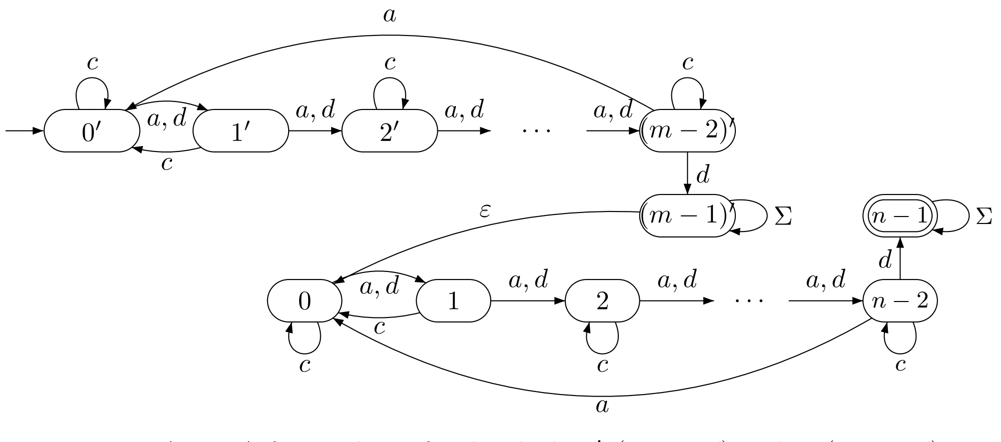


Figure 5: An NFA for product of right ideals $L'_m(a, -, c, d)$ and $L_n(a, -, c, d)$.

Restricted Complexity: To prove the bound is tight, consider the two dialects of the DFA of Definition 2 shown in Figure 5, where $\Sigma = \{a, c, d\}$. The

$m-1$ sets $\{p'\}$ for $p' \in Q'_{m-1}$ are reachable in $\mathcal{D}'_m$ by words in d^*, and $\{(m-1)', 0\}$ is reached by d^{m-1}. The 2^{n-2} sets of the form $\{(m-1)', 0\} \cup S$, where $S \subseteq Q_n \setminus \{0\}$, are reachable using words in $\{c, d\}^*$ as follows: To reach $\{(m-1)', 0\} \cup S$, where $S = \{q_1, \dots, q_k\}$, $1 \le q_1 < q_2 < \cdots < q_k \le n-1$, we have first $\{(m-1)', 0\}d = \{(m-1)', 0, 1\}$. State 1 will then be moved to the right by applying either d or dc repeatedly: If $q_{k-1} = q_k - 1$, use d; otherwise use dc. Repeating this process q_k times we eventually construct S. For example, to reach $\{(m-1)', 0\} \cup \{2, 5, 7, 8\}$ use $dd(dc)d(dc)(dc)d(dc)$. The 2^{n-2} sets $\{(m-1)', 0\} \cup S$ that contain $n-1$ all accept $\{a, c, d\}^*$; hence they are all equivalent.

The remaining states are pairwise distinguishable: States $\{p'\}$ and $\{q'\}$ with $0 \le p < q \le m-2$ are distinguished by $d^{m-1-q}d^{n-1}$, and $\{p'\}$ is distinguished from $\{(m-1)', 0\} \cup S$ by d^{n-1}. Two non-final states $\{(m-1)', 0\} \cup S$ and $\{(m-1)', 0\} \cup T$ with $q \in S \oplus T$ are distinguished by $a^{n-2-q}d$. Thus the product has complexity $m + 2^{n-2}$.

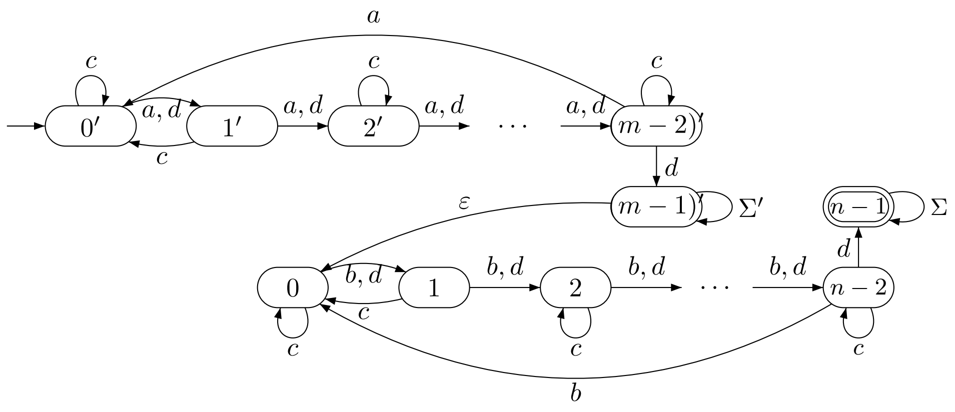


Figure 6: An NFA for product of right ideals $L'_m(a, -, c, d)$ and $L_n(b, -, c, d)$.

Unrestricted Complexity: Consider two dialects of the DFA of Definition 2 shown in Figure 6. Here $\Sigma' = \{a, c, d\}$ and $\Sigma = \{b, c, d\}$. By the restricted case, all states $\{p'\}$ for $p' \in Q'_{m-1}$ and $\{(m-1)', 0\} \cup S$ for $S \subseteq Q_n \setminus \{0\}$ are reachable by words in $\{c, d\}^*$. Apply b from $\{0'\}$ to reach the empty subset. By applying b to $\{(m-1)', 0\} \cup S$, $S \subseteq Q_n \setminus \{0\}$, we reach all $2^n - 1$ non-empty subsets of Q_n; hence all states are reachable. However, the 2^{n-1} sets $S \subseteq Q_n$ that contain $n-1$ all accept $\{b, c, d\}^*$ and are sent to the empty state by a; hence they are all equivalent. Similarly, the 2^{n-2} sets $\{(m-1)', 0\} \cup S$ that contain $n-1$ all accept $\{b, c, d\}^*$ and are sent to $\{(m-1)', 0\}$ by a; hence they are also equivalent.

The remaining states are pairwise distinguishable. States $\{p'\}$ and $\{q'\}$ with $0 \le p < q \le m-2$ are distinguished by $d^{m-1-q}d^{n-1}$, and $\{p'\}$ is distinguished from $\{(m-1)', 0\} \cup S$ or from S, where $\emptyset \subsetneq S \subseteq Q_n$, by d^{n-1}. Two states

$\{(m-1)', 0\} \cup S$ and $\{(m-1)', 0\} \cup T$ with $q \in S \oplus T$ are distinguished by $b^{n-2-q}d$, as are two states S and T with $q \in S \oplus T$. A state $\{(m-1)', 0\} \cup S$ is distinguishable from T where $S, T \subseteq Q_n$ by ad^{n-1}. Thus all $m + 2^{n-2} + 2^{n-1} + 1$ states are pairwise distinguishable.

At least three inputs to each DFA are required to achieve the bound in the unrestricted case: There must be a letter in Σ (like d) with a transition to $n-1$ to reach sets containing $n-1$, and this letter must be in Σ' in order to reach the sets that contain both $(m-1)'$ and $n-1$. However no single letter in $\Sigma' \cap \Sigma$ is sufficient to reach every set of the form $\{(m-1)', 0\} \cup S$, regardless of its behaviour on Q_n. For example, if the letter maps $0 \to q_1$ and $q_1 \to q_2$ then it is impossible to reach the state $\{(m-1)', 0, q_2\}$ by repeatedly applying the letter from $\{(m-1)', 0\}$, as it can never delete q_1. Hence there must be at least two letters in $\Sigma' \cap \Sigma$. Furthermore there must be some $\ell \in \Sigma \setminus \Sigma'$ to reach the empty state, and there must be some $\ell' \in \Sigma' \setminus \Sigma$ to distinguish $\{(m-1)', 0, n-1\}$ from $\{n-1\}$. Thus each alphabet must contain at least three letters to meet the bound.

7. Boolean Operations

Restricted Complexity: The bounds for right ideals were derived in [10]. We show that the DFAs $\mathcal{D}'_m(a, -, -, d)$ and $\mathcal{D}_n(-, -, d, a)$ shown in Figure 7 of the right ideals of Definition 2 meet the bounds.

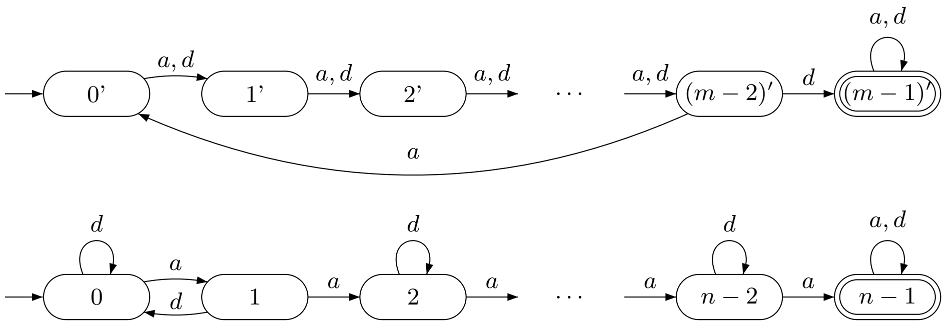


Figure 7: DFAs of $L'_m(a, -, -, d)$ and $L_n(-, -.d, a)$ for boolean operations.

Consider the direct product of $L'_m(a, -, -, d)$ and $L_n(-, -, d, a)$, illustrated in Figure 8 for $m = n = 4$. For $p' \in Q'_{m-1}$ state $(p', 0)$ is reached by d^p. Since the first column of $Q_{m-1} \times Q_n$ is reachable and $(p', q) \xrightarrow{a} ((p+1)', (q+1))$, where $p+1$ is taken mod $m-1$, we can reach every state in $Q_{m-1} \times Q_n$. State $((m-1)', q)$ is reached by $d^{m-1}a^q$; hence the states of $Q'_m \times Q_n$ are reachable.

We now check distinguishability, which depends on the final states of the

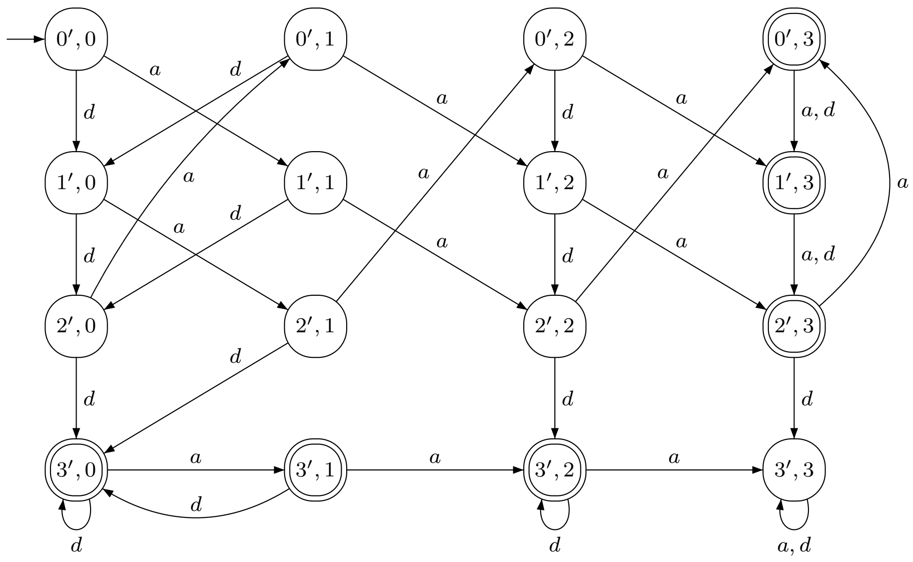


Figure 8: Partial illustration of direct product for $L'_4(a, -, -, d) \oplus L_4(-, -, d, a)$.

DFA. The direct product is made to recognize $L'_m(a, -, -, d) \circ L_n(-, -, d, a)$ by setting the final states to be $(\{(m-1)'\} \times Q_n) \circ (Q'_m \times \{n-1\})$.

For intersection and symmetric difference, all states are pairwise distinguishable. States that differ in the first coordinate are distinguished by words in d^*a^* and states that differ in the second coordinate are distinguished by words in a^*d^*. Hence the complexity is mn.

For difference, the states $\{(p', n-1) \mid p' \in Q'_m\}$ are all empty, and therefore equivalent. The remaining states are non-empty, and they are distinguished by words in d^* if they differ in the first coordinate or by words in a^*d^* if they differ in the second coordinate. Hence the complexity is $mn - m + 1$.

For union, the states $\{(p', n-1) \mid p' \in Q'_m\} \cup \{((m-1)', q) \mid q \in Q_n\}$ are all final and equivalent as they accept $\{a, d\}^*$. The remaining states are distinguished by words in d^* if they differ in the first coordinate or by words in a^* if they differ in the second coordinate. Hence the complexity is $mn - (m + n - 2)$.

As in regular languages, one letter in $\Sigma' \cap \Sigma$ is not sufficient to reach all the states of $Q'_{m-1} \times Q_{n-1}$ for all values of m and n; hence two letters are required to meet any of the bounds.

Unrestricted Complexity: The unrestricted bounds for right ideals are the same as those for arbitrary regular languages [6]. We show that the DFAs $\mathcal{D}'_m(a, -, c, d)$ and $\mathcal{D}_n(b, -, d, a)$ of Definition 2 meet the bounds.

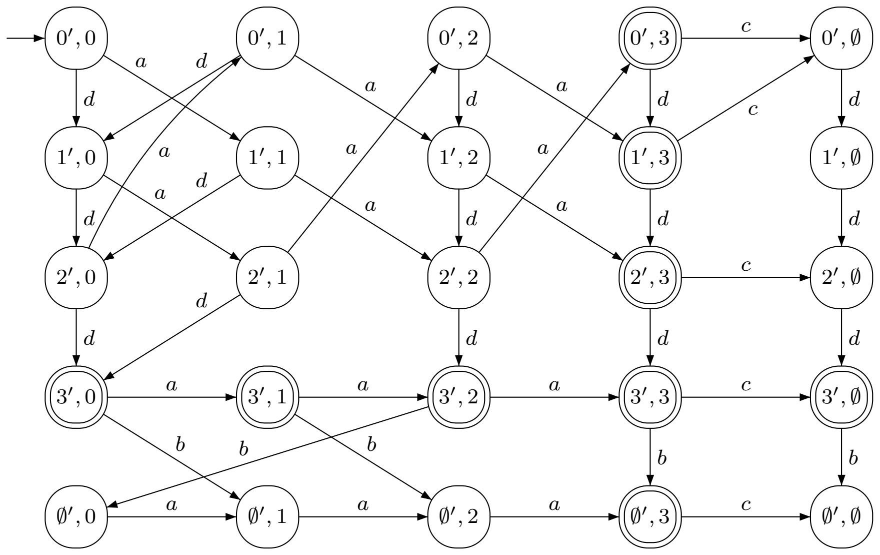


Figure 9: Partial illustration of the direct product for $L'_4(a,-,c,d) \cup L_4(b,-,d,a)$.

To compute $L'_m(a,-,c,d) \circ L_n(b,-,d,a)$, where $\circ$ is a boolean operation, add an empty state $\emptyset'$ to $\mathcal{D}'_m(a,-,c,d)$, and send all the transitions from any state of Q'_m under b to $\emptyset'$. Similarly, add an empty state $\emptyset$ to $\mathcal{D}_n(b,-,d,a)$ together with appropriate transitions; now the alphabets of the resulting DFAs are the same. The direct product of $L'_m(a,-,c,d)$ and $L_n(b,-,d,a)$ is illustrated in Figure 9 for $m = n = 4$.

As in the restricted case, the mn states of $Q'_m \times Q_n$ are reachable by words in $\{a,d\}^*$. The remaining states $(p',\emptyset)$ and $(\emptyset',q)$ are easily seen to be reachable using b and c, as well as a and d.

We now check distinguishability, which depends on the final states of the DFA. The direct product is made to recognize $L'_m(a,-,c,d) \circ L_n(b,-,d,a)$ by setting the final states to be $(\{(m-1)'\} \times Q_n \cup \{\emptyset\}) \circ (Q'_m \cup \{\emptyset'\} \times \{n-1\})$.

For union and symmetric difference, all states are pairwise distinguishable: States that differ in the first coordinate are distinguished by words in d^*c and states that differ in the second coordinate are distinguished by words in a^*b.

For difference, the final states are $((m-1)',q)$ for $q \neq n-1$. The alphabet of $L'_m(a,-,c,d) \setminus L_n(b,-,a,d)$ is $\{a,c,d\}$; hence we can omit b and delete all states $(\emptyset',q)$ and be left with a DFA recognizing the same language. The remaining states are distinguished by words in d^*c if they differ in the first coordinate or by words in a^*d^* if they differ in the second coordinate.

For intersection, the only final state is $((m-1)',n-1)$. The alphabet of $L'_m(a,-,c,d) \cap L_n(b,-,d,a)$ is $\{a,b\}$; hence we can omit b and c and delete all states $(p', \emptyset)$ and $(\emptyset', q)$. The remaining mn states are pairwise distinguishable as in the restricted case.

Note that the bound for difference is met by $L'_m(a, -, c, d) \setminus L_n(-, -, d, a)$, and that of intersection is met by $L'_m(a, -, -, d) \cap L_n(-, -, d, a)$. However the bounds for union and symmetric difference all require three letters in each dialect: There must be a letter in $\Sigma' \setminus \Sigma$ to reach states of the form $(p', \emptyset)$, and there must a letter in $\Sigma \setminus \Sigma'$ to reach states of the form $(\emptyset', q)$. As in regular languages, one letter in $\Sigma' \cap \Sigma$ is not sufficient to reach all the states of $Q'_m \times Q_n$ for all values of m and n; hence $|\Sigma' \cap \Sigma| \geq 2$ and so both Σ' and Σ must contain at least three letters.

It has been shown in [10] that at least two letters are needed for each right ideal that meets the bounds for star or reversal. Hence almost all our witnesses in Theorem 2 that meet the bounds for the common operations use minimal alphabets. □

6 Prefix-Closed Languages

The complexity of operations on prefix-closed languages was studied in [11], but most complex prefix-closed languages were not considered. As every prefix-closed language has an empty quotient, the restricted and unrestricted complexities are the same for binary operations.

Definition 3. *For $n \geq 4$, let $\mathcal{D}_n = \mathcal{D}_n(a, b, c, d) = (Q_n, \Sigma, \delta_n, 0, Q_n \setminus \{n-1\})$, where $\Sigma = \{a, b, c, d\}$, and δ_n is defined by the transformations a: $(0, \dots, n-2)$, b: $(0, 1)$, c: $(1 \to 0)$, and d: $\binom{0}{n-2}\, q \to q - 1 \pmod n$. Let $L_n = L_n(a, b, c, d)$ be the language accepted by $\mathcal{D}_n$. The structure of $\mathcal{D}_n(a, b, c, d)$ is shown in Figure 10.*

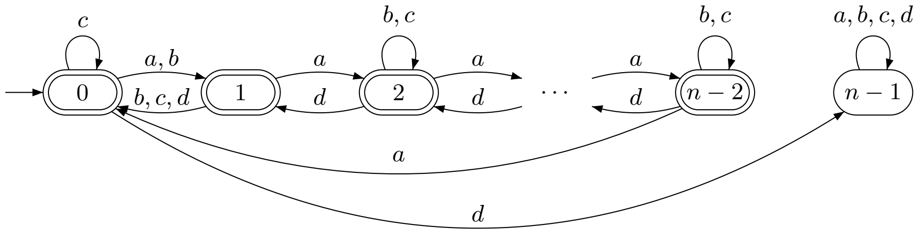


Figure 10: DFA of a most complex prefix-closed language.

Theorem 3 (Most Complex Prefix-Closed Languages)**.** *For each $n \geq 4$, the DFA of Definition 3 is minimal and $L_n(a, b, c, d)$ is a prefix-closed language of complexity n. The stream $(L_m(a, b, c, d) \mid m \geq 4)$ with some dialect streams is most complex in the class of prefix-closed languages. At least four letters are required to meet the bounds below.*

1. *The syntactic semigroup of $L_n(a,b,c,d)$ has cardinality n^{n-1}.*
2. *The quotients of $L_n(a,-,-,d)$ have complexity n, except for $\emptyset$, which has complexity 1.*
3. *The reverse of $L_n(a,-,-,d)$ has complexity 2^{n-1}, and $L_n(a,-,-,d)$ has 2^{n-1} atoms.*
4. *Each atom A_S of $L_n(a,b,c,d)$ has maximal complexity:*
$$\kappa(A_S) = \begin{cases} 2^{n-1}, & \text{if } S = \emptyset; \\ 1 + \sum_{x=1}^{n-|S|} \sum_{y=1}^{|S|} \binom{n-1}{x-1}\binom{n-x}{y}, & \text{if } \emptyset \subsetneq S \subsetneq Q_n. \end{cases}$$
5. *The star of $L_n(a,-,c,d)$ has complexity $2^{n-2}+1$.*
6. *The product $L'_m(a,b,c,d)L_n(a,d,b,c)$ has complexity $(m+1)2^{n-2}$.*
7. *For any proper binary boolean function $\circ$, the complexity of $L'_m(a,b,-,d) \circ L_n(b,a,-,d)$ is maximal. In particular, the complexity is mn if $\circ \in \{\cup, \oplus\}$, $mn-(n-1)$ if $\circ = \setminus$, and $mn-(m+n-2)$ if $\circ = \cap$.*

Proof. The DFA is minimal since state p rejects d^q if and only if $p < q$. It is prefix-closed because all non-empty states are final.

1. **Semigroup** Let d' induce the transformation $({}_0^{n-2}\, q \to q+1)$ (this was called d in the right-ideal section). Since $ada = d'$, the transition semigroup of the DFA of Figure 10 is the same as that of the DFA of the right ideal of Figure 4.
2. **Quotients** Obvious.
3. **Reversal** Since reversal commutes with complementation, we consider the complement of the language accepted by the DFA of Figure 10 restricted to the alphabet $\{a,d\}$. It was proved in [10] that the reverse of a right ideal has complexity at most 2^{n-1}, and in [17] that the number of atoms is the same as the complexity of the reverse. It remains to prove that all 2^{n-1} states of the DFA $\mathcal{D}^R$ obtained by the subset construction from the NFA $\mathcal{N}$ obtained by reversal of the DFA of the right ideal $\mathcal{D}$ are reachable and distinguishable. The proof is similar to that of [10]. Subset $\{n-1\}$ is the initial state of $\mathcal{N}$, and $n-1$ appears in every reachable state of $\mathcal{D}^R$. Every subset $\{n-1, q_2, q_3 \dots, q_k\}$ of size k, where $1 \le k \le n-2$ and $0 \le q_2 < q_3 < \cdots < q_k \le n-2$, is reached from the subset $\{n-1, q_3-(q_2+1), \dots, q_k-(q_2+1)\}$ of size $k-1$ by $da^{n-(q_2+1)}$. Since only state q, $0 \le q \le n-2$, accepts a^q, any two subsets differing by q are distinguishable by a^q.
4. **Atoms** Let L be a prefix-closed language with quotients $K_0, \dots, K_{n-1}$, $n \ge 4$. Recall that $\overline{L}$ is a right ideal with quotients $\overline{K_0}, \dots, \overline{K_{n-1}}$. For $S \subseteq \{0, \dots, n-1\}$, the atom of L corresponding to S is $A_S = \bigcap_{i \in S} K_i \cap \bigcap_{i \in \overline{S}} \overline{K_i}$. This can be rewritten as $\bigcap_{i \in \overline{S}} \overline{K_i} \cap \bigcap_{i \in \overline{\overline{S}}} \overline{\overline{K_i}}$, which is the atom of $\overline{L}$ corresponding

to $\overline{S}$; hence the sets of atoms of L and $\overline{L}$ are the same. The claim follows from the theorem for right ideals. The proof in [8] applies since the DFA that accepts the complement of the prefix-closed language of Figure 10 has all the transformations that fix $n-1$.

5. **Star** It was proved in [11] that $2^{n-2}+1$ is the maximal complexity of the star of a prefix-closed language. We now show that $L_n(a,-,c,d)$ meets this bound. Since $L_n(a,-,c,d)$ accepts ε, no new initial state is needed and it suffices to delete the empty state and add an ε-transition from each final state to the initial state to get an NFA $\mathcal{N}$ for L_n^*. In this NFA all 2^{n-2} subsets of Q_{n-1} containing 0 are reachable and pairwise distinguishable. Any non-empty set $\{0,q_2,q_3,\dots,q_k\}$ of size k with $0<q_2<q_3<\cdots<q_k\le n-2$ is reached from $\{0,q_3-q_2,\dots,q_k-q_2\}$ of size $k-1$ by $a(ac)^{q_2-1}$. Moreover, the empty set is reached from $\{0\}$ by d, giving the required bound. Sets $\{0\}\cup S$ and $\{0\}\cup T$ with $q\in S\oplus T$ are distinguished by $a^{n-2-q}d^{n-2}$.

6. **Product** It was shown in [11] that the complexity of the product of prefix-closed languages is $(m+1)2^{n-2}$. We now prove that our witness $L'_m(a,b,c,d)$ with minimal DFA $\mathcal{D}'_m(a,b,c,d)$ together with the dialect $L_n(a,d,b,c)$ with minimal DFA $\mathcal{D}_n(a,d,b,c)$ meets this bound. Construct the following NFA $\mathcal{N}$ for the product. Start with $\mathcal{D}'_m(a,b,c,d)$, but make all of its states non-final. Delete the empty state from $\mathcal{D}_n(a,d,b,c)$ and all the transitions to the empty state, add an ε-transition from each state $p'\in Q'_{m-1}$ to the initial state 0 of $\mathcal{D}_n(a,d,b,c)$. We will show that $(m-1)2^{n-2}$ states of the form $\{p',0\}\cup S$, where $S\subseteq Q_{n-1}\setminus\{0\}$, and 2^{n-1} states of the form $\{(m-1)'\}\cup S$, where $S\subseteq Q_{n-1}$ are reachable and pairwise distinguishable.

   The initial state of the subset automaton of $\mathcal{N}$ is $\{0',0\}$. State $\{1',0\}$ is reachable by b and $\{p',0\}$ for $2\le p\le m-2$ is reachable from $\{1',0\}$ by $(ab)^{p-1}$. State $\{p',0\}\cup S$ where $p'\in Q'_{m-1}$ and $S=\{q_1,\dots,q_k\}$ is reachable from $\{r',0,q_2-q_1,\dots,q_k-q_1\}$ by $a(ab)^{q_1-1}$ for some $r'\in Q'_{m-1}$. By induction, all $(m-1)2^{n-2}$ states $\{p',0\}\cup S$ are reachable. From $\{0',0\}\cup S$ by d^2 we reach $\{(m-1)',0\}\cup S$. Further apply ca to reach $\{(m-1)'\}\cup S$. Hence all 2^{n-1} subsets of the form $\{(m-1)'\}\cup S$ are reachable.

   We check that the states are pairwise distinguishable in four cases.

   a) $\{(m-1)'\}\cup S$ and $\{(m-1)'\}\cup T$ with $r\in S\oplus T$ are distinguished by $a^{n-2-r}c^{n-2}$.
   b) $\{p'\}\cup S$ and $\{p'\}\cup T$ with $r\in S\oplus T$ reduces to Case (a) by $a^{n-2-r}d^m$.
   c) $\{p'\}\cup S$ and $\{(m-1)'\}\cup T$ with $p'\in Q'_{m-1}$ are distinguished by c^n.
   d) $\{p'\}\cup S$ and $\{q'\}\cup T$ with $p<q\le m-2$ reduces to Case (c) by d^{p+1}.

7. **Boolean Operations** It is again convenient to consider the ideal languages defined by the complements of the prefix-closed languages of Figure 10 restricted to the alphabet $\{a,b,d\}$ and then use De Morgan's laws. Since every prefix-closed language has an empty quotient, it is sufficient to consider

boolean operations on languages over the same alphabet. The problems are the same as those in [9], except that there the transformation induced by d is $d:(n-2 \to n-1)$.

Let $\mathcal{D}_n(a,b,c,d)$ denote the DFA for the complement of the prefix-closed language of Definition 3 of complexity n and let L_n be the language accepted by $\mathcal{D}_n$. We consider boolean operations on right ideals L'_m and L_n.

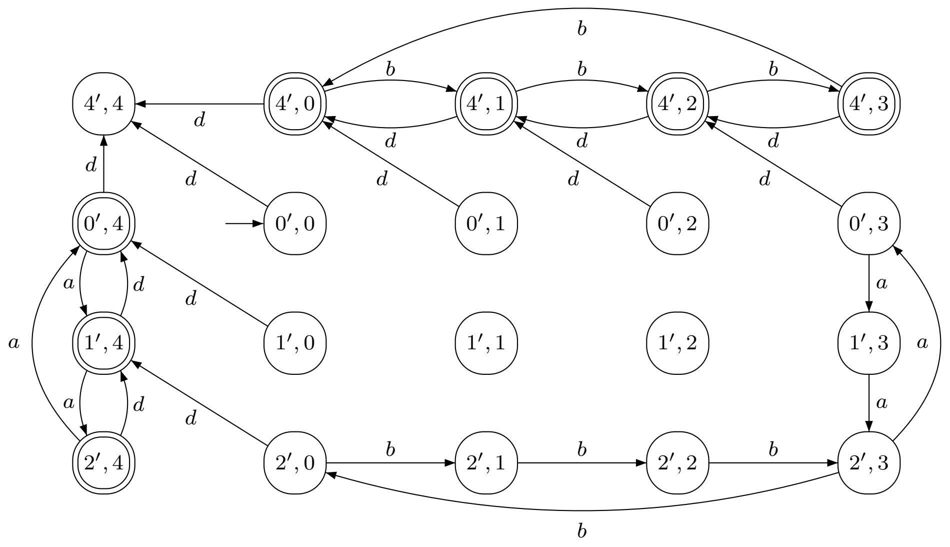


Figure 11: Partial illustration of the direct product for $L'_4(a,b,-,d) \oplus L_5(b,a,-,d)$.

The direct product is illustrated in Figure 11. The states in $Q'_{m-1} \times Q_{n-1}$ are reachable from the initial state $(0',0)$ by [2, Theorem 1] and computation in the case $m=n=4$. Then $((m-1)',0)$ is reached from $(0',1)$ by d and states of the form $((m-1)',q)$, $0 \le q \le n-2$, are then reached by words in b^*. Similarly, $(0',n-1)$ is reached from $(1',0)$ by d and states of the form $(p',n-1)$, $0 \le p \le m-2$, are then reached by words in a^*. Finally, $((m-1)',n-1)$ is reached from $((m-1)',0)$ by d. Hence all mn states are reachable.

Let $S = Q'_{m-1} \times Q_{n-1}$, $R = \{(m-1)'\} \times Q_n$, and $C = Q'_m \times \{n-1\}$. The final states of the direct product to recognize $L'_m(a,b,-,d) \circ L_n(b,a,-,d)$ are $R \circ C$.

Consider the following DFAs: $D'_{m-1}(a,b) = (Q'_{m-1}, \{a,b\}, \delta, 0', \{0'\})$ and $D_{n-1}(b,a) = (Q_{n-1}, \{a,b\}, \delta, 0, \{0\})$. By [2, Theorem 1], the states of S are pairwise distinguishable with respect to states $(\{0'\} \times Q_{n-1}) \circ \big(Q'_{m-1} \times \{0\}\big)$ for any $\circ \in \{\cup, \oplus, \setminus, \cap\}$. One can verify that if w distinguished two states of S with respect to $(\{0'\} \times Q_{n-1}) \circ \big(Q'_{m-1} \times \{0\}\big)$, then wd distinguishes them with respect to $R \circ C$ for each $\circ = \{\cup, \oplus, \setminus, \cap\}$. The rest of the argument depends on the operation $\circ \in \{\cup, \oplus, \setminus, \cap\}$.

$\cap, \oplus$: All mn states are pairwise distinguishable. The states of R are distinguished by words in d^*. The states of C are similarly distinguishable. The states of R are distinguished from the states of C by words in $\{a, d\}^*$. Every state of S is sent to a state of R by a word in $\{a, d\}^*$, and similarly to a state of C by a word in $\{b, d\}^*$; thus the states of S are distinguishable from the states of R or C.

$\setminus$: The states of C are all empty, leaving $m(n-1)+1$ distinguishable states. The states of R are distinguished by words in d^*.

$\cup$: The states of R and C are equivalent final states accepting all words, leaving $(m-1)(n-1)+1$ distinguishable states.

By De Morgan's laws we have $\kappa(L'_m \cup L_n) = \kappa(\overline{L'_m} \cap \overline{L_n})$, $\kappa(L'_m \oplus L_n) = \kappa(\overline{L'_m} \oplus \overline{L_n})$, $\kappa(L'_m \setminus L_n) = \kappa(\overline{L_n} \setminus \overline{L'_m})$, and $\kappa(L'_m \cap L_n) = \kappa(\overline{L'_m} \cup \overline{L_n})$. Thus the prefix-closed witness meets the bounds for boolean operations.

Since the semigroup of a prefix-closed language is the same as that of its complement, which is a right ideal, at least four letters are required to meet all the bounds. □

7 Prefix-Free Languages

The complexity of operations on prefix-free languages was studied in [19, 22, 23], but most complex prefix-free languages were not considered. As every prefix-free language has an empty quotient, the restricted and unrestricted complexities are the same for binary operations.

Definition 4. *For $n \geq 4$, let $\Sigma_n = \{a, b, c, d, e_0, \ldots, e_{n-3}\}$ and define the DFA $\mathcal{D}_n(\Sigma_n) = (Q_n, \Sigma_n, \delta_n, 0, \{n-2\})$, where δ_n is defined by the transformations a: $(n-2 \to n-1)(0, \ldots, n-3)$, b: $(n-2 \to n-1)(0, 1)$, c: $(n-2 \to n-1)(1 \to 0)$, d: $(0 \to n-2)(Q_n \setminus \{0\} \to n-1)$, e_q: $(n-2 \to n-1)(q \to n-2)$ for $q = 0, \ldots, n-3$. The transformations induced by a and b coincide when $n = 4$. Let $L_n(\Sigma_n)$ be the language accepted by $\mathcal{D}_n(\Sigma_n)$. The structure of $\mathcal{D}_n(\Sigma_n)$ is shown in Figure 12.*

Theorem 4. *For $n \geq 4$, the DFA of Definition 4 is minimal and $L_n(\Sigma)$ is a prefix-free language of complexity n. The stream $(L_n(a, b, c, d, e_0, \ldots, e_{n-3}) \mid n \geq 4)$ with dialect stream $(L_n(b, a, -, -, e_0, e_{n-3}) \mid n \geq 4)$ is most complex in the class of prefix-free languages. At least $n+2$ inputs are required to meet all the bounds below.*

1. *The syntactic semigroup of $L_n(a, b, c, -, e_0, \ldots, e_{n-3})$ has cardinality n^{n-2}. There is only one maximal transition semigroup of minimal DFAs accepting prefix-free languages. Moreover, fewer than $n+1$ inputs do not suffice to meet this bound.*

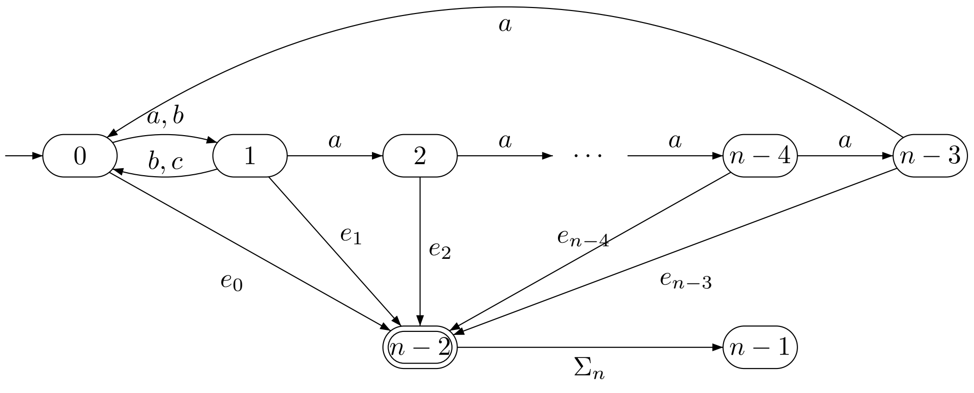


Figure 12: DFA of a most complex prefix-free language. Input d not shown; other missing transitions are self-loops.

2. *The quotients of $L_n(a,-,-,d)$ have complexity n, except for ε and $\emptyset$, which have complexity 2 and 1, respectively.*

3. *The reverse of $L_n(a,-,c,-,e_0)$ has complexity $2^{n-2}+1$, and $L_n(a,-,c,-,e_0)$ has $2^{n-2}+1$ atoms.*

4. *Each atom A_S of $L_n(a,b,c,-,e_0)$ has maximal complexity:*

$$\kappa(A_S) = \begin{cases} 2, & \textit{if } S = \{n-2\}; \\ 2^{n-1}, & \textit{if } S = \emptyset; \\ 2^{n-2}+1, & \textit{if } S = Q_{n-2}; \\ 2 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-2-|S|} \binom{n-2}{x}\binom{n-2-x}{y}, & \textit{if } \emptyset \subsetneq S \subsetneq Q_{n-2}. \end{cases}$$

5. *The star of $L_n(a,-,-,d)$ has complexity n.*

6. *The product $L'_m(a,-,-,d)L_n(a,-,-,d)$ has complexity $m+n-2$.*

7. *For $m,n \ge 4$ but $(m,n) \ne (4,4)$, and for any proper binary boolean function $\circ$, the complexity of $L_m(a,b,-,-,e_0,e_{m-3}) \circ L_n(b,a,-,-,e_0,e_{n-3})$ is maximal. In particular, these languages meet the bounds $mn-2$ for union and symmetric difference, $mn-2(m+n-3)$ for intersection, and $mn-(m+2n-4)$ for difference.*

Proof. Since only state q accepts $a^{n-2-q}d$ for $0 \le q \le n-3$, DFA $\mathcal{D}_n(a,-,-,d)$ is minimal. Since it has only one final state and that state accepts $\{\varepsilon\}$, $L_n(a,-,-,d)$ is prefix-free.

1. **Semigroup** The proof that the size of the semigroup is n^{n-2} is very similar to that in [12]. Inputs a, b, and c generate all transformations of Q_{n-2}. Moreover, any state $q \in Q_{n-2}$ can be sent to $n-2$ by e_q and to $n-1$ by $e_q e_q$.

Hence we have all n^{n-2} transformations of Q_n that fix $n-1$ and send $n-2$ to $n-1$. The maximal transition semigroup is unique, since it must contain all these transformations.

To prove that at least $n+1$ inputs are necessary, we see that $e_q\colon (n-2 \to n-1)(q \to n-2)$ is in the transition semigroup of $\mathcal{D}_n$. There are two types of states in $q \in Q_{n-2}$: those of Type 1, for which e_q is a generator (that is, the transformation of e_q is induced by a single letter), and those of Type 2, for which it is not. If e_q and e_p are generators, then clearly $e_p \neq e_q$.

If e_q is not a generator, then it must be a composition, $e_q = u_q v_q$, where u_q is in the semigroup and v_q is a generator. No state can be mapped by u_q to $n-2$ because then v_q would map $n-2$ to $n-1$. Hence u_q must be a permutation of Q_{n-2}. If $q \neq q'$ and e_q and $e_{q'}$ are not generators, then there exist u_q, v_q and $u_{q'}, v_{q'}$ as above, such that $e_q = u_q v_q$ and $e_{q'} = u_{q'} v_{q'}$. Then we must have $qu_q \neq q' u_{q'}$; otherwise both q and q' would be mapped to $n-2$. Hence $v_q \neq v_{q'}$ and all the generators of this type are distinct.

Finally, if e_q is a generator and $v_{q'}$ is as above, then $e_q \neq v_{q'}$, for otherwise $u_{q'}$ would be the identity and q' would be of Type 1. Therefore, $n-2$ generators are required in addition to those induced by a, b and c.

2. **Quotients** This is clear from the definition.
3. **Reversal** This was proved in [12].
4. **Atoms** First we establish an upper bound on the complexity of atoms of prefix-free languages. Let L be a prefix-convex language with n quotients $K_0, \dots, K_{n-1}$, in which K_{n-2} is final and $K_{n-1} = \emptyset$. Consider the intersection $A_S = \bigcap_{i \in S} K_i \cap \bigcap_{i \in \overline{S}} \overline{K_i}$, where $S \subseteq Q_n$, and $\overline{S} = Q_n \setminus S$. Clearly $n-1$ must be in $\overline{S}$ if A_S is an atom, for an atom must be non-empty. Since a prefix-free language has only one final state and that state accepts ε, if $n-2 \in S$, no other quotient is in S, for then A_S would not be an atom. Hence if $S = \{n-2\}$ then $A_S = \{\varepsilon\}$, and $\kappa(A_S) = 2$.

   Now suppose $S = \emptyset$; then $A_S = \bigcap_{i \in \overline{S}} \overline{K_i}$. Since K_{n-1} appears in every quotient of A_S, there are at most 2^{n-1} subsets of Q_{n-1} that can be reached from A_S together with $n-1$. Hence $\kappa(A_S) \le 2^{n-1}$.

   If $S = Q_{n-2}$, then $\overline{S} = \{n-2, n-1\}$ and $\bigcap_{i \in \overline{S}} \overline{K_i} = \Sigma^+$. If we reach $\overline{K_{n-1}} = \Sigma^*$, then any intersection which has $\{n-2, n-1\}$ in the complemented part is equivalent to one that has only $\{n-1\}$, since no quotient other than K_{n-2} contains ε. Hence we can reach at most $2^{n-2} - 1$ subsets of Q_{n-2}, along with the intersection $K_{n-2} \cap \overline{K_{n-1}} = \varepsilon$, and the empty quotient, for a total of $2^{n-2} + 1$ states.

   Finally, consider the case where $\emptyset \subsetneq S \subsetneq Q_{n-2}$. Then we have from 1 to $|S|$ uncomplemented quotients K_i with $i \in Q_{n-2}$, and from 1 to $n-2-|S|$ quotients K_i with $i \in Q_{n-2}$ in the complemented part; this leads to the formula given in the theorem.

It remains to be proved that the atoms of $L_n(a,b,c,-,e_0)$ meet these bounds. Atom $A_{\{n-2\}}$ is equal to $\{\varepsilon\}$ and thus has two quotients as required; assume now that $S \subseteq Q_{n-2}$. We are interested in the number of distinct quotients of $A_S = \bigcap_{i\in S} K_i \cap \bigcap_{i\in\overline{S}} \overline{K_i}$, where $S \subseteq Q_n \setminus \{n-1\}$. The quotients $w^{-1}A_S$ have the form $J_{X,Y} = \bigcap_{i\in X} K_i \cap \bigcap_{i\in Y} \overline{K_i}$ where $X = \{i \mid K_i = w^{-1}K_j \text{ for some } j \in S\}$ and $Y = \{i \mid K_i = w^{-1}K_j \text{ for some } j \in \overline{S}\}$. For brevity, we write $S \xrightarrow{w} X$ and $\overline{S} \xrightarrow{w} Y$; this notation is in agreement with the action of w on the states of $\mathcal{D}_n$ corresponding to S and $\overline{S}$.

Notice $J_{X,Y} = J_{X,Y\cup\{n-2\}}$ for all X and Y, except for the case $X = \{n-2\}$ in which $J_{X,Y} \in \{\{\varepsilon\},\emptyset\}$. Thus it is sufficient to assume $n-2 \notin Y$ from now on, as $\{J_{X,Y} \mid n-2 \notin Y, n-1 \in Y\}$ contains every quotient of A_S. We show that whenever $|X| \le |S|$, $|Y| \le |\overline{S}|$, $n-2 \notin Y$, and $n-1 \in Y$, there is a word $w \in \{a,b,c,e_0\}^*$ such that $S \xrightarrow{w} X$ and $\overline{S} \xrightarrow{w} Y$ and hence $J_{X,Y}$ is a quotient of A_S. When $S = Q_{n-2}$ we reach all quotients $J_{X,\{n-1\}}$ where $\emptyset \subsetneq X \subseteq Q_{n-2}$ by words in $\{a,b,c\}^*$, we reach $J_{\{n-2\},\{n-1\}}$ from $J_{\{0\},\{n-1\}}$ by e_0, and from there we reach the empty quotient by e_0. Similarly, when $\emptyset \subseteq S \subsetneq Q_{n-2}$, we reach $J_{X,Y}$ for $X \subseteq Q_{n-2}$ and $Y \cap Q_{n-2} \neq \emptyset$ by words in $\{a,b,c\}^*$, and the remaining quotients are easily reached using e_0.

It remains to show that non-empty quotients $J_{X,Y}$ and $J_{X',Y'}$ are distinct whenever $X \neq X'$ or $Y \neq Y'$. Notice $J_{X,Y} = \emptyset$ if either $X \cap Y \neq \emptyset$ or $\{n-2\} \subsetneq X$, and $J_{X,Y} = \{\varepsilon\}$ if and only if $X = \{n-2\}$. Apart from these special cases, every $J_{X,Y}$ is non-empty and does not contain ε.

For any $X \subseteq Q_{n-2}$, let w_X denote a word that maps $X \to \{n-2\}$ and $Q_n \setminus X \to \{n-1\}$; there is such a word in $\{a,b,c,e_0\}^*$ because $\{a,b,c\}^*$ contains u: $(n-2 \to n-1)(X \to n-3)(Q_{n-2}\setminus X \to 0)$, and then $w_X = ue_0ae_0$. Observe that $w_X \in K_i$ for all $i \in X$ and $w_X \notin K_j$ for all $j \notin X$. Hence, if $X \subseteq Q_{n-2}$ and $Y \subseteq Q_n \setminus X$, then $w_X \in J_{X,Y}$ and $w_{\overline{Y}\cap Q_{n-2}} \in J_{X,Y}$.

Let X' and Y' be any disjoint subsets of Q_n where $n-1 \in Y'$ and $J_{X',Y'} \neq \emptyset$. If $X' \neq X$ then either $w_X \notin J_{X',Y'}$ or $w_{X'} \notin J_{X,Y}$. Similarly, if $Y' \neq Y$ (and $Y \oplus Y' \neq \{n-2\}$) then either $w_{\overline{Y}\cap Q_{n-2}} \notin J_{X',Y'}$ or $w_{\overline{Y'}\cap Q_{n-2}} \notin J_{X,Y}$. Thus, any two quotients $J_{X,Y}$ and $J_{X',Y'}$, where $(X,Y) \neq (X',Y')$, are distinct.

When we established the upper bound on $\kappa(A_S)$, we counted the number of reachable, potentially distinct quotients $J_{X,Y}$ of each A_S. We have now shown that every reachable $J_{X,Y}$ is a quotient of A_S and determined all the cases when $J_{X,Y} = J_{X',Y'}$. It follows that every bound is met by $L_n(a,b,c,-,e_0)$.

5. **Star** Proved in [19]. For the purpose of proving that $n+2$ inputs are required for a most complex prefix-free witness, an outline of the proof is repeated here.

   Suppose that L is a prefix-free language with n quotients whose syntactic semigroup is maximal, and L^* has maximal complexity. We show that L requires an alphabet of size $n+2$. Towards a contradiction, let $\mathcal{D} = (Q_n, \Sigma, \delta, 0, \{n-2\})$ be a DFA for L where $|\Sigma| = n+1$. Assume $0,1,\dots,n-3$ are non-final, non-empty states, $n-2$ is the unique final state, and $n-1$ is the empty state. By [19], $\mathcal{D}$ must have this structure and $\delta(n-2, w) = n-1$ for any $w \in \Sigma^+$.

Since the syntactic semigroup of L is maximal, each letter of Σ has a specific role in $\mathcal{D}$ as described in **1** of this theorem. Three letters a', b', and c' are required to induce the transformations on Q_{n-2}; these letters cannot map any state of Q_{n-2} to $n-2$ or to $n-1$. An additional $n-2$ letters $v_0, v_1, \dots, v_{n-3}$ are required to generate $e_q\colon (n-2 \to n-1)(q \to n-2)$ for each $q \in Q_{n-2}$, where the action of e_q is induced by a word in $\{a', b', c'\}^* v_q$. Notice v_q cannot map any state of Q_{n-2} to $n-1$, since e_q does not. In summary, $\Sigma = \{a', b', c', v_0, \dots, v_{n-3}\}$ and for all $\ell \in \Sigma$ and $q \in Q_{n-2}$, $\delta(q, \ell) \neq n-1$.

An NFA for L^* is produced by adding to $\mathcal{D}$ a new initial state $0'$, which is final, adding an ε-transition from $n-2$ to 0, and deleting the empty state $n-1$. The transitions from $0'$ are exactly the same as the transitions from 0. Perform the subset construction on this NFA. The $n-1$ states $\{0'\}, \{0\}, \{1\}, \dots, \{n-3\}$ are all reachable and distinguishable by words in $\{a', b', c', v_0\}$. The only way to reach a set containing more than one state is by moving to $n-2$ and using the ε-transition. This leads to the state $\{0, n-2\}$, but applying any word $w \in \Sigma^+$ deletes $n-2$; thus, $\{0, n-2\}$ is the only reachable set with two or more states. However, $\{0'\}$ and $\{0, n-2\}$ are indistinguishable, since both are final and $\delta(\{0'\}, w) = \delta(\{0\}, w) = \delta(\{0, n-2\}, w)$ for $w \in \Sigma^+$.

So far, there are only $n-1$ reachable, distinguishable states in the subset construction. The remaining state is $\emptyset$, which can only be reached if there is a letter ℓ that moves from $q \in Q_{n-2}$ to $n-1$ in $\mathcal{D}$; a transition from $n-2$ to $n-1$ is not sufficient to reach the empty state. We showed that in our witness no $\ell \in \Sigma$ has $\delta(q, \ell) = n-1$. Therefore, $\kappa(L^*) \le n-1$, a contradiction. To achieve $\kappa(L^*) = n$, an additional letter is required. Therefore, any most complex prefix-free language stream requires $n+2$ inputs.

6. **Product** Proved in [19].

7. **Boolean Operations** Let $S = Q'_{m-2} \times Q_{n-2}$. For $0 \le p \le m-1$, let $R_p = \{(p', q) \mid q \in Q_n\}$, and for $0 \le q \le n-1$ let $C_q = \{(p', q) \mid p' \in Q'_m\}$. These are the sets of states in the rows and columns of Figure 13. The states in S are reachable from the initial state $(0', 0)$ by [2, Theorem 1]. Every other state in the direct product is reachable from some state in S, as illustrated in Figure 13.

   For $\circ \in \{\cup, \oplus, \setminus, \cap\}$, the direct product recognizes $L'_m \circ L_n$ if the final states are set to be $R_{m-2} \circ C_{n-2}$. Now we must determine which states are distinguishable with respect to $R_{m-2} \circ C_{n-2}$ for each value of $\circ$. Consider the DFAs $D'_m = (Q'_{m-2}, \{a, b\}, \delta, 0', \{(m-3)'\})$ and $D_n = (Q_{n-2}, \{b, a\}, \delta, 0, \{n-3\})$. By [2, Theorem 1], the states of S are pairwise distinguishable with respect to $(R_{m-3} \circ C_{n-3}) \cap S$. For any pair of states in S, let w be a word that distinguishes them in $(R_{m-3} \circ C_{n-3}) \cap S$; one verifies that further apply-

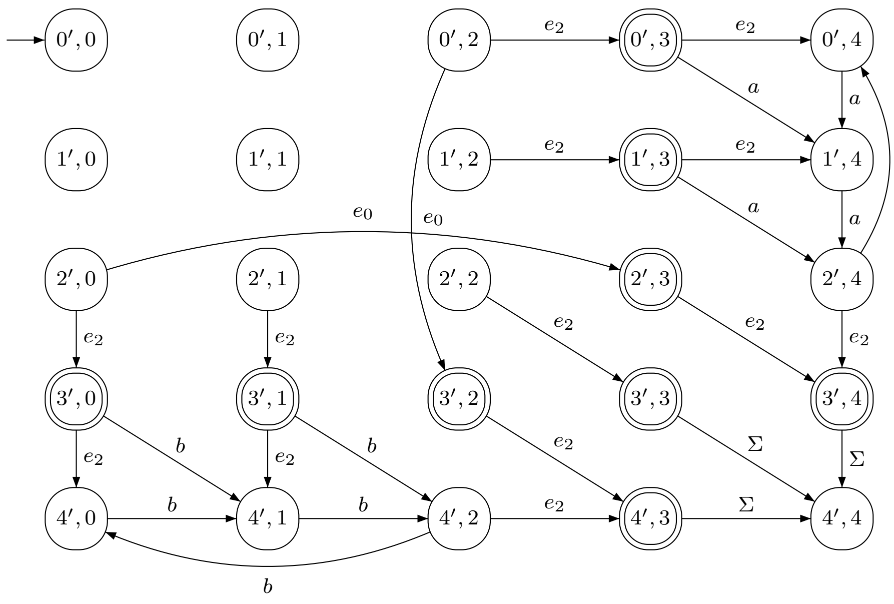


Figure 13: Partial illustration of the direct product for $L'_5(a, b, -, -, e_0, e_2) \cup L_5(b, a, -, -, e_0, e_2)$.

ing e_{m-3} distinguishes them with respect to $R_{m-2} \circ C_{n-2}$. The rest of the distinguishability argument depends on $\circ \in \{\cup, \oplus, \setminus, \cap\}$.

$\cup$: States $((m-1)', n-2)$, $((m-2)', n-1)$, and $((m-2)', n-2)$ are equivalent, since all three are final and any letter sends them to $((m-1)', n-1)$.

States of R_{m-1} are distinguished by words in b^*e_{m-3}. States of C_{n-1} are distinguished by words in a^*e_{m-3}. Excluding $((m-1)', n-2)$ and $((m-2)', n-1)$, which are equivalent, states of R_{m-1} are distinguished from states of C_{n-1} by words in a^*e_{m-3}.

States of $R_{m-2} \cup C_{n-2}$ are moved to states of $R_{m-1} \cup C_{n-1}$ by applying e_{m-3}. Excluding $((m-1)', n-2)$, $((m-2)', n-1)$, and $((m-2)', n-2)$, which are equivalent, every state is mapped by e_{m-3} to a different state of $R_{m-1} \cup C_{n-1}$; hence they are distinguishable.

Finally, we must show that states of S are distinguishable from the states of $R_{m-1} \cup C_{n-1}$. For any $(p', q) \in S$, there exists $w \in \{a, b\}^*$ such that $(p', q) \xrightarrow{w} (0', n-3)$, since both (p', q) and $(0', n-3)$ are reached from $(0', 0)$ by words in $\{a, b\}^*$, and a and b permute S. Then $(0', n-3) \xrightarrow{e_{m-3}} (0', n-2)$ and we have already shown that $(0', n-2)$ is distinguishable from all states in $R_{m-1} \cup C_{n-1}$. Thus, the $mn-2$ remaining states are pairwise distinguishable.

$\oplus$: States $((m-1)', n-2)$ and $((m-2)', n-1)$ are equivalent, and states

$((m-2)', n-2)$ and $((m-1)', n-1)$ are equivalent. The rest of the states are distinguishable by an argument similar to that of union.

$\cap$: State $((m-2)', n-2)$ is the only final state. The remaining non-final states of $R_{m-2} \cup R_{m-1} \cup C_{n-2} \cup C_{n-1}$ are all empty. Clearly the states of S are non-empty, since $((m-3)', n-3) \xrightarrow{e_{m-3}} ((m-2)', n-2)$. Thus, the remaining $mn - 2(m+n-3)$ states are pairwise distinguishable.

$\setminus$: The states of R_{m-1} and $((m-2)', n-2)$ are all equivalent. States $((m-1)', q)$ and $((m-2)', q)$ are equivalent for $0 \le q \le m-3$. The final states $(R_{m-2} \setminus \{((m-2)', n-2)\})$ are all equivalent.

The states of C_{n-1} are distinguished by words in a^*e_{m-3}. It remains to show that states of S are distinguishable from the states of C_{n-1}. Notice $((m-3)', n-3)$ is distinguished from $((m-3)', n-1)$ by e_{m-3}, and from every other state of C_{n-1} by be_{m-3}. For any state of S, there exists $w \in \{a, b\}^*$ that sends that state to $((m-3)', n-3)$, and notice $C_{n-1}w \subseteq C_{n-1}$. So all $mn - (m+2n-4)$ remaining states are pairwise distinguishable.

Note that stream $L_m(a, b, c, d, e_0, e_{m-3})$ with dialect $L_n(b, a, c, d, e_0, e_{m-3})$ meets the bounds for quotients, reversal, atomic complexity, star, product and boolean operations. □

Using some results from [22, 23] we define another prefix-free witness stream that meets all the bounds except those for syntactic complexity and atom complexity. *Moreover, all the bounds are met by dialects over minimal alphabets.*

Definition 5. *For $n \ge 4$, let $\mathcal{D}_n(a, c, d, e, f, g) = (Q_n, \Sigma, \delta_n, 0, \{n-2\})$, where $\Sigma = \{a, c, d, e, f, g\}$, and δ_n is defined by the transformations*

- a: $(n-2 \to n-1)(0, \dots, n-3)$,
- c: $(n-2 \to n-1)(1 \to 0)$.
- d: $(0 \to n-2)(Q_n \setminus \{0\} \to n-1)$,
- e: $(n-2 \to n-1)(n-3 \to n-2)$,
- f: $(n-2 \to n-1)(_0^{n-2}\ q \to q+1)$,
- g: $(n-2 \to n-1)$.

Note that b is not used, a, c, d, and e induce the same transformations as a, c, d, and e_{n-3} in Definition 4. DFA $\mathcal{D}_n(\Sigma)$ is shown in Figure 14. Let $L_n(\Sigma)$ be the language accepted by $\mathcal{D}_n(\Sigma)$.

Proposition 1. *For $n \ge 4$, the DFA of Definition 5 is minimal and $L_n(\Sigma)$ is a prefix-free language of complexity n. Moreover, all the witnesses for individual operations have minimal alphabets.*

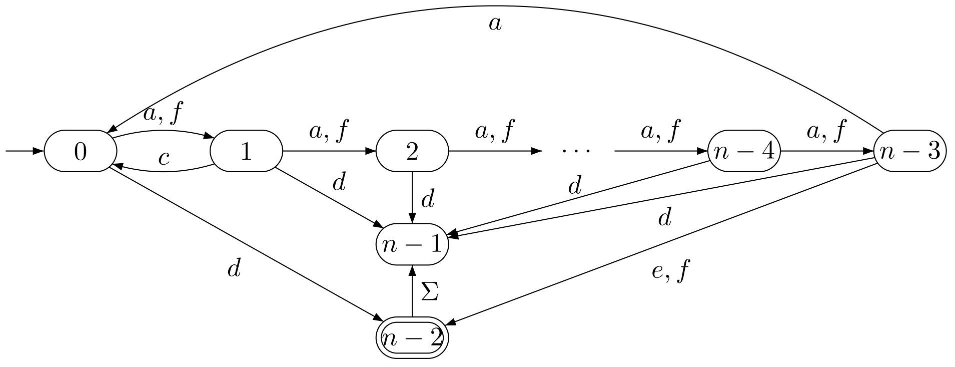


Figure 14: DFA $\mathcal{D}_n(\Sigma)$ of Definition 5; missing transitions are self-loops.

1. *The quotients of $L_n(a, -, -, -, f)$ have complexity n, except for the quotient ε and the empty quotient, which have complexity 2 and 1 respectively.*
2. *The reverse of $L_n(a, c, -, e)$ has complexity $2^{n-2} + 1$, and $L_n(a, -, c, d)$ has $2^{n-2} + 1$ atoms.*
3. *The star of $L_n(a, -, -, d)$ has complexity n.*
4. *For $m, n \geq 4$, $\kappa(L_m(-, -, -, -, f)L_n(-, -, -, -, f)) = m + n - 2$.*
5. *a) $\kappa(L_m(-, -, -, -, f, g) \cup L_n(-, -, -, -, g, f)) = \kappa(L_m(-, -, -, -, f, g) \oplus L_n(-, -, -, -, g, f)) = mn - 2$.*
   b) $\kappa(L_m(a, -, -, e, -, -) \setminus L_n(-, -, -, -, e, a)) = mn - (m + 2n - 4)$.
   c) $\kappa(L_m(a, -, -, e, -, -) \cap L_n(-, -, -, -, e, a)) = mn - 2(m + n - 3)$.

Proof. The first claim is obvious. The second and third claims were proved in Theorem 4. (A ternary witness was also used in [22] for the reverse, but it had more complicated transitions than our witness.) The fourth claim is from [22]. The results for union, symmetric difference and intersection were proved in [22], and that for difference in [23]. $\square$

8 Conclusions

Our results are summarized in Table 1. The largest bounds are shown in boldface type, and they are reached in the classes of ideal and closed languages. Recall that for regular languages we have the following results: semigroup: n^n; reverse: 2^n; star: $2^{n-1} + 2^{n-2}$; restricted product: $(m - 1)2^n + 2^{n-1}$; unrestricted product: $m2^n + 2^{n-1}$; restricted $\cup$ and $\oplus$: mn; unrestricted $\cup$ and $\oplus$: $(m + 1)(n + 1)$; restricted $\setminus$: mn; unrestricted $\setminus$: $mn + m$; restricted $\cap$: mn; unrestricted $\cap$: mn.

Table 1: Complexities of special prefix-convex languages

	Right-Ideal	Prefix-Closed	Prefix-Free
Semigroup	$\mathbf{n^{n-1}}$	$\mathbf{n^{n-1}}$	n^{n-2}
Reverse	$\mathbf{2^{n-1}}$	$\mathbf{2^{n-1}}$	$2^{n-2}+1$
Star	$n+1$	$\mathbf{2^{n-2}+1}$	n
Product restricted	$m+2^{n-2}$	$\mathbf{(m+1)2^{n-2}}$	$m+n-2$
Product unrestr.	$m+2^{n-1}+2^{n-2}+1$	$\mathbf{(m+1)2^{n-2}}$	$m+n-2$
$\cup$ restricted	$mn-(m+n-2)$	$\mathbf{mn}$	$mn-2$
$\cup$ unrestricted	$\mathbf{(m+1)(n+1)}$	mn	$mn-2$
$\oplus$ restricted	$\mathbf{mn}$	$\mathbf{mn}$	$mn-2$
$\oplus$ unrestricted	$\mathbf{(m+1)(n+1)}$	mn	$mn-2$
$\setminus$ restricted	$\mathbf{mn-(m-1)}$	$\mathbf{mn-(n-1)}$	$mn-(m+2n-4)$
$\setminus$ unrestricted	$\mathbf{mn+m}$	$\mathbf{mn-(n-1)}$	$mn-(m+2n-4)$
$\cap$ restr. and unrestr.	$\mathbf{mn}$	$mn-(m+n-2)$	$mn-2(m+n-3)$

References

[1] Ang, T. and Brzozowski, J. A. Languages convex with respect to binary relations, and their closure properties. *Acta Cybernet.*, 19(2):445–464, 2009.

[2] Bell, J., Brzozowski, J. A., Moreira, N., and Reis, R. Symmetric groups and quotient complexity of boolean operations. In Esparza, J. and et al., editors, *ICALP 2014*, volume 8573 of *LNCS*, pages 1–12. Springer Berlin / Heidelberg, 2014.

[3] Berstel, J., Perrin, D., and Reutenauer, C. *Codes and Automata (Encyclopedia of Mathematics and its Applications).* Cambridge University Press, 2010.

[4] Brzozowski, J. A. Quotient complexity of regular languages. *J. Autom. Lang. Comb.*, 15(1/2):71–89, 2010.

[5] Brzozowski, J. A. In search of the most complex regular languages. *Int. J. Found. Comput. Sci,*, 24(6):691–708, 2013.

[6] Brzozowski, J. A. Unrestricted state complexity of binary operations on regular languages. In C. Câmpeanu, F. Manea and Shallit, J., editors, *DCFS 2016*, volume 9777 of *LNCS*, pages 60–72. Springer Berlin / Heidelberg, 2016.

[7] Brzozowski, J. A. and Davies, G. Maximally atomic languages. In Ësik, Z. and Fülop, Z., editors, *Automata and Formal Languages (AFL 2014)*, pages 151–161. EPTCS, 2014.

[8] Brzozowski, J. A. and Davies, S. Quotient complexities of atoms in regular ideal languages. *Acta Cybernet.*, 22(2):293–311, 2015.

[9] Brzozowski, J. A., Davies, S., and Liu, B. Y. V. Most complex regular ideal languages. *Discrete Math. Theoret. Comput. Sc.*, 18(3), 2016. Paper #15.

[10] Brzozowski, J. A., Jirásková, G., and Li, B. Quotient complexity of ideal languages. *Theoret. Comput. Sci.*, 470:36–52, 2013.

[11] Brzozowski, J. A., Jirásková, G., and Zou, C. Quotient complexity of closed languages. *Theory Comput. Syst.*, 54:277–292, 2014.

[12] Brzozowski, J. A., Li, B., and Ye, Y. Syntactic complexity of prefix-, suffix-, bifix-, and factor-free regular languages. *Theoret. Comput. Sci.*, 449:37–53, 2012.

[13] Brzozowski, J. A. and Liu, B. Quotient complexity of star-free languages. *Int. J. Found. Comput. Sci.*, 23(6):1261–1276, 2012.

[14] Brzozowski, J. A. and Sinnamon, C. Unrestricted state complexity of binary operations on regular and ideal languages. *J. Autom. Lang. Comb.* To appear. See also `http://arxiv.org/abs/1609.04439`.

[15] Brzozowski, J. A., Szykuła, M., and Ye, Y. Syntactic complexity of regular ideals. `http://arxiv.org/abs/1509.06032`.

[16] Brzozowski, J. A. and Tamm, H. Quotient complexities of atoms of regular languages. *Int. J. Found. Comput. Sci.*, 24(7):1009–1027, 2013.

[17] Brzozowski, J. A. and Tamm, H. Theory of átomata. *Theoret. Comput. Sci.*, 539:13–27, 2014.

[18] Brzozowski, J. A. and Ye, Y. Syntactic complexity of ideal and closed languages. In Mauri, G. and Leporati, A., editors, *DLT 2011*, volume 6795 of *LNCS*, pages 117–128. Springer Berlin / Heidelberg, 2011.

[19] Han, Y.-S., Salomaa, K., and Wood, D. Operational state complexity of prefix-free regular languages. In Ésik, Z. and Fülöp, Z, editors, *Automata, Formal Languages, and Related Topics*, pages 99–115. Institute of Informatics, University of Szeged, Hungary, 2009.

[20] Holzer, M. and König, B. On deterministic finite automata and syntactic monoid size. *Theoret. Comput. Sci.*, 327(3):319–347, 2004.

[21] Iván, S. Complexity of atoms, combinatorially. *Inform. Process. Lett.*, 116(5):356–360, 2016.

[22] Jirásková, G. and Krausová, M. Complexity in prefix-free regular languages. In McQuillan, I., Pighizzini, G., and Trost, B., editors, *Proceedings of the 12th International Workshop on Descriptional Complexity of Formal Systems* (*DCFS*), pages 236–244. University of Saskatchewan, 2010.

[23] Krausová, M. Prefix-free regular languages: Closure properties, difference, and left quotient. In Kotásek, Z., Bouda, J., Cerná, I., Sekanina, L., Vojnar, T., and Antos, D., editors, *MEMICS*, volume 7119 of *Lecture Notes in Computer Science*, pages 114–122. Springer Berlin / Heidelberg, 2011.

[24] Krawetz, B., Lawrence, J., and Shallit, J. State complexity and the monoid of transformations of a finite set. In Domaratzki, M., Okhotin, A., Salomaa, K., and Yu, S., editors, *Proceedings of the Implementation and Application of Automata,* (*CIAA*), volume 3317 of *LNCS*, pages 213–224. Springer Berlin / Heidelberg, 2005.

[25] Myhill, J. Finite automata and representation of events. *Wright Air Development Center Technical Report*, 57–624, 1957.

[26] Pin, J.-E. Syntactic semigroups. In *Handbook of Formal Languages, vol. 1: Word, Language, Grammar*, pages 679–746. Springer, New York, NY, USA, 1997.

[27] Salomaa, A., Wood, D., and Yu, S. On the state complexity of reversals of regular languages. *Theoret. Comput. Sci.*, 320:315–329, 2004.

[28] Thierrin, G. Convex languages. In Nivat, M., editor, *Automata, Languages and Programming*, pages 481–492. North-Holland, 1973.

[29] Yu, S. State complexity of regular languages. *J. Autom. Lang. Comb.*, 6:221–234, 2001.

Acta Cybernetica **23** (2017) 43–59.

A Kleene Theorem for Weighted ω-Pushdown Automata[*]

Manfred Droste[a] and Werner Kuich[b]

Abstract

Weighted ω-pushdown automata were introduced as generalization of the classical pushdown automata accepting infinite words by Büchi acceptance. The main result in the proof of the Kleene Theorem is the construction of a weighted ω-pushdown automaton for the ω-algebraic closure of subsets of a continuous star-omega semiring.

1 Introduction

Weighted ω-pushdown automata were introduced by Droste, Kuich [4] as generalization of the classical pushdown automata accepting infinite words by Büchi acceptance (see Cohen, Gold [2]). To achieve the Kleene Theorem, the following result is needed.

Let S be a continuous star-omega semiring and let (s, v), $s, v \in S$, with $v = \sum_{1 \leq k \leq m} s_k t_k^\omega$ be a pair, where s, s_k, t_k, $1 \leq k \leq m$, are algebraic elements. Then an ω-pushdown automaton $\mathcal{P}$ can be constructed whose behavior $\|\mathcal{P}\|$ equals (s, v). The construction is split into three lemmas for the construction of t_k^ω, $s_k t_k^\omega$ and v.

This proves a Kleene Theorem that is in some aspects a generalization of Theorem 4.1.8 of Cohen, Gold [2].

The paper consists of this and three more sections. In Section 2 we refer the necessary preliminaries from the theories of semirings and semiring-semimodule pairs. In Section 3, we present some definitions and results from Droste, Kuich [4] that are needed in Section 4. In the last section, existing results in connection with the Kleene Theorem are quoted and the already mentioned constructions on ω-pushdown automata are performed.

[*]The second author was partially supported by Austrian Science Fund (FWF): grant no. I1661 - N25

[a]Institut für Informatik, Universität Leipzig, Leipzig, Germany, E-mail: `droste@informatik.uni-leipzig.de`

[b]Institut für Diskrete Mathematik und Geometrie, Technische Universität Wien, Wien, Austria, E-mail: `kuich@tuwien.ac.at`

DOI: 10.14232/actacyb.23.1.2017.4

2 Preliminaries

For the convenience of the reader, we quote definitions and results of Ésik, Kuich [6, 7, 9] from Ésik, Kuich [10]. The reader should be familiar with Sections 5.1-5.6 of Ésik, Kuich [10].

A semiring S is called *complete* if it is possible to define sums for all families $(a_i \mid i \in I)$ of elements of S, where I is an arbitrary index set, such that the following conditions are satisfied (see Conway [3], Eilenberg [5], Kuich [11]):

$$\text{(i)}\quad \sum_{i\in\emptyset} a_i = 0, \qquad \sum_{i\in\{j\}} a_i = a_j, \qquad \sum_{i\in\{j,k\}} a_i = a_j + a_k \text{ for } j \neq k\,,$$

$$\text{(ii)}\quad \sum_{j\in J}\Big(\sum_{i\in I_j} a_i\Big) = \sum_{i\in I} a_i\,, \text{ if } \bigcup_{j\in J} I_j = I \text{ and } I_j \cap I_{j'} = \emptyset \text{ for } j \neq j'\,,$$

$$\text{(iii)}\quad \sum_{i\in I}(c\cdot a_i) = c\cdot\Big(\sum_{i\in I} a_i\Big), \qquad \sum_{i\in I}(a_i\cdot c) = \Big(\sum_{i\in I} a_i\Big)\cdot c\,.$$

This means that a semiring S is complete if it is possible to define "infinite sums" (i) that are an extension of the finite sums, (ii) that are associative and commutative and (iii) that satisfy the distribution laws.

A semiring S equipped with an additional unary star operation $^* : S \to S$ is called a starsemiring. In complete semirings for each element a, the *star* a^* of a is defined by

$$a^* = \sum_{j\geq 0} a^j\,.$$

Hence, each complete semiring is a starsemiring, called a *complete starsemiring.* A *Conway semiring* (see Conway [3], Bloom, Ésik [1]) is a starsemiring S satisfying the *sum star identity*

$$(a+b)^* = a^*(ba^*)^*$$

and the *product star identity*

$$(ab)^* = 1 + a(ba)^*b$$

for all $a, b \in S$. Observe that by Ésik, Kuich [10], Theorem 1.2.24, each complete starsemiring is a Conway semiring.

Suppose that S is a semiring and V is a commutative monoid written additively. We call V a (left) S-semimodule if V is equipped with a (left) action

$$\begin{aligned} S\times V &\to V \\ (s,v) &\mapsto sv \end{aligned}$$

subject to the following rules:

$$s(s'v) = (ss')v\,, \quad (s+s')v = sv + s'v\,, \quad s(v+v') = sv + sv'\,,$$
$$1v = v\,, \quad 0v = 0\,, \quad s0 = 0\,,$$

for all $s, s' \in S$ and $v, v' \in V$. When V is an S-semimodule, we call (S, V) a *semiring-semimodule pair.*

Suppose that (S, V) is a semiring-semimodule pair such that S is a starsemiring and S and V are equipped with an omega operation ${}^\omega : S \to V$. Then we call (S, V) a *starsemiring-omegasemimodule pair.* Following Bloom, Ésik [1], we call a starsemiring-omegasemimodule pair (S, V) a *Conway semiring-semimodule pair* if S is a Conway semiring and if the omega operation satisfies the *sum omega identity* and the *product omega identity*:

$$(a+b)^\omega = (a^*b)^\omega + (a^*b)^*a^\omega \qquad \text{and} \qquad (ab)^\omega = a(ba)^\omega,$$

for all $a, b \in S$. It then follows that the *omega fixed-point equation* holds, i.e.

$$aa^\omega = a^\omega,$$

for all $a \in S$.

Ésik, Kuich [8] define a *complete semiring-semimodule pair* to be a semiring-semimodule pair (S, V) such that S is a complete semiring and V is a complete monoid with

$$s\big(\sum_{i\in I} v_i\big) = \sum_{i\in I} sv_i \qquad \text{and} \qquad \big(\sum_{i\in I} s_i\big)v = \sum_{i\in I} s_i v\,,$$

for all $s \in S$, $v \in V$, and for all families $(s_i)_{i\in I}$ over S and $(v_i)_{i\in I}$ over V; moreover, it is required that an *infinite product operation*

$$(s_1, s_2, \ldots) \;\mapsto\; \prod_{j\geq 1} s_j$$

is given mapping infinite sequences over S to V subject to the following three conditions:

$$\begin{aligned}
\prod_{i\geq 1} s_i \;&=\; \prod_{i\geq 1}(s_{n_{i-1}+1}\cdot \cdots \cdot s_{n_i})\\
s_1 \cdot \prod_{i\geq 1} s_{i+1} \;&=\; \prod_{i\geq 1} s_i\\
\prod_{j\geq 1}\sum_{i_j\in I_j} s_{i_j} \;&=\; \sum_{(i_1,i_2,\ldots)\in I_1\times I_2\times\ldots}\;\prod_{j\geq 1} s_{i_j}\,,
\end{aligned}$$

where in the first equation $0 = n_0 \leq n_1 \leq n_2 \leq \ldots$ and $I_1, I_2, \ldots$ are arbitrary index sets. Suppose that (S, V) is complete. Then we define

$$s^* \;=\; \sum_{i\geq 0} s^i \qquad \text{and} \qquad s^\omega \;=\; \prod_{i\geq 1} s\,,$$

for all $s \in S$. This turns (S, V) into a starsemiring-omegasemimodule pair. By Ésik, Kuich [8], each complete semiring-semimodule pair is a Conway semiring-semimodule pair. Observe that, if (S, V) is a complete semiring-semimodule pair, then $0^\omega = 0$.

A *star-omega semiring* is a semiring S equipped with unary operations * and $^\omega : S \to S$. A star-omega semiring S is called *complete* if (S, S) is a complete semiring semimodule pair, i.e., if S is complete and is equipped with an infinite product operation that satisfies the three conditions stated above.

A commutative monoid $(V, +, 0)$ is *continuous* (cf. Section 2.2 of [10]) if it is equipped with a a partial order $\leq$ such that the supremum of any chain exists and 0 is the least element. Moreover, the sum operation $+$ is continuous:

$$x + \sup Y \ = \ \sup(x + Y)$$

for all nonempty chains, where $x + Y = \{x + y : y \in Y\}$. (Actually this also holds when the set is empty.) It follows that the sum operation is monotonic: if $x \leq y$ in V, then $x + z \leq y + z$ for all $z \in V$.

Suppose now that $S = (S, +, \cdot, 0, 1)$ is a semiring. We say that S is a *continuous semiring* (cf. Section 2.2 of [10]) if $(S, +, 0)$ is a continuous commutative monoid equipped with a partial order $\leq$ and the product operation is continuous (hence, also monotonic), i.e., it preserves the supremum of nonempty chains in either argument:

$$\begin{aligned} (\sup X)y &= \sup(Xy) \\ y(\sup X) &= \sup(yX) \ , \end{aligned}$$

for all nonempty chains $X \subseteq S$, where $Xy = \{xy \ : \ x \in X\}$ and yX is defined in the same way.

By Corollary 2.2.2 of Ésik, Kuich [10] any continuous semiring is complete.

3 Weighted ω-pushdown automata

Weighted ω-pushdown automata were introduced by Droste, Kuich [4] as generalization of the classical pushdown automata accepting infinite words by Büchi acceptance (see Cohen, Gold [2]). In this section we refer to definitions and results of Droste, Kuich [4] that are needed for this paper.

Following Kuich, Salomaa [12] and Kuich [11], we introduce pushdown transitions matrices. Let Γ be an alphabet, called *pushdown alphabet* and let $n \geq 1$. A matrix $M \in (S^{n\times n})^{\Gamma^*\times\Gamma^*}$ is termed a *pushdown transition matrix* (with *pushdown alphabet* Γ and *stateset* $\{1, \dots, n\}$) if

(i) for each $p \in \Gamma$ there exist only finitely many blocks $M_{p,\pi}$, $\pi \in \Gamma^*$, that are unequal to 0;

(ii) for all $\pi_1, \pi_2 \in \Gamma^*$,

$$M_{\pi_1,\pi_2} = \begin{cases} M_{p,\pi} & \text{if there exist } p \in \Gamma, \pi, \pi' \in \Gamma^* \text{ with } \pi_1 = p\pi', \pi_2 = \pi\pi', \\ 0 & \text{otherwise.} \end{cases}$$

For the remaining of this paper, $M \in (S^{n\times n})^{\Gamma^*\times\Gamma^*}$ will denote a pushdown transition matrix with pushdown alphabet Γ and stateset $\{1,\dots,n\}$.

When we say "G is the graph with adjacency matrix $M \in (S^{n\times n})^{\Gamma^*\times\Gamma^*}$" then it means that G is the graph with adjacency matrix $M' \in S^{(\Gamma^*\times n)\times(\Gamma^*\times n)}$, where M' corresponds to M with respect to the canonical isomorphism between $((S^{n\times n})^{\Gamma^*\times\Gamma^*}$ and $S^{(\Gamma^*\times n)(\Gamma^*\times n)}$.

Let now M be a pushdown transition matrix and $0 \le k \le n$. Then $M^{\omega,k}$ is the column vector in $(S^n)^{\Gamma^*}$ defined as follows: For $\pi \in \Gamma^*$ and $1 \le i \le n$, let $((M^{\omega,k})_\pi)_i$ be the sum of all weights of paths in the graph with adjacency matrix M that have initial vertex (π, i) and visit vertices (π', i'), $\pi' \in \Gamma^*$, $1 \le i' \le k$, infinitely often. Observe that $M^{\omega,0} = 0$ and $M^{\omega,n} = M^\omega$.

Let $P_k = \{(j_1, j_2, \dots) \in \{1,\dots,n\}^\omega \mid j_t \le k$ for infinitely many $t \ge 1\}$.

Then for $\pi \in \Gamma^+$, $1 \le j \le n$, we obtain

$$((M^{\omega,k})_\pi)_j = \sum_{\pi_1,\pi_2,\dots\in\Gamma^+} \sum_{(j_1,j_2,\dots)\in P_k} (M_{\pi,\pi_1})_{j,j_1}(M_{\pi_1,\pi_2})_{j_1,j_2}(M_{\pi_2,\pi_3})_{j_2,j_3}\dots .$$

For the definition of an S'-algebraic system over a quemiring $S \times V$ we refer the reader to [10], page 136, and for the definition of quemirings to [10], page 110. Here we note that a quemiring T is isomorphic to a quemiring $S \times V$ determined by the semiring-semimodule pair (S, V), cf. [10], page 110.

Let $S' \subseteq S$, with $0, 1 \in S'$, and let $M \in (S'^{n\times n})^{\Gamma^*\times\Gamma^*}$ be a pushdown matrix. Consider the $S'^{n\times n}$-algebraic system over the complete semiring-semimodule pair $(S^{n\times n}, S^n)$

$$y_p = \sum_{\pi\in\Gamma^*} M_{p,\pi} y_\pi\,,\ p \in \Gamma\,. \tag{1}$$

(See Section 5.6 of Ésik, Kuich [10].) The variables of this system (1) are $y_p, p \in \Gamma$, and $y_\pi, \pi \in \Gamma^*$, is defined by $y_{p\pi} = y_p y_\pi$ for $p \in \Gamma$, $\pi \in \Gamma^*$ and $y_\varepsilon = 1$. Hence, for $\pi = p_1 \dots p_k$, $y_\pi = y_{p_1} \dots y_{p_k}$. The variables y_p are variables for $(S^{n\times n}, S^n)$.

Let $x = (x_p)_{p\in\Gamma}$, where x_p, $p \in \Gamma$, are variables for $S^{n\times n}$. Then, for $p \in \Gamma$, $\pi = p_1 p_2 \dots p_k$, $(M_{p,\pi} y_\pi)_x$ is defined to be

$$\begin{aligned}&(M_{p,\pi} y_\pi)_x\\ &= (M_{p,\pi} y_{p_1} \dots y_{p_k})_x\\ &= M_{p,\pi} z_{p_1} + M_{p,\pi} x_{p_1} z_{p_2} + \dots + M_{p,\pi} x_{p_1} \dots x_{p_{k-1}} z_{p_k}\,.\end{aligned}$$

Here z_p, $p \in \Gamma$, are variables for S^n.

We obtain, for $p \in \Gamma$, $\pi = p_1 \dots p_k$,

$$\begin{aligned}(M_{p,\pi} y_\pi)_x &= \sum_{p'\in\Gamma} \sum_{\substack{\pi=p_1\dots p_k\in\Gamma^+\\ p_j=p'}} M_{p,\pi} x_{p_1} \dots x_{p_{j-1}} z_{p'}\\ &= \sum_{\pi=p_1\dots p_k\in\Gamma^+} M_{p,\pi} \sum_{1\le j\le k} x_{p_1} \dots x_{p_{j-1}} z_{p_j}\,.\end{aligned}$$

The system (1) induces the following mixed ω-algebraic system:

$$x_p = \sum_{\pi \in \Gamma^*} M_{p\pi} x_\pi \, , \, p \in \Gamma, \tag{2}$$

$$z_p = \sum_{\pi \in \Gamma^*} (M_{p,\pi} y_\pi)_{(x_p)_{p\in\Gamma}} = \sum_{p' \in \Gamma} \sum_{\substack{\pi = p_1 \dots p_k \in \Gamma^+ \\ p_j = p'}} M_{p,\pi} x_{p_1} \dots x_{p_{j-1}} z_{p'} \, . \tag{3}$$

Here (2) is an ${S'}^{n\times n}$-algebraic system over the semiring $S^{n\times n}$ (see Section 2.3 of Ésik, Kuich [10]) and (3) is an $S^{n\times n}$-linear system over the semimodule S^n (see Section 5.5 of Ésik, Kuich [10]).

By Theorem 5.6.1 of Ésik, Kuich [10], $(A, U) \in ((S^{n\times n})^\Gamma, (S^n)^\Gamma)$ is a solution of (1) iff A is a solution of (2) and (A, U) is a solution of (3).

Theorem 3.1. *Let S be a complete star-omega semiring and $M \in ({S'}^{n\times n})^{\Gamma^*\times\Gamma^*}$ be a pushdown transition matrix. Then, for all $0 \le k \le n$,*

$$(((M^*)_{p,\varepsilon})_{p\in\Gamma}, ((M^{\omega,k})_p)_{p\in\Gamma})$$

is a solution of (1)*.*

We now introduce pushdown automata and ω-pushdown automata (see Kuich, Salomaa [12], Kuich [11], Cohen, Gold [2]).

Let S be a complete semiring and $S' \subseteq S$ with $0, 1 \in S'$. An *S'-pushdown automaton over S*

$$\mathcal{P} = (n, \Gamma, I, M, P, p_0)$$

is given by

(i) a finite set of *states* $\{1, \dots, n\}$, $n \ge 1$,

(ii) an alphabet Γ of *pushdown symbols*,

(iii) a *pushdown transition matrix* $M \in ({S'}^{n\times n})^{\Gamma^*\times\Gamma^*}$,

(iv) an *initial state vector* $I \in {S'}^{1\times n}$,

(v) a *final state vector* $P \in {S'}^{n\times 1}$,

(vi) an *initial pushdown symbol* $p_0 \in \Gamma$,

The *behavior* $\|\mathcal{P}\|$ *of* $\mathcal{P}$ is an element of S and is defined by $\|\mathcal{P}\| = I(M^*)_{p_0,\varepsilon} P$.

For a complete semiring-semimodule pair (S, V), an S'-ω-pushdown automaton (over (S, V))

$$\mathcal{P} = (n, \Gamma, I, M, P, p_0, k)$$

is given by an S'-pushdown automaton $(n, \Gamma, I, M, P, p_0)$ and an $k \in \{0, \dots, n\}$ indicating that the states $1, \dots, k$ are *repeated states*.

The *behavior* $\|\mathcal{P}\|$ of the S'-ω-pushdown automaton $\mathcal{P}$ is defined by

$$\|\mathcal{P}\| = I(M^*)_{p_0,\varepsilon}P + I(M^{\omega,k})_{p_0} \,.$$

Here $I(M^*)_{p_0,\varepsilon}P$ is the behavior of the S'-ω-pushdown automaton $\mathcal{P}_1 = (n, \Gamma, I, M, P, p_0, 0)$ and $I(M^{\omega,k})_{p_0}$ is the behavior of the S'-ω-pushdown automaton $\mathcal{P}_2 = (n, \Gamma, I, M, 0, p_0, k)$. Observe that $\mathcal{P}_2$ is an automaton with the Büchi acceptance condition: if G is the graph with adjacency matrix M, then only paths that visit the repeated states $1, \ldots, k$ infinitely often contribute to $\|\mathcal{P}_2\|$. Furthermore, $\mathcal{P}_1$ contains no repeated states and behaves like an ordinary S'-pushdown automaton.

Theorem 3.2. *Let S be a complete star-omega semiring and let $\mathcal{P} = (n, \Gamma, I, M, P, p_0, k)$ be an S'-ω-pushdown automaton over (S, S). Then $(\|\mathcal{P}\|, (((M^*)_{p,\varepsilon})_{p\in\Gamma}, ((M^{\omega,k})_p)_{p\in\Gamma}))$, $0 \leq k \leq n$, is a solution of the ${S'}^{n\times n}$-algebraic system*

$$y_0 = Iy_{p_0}P, y_p = \sum_{\pi\in\Gamma^*} M_{p,\pi}y_\pi,\ p \in \Gamma$$

over the complete semiring-semimodule pair $(S^{n\times n}, S^n)$.

Let now S be a continuous star-omega semiring and consider an S'-algebraic system $y = p(y)$ over (S, S). Then the least solution of the S'-algebraic system $x = p(x)$ over S, say σ, exists, and the components of σ are elements of $\mathfrak{Alg}\,(S')$. Moreover, write the $\mathfrak{Alg}\,(S')$-linear system $z = p_0(z)$ over S in the form $z = Mz$, where M is an $n\times n$-matrix. Then, by Theorem 5.6.1 of Ésik, Kuich [10], $(\sigma, M^{\omega,k})$, $0 \leq k \leq n$, is a solution of $y = p(y)$. Given a $k \in \{0, 1, \ldots, n\}$, we call this solution the solution of order k of $y = p(y)$. By ω-$\mathfrak{Alg}\,(S')$ we denote the collection of all components of solutions of all orders k of S'-algebraic systems over (S, S). (For details see Section 5.6 of Ésik, Kuich [10].)

4 The Kleene Theorem

The main result of this section is the following Kleene Theorem.

Theorem 4.1. *Let S be a continuous star-omega semiring. Then the following statements are equivalent for $(s, v) \in S \times S$:*

(i) $(s, v) = \|\mathfrak{A}\|$, *where* $\mathfrak{A}$ *is a finite* $\mathfrak{Alg}\,(S')$*-automaton over the quemiring* (S, S),

(ii) $(s, v) \in \omega\text{-}\mathfrak{Alg}\,(S')$,

(iii) $s \in \mathfrak{Alg}\,(S')$ *and* $v = \sum_{1\leq k\leq m} s_k t_k^\omega$, *where* $s_k, t_k \in \mathfrak{Alg}\,(S')\,, 1 \leq k \leq m$,

(iv) $(s, v) = \|\mathcal{P}\|$, *where* $\mathcal{P}$ *is an* S'*-*ω*-pushdown automaton.*

The proof of this Kleene Theorem is performed as follows:

1. The equivalence of (i), (ii) and (iii) is proved in [10], Theorem 5.4.9.

2. The implication *(iv)* $\Rightarrow$ *(ii)* is a simple corollary of Theorem 13 of [4].

3. The proof of the implication *(iii)* $\Rightarrow$ *(iv)* is performed by Lemmas 4.1, 4.2 and 4.3 proved in the following pages.

Lemma 4.1. *Let S be a complete star-omega semiring and $\mathcal{P}$ be an S'-pushdown automaton. Then there exists an S'-ω-pushdown automaton $\mathcal{P}'$ such that $\|\mathcal{P}'\| = \|\mathcal{P}\|^\omega$.*

Proof. Let $\mathcal{P} = (n, \Gamma, M, I, P, p_0)$. Then we construct $\mathcal{P}' = (2n, \Gamma', M', I', 0, p_0', n)$, $\Gamma' = \Gamma \cup \{p_0'\}$ as follows.

The pushdown transition matrix $M' \in \left(S'^{2n\times 2n}\right)^{\Gamma'^* \times \Gamma'^*}$ has, for $\pi \in \Gamma^*$, $1 \le j \le n$, the entries

$$\begin{aligned}(M'_{p_0',p_0'})_{n+i,j} &= (PI)_{i,j},\\ (M'_{p_0',\pi p_0'})_{i,n+j} &= (M_{p_0,\pi})_{i,j}\\ (M'_{p,\pi})_{n+i,n+j} &= (M_{p,\pi})_{i,j};\end{aligned}$$

all other entries of the matrices $M'_{p,\pi}, p \in \Gamma', \pi \in \Gamma'^*$, are 0.

The initial state vector $I' \in S'^{2n\times 1}$ has, for $1 \le i \le n$, the entries

$$I_i' = I_i, I_{n+i}' = 0.$$

We have to prove that

$$\|\mathcal{P}'\| = I' \left(M'^{\omega,n}\right)_{p_0'} = \|\mathcal{P}\|^\omega = \left(I \left(M^*\right)_{p_0,\varepsilon} P\right)^\omega .$$

The proof of this claim is as follows.

By definition, for $1 \le i \le 2n$,

$$\left(\left(M'^{\omega,n}\right)_{p_0'}\right)_i = \sum_{\pi_1,\pi_2,\ldots\in\Gamma'^*} \sum_{\substack{i_1,i_2,\ldots\in P_n\\ 1\le i_1,i_2,\ldots\le 2n}} \left(M'_{p_0',\pi_1}\right)_{i,i_1} \left(M'_{\pi_1,\pi_2}\right)_{i_1,i_2} \cdots$$

Inspection shows that a repeated state in the sequence $i_1, i_2, \ldots$ appears only if in the run $p_0', \pi_1, \pi_2, \ldots$ a transition from p_0' to p_0' appears.

Hence, we obtain, with $i_0^1 = i$, $\pi_0^t = \varepsilon$ for $t \geq 1$,

$$\begin{aligned}
&\Big((M'^{\omega,n})_{p_0'}\Big)_i \\
&= \prod_{t\geq 1} \sum_{k_t \geq 1} \sum_{1\leq i_0^t,\ldots,i_{k_t}^t \leq n} \sum_{\pi_1^t,\cdots,\pi_{k_t-1}^t \in \Gamma^*} \Big(M'_{p_0',\pi_1^t p_0'}\Big)_{i_0^t,n+i_1^t} \Big(M'_{\pi_1^t p_0',\pi_2^t p_0'}\Big)_{n+i_1^t,n+i_2^t} \cdots \\
&\qquad\qquad \Big(M'_{\pi_{k_t-1}^t p_0',p_0'}\Big)_{n+i_{k_t-1}^t,n+i_{k_t}^t} \Big(M'_{p_0',p_0'}\Big)_{n+i_{k_t}^t,i_0^{t+1}} \\
&= \prod_{t\geq 1} \sum_{k_t \geq 1} \sum_{1\leq i_0^t,\ldots,i_{k_t}^t \leq n} \sum_{\pi_1^t,\ldots,\pi_{k_t-1}^t \in \Gamma^*} \Big(M_{p_0,\pi_1^t}\Big)_{i_0^t,i_1^t} \Big(M_{\pi_1^t,\pi_2^t}\Big)_{i_1^t,i_2^t} \cdots \\
&\qquad\qquad \Big(M_{\pi_{k_t-1}^t,\varepsilon}\Big)_{i_{k_t-1}^t,i_{k_t}^t} (PI)_{i_{k_t}^t,i_0^{t+1}} \\
&= \prod_{t\geq 1} \sum_{k_t \geq 1} \sum_{1\leq i_0^t \leq n} \Big((M^{k_t})_{p_0,\varepsilon} PI\Big)_{i_0^t,i_0^{t+1}} \\
&= \prod_{t\geq 1} \sum_{1\leq i_0^t \leq n} \Big(\sum_{k_t\geq 1} (M^{k_t})_{p_0,\varepsilon} PI\Big)_{i_0^t,i_0^{t+1}} \\
&= \prod_{t\geq 1} \sum_{1\leq i_0^t \leq n} \Big((M^*)_{p_0,\varepsilon} PI\Big)_{i_0^t,i_0^{t+1}} \\
&= \Big((M^*)_{p_0,\varepsilon} PI\Big)^\omega_i .
\end{aligned}$$

Hence,

$$\begin{aligned}
\|\mathcal{P}'\| &= \sum_{1\leq i\leq 2n} I_i \Big((M'^{\omega,n})_{p_0'}\Big)_i \\
&= \sum_{1\leq i\leq n} I_i \Big((M'^{\omega,n})_{p_0'}\Big)_i \\
&= I\,(M'^{\omega,n})_{p_0'} \\
&= I\Big((M^*)_{p_0,\varepsilon} PI\Big)^\omega \\
&= \Big(I\,(M^*)_{p_0,\varepsilon} P\Big)^\omega \\
&= \|\mathcal{P}\|^\omega.
\end{aligned}$$

□

Lemma 4.2. *Let S be a complete star-omega semiring, $\mathcal{P}_1$ be an S'-ω-pushdown automaton and $\mathcal{P}_2$ be an S'-pushdown automaton. Then there exists an S'-ω-pushdown automaton $\mathcal{P}$ such that $\|\mathcal{P}\| = \|\mathcal{P}_2\|\|\mathcal{P}_1\|$.*

Proof. Let $\mathcal{P}_1 = (n_1, \Gamma_1, I_1, M_1, P_1, p_1, k)$ and $\mathcal{P}_2 = (n_2, \Gamma_2, I_2, M_2, P_2, p_2)$ with $\Gamma_1 \cap \Gamma_2 = \emptyset$. Then we construct $\mathcal{P} = (n_1 + n_2, \Gamma_1 \cup \Gamma_2, I, M, P, p_2, k)$ as follows.

Let $Q_1 = \{1, \ldots, n_1\}$ and $Q_2 = \{n_1 + 1, \ldots, n_2\}$. The pushdown transition matrix $M \in (S'^{(n_1+n_2)\times(n_1+n_2)})^{(\Gamma_1\cup\Gamma_2)^*\times(\Gamma_1\cup\Gamma_2)^*}$ has entries

1. transitions from Q_2 to Q_2

$$\begin{aligned}(M_{p_2,\pi p_1})_{i,j} &= \Big((M_2)_{p_2,\pi}\Big)_{i,j}, \quad i,j \in Q_2, \pi \in \Gamma_2^+,\\ (M_{p,\pi})_{i,j} &= \Big((M_2)_{p,\pi}\Big)_{i,j}, \quad i,j \in Q_2, p \in \Gamma_2, \pi \in \Gamma_2^+,\\ (M_{p,\varepsilon})_{i,j} &= \Big((M_2)_{p,\varepsilon}\Big)_{i,j}, \quad i,j \in Q_2, p \in \Gamma_2;\end{aligned}$$

2. transitions from Q_2 to Q_1

$$\begin{aligned}(M_{p_2,p_1})_{i,j} &= \Big((M_2)_{p_2,\varepsilon} P_2 I_1\Big)_{i,j}, \quad i \in Q_2, j \in Q_1,\\ (M_{p,\varepsilon})_{i,j} &= \Big((M_2)_{p,\varepsilon} P_2 I_1\Big)_{i,j}, \quad i \in Q_2, j \in Q_1, p \in \Gamma_2;\end{aligned}$$

3. transitions from Q_1 to Q_1

$$(M_{p,\pi})_{i,j} = \Big((M_1)_{p,\pi}\Big)_{i,j}, \quad i,j \in Q_1, p \in \Gamma_1, \pi \in \Gamma_1^*.$$

All other entries of the matrices $M_{p,\pi}$, $p \in \Gamma_1 \cup \Gamma_2$, $\pi \in (\Gamma_1 \cup \Gamma_2)^*$, are 0.

The initial state vector $I \in S'^{1\times(n_1+n_2)}$ and the final state vector $P \in S'^{(n_1+n_2)\times 1}$ have the entries

$$\begin{aligned}I_i &= 0,\ i \in Q_1, & I_i &= (I_2)_i,\ i \in Q_2;\\ P_i &= (P_1)_i,\ i \in Q_1, & P_i &= 0,\ i \in Q_2.\end{aligned}$$

We have to prove that

$$\begin{aligned}\|\mathcal{P}\| &= I\,(M^*)_{p_2,\varepsilon}\,P + I\,\big(M^{\omega,k}\big)_{p_2}\\ &= I_2\,(M_2^*)_{p_2,\varepsilon}\,P_2 I_1\,(M_1^*)_{p_1,\varepsilon}\,P_1 + I_2\,(M_2^*)_{p_2,\varepsilon}\,P_2 I_1\,\Big(M_1^{\omega,k}\Big)_{p_1}\\ &= \|\mathcal{P}_2\|\|\mathcal{P}_1\|.\end{aligned}$$

The proof of this claim is as follows.

By definition,

$$\Big(\big(M^{\omega,k}\big)_{p_2}\Big)_{i_0} = \sum_{\pi_1,\pi_2,\ldots\in(\Gamma_1\cup\Gamma_2)^*}\ \sum_{\substack{i_1,i_2,\ldots\in P_k\\ 1\le i_1,i_2,\ldots\le n_1+n_2}} (M_{p_2,\pi_1})_{i_0,i_1}\,(M_{\pi_1,\pi_2})_{i_1,i_2}\cdots, \quad i_0 \in Q_2,$$

$$\Big(\big(M^{\omega,k}\big)_{p_2}\Big)_{i_0} = 0, \quad i_0 \in Q_1.$$

As long as $\mathcal{P}$ remains in a state of Q_2, the contents of the pushdown tape is πp_1, $\pi \in \Gamma_2^*$. The transition from a state of Q_2 to a state of Q_1 is possible only in the following three situations:

(a) In the first step, the contents p_2 of the pushdown tape is replaced by p_1.

(b) The contents of the pushdown tape is pp_1, $p \in \Gamma_2$, and p is replaced by the empty word; so that after this replacement the contents is p_1.

(c) The contents of the pushdown tape is $p\pi p_1$, $p \in \Gamma_2$,$\pi \in \Gamma_2^+$, and p is replaced by the empty word. In this situation, no continuation of the computation of $\mathcal{P}$ is possible.

Since all the repeated states are states in Q_1, there must be a transition from a state of Q_2 to a state of Q_1.

As long as $\mathcal{P}$ remains in a state of Q_2 with πp_1, $\pi \in \Gamma_2^*$, on the pushdown tape, it simulates $\mathcal{P}_2$ up to situations (a) or (b). Then p_1 is the contents of the pushdown tape of $\mathcal{P}$, $\mathcal{P}$ is in a state of Q_1 and simulates $\mathcal{P}_1$, since there is no transition from a state of Q_1 to a state of Q_2.

Hence, we obtain, for $i_0 \in Q_2$,

$$\left((M^{\omega,k})_{p_2}\right)_{i_0}$$

$$= \sum_{\pi_1,\pi_2,\ldots\in\Gamma_1^+} \sum_{\substack{j_0,j_1,\ldots\in Q_1\\(j_0,j_1,\ldots)\in P_k}} (M_{p_2,p_1})_{i_0,j_0}(M_{p_1,\pi_1})_{j_0,j_1}(M_{\pi_1,\pi_2})_{j_1,j_2}\cdots +$$

$$\sum_{t\geq1}\sum_{\rho_1,\ldots,\rho_{t-1}\in\Gamma_2^+}\sum_{\rho_t\in\Gamma_2}\sum_{\pi_1,\pi_2,\ldots\in\Gamma_1^+}\sum_{i_1,\ldots,i_t\in Q_2}\sum_{\substack{j_0,j_1,\ldots\in Q_1\\(j_0,j_1,\ldots)\in P_k}} (M_{p_2,\rho_1p_1})_{i_0,i_1}$$

$$(M_{\rho_1p_1,\rho_2p_2})_{i_1,i_2}\cdots(M_{\rho_tp_1,p_1})_{i_t,j_0}(M_{p_1,\pi_1})_{j_0,j_1}(M_{\pi_1,\pi_2})_{j_1,j_2}\cdots$$

$$= \sum_{j_0\in Q_1}\left((M_2)_{p_2,\varepsilon}P_2I_1\right)_{i_0,j_0}\left(\left(M_1^{\omega,k}\right)_{p_1}\right)_{j_0} + \sum_{t\geq1}\sum_{\rho_1,\ldots,\rho_{t-1}\in\Gamma_2^+}$$

$$\sum_{\rho_t\in\Gamma_2}\sum_{\pi_1,\pi_2,\ldots\in\Gamma_1^+}\sum_{i_1,\ldots,i_t\in Q_2}\sum_{\substack{j_0,j_1,\ldots\in Q_1\\(j_0,j_1,\ldots)\in P_k}}\left((M_2)_{p_2,\rho_1}\right)_{i_0,i_1}\left((M_2)_{\rho_1,\rho_2}\right)_{i_1,i_2}\cdots$$

$$\left(\left((M_2)_{\rho_t,\varepsilon}\right)P_2I_1\right)_{i_t,j_0}\left((M_1)_{p_1,\pi_1}\right)_{j_0,j_1}\left((M_1)_{\pi_1,\pi_2}\right)_{j_1,j_2}\cdots$$

$$= \sum_{j_0\in Q_1}\sum_{t\geq0}\left((M_2^{t+1})_{p_2,\varepsilon}P_2I_1\right)_{i_0,j_0}\left(\left(M_1^{\omega,k}\right)_{p_1}\right)_{j_0}$$

$$= \left((M_2^*)_{p_2,\varepsilon}P_2I_1\left(M_1^{\omega,k}\right)_{p_1}\right)_{i_0}.$$

In the first equality, the first summand on the right side represents situation (a), while the second summand represents situation (b).

By definition,

$$\Big((M^*)_{p_2,\varepsilon}\Big)_{i_0,j} = \sum_{t\geq 1} \sum_{\pi_1,\dots,\pi_t \in (\Gamma_1\cup\Gamma_2)^*} \sum_{1\leq i_1,\dots,i_t\leq n_1+n_2} (M_{p_2,\pi_1})_{i_0,i_1}$$

$$(M_{\pi_1,\pi_2})_{i_1,i_2}\dots(M_{\pi_t,\varepsilon})_{i_t,j}\,, \quad i_0\in Q_2, j\in Q_1\cup Q_2,$$

$$\Big((M^*)_{p_2,\varepsilon}\Big)_{i_0,j} = 0, \quad i_0\in Q_1, j\in Q_1\cup Q_2.$$

Observe that $\pi_1 = \pi p_1$, $\pi\in\Gamma_2^*$. To obtain the empty tape, $\mathcal{P}$ has to replace eventually p_1 by some $\pi'\in\Gamma_1^*$. But this is possible only in situations (a) or (b).

Hence, we obtain, for $i_0\in Q_2, j\in Q_1$,

$$\begin{aligned}
\Big((M^*)_{p_2,\varepsilon}\Big)_{i_0,j} &= \sum_{j_0\in Q_1} (M_{p_2,p_1})_{i_0,j_0}\Big((M^*)_{p_1,\varepsilon}\Big)_{j_0,j} + \\
&\quad \sum_{t\geq 1}\sum_{\rho_1,\dots,\rho_{t-1}\in\Gamma_2^+}\sum_{\rho_t\in\Gamma_2}\sum_{i_1,\dots,i_t\in Q_2}\sum_{j_0\in Q_1}(M_{p_2,\rho_1p_1})_{i_0,i_1}\dots \\
&\quad (M_{\rho_tp_1,p_1})_{i_t,j_0}\Big((M^*)_{p_1,\varepsilon}\Big)_{j_0,j} \\
&= \sum_{j_0\in Q_1}\Big((M_2)_{p_2,\varepsilon}P_2I_1\Big)_{i_0,j_0}\Big((M_1^*)_{p_1,\varepsilon}\Big)_{j_0,j} + \\
&\quad \sum_{t\geq 1}\sum_{\rho_1,\dots,\rho_{t-1}\in\Gamma_2^+}\sum_{\rho_l\in\Sigma_2}\sum_{i_1,\dots,i_t\in Q_2}\sum_{j_0\in Q_1}\Big((M_2)_{p_2,\rho_1}\Big)_{i_0,i_1}\dots \\
&\quad \Big((M_2)_{\rho_{t-1},\rho_t}\Big)_{i_{t-1},i_t}\Big((M_2)_{\rho_t,\varepsilon}P_2I_1\Big)_{i_t,j_0}\Big((M_1^*)_{p_1,\varepsilon}\Big)_{j_0,j} \\
&= \Big((M_2)_{p_2,\varepsilon}P_2I_1(M_1^*)_{p_1,\varepsilon}\Big)_{j_0,j} + \\
&\quad \sum_{j_0\in Q_1}\sum_{t\geq 1}\Big((M_2^{t+1})_{p_2,\varepsilon}P_2I_1\Big)_{i_0,j_0}\Big((M_1^*)_{p_1,\varepsilon}\Big)_{j_0,j} \\
&= \sum_{t\geq 0}\Big((M_2^{t+1})_{p_2,\varepsilon}P_2I_1(M_1^*)_{p_1,\varepsilon}\Big)_{i_0,j} \\
&= \Big((M_2^*)_{p_2,\varepsilon}P_2I_1(M_1^*)_{p_1,\varepsilon}\Big)_{i_0,j}\,,
\end{aligned}$$

and, for $i_0\in Q_2, j\in Q_2$,

$$\Big((M^*)_{p_2,\varepsilon}\Big)_{i_0,j} = 0.$$

In the first equality, the first summand on the right side represents situation (a), while the second summand represents situation (b).

We obtain

$$\begin{aligned} I\left(M^*\right)_{p_2,\varepsilon} P &= \sum_{i\in Q_2}\sum_{j\in Q_1} (I_2)_i \left((M_2^*)_{p_2,\varepsilon} P_2 I_1 \left(M_1^*\right)_{p_1,\varepsilon}\right)_{i,j} (P_1)_j \\ &= I_2 \left(M_2^*\right)_{p_2,\varepsilon} P_2 I_1 \left(M_1^*\right)_{p_1,\varepsilon} P_1 \end{aligned}$$

and

$$\begin{aligned} I\left(M^{\omega,k}\right)_{p_2} &= \sum_{i\in Q_2} (I_2)_i \left((M_2^*)_{p_2,\varepsilon} P_2 I_1 \left(M_1^{\omega,k}\right)_{p_1}\right)_i \\ &= I_2 \left(M_2^*\right)_{p_2,\varepsilon} P_2 I_1 \left(M_1^{\omega,k}\right)_{p_1} . \end{aligned}$$

Hence,

$$\begin{aligned} \|\mathcal{P}\| &= I\left(M^*\right)_{p_2,\varepsilon} P + I\left(M^{\omega,k}\right)_{p_2} \\ &= I_2 \left(M_2^*\right)_{p_2,\varepsilon} P_2 \left(I_1 \left(M_1^*\right)_{p_1,\varepsilon} P_1 + I_1 \left(M_1^{\omega,k}\right)_{p_1}\right) \\ &= \|\mathcal{P}_2\|\|\mathcal{P}_1\|. \end{aligned}$$

□

Lemma 4.3. *Let S be a complete star-omega semiring and $\mathcal{P}_1$, $\mathcal{P}_2$ S'-ω-pushdown automata. Then there exists an S'-ω-pushdown automaton $\mathcal{P}$ such that $\|\mathcal{P}\| = \|\mathcal{P}_1\| + \|\mathcal{P}_2\|$.*

Proof. Let $\mathcal{P}_i = (n_i, \Gamma_i, I_i, M_i, P_i, p_i, k_i)$, $i = 1, 2$, with $\Gamma_1 \cap \Gamma_2 = \emptyset$. Then we construct $\mathcal{P} = (n_1 + n_2, \Gamma, I, M, P, p_0, k_1 + k_2)$, $\Gamma = \Gamma_1 \cup \Gamma_2 \cup \{p_0\}$.

The matrix $M \in \left(S'^{(n_1+n_2)\times(n_1+n_2)}\right)^{\Gamma^*\times\Gamma^*}$ is defined as follows. Let, for $\pi_1, \pi_2 \in \Gamma_1^*$, $(\pi_1, \pi_2) \neq (\varepsilon, \varepsilon)$,

$$(M_1)_{\pi_1,\pi_2} = \begin{pmatrix} a_{\pi_1,\pi_2} & b_{\pi_1,\pi_2} \\ c_{\pi_1,\pi_2} & d_{\pi_1,\pi_2} \end{pmatrix},$$

where the blocks are indexed by $\{1, \dots, k_1\}, \{k_1 + 1, \dots, n_1\}$, and, for $\pi_1, \pi_2 \in \Gamma_2^*$, $(\pi_1, \pi_2) \neq (\varepsilon, \varepsilon)$,

$$(M_2)_{\pi_1,\pi_2} = \begin{pmatrix} a_{\pi_1,\pi_2} & b_{\pi_1,\pi_2} \\ c_{\pi_1,\pi_2} & d_{\pi_1,\pi_2} \end{pmatrix},$$

where the blocks are indexed by $\{1, \dots, k_2\}, \{k_2 + 1, \dots, n_2\}$.

Then, we define, for $\pi \in \Gamma_1^*$,

$$M_{p_0,\pi} = \begin{pmatrix} a_{p_1,\pi} & 0 & b_{p_1,\pi} & 0 \\ 0 & 0 & 0 & 0 \\ c_{p_1,\pi} & 0 & d_{p_1,\pi} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix};$$

for $\pi \in \Gamma_2^*$,

$$M_{p_0,\pi} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & a_{p_1,\pi} & 0 & b_{p_1,\pi} \\ 0 & 0 & 0 & 0 \\ 0 & c_{p_1,\pi} & 0 & d_{p_1,\pi} \end{pmatrix};$$

for $p \in \Gamma_1, \pi \in \Gamma_1^*$,

$$M_{p,\pi} = \begin{pmatrix} a_{p,\pi} & 0 & b_{p,\pi} & 0 \\ 0 & 0 & 0 & 0 \\ c_{p,\pi} & 0 & d_{p,\pi} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix};$$

and for $p \in \Gamma_2, \pi \in \Gamma_2^*$,

$$M_{p,\pi} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & a_{p,\pi} & 0 & b_{p,\pi} \\ 0 & 0 & 0 & 0 \\ 0 & c_{p,\pi} & 0 & d_{p,\pi} \end{pmatrix}.$$

Here the blocks are indexed by $\{1, \ldots, k_1\}$, $\{k_1 + 1, \ldots, k_1 + k_2\}$,$\{k_1 + k_2 + 1, \ldots, k_2 + n_1\}$,$\{k_2 + n_1 + 1, \ldots, n_1 + n_2\}$.

The initial state vector $I \in S'^{1\times(n_1+n_2)}$ and the final state vector $P \in S'^{(n_1+n_2)\times 1}$ are defined by

$$I = \Big(((I_1)_i)_{1\le i\le k_1}, ((I_2)_i)_{1\le i\le k_2}, ((I_1)_i)_{k_1+1\le i\le n_1}, ((I_2)_i)_{k_2+1\le i\le n_2} \Big),$$

and

$$P = \Big(((P_1)_i)_{1\le i\le k_1}, ((P_2)_i)_{1\le i\le k_2}, ((P_1)_i)_{k_1+1\le i\le n_1}, ((P_2)_i)_{k_2+1\le i\le n_2} \Big)^\top,$$

with the same block indexing as before.

We have to prove that

$$\|\mathcal{P}\| = \|\mathcal{P}_1\| + \|\mathcal{P}_2\| = (I_1 M_1^* P_1 + I_2 M_2^* P_2) + (I_1 M_1^{\omega,k_1} + I_2 M_2^{\omega,k_2}).$$

The proof of this claim is as follows.

We obtain, for $1 \le i \le n_1 + n_2$,

$$\Big(\big(M^{\omega,k_1+k_2} \big)_{p_0} \Big)_i = \sum_{\pi_1,\pi_2,\ldots\in\Gamma^+} \sum_{\substack{(i_1,i_2,\ldots)\in P_{k_1+k_2} \\ 1\le i_1,i_2,\ldots\le n_1+n_2}} (M_{p_0,\pi_1})_{i,i_1} (M_{\pi_1,\pi_2})_{i_1,i_2} \cdots$$

For $1 \le i \le k_1$ and $k_1 + k_2 + 1 \le i \le n_1 + k_2$, and by deleting the 0-block rows and the corresponding 0-block columns, we obtain

$$
\begin{aligned}
&\left(\left(M^{\omega,k_1+k_2}\right)_{p_0}\right)_i \\
&= \sum_{\pi_1,\pi_2,\ldots\in\Gamma_1^+} \sum_{\substack{(i_1,i_2,\ldots)\in P_{k_1}\\ 1\le i_1,i_2,\ldots\le n_1}} \begin{pmatrix} a_{p_1,\pi_1} & b_{p_1,\pi_1}\\ c_{p_1,\pi_1} & d_{p_1,\pi_1}\end{pmatrix}_{i,i_1} \begin{pmatrix} a_{\pi_1,\pi_2} & b_{\pi_1,\pi_2}\\ c_{\pi_1,\pi_2} & d_{\pi_1,\pi_2}\end{pmatrix}_{i_1,i_2} \cdots \\
&= \sum_{\pi_1,\pi_2,\ldots\in\Gamma_1^+} \sum_{\substack{(i_1,i_2,\ldots)\in P_{k_1}\\ 1\le i_1,i_2,\ldots\le n_1}} \left((M_1)_{p_1,\pi_1}\right)_{i,i_1} \left((M_1)_{\pi_1,\pi_2}\right)_{i_1,i_2} \cdots \\
&= \left(\left(M_1^{\omega,k_1}\right)_{p_1}\right)_{i'},
\end{aligned}
$$

where $i' = i$ if $1 \le i \le k_1$ and $i' = i - k_2$ if $k_1 + k_2 + 1 \le i \le n_1 + k_2$.

A similar proof yields, for $k_1 + 1 \le i \le k_1 + k_2$ and $n_1 + k_2 + 1 \le i \le n_1 + n_2$, and by deleting the 0-block rows and the corresponding 0-block columns,

$$\left(\left(M^{\omega,k_1+k_2}\right)_{p_0}\right)_i = \left(\left(M_2^{\omega,k_2}\right)_{p_2}\right)_{i'},$$

where $i' = i - k_1$ if $k_1 + 1 \le i \le k_1 + k_2$ and $i' = i - n_1$ if $n_1 + k_2 + 1 \le i \le n_1 + n_2$.

By similar arguments, we obtain, for $1 \le i, j \le k_1$ and $k_1+k_2+1 \le i, j \le n_1+k_2$,

$$\left((M^*)_{p_0,\varepsilon}\right)_{i,j} = \left((M_1)^*_{p_1,\varepsilon}\right)_{i',j'},$$

where $i' = i$ if $1 \le i \le k_1$, $i' = i - k_2$ if $k_1 + k_2 + 1 \le i \le n_1 + k_2$, $j' = j$ if $1 \le j \le k_1$, and $j' = j - k_2$ if $k_1 + k_2 + 1 \le j \le n_1 + k_2$,

and for $k_1 + 1 \le i, j \le k_1 + k_2$ and $n_1 + k_2 + 1 \le i, j \le n_1 + n_2$,

$$\left((M^*)_{p_0,\varepsilon}\right)_{i,j} = \left((M_2)^*_{p_2,\varepsilon}\right)_{i',j'},$$

where $i' = i - k_1$ if $k_1 + 1 \le i \le k_1 + k_2$, $i' = i - n_1$ if $n_1 + k_2 + 1 \le i \le n_1 + n_2$, $j' = j - k_1$ if $k_1 + 1 \le j \le k_1 + k_2$, and $j' = j - n_1$ if $n_1 + k_2 + 1 \le j \le n_1 + n_2$.

Hence, we obtain

$$
\begin{aligned}
IM^*P = &\sum_{\substack{1\le i\le k_1\\ k_1+k_2+1\le i\le k_2+n_2}} \sum_{\substack{1\le j\le k_1\\ k_1+k_2+1\le j\le k_2+n_1}} I_i M^*_{i,j} P_j + \\
&\sum_{\substack{k_1+1\le i\le k_1+k_2\\ k_2+n_1+1\le i\le n_1+n_2}} \sum_{\substack{k_1+1\le j\le k_1+k_2\\ k_2+n_1+1\le j\le n_1+n_2}} I_i M^*_{i,j} P_j \\
= &\sum_{1\le i\le n_1} \sum_{1\le j\le n_1} (I_1)_i \left(M_1^*\right)_{i,j} (P_1)_j + \\
&\sum_{1\le i\le n_2} \sum_{1\le j\le n_2} (I_2)_i \left(M_2^*\right)_{i,j} (P_2)_j \\
= &\, I_1 M_1^* P_1 + I_2 M_2^* P_2,
\end{aligned}
$$

$$
\begin{aligned}
&IM^{\omega,k_1+k_2} \\
=&\sum_{\substack{1\le i\le k_1\\ k_1+k_2+1\le i\le k_2+n_1}} I_1\left(\left(M^{\omega,k_1+k_2}\right)_{p_0}\right)_i + \sum_{\substack{k_1+1\le i\le k_1+k_2\\ k_2+n_1+1\le i\le n_1+n_2}} \left(I_1\left(M^{\omega,k_1+k_2}\right)_{p_0}\right)_i \\
=&\sum_{1\le i\le n_1} (I_1)_i\left(\left(M_1^{\omega,k_1}\right)_{p_1}\right)_i + \sum_{1\le i\le n_2} (I_2)_i\left(\left(M_2^{\omega,k_2}\right)_{p_2}\right)_i \\
=&I_1\left(M_1^{\omega,k_1}\right)_{p_1} + I_2\left(M_2^{\omega,k_2}\right)_{p_2},
\end{aligned}
$$

and

$$
\begin{aligned}
\|\mathcal{P}\| &= IM^*P + IM^{\omega,k_1+k_2} \\
&= \left(I_1M_1^*P_1 + I_1\left(M_1^{\omega,k_1}\right)_{p_1}\right) + \left(I_2M_2^*P_2 + I_2\left(M_2^{\omega,k_2}\right)_{p_2}\right) \\
&= \|\mathcal{P}_1\| + \|\mathcal{P}_2\|.
\end{aligned}
$$

□

Proof of Theorem 4.1. We have only to prove implication (*iii*) ⇒ (*iv*). Since $s, s_k, t_k, 1 \le k \le m$, are in $\mathfrak{Alg}\,(S')$, there exist, by Theorem 6.8 of Kuich [11], S'-pushdown automata $\mathcal{P}_s$, $\mathcal{P}_{s_k}$, $\mathcal{P}_{t_k}$ with behaviors $\|\mathcal{P}_s\| = s$, $\|\mathcal{P}_{s_k}\| = s_k$, $\|\mathcal{P}_{t_k}\| = t_k$.

By Lemma 4.1, we can construct S'-ω-pushdown automata $\mathcal{P}'_k$ with behaviors $\|\mathcal{P}'_k\| = t_k^\omega$, $1 \le k \le m$; by Lemma 4.2 S'-ω-pushdown automata $\mathcal{P}_k$ with behaviors $\|\mathcal{P}_k\| = s_k t_k^\omega$, and by Lemma 4.3 an S'-ω-pushdown automaton $\mathcal{P}'$ with behavior $\|\mathcal{P}'\| = \sum_{1\le k\le m} s_k t_k^\omega$. Again by Lemma 4.3, we can construct an S'-ω-pushdown automaton $\mathcal{P}$ with behavior $\|\mathcal{P}\| = \left(s, \sum_{1\le k\le m} s_k t_k^\omega\right)$. □

Algebraic expressions denoting formal power series in $S^{alg}\langle\langle\Sigma^*\rangle\rangle$, S a continuous commutative semiring and Σ an alphabet, are defined in Section 3.5 of Ésik, Kuich [10]. By help of Theorem 4.1 (iii) ω-algebraic expressions denoting pairs $(s, v) \in \omega$-$\mathfrak{Alg}\,(S')$, $S' = S\langle\Sigma \cup \{\varepsilon\}\rangle$, S a continuous star-omega semiring and Σ an alphabet, can easily be defined.

References

[1] Bloom, S. L., Ésik, Z.: Iteration Theories. EATCS Monographs on Theoretical Computer Science. Springer, 1993.

[2] Cohen, R. S., Gold, A. Y.: Theory of ω-languages I: Characterizations of ω-context-free languages. JCSS 15(1977) 169–184.

[3] Conway, J. H.: Regular Algebra and Finite Machines. Chapman & Hall, 1971.

[4] Droste, M., Kuich, W.: The triple-pair construction for weighted ω-pushdown automata, to appear

[5] Eilenberg, S.: Automata, Languages and Machines. Vol. A. Academic Press, 1974.

[6] Ésik, Z., Kuich, W.: A semiring-semimodule generalization of ω-context-free languages, Theory is Forever, LNCS 3113, Springer, 2004, 68–80.

[7] Ésik, Z., Kuich, W.: A semiring-semimodule generalization of ω-regular languages II. Journal of Automata, Languages and Combinatorics 10 (2005) 243–264.

[8] Ésik, Z., Kuich, W.: On iteration semiring-semimodule pairs. Semigroup Forum 75 (2007), 129–159.

[9] Ésik, Z., Kuich, W.: A semiring-semimodule generalization of transducers and abstract ω-families of power series. Journal of Automata, Languages and Combinatorics, 12 (2007), 435–454.

[10] Ésik, Z., Kuich, W.: Modern Automata Theory. `http://www.dmg.tuwien.ac.at/kuich`

[11] Kuich, W.: Semirings and formal power series: Their relevance to formal languages and automata theory. In: Handbook of Formal Languages (Eds.: G. Rozenberg and A. Salomaa), Springer, 1997, Vol. 1, Chapter 9, 609–677.

[12] Kuich, W., Salomaa, A.: Semirings, Automata, Languages. EATCS Monographs on Theoretical Computer Science, Vol. 5. Springer, 1986.

Acta Cybernetica **23** (2017) 61–79.

Continuous Semiring-Semimodule Pairs and Mixed Algebraic Systems[*]

Zoltán Ésik[a] and Werner Kuich[b]

Abstract

We associate with every commutative continuous semiring S and alphabet Σ a category whose objects are all sets and a morphism $X \to Y$ is determined by a function from X into the semiring of formal series $S\langle\langle(Y \uplus \Sigma)^*\rangle\rangle$ of finite words over $Y \uplus \Sigma$, an $X \times Y$-matrix over $S\langle\langle(Y \uplus \Sigma)^*\rangle\rangle$, and a function from X into the continuous $S\langle\langle(Y \uplus \Sigma)^*\rangle\rangle$-semimodule $S\langle\langle(Y \uplus \Sigma)^\omega\rangle\rangle$ of series of ω-words over $Y \uplus \Sigma$. When S is also an ω-semiring (equipped with an infinite product operation), then we define a fixed point operation over our category and show that it satisfies all identities of iteration categories. We then use this fixed point operation to give semantics to recursion schemes defining series of finite and infinite words. In the particular case when the semiring is the Boolean semiring, we obtain the context-free languages of finite and ω-words.

1 Introduction

Suppose that S is a continuous semiring and Σ and X are sets. Let X^* and X^ω respectively denote the sets of all finite and ω-words over X. We can form the continuous semiring $S\langle\langle X^*\rangle\rangle$ of series over X^* with coefficients in S and the continuous $S\langle\langle X^*\rangle\rangle$-semimodule $S\langle\langle X^\omega\rangle\rangle$ of series over X^ω with coefficients in S. If S is equipped with an infinite product operation $S^\omega \to S, s_1 s_2 \cdots \mapsto \prod_{n\geq 1} s_n$, satisfying certain axioms including a sort of continuity described in the sequel, then we can also define an infinite product operation $(S\langle\langle X^*\rangle\rangle)^\omega \to S\langle\langle X^\omega\rangle\rangle$. In our first result, we show that the construction of the 'continuous ω-semiring-semimodule pair' $(S\langle\langle X^*\rangle\rangle, S\langle\langle X^\omega\rangle\rangle)$ enjoys a universal property, cf. Theorem 5.1.

In the second part of the paper we use the above universality result to give an algebraic treatment of recursion schemes defining series of finite and infinite words over Σ. To this end, we will restrict ourselves to commutative continuous

[*]The first author received support from NKFI grant no. ANN 110883. The second author was partially supported by Austrian Science Fund (FWF): grant no. I1661 -N25.

[a]Dept. of Foundations of Computer Science, University of Szeged, Szeged, Hungary
[b]Inst. of Discrete Mathematics and Geometry, Technical University of Vienna, Vienna, Austria
E-mail: kuich@tuwien.ac.at

DOI: 10.14232/actacyb.23.1.2017.5

ω-semirings S. Suppose that

$$x_i = p_i \; i \in I \; (R)$$

is a finite or infinite recursion scheme (or system of fixed point equations), where each p_i is a series in $S\langle\langle (X \uplus \Sigma)^* \rangle\rangle$. Then we associate with R in a natural way a function

$$F_R : \; S\langle\langle \Sigma^* \rangle\rangle \times S\langle\langle S^\omega \rangle\rangle \to S\langle\langle \Sigma^* \rangle\rangle \times S\langle\langle S^\omega \rangle\rangle$$

and define the semantics of R as a 'canonical' fixed point of F_R.

In order to facilitate the construction of the canonical fixed point, we introduce a category whose objects are sets (of recursion variables) and a morphism $f : X \to Y$ has three components:

- a function $f_0 : X \to S\langle\langle (Y \uplus \Sigma)^* \rangle\rangle$,
- a matrix $f_M \in (S\langle\langle (Y \uplus \Sigma)^* \rangle\rangle)^{X \times Y}$,
- a function $f_\omega : X \to S\langle\langle (Y \uplus \Sigma)^\omega \rangle\rangle$.

We define composition and the identity morphisms in a natural way to obtain a category $\mathbf{Ser}^\omega_{S,\Sigma}$. By taking the first components of morphisms, this category can be projected onto the category $\mathbf{Ser}_{S,\Sigma}$ having sets as objects and functions $X \to S\langle\langle (Y \uplus \Sigma)^* \rangle\rangle$ as morphisms $X \to Y$. It will be clear from the definition that the coproduct $X_1 \oplus \cdots \oplus X_n$ of any finite sequence $X_1, \ldots, X_n$ of objects exists in $\mathbf{Ser}^\omega_{S,\Sigma}$, in fact it will be given by disjoint union. (Actually all coproducts will exist, but this fact is not important for the paper.)

Next we define a dagger operation mapping a morphism $f : X \to X \oplus Y$ to a morphism $f^\dagger : X \to Y$. We prove that for any $f : X \to X \oplus Y$, the morphism $f^\dagger : X \to Y$ is a solution of the fixed point equation

$$\xi = f \circ \langle \xi, \mathbf{id}_Y \rangle.$$

(Here, $\langle -, - \rangle$ denotes the source pairing operation determined by the coproduct structure.) Intuitively, f represents a system of fixed point equations (recursion scheme) in the variables X and parameters Y, and $f^\dagger$ is its canonical solution. In particular, the function F_R associated with a system of fixed point equations (R) can be seen as a morphism $f : X \to X$ in $\mathbf{Ser}^\omega_{S,\Sigma}$, and the canonical fixed point of F_R can be derived from $f^\dagger : X \to \varnothing$. Indeed the three components of $f^\dagger$ are a function $X \to S\langle\langle \Sigma^* \rangle\rangle$, the empty $X \times \varnothing$-matrix over $S\langle\langle \Sigma^* \rangle\rangle$, and a function $X \to S\langle\langle \Sigma^\omega \rangle\rangle$. The first and third components form the canonical fixed point of F_R. Our approach generalizes the construction of context-free languages of finite and ω-words [10].

Categories with finite coproducts (or dually, products) equipped with a parametric fixed point operation have been studied since the late 1960's. A class of structures, called iteration theories, or iteration categories, have been identified. It has been shown that most (if not all) of the major fixed point structures used in computer science give rise to iteration categories. As a main technical contribution, we prove that for each commutative continuous semiring S and alphabet $\Sigma, \mathbf{Ser}^{\omega}_{S,\Sigma}$ is also an iteration category, cf. Theorem 7.1. A few consequences of this fact are discussed in the conclusion.

2 Continuous semirings

Recall that a commutative monoid $(V, +, 0)$ is *continuous* (cf. Section 2.2 of [6]) if it is equipped with a a partial order $\leq$ such that the supremum of any chain (or equivalently, directed set [12]) exists and 0 is the least element. Moreover, the sum operation + is continuous:

$$x + \sup Y \ = \ \sup(x + Y)$$

for all nonempty chains or nonempty directed sets $Y \subseteq V$, where $x + Y = \{x + y : y \in Y\}$. (Actually this also holds when the set is empty.) It follows that the sum operation is monotonic: if $x \leq y$ in V, then $x + z \leq y + z$ for all $z \in V$.

Suppose that V is a continuous commutative monoid and $x_i \in V$ for all $i \in I$. We define $\sum_{i \in I} x_i$ as the supremum of all finite sums $x_{i_1} + \ldots + x_{i_n}$ where $i_1, \ldots, i_n$ are pairwise different elements of I. It is well-known that this summation operation is completely associative and commutative:

$$\sum_{i \in I} x_i \ = \ \sum_{j \in J} \sum_{i \in I_j} x_i,$$

whenever I is the disjoint union of the sets $I_j, j \in J$ and $x_i \in V$ for all $i \in I$. Moreover,

$$\sum_{i \in I} x_i \ = \ \sum_{i \in I} x_{\pi(i)},$$

whenever π is a permutation $I \to I$ and $x_i \in V$ for all $i \in I$.

Continuous commutative monoids are closed under several constructions including direct product. Suppose that V_i is a continuous commutative monoid for all $i \in I$. Then $V = \prod_{i \in I} V_i$, equipped with the pointwise sum operation and pointwise ordering, is also a continuous commutative monoid. It follows that the summation operation in the product V is the pointwise summation. In particular, if V is a continuous commutative monoid, then so is V^I for any set I.

Suppose now that $S = (S, +, \cdot, 0, 1)$ is a semiring [8, 11]. We say that S is a *continuous semiring* (cf. Section 2.2 of [6]) if $(S, +, 0)$ is a continuous commutative monoid equipped with a partial order $\leq$[1] and the product operation is continuous (hence, also monotonic), i.e., it preserves the supremum of nonempty chains (or nonempty directed subsets) in either argument:

$$(\sup X)y = \sup(Xy)$$
$$y(\sup X) = \sup(yX) \ ,$$

[1]Unlike at some other places, we do not require that $x \leq y$ iff there is some z with $x + z = y$.

for all nonempty chains (or directed sets) $X \subseteq S$, where $Xy = \{xy \ : \ x \in X\}$ and yX is defined in the same way. It follows that product distributes over all sums:

$$(\sum_{i \in I} x_i)y = \sum_{i \in I} x_i y$$
$$y(\sum_{i \in I} x_i) = \sum_{i \in I} y x_i,$$

whenever $x_i \in S$ for all $i \in I$.

Suppose that S is a continuous semiring and I is a set. Then the *matrix semiring* $S^{I \times I}$, equipped with the pointwise sum operation, the usual matrix product operation and the pointwise ordering is also a continuous semiring. The matrix product is meaningful since the sum of any I-indexed family of elements of S exists.

Continuous semirings are also closed under the formation of *power series semirings.* Suppose that S is a continuous semiring and X is a set. As usual, let $S\langle\langle X^* \rangle\rangle$ denote the semiring of all power series $s = \sum_{u \in X^*} \langle s, u \rangle u$ over X with coefficients in S. Each series s may be viewed as a function $X^* \to S$ mapping a word $u \in X^*$ to $\langle s, u \rangle$. Equipped with the pointwise order relation $s \leq s'$ iff $\langle s, u \rangle \leq \langle s', u \rangle$ for all $u \in X^*, S\langle\langle X^* \rangle\rangle$ is a continuous semiring. The sum of any family of series is the pointwise sum.

Theorem 2.1. *Suppose that S is a continuous semiring. Then for each set X, the continuous semiring $S\langle\langle X^* \rangle\rangle$ has the following universal property. Given a continuous semiring S', a continuous semiring morphism $h_S : S \to S'$ and any function $h_X : X \to S'$ such that the elements of $h_S(S)$ commute with the elements of S', there is a unique continuous semiring morphism $h^\sharp : S\langle\langle X^* \rangle\rangle \to S'$ extending h_S and h_X.*

Proof. We provide an outline of the proof. For details, see [7].

First we extend h_X to a monoid morphism $X^* \to S'$, denoted just h. Then, for a series $s \in S\langle\langle X^* \rangle\rangle$, we define $h^\sharp(s) = \sum_{u \in X^*} h_S(\langle s, \ u \rangle) h(u)$. It is clear that $h^\sharp$ extends h_S and h_X and preserves the constants 0 and 1. Then we prove that $h^\sharp$ is continuous and preserves the binary sum operation. It then follows that $h^\sharp$ preserves all finite and infinite sums. Last, we prove that $h^\sharp$ preserves the product operation. Since the definition of $h^\sharp$ was forced, it is unique. □

3 Continuous semiring-semimodule pairs

Suppose now that S is a semiring and V is a commutative monoid as above. We call V a (left) *S-semimodule* [9] if there is an action $S \times V \to S$ subject to the usual associativity, distributivity and unitary conditions:

$$\begin{aligned}(s_1s_2)v &= s_1(s_2v)\\ (s_1+s_2)v &= s_1v+s_2v\\ s(v_1+v_2) &= sv_1+sv_2\\ 1v &= v\\ 0v &= 0\\ s0 &= 0\end{aligned}$$

for all $s, s_1, s_2 \in S$ and $v, v_1, v_2 \in V$. When V is an S-semimodule, we also say that (S, V) is a *semiring-semimodule* pair.

We call a semiring-semimodule pair (S, V) continuous if S is a continuous semiring and V is a continuous commutative monoid such that the action is continuous in either argument:

$$\begin{aligned}(\sup X)v &= \sup(Xv)\\ x(\sup Y) &= \sup(xY)\end{aligned}$$

for all $x \in S, y \in V$ and nonempty chains (or nonempty directed sets) $X \subseteq S$ and $Y \subseteq V$. Of course, $Xv = \{sv : s \in X\}$ and $xY = \{xw : w \in Y\}$. It follows that action distributes over summation on either side:

$$\begin{aligned}(\sum_{i\in I} x_i)v &= \sum_{i\in I} x_iv\\ x(\sum_{i\in I} v_i) &= \sum_{i\in I} xv_i\end{aligned}$$

for all $x, x_i \in S, v, v_i \in V, i \in I$, where I is any index set.

Moreover, we call a continuous semiring-semimodule pair (S, V) a continuous ω-semiring-semimodule pair if it is equipped with an *infinite product operation* $\prod_{n\geq 1} x_n$ mapping an ω-sequence (or ω-word) $x_1x_2\cdots \in S^\omega$ to $\prod_{n\geq 1} x_n \in V$. The infinite product is subject to the following axioms:

Ax1

$$x\prod_{n\geq 1} x_n = \prod_{n\geq 1} y_n,$$

where $y_1 = x$ and $y_{n+1} = x_n$ for all $n \geq 1$.

Ax2

$$\prod_{n\geq 1} x_n = \prod_{n\geq 1} x_{i_n}\cdots x_{i_{n+1}-1}$$

where the sequence $i_1 = 1 \leq i_2 \leq \cdots$ increases without a bound and the product of an empty family is 1.

Ax3

$$\prod_{n\geq 1}(x_n+y_n) = \sum_{z_n=x_n \text{ or } z_n=y_n} \prod_{n\geq 1} z_n$$

Ax4

$$\prod_{n\geq 1} \sup X_n = \sup_{x_n \in X_n} \prod_{n\geq 1} x_n$$

where for each $n, X_n \subseteq S$ is a nonempty chain (or a nonempty directed set).

It follows that

$$\prod_{n\geq 1} \sum_{i_n \in I_n} x_{i_n} = \sum_{i_1 \in I_1, i_2 \in I_2, \ldots} \prod_{n \geq 1} x_{i_n} \tag{1}$$

where $\{x_{i_n} : i_n \in I_n\}$ is a family of elements of S for all $n \geq 1$. Indeed,

$$\begin{aligned}
\prod_{n\geq 1} \sum_{i_n \in I_n} x_{i_n} &= \prod_{n\geq 1} \sup_{F_n \subseteq I_n} \{ \sum_{i_n \in F_n} x_{i_n} \} \\
&= \sup_{F_n \subseteq I_n} \prod_{n \geq 1} \{ \sum_{i_n \in F_n} x_{i_n} \} \\
&= \sup_{F_n \subseteq I_n} \sum_{i_n \in F_n} \prod_{n\geq 1} x_{i_n} \\
&= \sup_{F \subseteq I_1 \times I_2 \times \ldots} \sum_{(i_1, i_2, \ldots) \in F} \prod_{n \geq 1} x_{i_n} \\
&= \sum_{(i_1, i_2, \ldots) \in I_1 \times I_2 \times \ldots} \prod_{n \geq 1} x_{i_n} .
\end{aligned}$$

In particular, note that $\prod_{n\geq 1} x_n = 0$ whenever there is some m such that x_m is 0. (This also follows from Ax1.) Moreover, infinite product is monotonic: if $x_n \leq y_n$ in S for all $n \geq 1$, then $\prod_{n\geq 1} x_n \leq \prod_{n \geq 1} y_n$.

Suppose that S is a continuous semiring. Then we define a star operation $S \to S$ as usual: $s^* = \sum_{n\geq 0} s^n$ for all $s \in S$. It is known that s^* is the least solution of the fixed point equation $x = sx + 1$ (and also of $x = xs + 1$) over S. And if (S, V) is a continuous ω-semiring-semimodule pair, we define an omega operation $S^\omega \to V$ by $s^\omega = \prod_{n \geq 1} s$ for all $s \in S$. It is known that for each $s \in S, s^\omega$ is a solution of the equation $v = sv$ over V.

Complete equational and quasi-equational axiomatization of the equational properties of the star operation in continuous semirings has been given in [2]. Among the identities satisfied by continuous semirings are the sum star and product star identities [3, 1]:

$$\begin{aligned}
(x+y)^* &= (x^* y)^* x^* \\
(xy)^* &= 1 + x(yx)^* y
\end{aligned}$$

Also $0^* = 1$ and $1^* = 1^{**}$ hold.

When (S, V) is a continuous ω-semiring-semimodule pair, the omega operation satisfies the sum omega and product omega identities [1]:

$$\begin{aligned}
(x+y)^\omega &= (x^* y)^\omega + (x^* y)^* x^\omega \\
(xy)^\omega &= x(yx)^\omega
\end{aligned}$$

for all $x, y \in S$. Also $0^\omega = 0$.

4 Matrices and series

4.1 Matrices

Suppose that (S, V) is a continuous ω-semiring-semimodule pair and I is a set. Then, as mentioned above, $S^{I\times I}$ is a continuous semiring, and V^I is a continuous commutative monoid. There is a natural left action of $S^{I\times I}$ on V^I defined similarly to matrix multiplication:

$$(MN)_i = \sum_{j\in I} M_{i,j} N_j,\ i \in I,$$

for all $M \in S^{I\times I}$ and $N \in S^I$. Moreover, we may define an infinite product operation by

$$(\prod_{n\geq 1} M_n)_i = \sum_{i=i_1,i_2,\ldots} \prod_{n\geq 1} (M_n)_{i_n,i_{n+1}},\ i \in I.$$

Proposition 4.1. *Suppose that (S, V) is a continuous ω-semiring-semimodule pair and I is a set. Then, equipped with the pointwise orderings, $(S^{I\times I}, V^I)$ is also a continuous ω-semiring-semimodule pair.*

Proof. The fact that $S^{I\times I}$ is a continuous semiring is proved in [7]. It is clear that V^I is a continuous commutative monoid and that equipped with the action, $(S^{I\times I}, V^I)$ is a semiring-semimodule pair. The action is continuous in either argument, so that $(S^{I\times I}, V^I)$ is also a continuous semiring-semimodule pair. This can be proved by an argument similar to that used in [7] to establish that product in $S^{I\times I}$ is continuous in either argument.

In order to conclude, we still need to prove that the infinite product over matrices satisfies the axioms Ax1-Ax4. Let $M_n \in S^{I\times I}$ for all $n \geq 1$, and define $M' = M_1$ and $M'_n = M_{n+1}$ for all $n \geq 1$. Then for every $i \in I$, the ith component of $\prod_{n\geq 1} M'_n$ is

$$\sum_{j\in I} (M_1)_{i,j} \sum_{j=j_1,j_2,\ldots\in I} (\prod_{n\geq 1} (M_{n+1})_{j_n,j_{n+1}}) = \sum_{i=i_1,i_2,\ldots\in I} \prod_{n\geq 1} (M_n)_{i_n,i_{n+1}}$$

which is the ith component of $\prod_{n\geq 1} M_n$. Hence, Ax1 holds.

Suppose now that the sequence $1 = k_1 \leq k_2 \leq \cdots$ increases without a bound and define $M'_n = M_{i_n} \cdots M_{i_{n+1}-1}$ for all $n \geq 1$. Then for each $i \in I$, the jth component of $\prod_{n\geq 1} M'_n$ is

$$\begin{aligned}
&\sum_{j=j_1,j_2,\ldots\in I} \prod_{n\geq 1} (M')_{j_n,j_{n+1}} \\
&= \sum_{j=j_1,j_2,\ldots\in I} \prod_{n\geq 1} \sum_{j_n=\ell_1,\ell_2,\ldots,\ell_{i_{n+1}-i_n}=j_{n+1}} (M_{i_n})_{\ell_1,\ell_2} \cdots (M_{i_{n+1}-1})_{\ell_{i_{n+1}-i_n-1},\ell_{i_{n+1}-i_n}} \\
&= \sum_{j=j_1,j_2,\ldots} \prod_{n\geq 1} (M_j)_{j_n,j_{n+1}},
\end{aligned}$$

proving Ax2.

In order to prove that Ax3 holds, let $M_n, M'_n \in S^{I\times I}$ for all $n \geq 1$. Then for every $i \in I$, the ith component of $\prod_{n\geq 1}(M_n + M'_n)$ is

$$\begin{aligned}\sum_{i=i_1,i_2,\ldots}\prod_{n\geq 1}(M_n + M'_n)_{i_n,i_{n+1}} &= \sum_{P_n=M_n \text{ or } P_n=M'_n}\ \sum_{i=i_1,i_2,\ldots}\prod_{n\geq 1} P_{i_n,i_{n+1}} \\ &= (\sum_{P_n=M_n \text{ or } P_n=M'_n}\ \prod_{n\geq 1} P_n)_i,\end{aligned}$$

i.e., the ith component $\sum_{P_n=M_n \text{ or } P_n=M'_n}\prod_{n\geq 1} P_n$. This proves Ax3.

Suppose now that for each $n \geq 1, \mathcal{M}_n$ is a nonempty chain w.r.t. the pointwise ordering of matrices in $S^{I\times I}$. We want to prove that

$$\prod_{n\geq 1}\sup \mathcal{M}_n = \sup_{M_n\in\mathcal{M}_n}\prod_{n\geq 1} M_n. \tag{2}$$

Let $i \in I$ be fixed. Then the ith component of $\prod_{n\geq 1}\sup\mathcal{M}_n$ is

$$\begin{aligned}\sum_{i=i_1,i_2,\ldots}\prod_{n\geq 1}(\sup\mathcal{M})_{i_n,i_{n+1}} &= \sum_{i=i_1,i_2,\ldots}\ \sup_{M_n\in\mathcal{M}_n}\prod_{n\geq 1}(M_n)_{i_n,i_{n+1}} \\ &= \sup_{M_n\in\mathcal{M}_n}\ \sum_{i=i_1,i_2,\ldots}\prod_{n\geq 1}(M_n)_{i_n,i_{n+1}} \\ &= \sup_{M_n\in\mathcal{M}_n}\ (\prod_{n\geq 1} M_n)_i.\end{aligned}$$

Here, we used the fact, proved in [7], that summation is continuous. It follows that (2) holds. □

Hence, if (S, V) is a continuous ω-semiring-semimodule pair, then $(S^{I\times I}, V^I)$ comes with a star operation and an omega operation.

4.2 Series

We may also construct continuous ω-semiring-semimodule pairs of power series. To this end, we assume that S is a continuous ω-semiring equipped with an infinite product $S^\omega \to S$ subject to axioms similar to those defining continuous ω-semiring-semimodule pairs. This amounts to requiring that, equipped with left multiplication as the action and the infinite product, (S, S) is a continuous ω-semiring-semimodule pair.

Let X be any set. We already know that $S\langle\langle X^*\rangle\rangle$ is a continuous semiring. In a similar way, $S\langle\langle X^\omega\rangle\rangle$, equipped with the pointwise sum operation and pointwise ordering, is a continuous commutative monoid, and the action of $S\langle\langle X^*\rangle\rangle$ on $S\langle\langle X^\omega\rangle\rangle$ defined by

$$sr = \sum_{w\in A^\omega}\sum_{w=uv}\langle s, u\rangle\langle r, v\rangle uv$$

turns $S\langle\langle X^\omega\rangle\rangle$ into an $S\langle\langle X^*\rangle\rangle$-semimodule. Moreover, the action is continuous in either argument and it is easy to check that Ax1-Ax4 hold.

Let $s_n \in S\langle\langle X^* \rangle\rangle$ for all $n \geq 1$. We define $r = \prod_{n\geq 1} s_n \in S\langle\langle X^\omega \rangle\rangle$ as follows. Given $v \in X^\omega$, we define

$$\langle r, v\rangle = \sum_{v=v_1v_2\ldots} \prod_{n\geq 1} \langle s_n, v_n\rangle$$

Proposition 4.2. *Let S be a continuous ω-semiring and X be a set. Then $(S\langle\langle X^* \rangle\rangle, S\langle\langle X^\omega \rangle\rangle)$ is a continuous ω-semiring-semimodule pair.*

Proof. First we establish Ax1. Let $s \in S\langle\langle X^* \rangle\rangle$ and $s_n \in S\langle\langle X^* \rangle\rangle$ for all $n \geq 1$. Then, for all $w \in X^\omega$,

$$\begin{aligned}
\langle s \prod_{n\geq 1} s_n, w\rangle &= \sum_{w=uv} \langle s, u\rangle \langle \prod_{n\geq 1} s_n, v\rangle \\
&= \sum_{w=uv} \langle s, u\rangle \sum_{v=v_1v_2\ldots} \prod_{n\geq 1} \langle s_n, v_n\rangle \\
&= \sum_{w=uv_1v_2\ldots} \langle s, u\rangle \prod_{n\geq 1} \langle s_n, v_n\rangle \\
&= \sum_{w=v_1v_2\ldots} \langle s'_n, v_n\rangle \\
&= \langle \prod_{n\geq 1} s'_n, w\rangle,
\end{aligned}$$

where $s'_1 = s$ and $s'_{n+1} = s_n$ for all $n \geq 1$.

Let again $s_n \in S\langle\langle X^* \rangle\rangle$ for all $n \geq 1$. Suppose that the sequence $i_1 = 1 \leq i_2 \leq \cdots$ increases without a bound. For each $n \geq 1$, define $s'_n = s_{i_n} \cdots s_{i_{n+1}-1}$. Then for all $w \in X^\omega$,

$$\begin{aligned}
\langle \prod_{n\geq 1} s_n, w\rangle &= \sum_{w=v_1v_2\ldots} \prod_{n\geq 1} \langle s_n, v_n\rangle \\
&= \sum_{w=v_1v_2\ldots} \prod_{n\geq 1} \langle s_{i_n}, v_{i_n}\rangle \cdots \langle s_{i_{n+1}-1}, v_{i_{n+1}-1}\rangle \\
&= \sum_{w=u_1u_2\cdots} \prod_{n\geq 1} \langle s'_n, u_n\rangle \\
&= \langle \prod_{n\geq 1} s'_n, w\rangle,
\end{aligned}$$

proving Ax2.

Next, suppose that $s_n, s'_n \in S\langle\langle X^* \rangle\rangle$ for all $n \geq 1$. Then for all $w \in X^\omega$,

$$\begin{aligned}
\langle \prod_{n\geq 1} (s_n + s'_n), w\rangle &= \sum_{w=v_1v_2\ldots} \prod_{n\geq 1} \langle s_n + s'_n, v_n\rangle \\
&= \sum_{w=v_1v_2\cdots} \sum_{r_n=s_n \text{ or } r_n=s'_n} \prod_{n\geq 1} \langle r_n, v_n\rangle \\
&= \sum_{r_n=s_n \text{ or } r_n=s'_n} \langle \prod_{n\geq 1} r_n, w\rangle,
\end{aligned}$$

proving Ax3.

Finally, suppose that $(I_n, \leq)$ is a nonempty directed partially ordered set for each $n \geq 1$ and $s_i \in S\langle\langle X^* \rangle\rangle$ for all $i \in I_n, n \geq 1$ such that $s_i \leq s_j$ whenever $i \leq j$ in I_n. We want to prove that

$$\prod_{n\geq 1} \sup_{i\in I_n} s_i = \sup_{i_1\in I_1, i_2\in I_2,\ldots} \prod_{n\geq 1} s_{i_n}.$$

Let $w \in X^\omega$. Then

$$\begin{aligned}
\langle \prod_{n\geq 1} \sup_{i\in I_n} s_i, w\rangle &= \sum_{w=v_1v_2\ldots} \prod_{n\geq 1} \langle \sup_{i\in I_n} s_i, v_n\rangle \\
&= \sum_{w=v_1v_2\ldots} \prod_{n\geq 1} \sup_{i\in I_n} \langle s_i, v_n\rangle \\
&= \sum_{w=v_1v_2\ldots} \sup_{i_1\in I_1, i_2\in I_2,\ldots} \prod_{n\geq 1} \langle s_{i_n}, v_n\rangle \\
&= \sup_{i_1\in I_1, i_2\in I_2,\ldots} \sum_{w=v_1v_2\ldots} \prod_{n\geq 1} \langle s_{i_n}, v_n\rangle \\
&= \sup_{i_1\in I_1, i_2\in I_2,\ldots} \langle \prod_{n\geq 1} s_{i_n}, w\rangle \\
&= \langle \sup_{i_1\in I_1, i_2\in I_2,\ldots} \prod_{n\geq 1} s_{i_n}, w\rangle,
\end{aligned}$$

proving Ax4. □

Hence, if S is a continuous semiring, then for any set X, $(S\langle\langle X^* \rangle\rangle,\ S\langle\langle X^\omega \rangle\rangle)$ has a star and an omega operation.

5 Freeness

Suppose now that (S, V) and (S', V') are continuous ω-semiring-semimodule pairs. We say that a pair of functions $h = (h_S, h_V)$ with $h_S : S \to S'$ and $h_V : V \to V'$ is a continuous ω-semiring-semimodule pair morphism if h_S is a continuous semiring homomorphism, h_V is a continuous monoid homomorphism, h_S and h_V jointly preserve the action, moreover, infinite product is preserved:

$$h_V(\prod_{n\geq 1} x_n) = \prod_{n\geq 1} h_S(x_n)$$

for all $x_n \in S, n \geq 1$. In this section we prove:

Theorem 5.1. *Suppose that S is a continuous ω-semiring. Then for each set X, the continuous ω-semiring-semimodule pair $(S\langle\langle X^* \rangle\rangle, S\langle\langle X^\omega \rangle\rangle)$ has the following universal property. Let (S', V') be a continuous ω-semiring-semimodule pair, $h_S : S \to S'$ a continuous semiring morphism and $h_X : X \to S'$ a function. Suppose that the elements of $h_S(S)$ commute with the elements of S' and the following infinite commutativity holds:*

$$\prod_{n\geq 1} s_n s'_n = (\prod_{n\geq 1} s_n)(\prod_{n\geq 1} s'_n)$$

for all $s_n \in h_S(S)$ and $s'_n \in S'$. Then there is a unique continuous semiring-semimodule morphism $h^\sharp = (h^\sharp_S,\ h^\sharp_V)$ extending h_S and h_X.

Proof. We have already shown that $(S\langle\langle X^*\rangle\rangle, S\langle\langle X^\omega\rangle\rangle)$ is a continuous semiring-semimodule pair. Let us first extend h to a function $X^* \to S$, denoted just h, so that it becomes a (multiplicative) monoid homomorphism. Then let $h^\sharp_S : S\langle\langle X^*\rangle\rangle \to S'$ be defined by

$$h^\sharp_S(s) = \sum_{u\in X^*} h_S(\langle s,u\rangle)h(u).$$

It is known that $h^\sharp_S$ is a continuous semiring homomorphism.

Next we extend h to a function $X^\omega \to V'$ by defining $h(v) = \prod_{i\ge 1} h_X(x_i)$ for each $v = x_1x_2\ldots$ in X^ω. Finally, when $s \in S\langle\langle X^\omega\rangle\rangle$, let $h^\sharp_V(s) = \sum_{v\in X^\omega} h_S(\langle s,v\rangle)h(v)$.

Suppose that $s_i \in S\langle\langle X^\omega\rangle\rangle$ for all $i \in I$, where I is nonempty directed partially ordered set, ordered by the relation $\le$. Moreover, suppose that $s_i \le s_j$ whenever $i \le j$ in I and let $s = \sup_{i\in I} s_i$. Then

$$\begin{aligned} h^\sharp_V(s) &= \sum_{v\in X^\omega} h_S(\langle s,v\rangle)h(v) \\ &= \sum_{v\in X^\omega} \sup_{i\in I} h_S(\langle s_i,v\rangle)h(v) \\ &= \sup_{i\in I} \sum_{v\in X^\omega} h_S(\langle s_i,v\rangle)h(v) \\ &= \sup_{i\in I} h^\sharp_V(s_i), \end{aligned}$$

proving that $h^\sharp_V$ is continuous. To prove that $h^\sharp_V$ preserves the sum operation, let $s_1, s_2 \in S\langle\langle X^\omega\rangle\rangle$. Then

$$\begin{aligned} h^\sharp_V(s_1+s_2) &= \sum_{v\in X^\omega} h_S(\langle s_1+s_2,v\rangle)h(v) \\ &= \sum_{v\in X^\omega} h_S(\langle s_1,v\rangle)h(v) + h_S(\langle s_2,v\rangle)h(v) \\ &= \sum_{v\in X^\omega} h_S(\langle s_1,v\rangle)h(v) + \sum_{v\in X^\omega} h_S(\langle s_2,v\rangle)h(v) \\ &= h^\sharp_V(s_1) + h^\sharp_V(s_2). \end{aligned}$$

It is clear that $h^\sharp_V$ preserves 0. In order to prove that $h^\sharp_S$ and $h^\sharp_V$ jointly preserve the action, let $s \in S\langle\langle X^*\rangle\rangle$ and $r \in S\langle\langle X^\omega\rangle\rangle$. Then

$$\begin{aligned} h^\sharp_V(sr) &= \sum_{v\in X^\omega} h_S(\langle sr,v\rangle)h(v) \\ &= \sum_{v\in X^\omega}\sum_{v=uw} h_S(\langle s,u\rangle)h_S(\langle s,w\rangle)h(u)h(w) \\ &= \sum_{v\in X^\omega}\sum_{v=uw} h_S(\langle s,u\rangle)h(u)h_S(\langle r,w\rangle)h(w) \\ &= \sum_{u\in X^*} h_S(\langle s,u\rangle)h(u) \sum_{w\in X^\omega} h_S(\langle r,w\rangle)h(w) \\ &= h^\sharp_S(s)h^\sharp_V(r). \end{aligned}$$

Finally, we prove that $h_V^\sharp$ preserves the infinite product. To this end, let $s_n \in S\langle\langle X^*\rangle\rangle$ for all $n \geq 1$. We want to prove that $h_V^\sharp(\prod_{n\geq 1} s_n) = \prod_{n\geq 1} h_S^\sharp(s_n)$.

$$\begin{aligned}
h_V^\sharp(\prod_{n\geq 1} s_n) &= \sum_{v\in X^\omega} h_S((\prod_{n\geq 1} s_n, v))h(v) \\
&= \sum_{v\in X^\omega} \sum_{v=v_1v_2\dots} \prod_{n\geq 1} h_S((s_n, v_n))h(v_n) \\
&= \sum_{v\in X^\omega} \sum_{v=v_1v_2\dots} \prod_{n\geq 1} h_S((s_n, v_n)) \prod_{n\geq 1} h(v_n) \\
&= \prod_{n\geq 1}(\sum_{v_n\in X^*} h_S((s_n, v_n))h(v_n)) \\
&= \prod_{n\geq 1} h_S(s_n).
\end{aligned}$$

It is clear that h_S extends h. Since the definitions of h_S and h_V were forced, they are unique. □

6 The category $\mathbf{Matr}_{(S,V)}$

All categories $\mathcal{C}$ in the paper will have sets as objects. The composition of morphisms $f : X \to Y$ and $g : Y \to Z$ will be denoted $f \circ g$. We usually let $\mathbf{id}_X$ denote the identity morphism $X \to X$.

Our categories will have finite coproducts. For a sequence $X_1, \dots, X_n$ of objects, the coproduct $X_1 \oplus \dots \oplus X_n$ will be given by disjoint union $X_1 \uplus \dots \uplus X_n$. In particular, the empty set $\varnothing$ will serve as initial object.

Let $X_1, \dots, X_n$ be objects. For each $i = 1, \dots, n$, the ith coproduct injection $\mathbf{in}_{X_i} : X_i \to X_1 \oplus \dots \oplus X_n$ will always be determined by the embedding of X_i into $X_1 \uplus \dots \uplus X_n$. We will let $!_X$ denote the unique morphism $\varnothing \to X$. Moreover, if $f_i : X_i \to X$ for $i = 1, \dots, n$, then we will let $\langle f_1, \dots, f_n\rangle$ denote the unique morphism $f : X_1 \oplus \dots \oplus X_n \to X$ with $\mathbf{in}_{X_i} \circ f = f_i$ for all i. And when $f_i : X_i \to Y_i$, where $i \in \{1, \dots, n\}$, then we let $f_1 \oplus \dots \oplus f_n$ denote the unique morphism $f : X_1 \oplus \dots \oplus X_n \to Y_1 \oplus \dots \oplus Y_n$ with $\mathbf{in}_{X_i} \circ f = f_i \circ \mathbf{in}_{Y_i}$ for all i.

For any X, Y, the hom-set $\mathcal{C}(X, Y)$ of morphisms $X \to Y$ will be both a complete partial order $(\mathcal{C}, \leq)$ and a commutative monoid $(\mathcal{C}(X, Y), +, 0_{X,Y})$ such that the zero morphism $0_{X,Y}$ is also least w.r.t. $\leq$ and the operation $+$ is continuous in both of its arguments. Also, the operation of composition will be continuous in both arguments. Moreover, the partial order will be related to the coproduct structure so that for any $f, g : X_1 \oplus \dots \oplus X_n \to Y$, $f \leq g$ iff $\mathbf{in}_{X_i} \circ f \leq \mathbf{n}_{X_i} \circ g$. It follows that when $f_i : X_i \to Y_i$, where $i \in \{1, \dots, n\}$, then $f_1 \oplus \dots \oplus f_n \leq g_1 \oplus \dots \oplus g_n$ iff $f_i \leq g_i$ for all i.

The following identities will hold for all $f, g : X \to Y$ and $h : Y \to Z$:

$$(f + g) \circ h = f \circ h + g \circ h$$
$$0_{X,Y} \circ h = 0_{X,Z}$$

Finally, our categories will be equipped with a dagger operation mapping a morphism $f : X \to X \oplus Y$ to a morphism $f^\dagger : X \to Y$. This operation will always be a fixed point operation, so that the following fixed point identity will hold:

$$f^\dagger = f \circ \langle f^\dagger, \mathbf{id}_Y \rangle$$

for all $f : X \to X \oplus Y$.

Iteration categories are categories with finite coproducts and a dagger operation satisfying certain identities including the above fixed point identity, the parameter identity

$$(f \circ (\mathbf{id}_X \oplus g))^\dagger = f^\dagger \circ g$$

where $f : X \to X \oplus Y$ and $g : Y \to Z$, the double dagger identity

$$f^{\dagger\dagger} = (f \circ (\langle \mathbf{id}_X, \mathbf{id}_X \rangle \oplus \mathbf{id}_Y))^\dagger,$$

where $f : X \to X \oplus X \oplus Y$, to name a few, and some other identities including the group identities that we will described later. All of our categories will be iteration categories.

Suppose now that (S, V) is a continuous ω-semiring-semimodule pair. Then (S, V) determines a category $\mathbf{Matr}_{(S,V)}$ whose objects are all sets and a morphism $I \to J$ is an ordered pair (A, u) , where $A \in S^{I\times J}$ and $u \in V^I$. Hence a morphism $I \to I$ is an element of the semiring-semimodule pair $(S^{I\times I}, V^I)$.

Composition is defined as follows. Suppose that $(A, u) : I \to J$ and $(B, v) : J \to K$. Then we define $(A, u) \circ (B, v) = (AB, u + Av) : I \to K$. It is easy to check that composition is associative with the morphisms $(E^{I\times I},\ 0^I) : I \to I$ serving as identities, where $E^{I\times I}$ is the unit matrix in $S^{I\times I}$ and 0^I denotes the zero element of V^I. (For finite sets, this category is defined in [1].)

The partial order $\le$ on a hom-set of $\mathbf{Matr}_{(S,V)}$ is defined pointwise, so that when $(A, u), (B, v) : I \to J$, then $(A, u) \le (B, v)$ iff $A_{i,j} \le B_{i,j}$ and $u_i \le v_i$ for all $i \in I$ and $j \in J$. Clearly, each hom-set forms a complete partial order, and it is not difficult to verify that composition is continuous.

We can also impose a commutative monoid structure on the hom-sets by defining $(A, u) + (B, v)$ pointwise, for all $(A, u), (B, v) : I \to J$. Hence $(A, u) + (B, v) = (C, w)$ with $C_{i,j} = A_{i,j} + B_{i,j}$ and $w_i = u_i + v_i$ for all $i \in I$ and $j \in J$. The zero morphism $I \to J$ is the morphism $(0^{I\times J}, 0^I)$ consisting of two zero matrices. We denote it by $0_{I,J}$, or just 0. We have

$$\begin{aligned}((A, u) + (B, v)) \circ (C, w) &= (A, u) \circ (C, w) + (B, v) \circ (C, w)\\ 0_{I,j} \circ (C, w) &= 0_{I,K}\end{aligned}$$

for all $(C, w) : J \to K$. It is not difficult to prove that composition is continuous.

Coproduct is given by disjoint union on objects. When $X_1, \dots, X_n$ is a sequence of sets and $i \in \{1, \dots, n\}$, then ith coproduct embedding $\mathbf{in}_i$ consists of an $X_i \times (X_1 \uplus \cdots \uplus X_n)$ matrix whose $x_i \times X_i$ submatrix is an identity matrix and whose $X_i \times X_j$ matrices are all zero matrices for $j \ne i$, together with the column matrix $0^{X_i} \in V^{X_i}$.

We have already noted that for each set I, $(S^{I\times I}, V^I)$ is a continuous ω-semiring-semimodule pair. Hence it comes with a star and an omega operation: For each $A \in S^{I\times I}, A^* = \sum_{n\geq 0} A^n \in S^{I\times I}$ and $A^\omega = \prod_{n\geq 1} A$ in V^I. These operations satisfy the identities mentioned above. And in fact,

$$\begin{aligned}(A+B)^* &= (A^*B)^*A^*, \quad A, B \in S^{I\times I}\\ (AB)^* &= E_I + A(BA)^*B, \quad A \in S^{I\times J}, B \in S^{J\times I}\end{aligned}$$

and

$$\begin{aligned}(A+B)^\omega &= (A^*B)^*A^\omega + (A^*B)^\omega, \quad A, B \in S^{I\times I}\\ (AB)^\omega &= A(BA)^\omega, \quad A \in S^{I\times J}, B \in S^{J\times I}\end{aligned}$$

Suppose now that $I = J \uplus K$ and $M \in S^{I\times I}$ is partitioned as

$$M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

Then

$$M^* = \begin{pmatrix} (a+bd^*c)^* & (a+bd^*c)^*bd^* \\ (d+ca^*b)^*ca^* & (d+ca^*b)^* \end{pmatrix}$$

and

$$M^\omega = \begin{pmatrix} (a+bd^*c)^\omega + (a+bd^*c)^*bd^\omega \\ (d+ca^*b)^\omega + (d+ca^*b)^*ca^\omega \end{pmatrix}$$

The star and omega operations together give rise to a dagger operation over $\mathbf{Matr}_{(S,V)}$ that map a morphism $X \to X \oplus Y$ to a morphism $X \to Y$. To define it, let $(A, u) : X \to I \oplus J$, and partition A as (a, b) with $a \in S^{I\times I}$ and $b \in S^{I\times J}$. Then we define $(A, u)^\dagger = (a^*b,\ a^\omega + a^*v) : I \to J$. Clearly, $(A, u)^\dagger$ is a solution of the equation

$$(X, x) = (A, u)\begin{pmatrix} (X, x) \\ (E_J, 0) \end{pmatrix} = (aX + b, ax + u)$$

where (X, x) ranges over the morphisms $I \to J$. It is known that equipped with dagger, $\mathbf{Matr}_{(S,V)}$ is an iteration category.

7 Categories of Series

7.1 The category $\mathbf{Ser}_{S,\Sigma}$

Suppose that S is a commutative continuous semiring and Σ is a set. We define the category $\mathbf{Ser}_{S,\Sigma}$ whose objects are all sets and a morphism $X \to Y$ is a function $f : X \to S\langle\langle (Y \uplus \Sigma)^* \rangle\rangle$, or alternatively, a tuple $(f_x)_{x\in X}$ of series $f_x \in S\langle\langle (Y \uplus \Sigma)^* \rangle\rangle$. Suppose that $f : X \to Y$ and $g : Y \to Z$. Then we define $f \circ g$ as the function composition of f and $g^\sharp$, the extension of the function $Y \uplus \Sigma$ to $S\langle\langle (Z \uplus \Sigma)^* \rangle\rangle$ which agrees with g on Y and is the identity function on Σ to a continuous semiring homomorphism $S\langle\!\langle (Y \uplus \Sigma)^* \rangle\!\rangle \to S\langle\!\langle (Z \uplus \Sigma)^* \rangle\!\rangle$. For each X, the identity morphism $\mathbf{id}_X$ is the embedding of X into $X \uplus \Sigma$.

Each hom-set of morphisms $X \to Y$ of the category $\mathbf{Ser}_{S,\Sigma}$ has the structure of a complete partial order and commutative monoid. For any $f, g : X \to Y$, we define $f \leq g$ iff $f_x \leq g_x$ for all x, and similarly, $(f+g)_x = f_x + g_x$ for all x. The neutral element is the series $0_{X,Y}$ whose components are all 0. This is also the least morphism $X \to Y$. Composition of morphisms is continuous as is the sum operation.

Also, $\mathbf{Ser}_S$ has finite coproducts are given by disjoint sum on objects. The coproduct $X_1 \oplus \cdots \oplus X_n$ of a sequence $X_1, \ldots, X_n$ of sets is given by disjoint union, and for each $i \in \{1, \ldots, n\}$, $\mathbf{in}_{X_i} : X_i \to X_1 \oplus \cdots \oplus X_n$ is the embedding of X_i into $S\langle\!\langle (X_1 \uplus \cdots \uplus X_n \uplus \Sigma)^* \rangle\!\rangle$.

We define a dagger operation on $\mathbf{Ser}_{S,\Sigma}$ which maps a morphism $f : X \to X \oplus Y$ to $f^\dagger : X \to Y$. The morphism $f^\dagger$ is given as the least solution of the fixed point equation

$$\xi = f \circ \langle \xi, \mathbf{id}_Y \rangle.$$

In more detail, $f^\dagger = \sup_{n \geq 0} f^{(n)}$, where $f^{(0)} = 0$ and $f^{(n+1)} = f \circ \langle f^{(n)}, \mathbf{id}_Y \rangle$. It is known that equipped with this dagger operation, $\mathbf{Ser}_{S,\Sigma}$ is an iteration category.

7.2 The category $\mathbf{Ser}^\omega_{S,\Sigma}$.

Suppose now that S is a commutative continuous ω-semiring satisfying the infinite commutativity identity. Then we define another category $\mathbf{Ser}^\omega_{S,\Sigma}$ with sets as objects as above. However, a morphism $f : X \to Y$ is now a triplet (f_0, f_M, f_ω) with $f_0 : X \to Y$ in $\mathbf{Ser}_{S,\Sigma}$ and $(f_M, f_\omega) : X \to Y$ in $\mathbf{Matr}_{(S\langle\!\langle (Y \uplus \Sigma)^* \rangle\!\rangle, S\langle\!\langle (Y \uplus \Sigma)^\omega \rangle\!\rangle)}$. Hence, $f_0 : X \to S\langle\!\langle (Y \uplus \Sigma)^* \rangle\!\rangle$, $f_M \in S\langle\!\langle (Y \uplus \Sigma)^* \rangle\!\rangle^{X \times Y}$ and $f_\omega : X \to S\langle\!\langle (Y \uplus \Sigma)^\omega \rangle\!\rangle$.

Composition is defined as follows. Let $f : X \to Y$ and $g : Y \to Z$. Then the components of $h = f \circ g : X \to Z$ are given by

- $h_0 = f_0 \circ g_0$, where the composition is taken from $\mathbf{Ser}_{S,\Sigma}$, and
- $(h_M, h_\omega) = g_0^\sharp((f_M, f_\omega)) \circ (g_M, g_\omega) = (g_0^\sharp(f_M), g_0^\sharp(f_\omega)) \circ (g_M, g_\omega)$ where the composition is taken from the category $\mathbf{Matr}_{(S\langle\!\langle (Z \uplus \Sigma)^* \rangle\!\rangle, S\langle\!\langle (Z \uplus \Sigma)^\omega \rangle\!\rangle)}$.

Note that the definition is legitimate, since $(S\langle\!\langle (Z \uplus \Sigma)^* \rangle\!\rangle, S\langle\!\langle (Z \uplus \Sigma)^\omega \rangle\!\rangle)$ is a continuous ω-semiring-semimodule pair and $g_0^\sharp((f_M, f_\omega)) = (g_0^\sharp(f_M), g_0^\sharp(f_\omega))$ is a morphism $X \to Y$ and (g_M, g_ω) is a morphism $Y \to Z$ in $\mathbf{Matr}_{(S\langle\!\langle (Z \uplus \Sigma)^* \rangle\!\rangle, S\langle\!\langle (Z \uplus \Sigma)^\omega \rangle\!\rangle)}$. Of course, $g_0^\sharp$ is the extension of g_0 to a continuous ω-semiring-semimodule morphism and $g_0^\sharp(f_M)$ and $g_0^\sharp(f_\omega)$ are formed component-wise. It is a routine matter to verify that composition is associative. The identity morphism $X \to X$ is determined by the corresponding identity morphisms in $\mathbf{Ser}_{S,\Sigma}$ and $\mathbf{Matr}_{(S\langle\!\langle (X \uplus \Sigma)^* \rangle\!\rangle, S\langle\!\langle (X \uplus \Sigma)^\omega \rangle\!\rangle)}$.

Each hom-set of morphisms $X \to Y$ is partially ordered by the component-wise order inherited from $\mathbf{Ser}_{(S,\Sigma)}$ and $\mathbf{Matr}_{(S\langle\!\langle (Y \uplus \Sigma)^* \rangle\!\rangle, S\langle\!\langle (Y \uplus \Sigma)^\omega \rangle\!\rangle)}$. Also, each hom-set has the structure of a commutative monoid. For any morphisms $f = (f_0, f_M, f_\omega)$ and $g = (g_0, g_M, g_\omega) : X \to Y$, we define

$$f + g = (f_0 + g_0, f_M + g_M, f_\omega + g_\omega).$$

The components of the morphism $0_{X,Y}$ are the respective zero morphisms.

The category $\mathbf{Ser}^\omega_{S,\Sigma}$ has finite coproducts. On objects, coproduct is again given by disjoint sum.

We now define a dagger operation. Let $f : X \to X \oplus Y, f = (f_0, f_M, f_\omega)$. Then we define the components of $f^\dagger : X \to Y$ as $f_0^\dagger : X \to Y$ in $\mathbf{Ser}_{S,\Sigma}$ and

$$((f^\dagger)_M, (f^\dagger)_\omega) = \langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M,\ f_\omega)^\dagger) = (\langle f_0, \mathbf{id}_Y\rangle^\sharp(f_M,\ f_\omega)^\dagger) : X \to Y$$

in $\mathbf{Matr}_{(S\langle\langle (Y \uplus \Sigma)^*\rangle\rangle, S\langle\langle (Y\uplus\Sigma)^\omega\rangle\rangle)}$.

We prove that the fixed point identity holds. To this end, let $f : X \to X \oplus Y$ as above. Then

$$f \circ \langle f^\dagger, \mathbf{id}_Y\rangle = (\langle f_0 \circ \langle f_0^\dagger, \mathbf{id}_Y\rangle, \langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega)) \circ \langle\langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega)^\dagger), \mathbf{id}_Y\rangle.$$

Hence, the first component of $f^\dagger$ is $f_0^\dagger$. The second and third are given by

$$\begin{aligned}\langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp&((f_M, f_\omega)) \circ \langle\langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega)^\dagger)), \mathbf{id}_Y\rangle\\ &= \langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega) \circ \langle (f_M, f_\omega)^\dagger), \mathbf{id}_Y\rangle)\\ &= \langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M,\ f_\omega)^\dagger).\end{aligned}$$

Hence $f \circ \langle f^\dagger, \mathbf{id}_Y\rangle = f^\dagger$.

Theorem 7.1. $\mathbf{Ser}^\omega_{(S,\Sigma)}$ *is an iteration category.*

Proof. We have already proved that the fixed point identity holds. In order to complete the proof, we establish the parameter, double dagger and group identities.

First we consider the parameter identity. Let $f = (f_0, f_M, f_\omega) : X \to X \oplus Y$ and $g = (g_0, g_M, g_\omega) : Y \to Z$. We want to prove that $(f \circ (\mathbf{id}_X \oplus g))^\dagger = f^\dagger \circ g$.

It is clear that the first components of $(f \circ (\mathbf{id}_X \oplus g))^\dagger$ and $f^\dagger \circ g$ are $(f_0 \circ (\mathbf{id}_X \oplus g_0))^\dagger$ and $f_0^\dagger \circ g$, respectively. Since the parameter identity holds in $\mathbf{Ser}_{S,\Sigma}$, we conclude that the first components are equal.

The second and third components of $f \circ (\mathbf{id}_X \oplus g)$ are given by

$$(\mathbf{id}_X \oplus g_0)^\sharp((f_M, f_\omega)) \circ (\mathbf{id}_X \oplus (g_M, g_\omega)),$$

hence the corresponding components of $(f \circ (\mathbf{id}_X \oplus g))^\dagger$ are given by

$$\langle (f_0 \circ (\mathbf{id}_X \oplus g_0))^\dagger, \mathbf{id}_Z\rangle^\sharp(((\mathbf{id}_X \oplus g_0)^\sharp((f_M, f_\omega)) \circ (\mathbf{id}_X \oplus (g_M, g_\omega)))^\dagger)$$

which is

$$\begin{aligned}&\langle f_0^\dagger \circ g_0, \mathbf{id}_Z\rangle^\sharp(((\mathbf{id}_X \oplus g_0)^\sharp((f_M, f_\omega)))^\dagger \circ (g_M, g_\omega))\\ &= ((\mathbf{id}_X \oplus g_0) \circ \langle f_0^\dagger \circ g_0, \mathbf{id}_Z\rangle)^\sharp((f_M, f_\omega)^\dagger) \circ \langle f_0^\dagger \circ \langle g_0, \mathbf{id}_Z\rangle^\sharp(g_M, g_\omega)\\ &= ((\mathbf{id}_X \oplus g_0) \circ \langle f_0^\dagger \circ g_0, \mathbf{id}_Z\rangle)^\sharp((f_M, f_\omega)^\dagger) \circ (g_M, g_\omega).\end{aligned}$$

On the other hand, the second and third components of $f^\dagger$ and $f^\dagger \circ g$ are respectively given by $\langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega)^\dagger)$ and

$$g_0^\sharp(\langle f_0^\dagger, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega)^\dagger)) \circ (g_M, g_\omega) = (\langle f_0^\dagger, \mathbf{id}_Y\rangle \circ g_0)^\sharp((f_M, f_\omega)^\dagger) \circ (g_M, g_\omega).$$

But $(\mathbf{id}_X \oplus g_0) \circ \langle f_0^\dagger \circ g_0, \mathbf{id}_Z\rangle = \langle f_0^\dagger \circ g_0, g_0\rangle = \langle f_0^\dagger, \mathbf{id}_Y\rangle \circ g_0$, so that the second and third components are also equal.

We prove that the double dagger identity holds. To this end, let $f = (f_0, f_M, f_\omega) : X \to X \oplus X \oplus Y$ and $\tau = \langle \mathbf{id}_X, \mathbf{id}_X\rangle \oplus \mathbf{id}_Y$. Then the first component of $f^{\dagger\dagger}$ is $f_0^{\dagger\dagger}$, and the first component of $(f \circ \tau)^\dagger$ is $(f_0 \circ \tau)^\dagger$. These are equal since the double dagger identity holds in $\mathbf{Ser}_{S,\Sigma}$.

The second and third components of $f^\dagger$ and $f^{\dagger\dagger}$ are

$$\langle f_0^\dagger, \mathbf{id}_{X\oplus Y}\rangle^\sharp((f_M, f_\omega)^\dagger)$$

and

$$\begin{aligned}
&\langle f_0^{\dagger\dagger}, \mathbf{id}_Y\rangle^\sharp(\langle f_0^\dagger, \mathbf{id}_{X\oplus Y}\rangle^\sharp((f_M, f_\omega)^\dagger)))\\
&= (\langle f_0^\dagger, \mathbf{id}_{X\oplus Y}\rangle \circ \langle f_0^{\dagger\dagger}, \mathbf{id}_Y\rangle)^\sharp((f_M, f_\omega)^{\dagger\dagger})\\
&= (\langle f_0^\dagger \circ \langle f_0^{\dagger\dagger}, \mathbf{id}_Y\rangle, f_0^{\dagger\dagger}, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega)^{\dagger\dagger})\\
&= \langle f_0^{\dagger\dagger}, f_0^{\dagger\dagger}, \mathbf{id}_Y\rangle^\sharp((f_M, f_\omega)^{\dagger\dagger})\\
&= (\tau \circ \langle f^{\dagger\dagger}, \mathbf{id}_Y\rangle)^\sharp((f_M, f_\omega)^{\dagger\dagger}),
\end{aligned}$$

respectively. Now the second and third components of $f \circ \tau$ are given by $\tau^\sharp((f_M, f_\omega)) \circ \tau$ and thus the second and third components of $(f \circ \tau)^\dagger$ are given by

$$\begin{aligned}
&\langle (f \circ \tau)^\dagger, \mathbf{id}_Y\rangle^\sharp((\tau^\sharp((f_M, f_\omega)) \circ \tau)^\dagger)\\
&= \langle f^{\dagger\dagger}, \mathbf{id}_Y\rangle^\sharp((\tau^\sharp((f_M, f_\omega)) \circ \tau)^\dagger)\\
&= (\langle f^{\dagger\dagger}, \mathbf{id}_Y\rangle)^\sharp((\tau^\sharp((f_M, f_\omega)))^{\dagger\dagger})\\
&= (\langle f^{\dagger\dagger}, \mathbf{id}_Y\rangle)^\sharp(\tau^\sharp(((f_M, f_\omega))^{\dagger\dagger}))\\
&= (\tau \circ \langle f^{\dagger\dagger}, \mathbf{id}_Y\rangle)^\sharp((f_M, f_\omega)^{\dagger\dagger}).
\end{aligned}$$

Hence the double dagger identity holds.

Our next task is to prove that the (simplified) composition identity holds:

$$(f \circ g)^\dagger = f \circ (g \circ (f \oplus \mathbf{id}_Z))^\dagger,$$

where $f : X \to Y$ and $g : Y \to X \oplus Z$. We will establish this identity only in the case when Z is the initial object $\varnothing$. In this case, the identity takes the following form:

$$(f \circ g)^\dagger = f \circ (g \circ f)^\dagger,$$

where $f : X \to Y$ and $g : Y \to X$.

It is again clear that the first components of the two sides are equal, since the identity holds in $\mathbf{Ser}_{S,\Sigma}$.

Now the second and third components of $(f \circ g)^\dagger$ are

$$\begin{aligned}
&((f_0 \circ g_0)^\dagger)^\sharp((g_0^\sharp((f_M, f_\omega)) \circ (g_M, g_\omega))^\dagger)\\
&= ((g_0 \circ (f_0 \circ g_0)^\dagger)^\sharp((f_M, f_\omega)) \circ (f_0 \circ g_0)^\dagger)^\sharp((g_M, g_\omega)))^\dagger\\
&= (((g_0 \circ f_0)^\dagger)^\sharp((f_M, f_\omega)) \circ ((f_0 \circ g_0)^\dagger)^\sharp((g_M, g_\omega)))^\dagger.
\end{aligned}$$

The corresponding components of $f \circ (g \circ f)^\dagger$ are

$$\begin{aligned}
&((g_0 \circ f_0)^\dagger)^\sharp((f_M, f_\omega)) \circ ((g_0 \circ f_0)^\dagger)^\sharp((f_0^\sharp((g_M, g_\omega)) \circ (f_M, f_\omega))^\dagger)\\
&= ((g_0 \circ f_0)^\dagger)^\sharp((f_M, f_\omega)) \circ (((f_0 \circ g_0)^\dagger)^\sharp((g_M, g_\omega)) \circ ((g_0 \circ f_0)^\dagger)^\sharp((f_M, f_\omega)))^\dagger\\
&= (((g_0 \circ f_0)^\dagger)^\sharp((f_M, f_\omega)) \circ ((f_0 \circ g_0)^\dagger)^\sharp((g_M, g_\omega)))^\dagger.
\end{aligned}$$

Hence, the second and third components are also equal.

Our last task is to prove that the group identities hold. Suppose that G is a finite group of order n whose elements are the integers $\{1, \ldots, n\}$, say. Let $g : X \to X \oplus \cdots \oplus X \oplus Y$, where there are n occurrences of X in the target. For each $i \in \{1, \ldots, n\}$, let

$$\rho_i = \langle \mathbf{in}_{i\cdot 1}, \ldots, \mathbf{in}_{i\cdot n} \rangle \oplus \mathbf{id}_Y : X \oplus \cdots \oplus X \oplus Y \to X \oplus \cdots \oplus X \oplus Y.$$

Moreover, let

$$\tau = \langle \mathbf{id}_X, \ldots, \mathbf{id}_X \rangle : X \oplus \cdots \oplus X \to X.$$

The group identity associated with G is

$$\langle g \circ \rho_1, \ldots, g \circ \rho_n \rangle^\dagger = \tau \circ (g \circ (\tau \oplus \mathbf{id}_Y))^\dagger.$$

We will prove this only in the case when $Y = \varnothing$. The first components of the two sides are again equal. The second and third components of the morphism on the left hand side are given by

$$\begin{aligned}
&(\langle g_0 \circ \rho_1, \ldots, g_0 \circ \rho_n \rangle^\dagger)^\sharp(\langle (g_M, g_\omega) \circ \rho_1, \ldots, (g_M, g_\omega) \circ \rho_n \rangle^\dagger)\\
&= (\tau \circ (g_0 \circ \tau)^\dagger)^\sharp(\langle (g_M, g_\omega) \circ \rho_1, \ldots, (g_M, g_\omega) \circ \rho_n \rangle^\dagger)\\
&= \langle (\tau \circ (g_0 \circ \tau)^\dagger)^\sharp((g_M, g_\omega)) \circ \rho_1, \ldots, (\tau \circ (g_0 \circ \tau)^\dagger)^\sharp((g_M, g_\omega)) \circ \rho_n \rangle^\dagger\\
&= \tau \circ ((\tau \circ (g_0 \circ \tau)^\dagger)^\sharp((g_M, g_\omega)) \circ \tau)^\dagger.
\end{aligned}$$

But this is exactly the morphism determined by the second and third components of the morphism on the right hand side of the group identity associated with G. This completes the proof of the theorem. $\square$

Remark of the second author.

From May 17 to May 20, 2016, Zoltan stayed and worked in my home and almost finished his presumably final paper as coauthor. He wrote this paper in his typical style of working: After extensive discussions he typed the tex file directly into the computer without using a concept or notes. This paper is complete except for the announced "Conclusion". His planned content of the "Conclusion" is unknown to me and I have not tried to reconstruct it. Therefore, this paper appears according to Zoltan's conceptions.

References

[1] S.L. Bloom and Z. Ésik: *Iteration Theories.* Springer, 1993.

[2] S.L. Bloom, Z. Ésik: Axiomatizing rational power series over natural numbers. Inf. Comput. 207(2009), 793–811.

[3] J.H. Conway: *Regular Algebra and Finite Machines*, Chapman and Hall, Ltd., 1971.

[4] Z. Ésik: Group axioms for iteration, *Information and Computation*, 148(1999), 131– 180.

[5] Z. Ésik: Equational properties of fixed point operations in cartesian categories: An overview. In: *MFCS 2015*, Springer, LNCS 9234, 2015, 18–37.

[6] Z. Ésik and W. Kuich, *Modern Automata Theory,* available from http://www.dmg.tuwien.ac.at/kuich/

[7] Z. Ésik and W. Kuich: Solving fixed point equations over complete semirings, to appear in Festschrift for Janusz Brzozowski's 80th birthday, World Scientific, 2017.

[8] J. Golan: *Semirings and their Applications*, Springer, 1999.

[9] J. Golan: *Semirings and Affine Equations over Them*: *Theory and Applications*, Springer, 2003.

[10] R.S. Cohen, A.Y. Gold: Theory of omega-Languages. I. Characterizations of omega-Context-Free Languages. J. Comput. Syst. Sci., 15(1977), 169–184.

[11] W. Kuich and A. Salomaa: *Semirings, Automata, Languages*, Springer, 1986.

[12] G. Markowsky: Chain complete posets and directed sets with applications, *Algebra Universalis*, 6(1976), 53–68.

Acta Cybernetica **23** (2017) 81–90.

Trace Simulation Semantics is not Finitely Based over BCCSP[*]

Luca Aceto[a], David de Frutos Escrig[b], and Anna Ingólfsdóttir[a]

Abstract

This note shows that the trace simulation preorder does not have a finite inequational basis over the language BCCSP. Indeed, no collection of sound inequations of bounded depth is ground-complete with respect to the trace simulation preorder over BCCSP even over a singleton set of actions.

Keywords: trace simulation preorder, complete axiomatizations, BCCSP

1 Introduction

The study of the equational theory of several algebraic structures has been one of the main research interests of the late Zoltán Ésik—see, for instance, the references [2, 3, 8, 9, 10, 13, 14, 16] for a small sample of his work in this area.

In the setting of process algebras, the study of complete axiomatizations of behavioural equivalences can be traced back to the early contributions of Hennessy and Milner [18], and Bergstra and Klop [7]. Since then, the investigation of the equational theory of various process algebras has been a major topic of research and Zoltán Ésik has contributed to this field in many ways—see, for instance, [1, 11, 15]

A complete axiomatization of a behavioural congruence yields a purely syntactic characterization, independent of the actual details of the chosen semantic model for processes and of the definition of the behavioural equivalence, of the semantics of a process algebra. This bridge between syntax and semantics plays an important role in both the practice and the theory of process algebras. From the point of view of practice, these proof systems can be used to perform system verifications in a purely syntactic way using general purpose theorem provers or proof checkers, and form the basis of purpose built axiomatic verification tools. From the theoretical point of view, complete axiomatizations of behavioural equivalences capture the essence of

[*]Luca Aceto and Ingólfsdóttir have been partially supported by the project 'Nominal Structural Operational Semantics' (nr. 141558-041) of the Icelandic Research Fund.

[a]ICE-TCS, School of Computer Science, Reykjavik University, Iceland, E-mail: `{luca,annai}@ru.is`

[b]Departamento de Sistemas Informáticos y Computación, Universidad Complutense de Madrid, Spain, E-mail: `defrutos@sip.ucm.es`

DOI: 10.14232/actacyb.23.1.2017.6

different notions of semantics for processes in terms of a basic collection of identities, and this often allows one to compare semantics which may have been defined in very different styles and frameworks. A review of existing complete equational axiomatizations for many of the behavioural semantics in van Glabbeek's spectrum is offered in [26]. The equational axiomatizations offered in that reference are over the language BCCSP, a common fragment of Milner's CCS [21, 22] and Hoare's CSP [19] suitable for describing finite synchronization trees, and characterize the differences between behavioural semantics in terms of a few revealing axioms.

In this paper, we contribute to the study of the equational theory of semantic equivalences over BCCSP by showing that the trace simulation preorder does not have a finite inequational basis over the language BCCSP (Theorem 1). Indeed, no collection of sound inequations of bounded depth is ground-complete with respect to the trace simulation preorder over BCCSP even over a singleton set of actions (Theorem 2). The proof of our main result is proof theoretic. We are sure that Zoltán Ésik would have preferred to see a model-theoretic argument, like those he used with two of the authors of this paper in joint work on the max-plus algebra of the natural numbers and on the equational theory of tropical semirings [2, 3], but we hope that he would have found our result and its proof appealing nonetheless.

The paper is organized as follows. Section 2 presents preliminaries on the syntax and semantics of BCCSP, the behavioural equivalences and preorders we study and inequational logic. Section 3 introduces our main result, whose proof is given in Section 3.1.

2 Preliminaries

Syntax of BCCSP We work with BCCSP [26, 19, 22] over the action set A. This language is a basic process algebra for expressing finite process behaviour. Its syntax consists of closed (process) terms p, q that are constructed from a constant $\mathbf{0}$, a binary operator $\_+\_$ called *alternative composition*, and the unary *prefix* operators $a\_$ with $a \in A$. Open terms t, u can, moreover, contain occurrences of variables from a countably infinite set V (with typical elements x, y, z).

In what follows, for each $n \geq 0$, we use $a^n\mathbf{0}$ to stand for the term $\mathbf{0}$ if $n = 0$, and for $a(a^{n-1}\mathbf{0})$ if $n > 0$.

A (closed) substitution maps variables in V to (closed) terms. For every term t and substitution σ, the term $\sigma(t)$ is obtained by replacing every occurrence of a variable x in t by $\sigma(x)$. Note that $\sigma(t)$ is closed if σ is a closed substitution.

Transition rules Closed BCCSP terms denote finite process behaviours, where $\mathbf{0}$ does not exhibit any behaviour, $p + q$ is the nondeterministic choice between the behaviours of p and q, and ap executes action a to transform into p. This intuition is captured, in the style of Plotkin [25], by the transition rules below, which give rise to a-labelled transitions, with $a \in A$, between closed terms.

$$\frac{}{ax \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

The operational semantics is extended to open terms by assuming that variables do not exhibit any behaviour. We write $t \not\rightarrow$ if there are no action a and term t' such that $t \xrightarrow{a} t'$ holds.

For each $s \in A^*$, the transition relation $\xrightarrow{s}$ is defined thus, where ε denotes the empty string:

- $t \xrightarrow{\varepsilon} t'$ if, and only if, $t = t'$;
- $t \xrightarrow{as} t'$ if, and only if, there is some t'' such that $t \xrightarrow{a} t'' \xrightarrow{s} t'$.

If $t \xrightarrow{s} t'$, then we say that s is a *trace* of t. Such a trace is *complete* if $t' \not\rightarrow$.

For each BCCSP term t, we define

$$T(t) = \{s \mid t \xrightarrow{s} t' \text{ for some } t'\}.$$

The *depth* of a term t, written depth(t), is the length of a longest trace $s \in T(t)$. The *norm* of a term t, written norm(t), is the length of a shortest complete trace $s \in T(t)$. (The notion of norm stems from [6].) For example, the closed term $a^2 + a^3$ has norm two and depth three.

Simulation, bisimulation and trace simulation We define the following three variations on the notion of simulation over closed BCCSP terms.

Definition 1 (Simulations). *A binary relation $\mathcal{R}$ over closed BCCSP terms is:*

- *a* simulation *[20, 24] if $p \mathrel{\mathcal{R}} q$ and $p \xrightarrow{a} p'$ imply $q \xrightarrow{a} q'$ for some q' with $p' \mathrel{\mathcal{R}} q'$;*
- *a* bisimulation *[22, 24] if it is a simulation whose inverse is also a simulation;*
- *a* trace simulation *if it is a simulation that satisfies the following condition:*

  $p \mathrel{\mathcal{R}} q$ *implies* $T(q) = T(p)$.

We write $p \precsim_{TS} q$ if there is a trace simulation $\mathcal{R}$ with $p \mathrel{\mathcal{R}} q$, and $p \mathrel{\underline{\leftrightarrow}} q$ if there is a bisimulation $\mathcal{R}$ with $p \mathrel{\mathcal{R}} q$. We will refer to $\precsim_{TS}$ as the trace simulation preorder*, and to $\underline{\leftrightarrow}$ as* bisimilarity*.*

Let $\precsim \in \{\precsim_{TS}, \underline{\leftrightarrow}\}$. We define $t \precsim u$ if $\sigma(t) \precsim \sigma(u)$ for each closed substitution σ.

It is well known that $\precsim_{TS}$ is a preorder and $\underline{\leftrightarrow}$ is an equivalence relation. Moreover, both relations are preserved by the operators of the language BCCSP.

Inequational logic An *inequation* (respectively, an *equation*) over the language BCCSP is a formula of the form $t \leq u$ (respectively, $t = u$), where t and u are BCCSP terms. An *(in)equational axiom system* is a collection of (in)equations over the language BCCSP. An equation $t = u$ is derivable from an equational axiom system E if it can be proven from the axioms in E using the rules of equational logic (viz. reflexivity, symmetry, transitivity, substitution and closure under BCCSP contexts).

$$t = t \quad \frac{t = u}{u = t} \quad \frac{t = u \quad u = v}{t = v} \quad \frac{t = u}{\sigma(t) = \sigma(u)} \quad \frac{t = u}{at = au} \quad \frac{t = u \quad t' = u'}{t + t' = u + u'}$$

For the derivation of an inequation $t \leq u$ from an inequational axiom system E, the rule for symmetry is omitted.

It is well known that, without loss of generality, one may assume that substitutions happen first in (in)equational proofs, i.e., that the fourth rule may only be used when its premise is one of the (in)equations in E. Moreover, by postulating that for each equation in E also its symmetric counterpart is present in E, one may assume that applications of symmetry happen first in equational proofs, i.e., that the second rule is never used in equational proofs. (See, e.g., [12, page 497] for a thorough discussion of this 'normalized equational proofs'.) In the remainder of this paper, we shall always tacitly assume that equational axiom systems are closed with respect to symmetry. Note that, with this assumption, there is no difference between the rules of inference of equational and inequational logic. In what follows, we shall consider an equation $t = u$ as a shorthand for the pair of inequations $t \leq u$ and $u \leq t$.

The depth of $t \leq u$ and $t = u$ is the maximum of the depths of t and u. The depth of a collection of (in)equations is the supremum of the depths of its elements.

An inequation $t \leq u$ is *sound* with respect to $\precsim_{TS}$ if $t \precsim_{TS} u$ holds. For example, as our readers can readily check, the inequation

$$ax \leq ax + x \tag{1}$$

is sound with respect to $\precsim_{TS}$ if $A = \{a\}$ and is unsound otherwise.

An (in)equational axiom system E is sound with respect to $\precsim_{TS}$ if so is each (in)equation in E. It is *complete* if each valid inequation $t \precsim_{TS} u$ can be derived from E, and it is *ground complete* if each valid inequation $t \precsim_{TS} u$ relating *closed terms* can be derived from E. A set of complete and sound (in)equations is sometimes referred to as an *(in)equational basis.*

The core axioms A1–A4 for BCCSP given below are classic and stem from [18]. They are complete [23], and sound and ground complete [18, 22], over BCCSP (over any nonempty set of actions) modulo bisimulation equivalence [22, 24], which is the finest semantics in van Glabbeek's spectrum [26].

$$\begin{array}{llcl}
\text{A1} & x + y & \approx & y + x \\
\text{A2} & (x + y) + z & \approx & x + (y + z) \\
\text{A3} & x + x & \approx & x \\
\text{A4} & x + \mathbf{0} & \approx & x
\end{array}$$

In what follows, for notational convenience, we consider terms up to the least congruence generated by axioms A1–A4, that is, up to bisimulation equivalence.

3 The negative result

Our aim in what follows is to show the following theorem.

Theorem 1. *The (in)equational theory of $\precsim_{TS}$ over BCCSP does not have a finite inequational basis. In particular, no finite set of sound inequations over BCCSP modulo $\precsim_{TS}$ can prove all of the sound inequations in the family*

$$a^{2m} \leq a^{2m} + a^m \qquad (m \geq 0).$$

In what follows, we shall present a proof of the above result, which has proof has a 'proof-theoretic' flavour.

Remark 1. The family of inequations in the statement of Theorem 1 was used in [5, 4] to prove that the 2-nested simulation preorder from [17] does not afford a finite ground-complete inequational axiomatization over BCCSP.

3.1 A proof-theoretic argument for Theorem 1

Our proof of Theorem 1 is based on obtaining that result as a corollary of the following one.

Theorem 2. *Let E be a collection of inequations whose elements are sound modulo $\precsim_{TS}$ and have depth smaller than m. Suppose furthermore that the closed inequation $p \leq q$ is derivable from E, that $q \precsim_{TS} a^{2m} + a^m$ and* $\text{norm}(p) = 2m$. *Then* $\text{norm}(q) = 2m$.

Having shown the above result, Theorem 1 can be proved as follows. Let E be a finite inequational axiom system that is sound modulo $\precsim_{TS}$. Pick m larger than the depth of E. (Such an m exists since E is finite.) Then, by Theorem 2, E cannot prove the valid inequation

$$a^{2m} \leq a^{2m} + a^m,$$

and is therefore incomplete. Indeed, a^{2m} has norm $2m$, but $a^{2m} + a^m$ has norm m.

In the remainder of this section, we shall present a proof of Theorem 2. In order to show that result, we shall first prove that the property mentioned in that statement holds true for instantiations of sound inequations whose depth is smaller than m. Next we use this fact to argue that the stated property is preserved by arbitrary inequational derivations from a collection of inequations whose elements have depth smaller than m and are sound modulo $\precsim_{TS}$.

Definition 2. *We say that a term t has an occurrence of variable x reachable via a sequence of actions s if there is some term t' such that $t \xrightarrow{s} x + t'$.*

For example, $ax + a\mathbf{0}$ has an occurrence of x reachable via a because $ax + a\mathbf{0} \xrightarrow{a} x$ and $x = x + \mathbf{0}$.

Lemma 1. *Assume that $t \precsim_{TS} u$ and that u has an occurrence of variable x reachable via a sequence of actions s. Then t also has an occurrence of variable x reachable via some sequence of actions s'.*

Proof. Assume that $t \precsim_{TS} u$ and that u has an occurrence of variable x reachable via a sequence of actions s. Let m be larger than the depth of t. Consider the closed substitution σ mapping x to a^m and every other variable to $\mathbf{0}$. Since u has an occurrence of variable x reachable via s, it is easy to see that $\sigma(u) \xrightarrow{sa^m} \mathbf{0}$. As $\sigma(t) \precsim_{TS} \sigma(u)$ because $t \precsim_{TS} u$ by assumption, it must be the case that $\sigma(t) \xrightarrow{sa^m} p$ for some p. As the depth of t is smaller than m, the substitution σ maps all variables different from x to $\mathbf{0}$ and $\sigma(u) \xrightarrow{sa^m} p$, it follows that $t \xrightarrow{s'} x + t'$ for some t', which was to be shown. $\square$

Remark 2. Note that, in general, the traces s and s' mentioned in the statement of the above lemma need not be equal. For instance, as we observed previously, the inequation

$$ax \leq ax + x$$

is sound with respect to $\precsim_{TS}$ if $A = \{a\}$ and the term $ax + x$ has an occurrence of variable x reachable via the sequence of actions ε. However, the only occurrence of x in the term ax is reachable via the sequence of actions a.

The following lemma is the first stepping stone towards the proof of Theorem 2. It establishes that the property mentioned in that statement holds true for instantiations of sound inequations whose depth is smaller than m.

Lemma 2. *Suppose that $t \precsim_{TS} u$ and that m is larger than the depth of u. Let σ be a closed substitution. Suppose, furthermore, that $\sigma(u) \precsim_{TS} a^{2m} + a^m$ and* $\mathrm{norm}(\sigma(t)) = 2m$. *Then* $\mathrm{norm}(\sigma(u)) = 2m$.

Proof. The assumption that $\sigma(u) \precsim_{TS} a^{2m} + a^m$ yields that $\mathrm{norm}(\sigma(u)) = 2m$ or $\mathrm{norm}(\sigma(u)) = m$. Assume, towards a contradiction, that $\mathrm{norm}(\sigma(u)) = m$. Then, since $\mathrm{depth}(u) < m$, there are some $i < m$ and some variable x such that u has an occurrence of variable x reachable via a^i and $\sigma(x) \xrightarrow{a^{m-i}} \mathbf{0}$. Since $t \precsim_{TS} u$ and $\mathrm{depth}(t) < m$ too (because $t \precsim_{TS} u$ clearly implies that $\mathrm{depth}(t) = \mathrm{depth}(u)$ and $\mathrm{depth}(u) < m$ by our assumption), there is some $j < m$ such that t has an occurrence of variable x reachable via a^j. But then $\sigma(t)$ has a trace of length $j + (m - i) < 2m$ leading to $\mathbf{0}$. This contradicts the assumption that $\mathrm{norm}(\sigma(t)) = 2m$. Therefore $\mathrm{norm}(\sigma(u)) = 2m$, as claimed. $\square$

We will now argue that the property stated in Theorem 2 is preserved by arbitrary inequational derivations from a collection of inequations whose elements are sound modulo $\precsim_{TS}$ and have depth smaller than m. The following lemma will allow us to handle closure under action prefixing in that proof.

Lemma 3. *Assume that $aq \precsim_{TS} a^{2m} + a^m$. Then* $\mathrm{norm}(aq) = 2m$.

Proof. By our assumptions, it follows that $m \geq 1$, $\text{depth}(aq) = 2m$ and that $\text{norm}(aq) = 2m$ or $\text{norm}(aq) = m$.

Assume, towards a contradiction, that $\text{norm}(aq) = m$. Then q has depth $2m-1$ and norm $m-1$. Since $aq \precsim_{TS} a^{2m} + a^m$ and $\text{depth}(q) = 2m-1$, it must be the case that $q \precsim_{TS} a^{2m-1}$. But this is impossible since q can terminate in $m-1$ steps and a^{2m-1} cannot. Therefore aq has norm $2m$, as claimed. □

We now have all the necessary ingredients to complete our proof of Theorem 2, and therefore of Theorem 1.

Proof. (of Theorem 2) Assume that E is a collection of inequations whose elements are sound modulo $\precsim_{TS}$ and have depth smaller than m. Suppose furthermore that

- the inequation $p \leq q$ is derivable from E,
- $q \precsim_{TS} a^{2m} + a^m$, and
- $\text{norm}(p) = 2m$.

(Observe that m is positive because it is larger than the depth of E.) We shall prove that $\text{norm}(q) = 2m$ by induction on a closed derivation of $p \leq q$ from E. We proceed by examining the last rule used in the proof of $p \leq q$ from E. The case of reflexivity is trivial and that of transitivity follows by applying the inductive hypothesis twice. If $p \leq q$ is proved by instantiating an inequation in E, then the claim follows by Lemma 2. We are therefore left with the congruence rules, which we examine separately below.

- Suppose that E proves $p \leq q$ because $p = ap'$, $u = aq'$ and E proves $p' \leq q'$ by a shorter inference. By the soundness of E and the proviso of the theorem, we have that
$$p = ap' \precsim_{TS} u = aq' \precsim_{TS} a^{2m} + a^m$$
and $\text{norm}(p) = 2m$. Lemma 3 now yields $\text{norm}(q) = 2m$, as required.

- Suppose that E proves $p \leq q$ because $p = p_1 + p_2$, $q = p_1 + p_2$ and E proves $p_i \leq q_i$, $1 \leq i \leq 2$, by shorter inferences. Since p has norm $2m$ and m is positive, we may assume, without loss of generality, that p_1 has norm $2m$. Moreover, the depth of p_1 is also $2m$, since
$$p = p_1 + p_2 \precsim_{TS} q_1 + q_2 = q \precsim_{TS} a^{2m} + a^m.$$
Therefore q_1 has depth $2m$ because E is sound. Since $q_1 + q_2 \precsim_{TS} a^{2m} + a^m$, for each q_1' such that $q_1 \xrightarrow{a} q_1'$ we have that $q_1' \precsim_{TS} a^{2m-1}$ or $q_1' \precsim_{TS} a^{m-1}$. As q_1 has positive depth, this means that $q_1 \precsim_{TS} a^{2m} + a^m$. We may therefore apply the induction hypothesis to obtain that $\text{norm}(q_1) = 2m$. If p_2 is $\mathbf{0}$ then we are done since, in that case, q_2 is also $\mathbf{0}$ by the soundness of E. If p_2 is not $\mathbf{0}$, then its norm is also $2m$, because p has norm and depth equal to $2m$. But then, reasoning as above, we may infer that $\text{norm}(q_2) = 2m$. Since $q = q_1 + q_2$ and $\text{norm}(q_1) = \text{norm}(q_2) = 2m$, we have that $\text{norm}(q) = 2m$, which was to be shown.

This completes the proof. □

Dedication Luca Aceto and Anna Ingólfsdóttir dedicate this paper to the memory of their collaborator and friend Zoltán Ésik, from whom they have learned much and with whom they have shared many pleasant days. They will miss him.

References

[1] Aceto, Luca, Ésik, Zoltán, and Ingólfsdóttir, Anna. Equational axioms for probabilistic bisimilarity. In Kirchner, Hélène and Ringeissen, Christophe, editors, *Algebraic Methodology and Software Technology, 9th International Conference, AMAST 2002, Saint-Gilles-les-Bains, Reunion Island, France, September 9–13, 2002, Proceedings*, volume 2422 of *Lecture Notes in Computer Science*, pages 239–253. Springer, 2002.

[2] Aceto, Luca, Ésik, Zoltán, and Ingólfsdóttir, Anna. Equational theories of tropical semirings. *Theoretical Computer Science*, 298(3):417–469, 2003.

[3] Aceto, Luca, Ésik, Zoltán, and Ingólfsdóttir, Anna. The max-plus algebra of the natural numbers has no finite equational basis. *Theoretical Computer Science*, 293(1):169–188, 2003.

[4] Aceto, Luca, Fokkink, Wan, and Ingólfsdóttir, Anna. 2-nested simulation is not finitely equationally axiomatizable. In *STACS 2001 – 18th Annual Symposium on Theoretical Aspects of Computer Science,*, volume 2010 of *Lecture Notes in Computer Science*, pages 39–50. Springer, 2001.

[5] Aceto, Luca, Fokkink, Wan, van Glabbeek, Rob, and Ingólfsdóttir, Anna. Nested semantics over finite tree are equationally hard. *Information and Computation*, 191(2):203–232, 2004.

[6] Baeten, Jos, Bergstra, Jan A., and Klop, Jan Willem. Decidability of bisimulation equivalence for processes generating context-free languages. *Journal of the ACM*, 40(3):653–682, 1993.

[7] Bergstra, Jan A. and Klop, Jan Willem. Process algebra for synchronous communication. *Information and Control*, 60(1-3):109–137, 1984.

[8] Bloom, Stephen L. and Ésik, Zoltán. *Iteration Theories - The Equational Logic of Iterative Processes.* EATCS Monographs on Theoretical Computer Science. Springer, 1993.

[9] Bloom, Stephen L. and Ésik, Zoltán. Nonfinite axiomatizability of shuffle inequalities. In Mosses, Peter D., Nielsen, Mogens, and Schwartzbach, Michael I., editors, *TAPSOFT'95: Theory and Practice of Software Development, 6th International Joint Conference CAAP/FASE, Aarhus, Denmark, May 22–26, 1995, Proceedings*, volume 915 of *Lecture Notes in Computer Science*, pages 318–333. Springer, 1995.

[10] Bloom, Stephen L. and Ésik, Zoltán. Iteration algebras are not finitely axiomatizable. extended abstract. In Gonnet, Gaston H., Panario, Daniel, and Viola, Alfredo, editors, *LATIN 2000: Theoretical Informatics, 4th Latin American Symposium, Punta del Este, Uruguay, April 10–14, 2000, Proceedings*, volume 1776 of *Lecture Notes in Computer Science*, pages 367–376. Springer, 2000.

[11] Bloom, Stephen L., Ésik, Zoltán, and Taubner, Dirk. Iteration theories of synchronization trees. *Information and Computation*, 102(1):1–55, 1993.

[12] Chen, Taolue, Fokkink, Wan, Luttik, Bas, and Nain, Sumit. On finite alphabets and infinite bases. *Information and Computation*, 206(5):492–519, 2008.

[13] Crvenkovic, Sinisa, Dolinka, Igor, and Ésik, Zoltán. The variety of Kleene algebras with conversion is not finitely based. *Theoretical Computer Science*, 230(1–2):235–245, 2000.

[14] Ésik, Zoltán. Group axioms for iteration. *Information and Computation*, 148(2):131–180, 1999.

[15] Ésik, Zoltán. Continuous additive algebras and injective simulations of synchronization trees. *Journal of Logic and Computation*, 12(2):271–300, 2002.

[16] Ésik, Zoltán and Bertol, Michael. Nonfinite axiomatizability of the equational theory of shuffle. In Fülöp, Zoltán and Gécseg, Ferenc, editors, *Automata, Languages and Programming, 22nd International Colloquium, ICALP95, Szeged, Hungary, July 10–14, 1995, Proceedings*, volume 944 of *Lecture Notes in Computer Science*, pages 27–38. Springer, 1995.

[17] Groote, Jan Friso and Vaandrager, Frits Willem. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, 1992.

[18] Hennessy, Matthew and Milner, Robin. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32:137–161, 1985.

[19] Hoare, Charles Antony Richard. *Communicating Sequential Processes.* Prentice Hall, 1985.

[20] Milner, Robin. An algebraic definition of simulation between programs. In *Proceedings 2nd Joint Conference on Artificial Intelligence*, pages 481–489. BCS, 1971. Also available as Report No. CS-205, Computer Science Department, Stanford University.

[21] Milner, Robin. *A Calculus of Communicating Systems.* LNCS 92. Springer, 1980.

[22] Milner, Robin. *Communication and Concurrency.* Prentice Hall, 1989.

[23] Moller, Faron. *Axioms for Concurrency.* PhD thesis, Report CST-59-89, Department of Computer Science, University of Edinburgh, 1989.

[24] Park, David M.R. Concurrency and automata on infinite sequences. In *Theoretical Computer Science, 5th GI-Conference*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer, 1981.

[25] Plotkin, Gordon D. A structural approach to operational semantics. *Journal of Logic and Algebraic Programming*, 60–61:17–139, 2004.

[26] van Glabbeek, Rob. The linear time – branching time spectrum I; the semantics of concrete, sequential processes. In Bergstra, J.A., Ponse, A., and Smolka, S.A., editors, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier, 2001. Available at `http://boole.stanford.edu/pub/spectrum1.ps.gz`.

Acta Cybernetica **23** (2017) 91–111.

Commutative Positive Varieties of Languages*

Jorge Almeida[a], Zoltán Ésik[b], and Jean-Éric Pin[c]

To the memory of Zoltán Ésik.

Abstract

We study the commutative positive varieties of languages closed under various operations: shuffle, renaming and product over one-letter alphabets.

Most monoids considered in this paper are finite. In particular, we use the term *variety of monoids* for *variety of finite monoids*. Similarly, all languages considered in this paper are regular languages and hence their syntactic monoid is finite.

1 Introduction

Eilenberg's variety theorem [12] and its ordered version [17] provide a convenient setting for studying classes of regular languages. It states that positive varieties of languages are in one-to-one correspondence with varieties of finite ordered monoids.

There is a large literature on operations on regular languages. For instance, the closure of [positive] varieties of languages under various operations has been extensively studied: Kleene star [16], concatenation product [7, 19, 25], renaming [1, 4, 8, 23, 26] and shuffle [6, 10, 14]. The ultimate goal would be the complete classification of the positive varieties of languages closed under these operations.

*The last two authors acknowledge support from the cooperation programme CNRS/Magyar Tudomanyos Akadémia. The first author was partially supported by CMUP (UID/MAT/00144/2013), which is funded by FCT (Portugal) with national (MCTES) and European structural funds (FEDER), under the partnership agreement PT2020. The third author was partially funded from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 670624) and by the DeLTA project (ANR-16-CE40-0007)

[a]CMUP, Dep. Matemática, Faculdade de Ciências, Universidade do Porto, Rua do Campo Alegre 687, 4169-007 Porto, Portugal. E-mail: `jalmeida@fc.up.pt`

[b]Dept. of Foundations of Computer Science, University of Szeged, Árpád tér 2, H-6720 Szeged, P.O.B. 652 Hungary.

[c]IRIF, CNRS and Université Paris-Diderot, Case 7014, 75205 Paris Cedex 13, France.E-mail: `Jean-Eric.Pin@irif.fr`

DOI: 10.14232/actacyb.23.1.2017.7

The first step in this direction is to understand the commutative case, which is the goal of this paper.

We first show in Theorem 5.1 that every commutative positive *ld*-variety of languages is a positive variety of languages. This means that if a class of commutative languages is closed under Boolean operations and under inverses of length-decreasing morphisms then it is also closed under inverses of morphisms. This result has a curious application in weak arithmetic, stated in Proposition 5.4.

Next we study two operations on languages, shuffle and renaming. These two operations are closely related to the so-called *power operator* on monoids, which associates with each monoid the monoid of its subsets. In its ordered version, it associates with each ordered monoid the ordered monoid of its downsets. We give four equivalent conditions characterizing the commutative positive varieties of languages closed under shuffle (Proposition 6.1) or under renaming (Proposition 6.2).

In order to keep the paper self-contained, prerequisites are presented in some detail in Section 2. Inequalities form the topic of Section 3. We start with their formal definitions, describe their various interpretations and establish some of their properties. General results on renaming are given in Section 4 and more specific results on commutative varieties are proposed in Section 5, including our previously mentioned result on *ld*-varieties. Our characterizations of the positive varieties of languages closed under shuffle or renaming form the meat of Section 6 and are illustrated by three examples in Section 7. Finally, a few research directions are suggested in Section 8.

2 Prerequisites

In this section, we briefly recall the following notions: lattices and (positive) varieties of languages, syntactic ordered monoids, varieties of ordered monoids, stamps, downset monoids, free profinite monoids.

2.1 Languages

Let A be a finite alphabet. Let $[u]$ be the *commutative closure* of a word u, that is, the set of words commutatively equivalent to u. For instance, $[aab] = \{aab, aba, baa\}$. A language L is *commutative* if, for every word $u \in L$, $[u]$ is contained in L.

A *lattice of languages* is a set $\mathcal{L}$ of regular languages of A^* containing $\emptyset$ and A^* and closed under finite union and finite intersection. It is *closed under quotients* if, for each $L \in \mathcal{L}$ and $u \in A^*$, the languages $u^{-1}L$ and Lu^{-1} are also in $\mathcal{L}$.

The *shuffle product* (or simply *shuffle*) of two languages L_1 and L_2 over A is the language

$$\begin{aligned} L_1 \sqcup\!\sqcup L_2 = \{w \in A^* \mid w = u_1v_1 \cdots u_nv_n \text{ for some words } u_1, \ldots, u_n \\ v_1, \ldots, v_n \text{ of } A^* \text{ such that } u_1 \cdots u_n \in L_1 \text{ and } v_1 \cdots v_n \in L_2\} \end{aligned}$$

The shuffle product defines a commutative and associative operation on the set of languages over A.

A *renaming* or *length-preserving morphism* is a morphism φ from A^* into B^*, such that, for each word u, the words u and $\varphi(u)$ have the same length. It is equivalent to require that, for each letter a, $\varphi(a)$ is also a letter, that is, $\varphi(A) \subseteq B$. Similarly, a morphism is *length-decreasing* if the image of each letter is either a letter or the empty word.

A *class of languages* is a correspondence $\mathcal{C}$ which associates with each alphabet A a set $\mathcal{C}(A^*)$ of regular languages of A^*.

A *positive variety of languages* is a class of regular languages $\mathcal{V}$ such that:

(1) for every alphabet A, $\mathcal{V}(A^*)$ is a lattice of languages closed under quotients,

(2) if $\varphi : A^* \to B^*$ is a morphism, $L \in \mathcal{V}(B^*)$ implies $\varphi^{-1}(L) \in \mathcal{V}(A^*)$.

A *variety of languages* is a positive variety $\mathcal{V}$ such that each lattice $\mathcal{V}(A^*)$ is closed under complement. We shall also use two slight variations of these notions. A *positive ld-variety* [*lp-variety*] *of languages* [13, 19] is a class of regular languages $\mathcal{V}$ satisfying (1) and

(2′) if $\varphi : A^* \to B^*$ is a length-decreasing [length-preserving] morphism, then $L \in \mathcal{V}(B^*)$ implies $\varphi^{-1}(L) \in \mathcal{V}(A^*)$.

2.2 Syntactic ordered monoids

An *ordered monoid* is a monoid M equipped with a partial order $\leqslant$ compatible with the product on M: for all $x, y, z \in M$, if $x \leqslant y$ then $zx \leqslant zy$ and $xz \leqslant yz$.

The *ordered syntactic monoid* of a language was first introduced by Schützenberger in [24, p. 10]. Let L be a language of A^*. The *syntactic preorder* of L is the relation $\leqslant_L$ defined on A^* by $u \leqslant_L v$ if, for every $x, y \in A^*$, $xuy \in L$ implies $xvy \in L$. When the language L is clear from the context, we may write $\leqslant$ instead of $\leqslant_L$. As is standard in preorder notation, we write $u < v$ to mean that $u \leqslant v$ holds but $v \leqslant u$ does not.

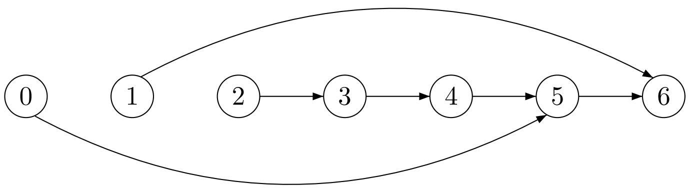
For instance, let $A = \{a\}$. If $L = a + a^3$, then $a^3 \leqslant_L a$, but if $L = a + a^3a^*$, then $a \leqslant_L a^3$.

The associated equivalence relation $\sim_L$, defined by $u \sim_L v$ if $u \leqslant_L v$ and $v \leqslant_L u$, is the *syntactic congruence* of L and the quotient monoid $M(L) = A^*/{\sim_L}$ is the *syntactic monoid* of L. The natural morphism $\eta : A^* \to A^*/{\sim_L}$ is the *syntactic stamp* of L. The *syntactic image* of L is the set $P = \eta(L)$.

The *syntactic order* $\leqslant$ is defined on $M(L)$ as follows: $u \leqslant v$ if and only if for all $x, y \in M$, $xuy \in P$ implies $xvy \in P$. The partial order $\leqslant$ is compatible with multiplication and the resulting ordered monoid $(M, \leqslant)$ is called the *ordered syntactic monoid* of L.

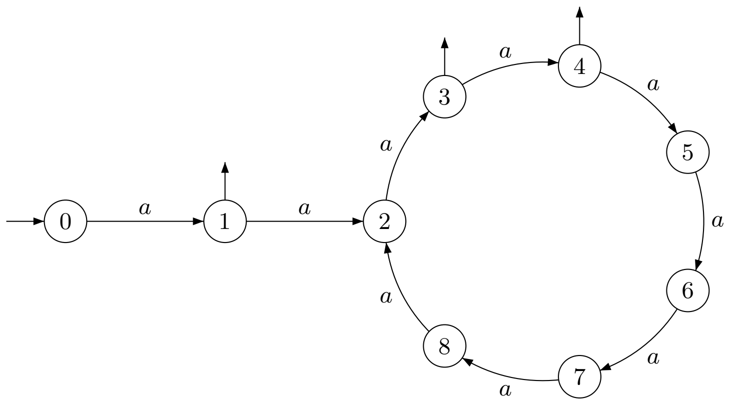
Example 2.1. Let L be the language $1 + a$. The syntactic monoid of L is the commutative monoid $\{1, a, 0\}$ satisfying $a^2 = 0$. The syntactic order is $0 < a < 1$. Indeed, one has $a \leqslant 1$ since, for each $r \geqslant 0$, the condition $a^r a \in L$ implies $a^r \in L$. Similarly, one has $0 \leqslant a$ since, for each $r \geqslant 0$, the condition $a^r a^2 \in L$ implies $a^r a \in L$. However, $1 \nleqslant a$ and $a \nleqslant 0$ since $a \in L$ but $a^2 \notin L$.

Example 2.2. Let L be the language $a + a^6a^*$. The syntactic monoid of L may be identified with the commutative monoid $\{0, 1, \ldots, 6\}$ equipped with the operation $xy = \min\{x + y, 6\}$. In particular, 0 and 6 are the unique idempotents. The syntactic order is represented as follows (a path from i to j means that $i < j$):



For instance, one has $1 < 6$ since, for each $r \geqslant 0$, the condition $aa^r \in L$ implies $a^6a^r \in L$. Similarly, one has $0 < 5$ since, for each $r \geqslant 0$, the condition $a^r \in L$ implies $a^5a^r \in L$. But $1 \not< 5$ since $a \in L$ but $a^5 \notin L$.

Example 2.3. Let L be the language $a + (a^3 + a^4)(a^7)^*$. Its minimal automaton is represented below.



The syntactic monoid of L is the monoid presented by $\langle a \mid a^9 = a^2 \rangle$. The syntatic order is the equality relation.

2.3 Stamps

Monoids and ordered monoids are used to recognise languages, but there is a slightly more restricted notion. A *stamp* is a surjective monoid morphism $\varphi : A^* \to M$ from a finitely generated free monoid A^* onto a finite monoid M. If M is an ordered monoid, φ is called an *ordered stamp.*

The *restricted direct product* of two [ordered] stamps $\varphi_1 : A^* \to M_1$ and $\varphi_2 : A^* \to M_2$ is the stamp φ with domain A^* defined by $\varphi(a) = (\varphi_1(a), \varphi_2(a))$ (see Figure 1). The image of φ is an [ordered] submonoid of the [ordered] monoid $M_1 \times M_2$.

$$\begin{tikzcd}
 & & M_2 \\
A^* \arrow[rru, "\varphi_2"] \arrow[r, "\varphi"] \arrow[rrd, "\varphi_1"'] & \mathrm{Im}(\varphi) \subseteq M_1 \times M_2 \arrow[ru, "\pi_2"'] \arrow[rd, "\pi_1"] & \\
 & & M_1
\end{tikzcd}$$

Figure 1: The restricted direct product of two stamps.

Recall that an *upset* of an ordered set E is a subset U of E such that the conditions $x \in U$ and $x \leqslant y$ imply $y \in U$. A language L of A^* is *recognised by a stamp* $\varphi : A^* \to M$ if there exists a subset P of M such that $L = \varphi^{-1}(P)$. It is *recognised by an ordered stamp* $\varphi : A^* \to M$ if there exists an upset U of M such that $L = \varphi^{-1}(U)$.

It is easy to see that if two languages L_0 and L_1 of A^* are recognised by the [ordered] stamps φ_0 and φ_1, respectively, then $L_0 \cap L_1$ and $L_0 \cup L_1$ are both recognised by the restricted product of φ_0 and φ_1.

2.4 Varieties

Varieties of languages and their avatars all admit an algebraic characterization. We first describe the corresponding algebraic objects and summarize the correspondence results at the end of this section. See [18] for more details.

[Positive] varieties of languages correspond to varieties of [ordered] monoids. A *variety of monoids* is a class of monoids closed under taking submonoids, quotients and finite direct products. *Varieties of ordered monoids* are defined analogously.

The description of the algebraic objects corresponding to positive *lp*- and *ld*-varieties of languages is more complex and relies on the notion of stamp defined in Section 2.3. An *lp-morphism* from a stamp $\varphi : A^* \to M$ to a stamp $\psi : B^* \to N$ is a pair (f, α), where $f : A^* \to B^*$ is length-preserving, $\alpha : M \to N$ is a morphism of [ordered] monoids, and $\psi \circ f = \alpha \circ \varphi$.

$$\begin{tikzcd}
A^* \arrow[r, "f"] \arrow[d, "\varphi"'] & B^* \arrow[d, "\psi"] \\
M \arrow[r, "\alpha"] & N
\end{tikzcd}$$

The *lp*-morphism (f, α) is an *lp-projection* if f is surjective. It is an *lp-inclusion* if α is injective.

An [*ordered*] *lp-variety of stamps* is a class of [ordered] stamps closed under *lp*-projections, *lp*-inclusions and finite restricted direct products. [*Ordered*] *ld*-

varieties of stamps are defined in the same way, just by replacing lp by ld and length-preserving by length-decreasing everywhere in the definition.

Here are the announced correspondence results. Eilenberg's variety theorem [12] and its ordered counterpart [17] give a bijective correspondence between varieties of [ordered] monoids and positive varieties of languages. Let $\mathbf{V}$ be a variety of finite [ordered] monoids and, for each alphabet A, let $\mathcal{V}(A^*)$ be the set of all languages of A^* whose [ordered] syntactic monoid is in $\mathbf{V}$. Then $\mathcal{V}$ is a [positive] variety of languages. Furthermore, the correspondence $\mathbf{V} \to \mathcal{V}$ is a bijection between varieties of [ordered] monoids and [positive] varieties of languages.

There is a similar correspondence for lp-varieties of [ordered] stamps [13, 27]. Let $\mathbf{V}$ be an lp-variety of [ordered] stamps. For each alphabet A, let $\mathcal{V}(A^*)$ be the set of all languages of A^* whose [ordered] syntactic stamp is in $\mathbf{V}$. Then $\mathcal{V}$ is a [positive] lp-variety of languages. Furthermore, the correspondence $\mathbf{V} \to \mathcal{V}$ is a bijection between lp-varieties of [ordered] stamps and [positive] lp-varieties of languages.

Finally, there is a similar statement for ld-varieties of [ordered] stamps.

2.5 Downset monoids

Let $(M, \leqslant)$ be an ordered monoid. A *downset* of M is a subset F of M such that if $x \in F$ and $y \leqslant x$ then $y \in F$. The *product of two downsets* X and Y is the downset

$$XY = \{z \in M \mid \text{there exist } x \in X \text{ and } y \in Y \text{ such that } z \leqslant xy\}$$

This operation makes the set of nonempty downsets of M a monoid, denoted by $\mathcal{P}^{\downarrow}(M)$ and called the *downset monoid* of M. Its identity element is $\downarrow 1$. If one also considers the empty set, one gets a monoid with zero, denoted $\mathcal{P}_0^{\downarrow}(M)$, in which the empty set is the zero. For instance, if M is the trivial monoid, $\mathcal{P}_0^{\downarrow}(M)$ is isomorphic to the ordered monoid $\{0, 1\}$, consisting of an identity 1 and a zero 0, ordered by $0 < 1$. This monoid will be denoted by $U_1^{\downarrow}$.

The monoids $\mathcal{P}_0^{\downarrow}(M)$ and $\mathcal{P}^{\downarrow}(M)$ are closely related. First, $\mathcal{P}^{\downarrow}(M)$ is a submonoid of $\mathcal{P}_0^{\downarrow}(M)$. Secondly, as shown in [10, Proposition 5.1, p. 452], $\mathcal{P}_0^{\downarrow}(M)$ is isomorphic to a quotient monoid of $\mathcal{P}^{\downarrow}(M) \times U_1^{\downarrow}$.

The monoids $\mathcal{P}^{\downarrow}(M)$ and $\mathcal{P}_0^{\downarrow}(M)$ are naturally ordered by inclusion, denoted by $\leqslant$. Note that $X \leqslant Y$ if and only if, for each $x \in X$, there exists $y \in Y$ such that $x \leqslant y$.

Given a variety of ordered monoids $\mathbf{V}$, let $\mathbf{P}^{\downarrow}\mathbf{V}$ $[\mathbf{P}_0^{\downarrow}\mathbf{V}]$ denote the variety of ordered monoids generated by the monoids of the form $\mathcal{P}^{\downarrow}(M)$ $[\mathcal{P}_0^{\downarrow}(M)]$, where $M \in \mathbf{V}$. The operator $\mathbf{P}^{\downarrow}$ was intensively studied in [4]. In particular, it is known that both $\mathbf{P}^{\downarrow}$ and $\mathbf{P}_0^{\downarrow}$ are idempotent operators.

The hereinabove relation between $\mathcal{P}_0^{\downarrow}(M)$ and $\mathcal{P}^{\downarrow}(M)$ can be extended to varieties as follows. Let $\mathbf{Sl}^{\downarrow}$ be the variety of ordered monoids generated by $U_1^{\downarrow}$. It is a well-known fact that $\mathbf{Sl}^{\downarrow} = [\![xy = yx, x = x^2, x \leqslant 1]\!]$. Moreover, the equality

$$\mathbf{P}_0^{\downarrow}\mathbf{V} = \mathbf{P}^{\downarrow}\mathbf{V} \vee \mathbf{Sl}^{\downarrow} \tag{1}$$

holds for any variety of ordered monoids $\mathbf{V}$.

2.6 Free profinite monoid

We refer the reader to [1, 2, 3, 28] for detailed information on profinite completions and we just recall here a few useful facts. Let d be the profinite metric on the free monoid A^*. We let $\widehat{A^*}$ denote the completion of the metric space (A^*, d). The product on A^* is uniformly continuous and hence has a unique continuous extension to $\widehat{A^*}$. It follows that $\widehat{A^*}$ is a compact monoid, called the *free profinite monoid* on A. Furthermore, every stamp $\varphi : A^* \to M$ admits a unique continuous extension $\widehat{\varphi} : \widehat{A^*} \to M$. Similarly, every morphism $f : A^* \to B^*$ admits a unique continuous extension $\widehat{f} : \widehat{A^*} \to \widehat{B^*}$. In the sequel, $\overline{L}$ denotes the closure in $\widehat{A^*}$ of a subset L of A^*.

The length of a word u is denoted by $|u|$. The length map $u \to |u|$ defines a morphism from A^* to the additive semigroup $\mathbb{N}$. If $A = \{a\}$, this morphism is actually an isomorphism, which maps a^n to n. In other words, $(\mathbb{N}, +, 0)$ is the free monoid with a single generator. We let $\widehat{\mathbb{N}}$ denote the profinite completion of $\mathbb{N}$, which is of course isomorphic to $\widehat{a^*}$.

This allows one to define the length $|u|$ of an element u of $\widehat{A^*}$ simply by extending by continuity the length map defined on A^*. The length map is actually a morphism, that is, $|1| = 0$ and $|uv| = |u| + |v|$ for all $u, v \in \widehat{A^*}$.

3 Inequalities and identities

The inequalities [equalities] occurring in this paper are of the form $u \leqslant v$ [$u = v$], where u and v are both in $\widehat{A^*}$ for some alphabet A. In an ordered context, $u = v$ is often viewed as a shortcut for $u \leqslant v$ and $v \leqslant u$.

However, these inequalities are interpreted in several different contexts, which may confuse the reader. Let us clarify matters by giving precise definitions for each case.

3.1 Inequalities

Ordered monoids. Let M be an ordered monoid, let X be an alphabet and let $u, v \in \widehat{X^*}$. Then M *satisfies the inequality* $u \leqslant v$ if, for each morphism $\psi : X^* \to M$, $\widehat{\psi}(u) \leqslant \widehat{\psi}(v)$.

This is the formal definition but in practice, it is easier to think of u and v as terms in which one substitutes each symbol $x \in X$ for an element of M. For instance, M satisfies the inequality $xy^{\omega+1} \leqslant x^\omega y$ if, for all $x, y \in M$, $xy^{\omega+1} \leqslant x^\omega y$.

Varieties of ordered monoids. Let $\mathbf{V}$ be a variety of ordered monoids, let X be an alphabet and let $u, v \in \widehat{X^*}$. Then $\mathbf{V}$ *satisfies an inequality* $u \leqslant v$ if each ordered monoid of $\mathbf{V}$ satisfies the inequality. In this context, equalities of the form $u = v$ are often called *identities*.

It is proved in [20] that any variety of ordered monoids may be defined by a (possibly infinite) set of such inequalities. This result extends to the ordered case the classical result of Reiterman [22] and Banaschewski [5]: any variety of monoids may be defined by a (possibly infinite) set of identities.

The case of lp-varieties and ld-varieties of ordered stamps. Let $\mathbf{V}$ be an lp-variety [ld-variety] of ordered stamps, let X be an alphabet and let $u, v \in \widehat{X^*}$. Then $\mathbf{V}$ *satisfies the inequality* $u \leqslant v$ if, for each stamp $\varphi : A^* \to M$ of $\mathbf{V}$ and for every length-preserving [length-decreasing] morphism $f : X^* \to A^*$, $\widehat{\varphi}(\widehat{f}(u)) \leqslant \widehat{\varphi}(\widehat{f}(v))$.

The difficulty is to interpret correctly $\widehat{f}(u)$. If f is length-preserving, $\widehat{f}(u)$ is obtained by replacing each symbol $x \in X$ by a letter of A. For instance, an lp-variety $\mathbf{V}$ satisfies the inequality $xy^{\omega+1} \leqslant x^\omega y$ if, for each stamp $\varphi : A^* \to M$ of $\mathbf{V}$ and for all letters $a, b \in A$, $\widehat{\varphi}(ab^{\omega+1}) \leqslant \widehat{\varphi}(a^\omega b)$.

It is proved in [15, 19] that any ordered lp-variety of stamps may be defined by a (possibly infinite) set of such inequalities.

If f is length-decreasing, this is even more tricky. Then $\widehat{f}(u)$ is obtained by replacing each symbol $x \in X$ by either a letter of A or by the empty word. For instance, an ld-variety $\mathbf{V}$ satisfies the inequality $xy^{\omega+1} \leqslant x^\omega y$ if, for each stamp $\varphi : A^* \to M$ of $\mathbf{V}$ and for all letters $a, b \in A$, $\widehat{\varphi}(ab^{\omega+1}) \leqslant \widehat{\varphi}(a^\omega b)$, $\widehat{\varphi}(b^{\omega+1}) \leqslant \widehat{\varphi}(b)$ and $\widehat{\varphi}(a) \leqslant \widehat{\varphi}(a^\omega)$.

It is proved in [15, 19] that any ordered ld-variety of stamps may be defined by a (possibly infinite) set of such inequalities.

We will also need the following elementary result. Recall that a variety of [ordered] monoids is *aperiodic* if it satisfies the identity $x^\omega = x^{\omega+1}$.

Proposition 3.1. *Let* $\mathbf{V}$ *be an aperiodic variety of ordered monoids. Then, for each* $\alpha \in \widehat{\mathbb{N}}$, $\mathbf{V}$ *satisfies the identity* $x^\omega = x^\omega x^\alpha$.

Proof. Let $\alpha \in \widehat{\mathbb{N}}$. Then $\alpha = \lim_{n\to\infty} k_n$ for some sequence $(k_n)_{n\geqslant 0}$ of nonegative integers. Since $\mathbf{V}$ is aperiodic, it satisfies the identity $x^{\omega+k_n} = x^\omega$ for all n, and hence it also satisfies the identity $x^\omega x^\alpha = x^\omega$. $\square$

4 Renaming

In this section, we give some general results on renaming.

Since any map may be written as the composition of an injective map with a surjective map, one gets immediately:

Lemma 4.1. *A class of languages is closed under renaming if and only if it is closed under injective and surjective renamings.*

The next two results give a simple description of the positive lp-varieties [ld-varieties] of languages closed under injective renaming:

Proposition 4.1. *The following conditions are equivalent for a positive lp-variety of languages* $\mathcal{V}$*:*

(1) *$\mathcal{V}$ is closed under injective renaming,*

(2) *for each alphabet A and each nonempty set $B \subseteq A$, B^* belongs to $\mathcal{V}(A^*)$,*

(3) *for each alphabet A and each set $B \subseteq A$, B^* belongs to $\mathcal{V}(A^*)$.*

Proof. (1) implies (3). Suppose that $\mathcal{V}$ is closed under injective renaming. Let B be a subset of an alphabet A. Since $B^* \in \mathcal{V}(B^*)$ and since the embedding of B^* into A^* is an injective renaming, one also has $B^* \in \mathcal{V}(A^*)$.

(3) implies (2) is trivial.

(2) implies (3). We have to show that for any alphabet A, $\{1\} \in \mathcal{V}(A^*)$. First assume that A has at least two elements. If $A = B_1 \cup B_2$ is a partition of A into two disjoint nonempty sets B_1 and B_2, then both B_1^* and B_2^* are in $\mathcal{V}(A^*)$, so that $\{1\} = B_1^* \cap B_2^*$ is also in $\mathcal{V}(A^*)$. Now consider a one-letter alphabet a and the two-letter alphabet $\{a, b\}$. The inclusion $h : a^* \to \{a, b\}^*$ is length preserving and thus $\{1\} = h^{-1}(\{1\})$ is in $\mathcal{V}(a^*)$. Finally, the result is trivial if A is empty.

(3) implies (1). Suppose that, for each alphabet A and nonempty set $B \subseteq A$, $B^* \in \mathcal{V}(A^*)$. Let $h : B^* \to A^*$ be an injective renaming. Then there is a renaming $f : A^* \to B^*$ such that $f \circ h$ is the identity function on B^*. Since for any $L \subseteq B^*$, $h(L) = f^{-1}(L) \cap (h(B))^*$, we conclude that $h(L) \in \mathcal{V}(A^*)$ whenever $L \in \mathcal{V}(B^*)$. □

Proposition 4.2. *An ld-variety $\mathcal{V}$ is closed under injective renaming if and only if for each one-letter alphabet a, $\{1\}$ belongs to $\mathcal{V}(a^*)$.*

Proof. Since each *ld*-variety is an *lp*-variety, Proposition 4.1 shows that $\mathcal{V}$ is closed under injective renaming if and only if, for each alphabet A and each subset B of A, B^* belongs to $\mathcal{V}(A^*)$. In particular, if $\mathcal{V}$ is closed under injective renaming, then $\{1\}$ belongs to $\mathcal{V}(a^*)$.

Suppose now that $\mathcal{V}(a^*)$ contains $\{1\}$. Let A be any alphabet and let B be a subset of A. The morphism $h : A^* \to a^*$ that maps each element of B to 1 and all elements of $A \setminus B$ to a is length-decreasing. Since $\mathcal{V}$ is an *ld*-variety and $\{1\}$ belongs to $\mathcal{V}(a^*)$, $h^{-1}(\{1\})$ also belongs to $\mathcal{V}(a^*)$. But $B^* = h^{-1}(\{1\})$, and hence $\mathcal{V}(A^*)$ contains B^* as required. □

Let $\mathbf{V}$ be a variety of ordered monoids and let $\mathcal{V}$ be the corresponding positive variety of languages. A description of the positive variety of languages corresponding to $\mathbf{P}^{\downarrow}\mathbf{V}$ was given by Polák [21, Theorem 4.2] and by Cano and Pin [9] and [10, Proposition 6.3]. The following stronger version[1] was given in [8]. For each alphabet A, let us denote by $\Lambda\mathcal{V}(A^*)$ $[\Lambda'\mathcal{V}(A^*)]$ the set of all languages of A^* of the form $\varphi(K)$, where φ is a [surjective] renaming from B^* to A^*, B is an arbitrary finite alphabet, and K is a language of $\mathcal{V}(B^*)$.

Theorem 4.1. *The class $\Lambda\mathcal{V}$ $[\Lambda'\mathcal{V}]$ is a positive variety of languages and the corresponding variety of ordered monoids is $\mathbf{P}_0^{\downarrow}\mathbf{V}$ $[\mathbf{P}^{\downarrow}\mathbf{V}]$.*

Corollary 4.1. *A positive variety of languages $\mathcal{V}$ is closed under [surjective] renaming if and only if $\mathbf{V} = \mathbf{P}_0^{\downarrow}\mathbf{V}$ $[\mathbf{V} = \mathbf{P}^{\downarrow}\mathbf{V}]$.*

[1]We warn the reader that a different notation was used in [8].

5 Commutative varieties

A stamp $\varphi : A^* \to M$ is said to be *commutative* if M is commutative. An ld-variety is *commutative* if all its stamps are commutative. A stamp $\varphi : A^* \to M$ is called *monogenic* if A is a singleton alphabet.

Proposition 5.1. *Every commutative ld-variety of* [*ordered*] *stamps is generated by its monogenic* [*ordered*] *stamps.*

Proof. We first give the proof in the unordered case. Let $\mathbf{V}$ be a commutative ld-variety of stamps and let $\varphi : A^* \to M$ be a stamp of $\mathbf{V}$. For each $a \in A$, denote by M_a the submonoid of M generated by $\varphi(a)$ and let $\gamma_a : A^* \to M_a$ be the stamp defined by $\gamma_a(a) = \varphi(a)$ and $\gamma_a(c) = 1$ for $c \neq a$. Let $\mathbf{W}$ be the ld-variety of stamps generated by the stamps γ_a, for $a \in A$. We claim that $\mathbf{V} = \mathbf{W}$.

Let $\pi_a : A^* \to A^*$ be the length-decreasing morphism defined by $\pi_a(a) = a$ and $\pi_a(c) = 1$ for $c \neq a$. Denoting by ι_a the natural embedding from M_a into M, one gets the following commutative diagram:

$$\begin{tikzcd}
A^* \arrow[r, "\pi_a"] \arrow[d, "\gamma_a"'] & A^* \arrow[d, "\varphi"] \\
M_a \arrow[r, "\iota_a"] & M
\end{tikzcd}$$

Therefore (π_a, ι_a) is an ld-inclusion and each stamp γ_a belongs to $\mathbf{V}$. Thus $\mathbf{W} \subseteq \mathbf{V}$.

The restricted product γ of the stamps γ_a also belongs to $\mathbf{W}$. Note that γ is a surjective morphism from A^* onto $\prod_{a\in A} M_a$. Moreover, the function $\alpha : \prod_{a\in A} M_a \to M$ which maps each family $(m_a)_{a\in A}$ onto the product $\prod_{a\in A} m_a$ is a surjective morphism. Since $\alpha \circ \gamma = \varphi$, the stamp φ belongs to $\mathbf{W}$. Thus $\mathbf{V} \subseteq \mathbf{W}$. This proves the claim and the proposition.

In the ordered case, each M_a is an ordered submonoid of M and thus each γ_a is an ordered stamp. Since ι_a clearly preserves the order, the same argument shows that each γ_a is in $\mathbf{V}$ and thus $\mathbf{W} \subseteq \mathbf{V}$. For the reverse inclusion, one basically needs to observe that $\prod_{a\in A} M_a$ is equipped with the product order, and that the map α preserves the order, since M is an ordered monoid. □

A similar but simpler proof would give the following result:

Proposition 5.2. *Every commutative variety of* [*ordered*] *monoids is generated by its monogenic* [*ordered*] *monoids.*

Proposition 5.1 has an interesting consequence in terms of languages. Equivalently, a language is *commutative* if its syntactic monoid is commutative.

Corollary 5.1. *Let $\mathcal{V}_1$ and $\mathcal{V}_2$ be two positive ld-varieties of commutative languages. Then $\mathcal{V}_1 \subseteq \mathcal{V}_2$ if and only if $\mathcal{V}_1(a^*) \subseteq \mathcal{V}_2(a^*)$.*

Corollary 5.1 shows that a positive commutative *ld*-variety of languages is entirely determined by its languages on a one-letter alphabet. Here is a more explicit version of this result.

Proposition 5.3. *Let $\mathcal{V}$ be a commutative positive ld-variety of languages. Then for each alphabet $A = \{a_1, \ldots, a_k\}$, $\mathcal{V}(A^*)$ consists of all finite unions of languages of the form $L_1 ⧢ \cdots ⧢ L_k$ where, for $1 \leqslant i \leqslant k$, $L_i \in \mathcal{V}(a_i^*)$.*

Proof. Let $A = \{a_1, \ldots, a_k\}$ be an alphabet. Let $\mathcal{W}(A^*)$ consist of all finite unions of languages of the form $L_1 ⧢ \cdots ⧢ L_k$ where, for $1 \leqslant i \leqslant k$, $L_i \in \mathcal{V}(a_i^*)$. Let us first prove a lemma.

Lemma 5.1. *The class $\mathcal{W}$ is a commutative positive ld-variety of languages.*

Proof. By construction, every language of $\mathcal{W}$ is commutative. Furthermore, $\mathcal{W}(A^*)$ is closed under union. To prove that $\mathcal{W}(A^*)$ is closed under intersection, it suffices to show that the intersection of any two languages $L = L_1 ⧢ \cdots ⧢ L_k$ and $L' = L'_1 ⧢ \cdots ⧢ L'_k$ with $L_i, L'_i \in \mathcal{V}(a_i^*)$ is in $\mathcal{W}(A^*)$. We claim that

$$L \cap L' = (L_1 \cap L'_1) ⧢ \cdots ⧢ (L_k \cap L'_k) \tag{2}$$

Let R be the right hand side of (2). The inclusion $R \subseteq L \cap L'$ is clear. Moreover, if $u \in L \cap L'$, then $u \in (a_1^{n_1} ⧢ \cdots ⧢ a_k^{n_k}) \cap (a_1^{n'_1} ⧢ \cdots ⧢ a_k^{n'_k})$, with $a_i^{n_i} \in L_i$ and $a_i^{n'_i} \in L'_i$ for $1 \leqslant i \leqslant k$. This forces $n_i = n'_i$ and hence $u \in R$, which proves the claim.

Let us prove that $\mathcal{W}(A^*)$ is closed under quotient by any word u. Setting $n_i = |u|_{a_i}$ for $1 \leqslant i \leqslant k$, it suffices to observe that

$$u^{-1}(L_1 ⧢ \cdots ⧢ L_k) = (a_1^{n_1})^{-1}L_1 ⧢ \cdots ⧢ (a_k^{n_k})^{-1}L_k$$

Finally, let $\alpha : B^* \to A^*$ be a length-decreasing morphism. It is proved in [6, Proposition 1.1] that

$$\alpha^{-1}(L_1 ⧢ \cdots ⧢ L_k) = \alpha^{-1}(L_1) ⧢ \cdots ⧢ \alpha^{-1}(L_k) \tag{3}$$

It follows that $\mathcal{W}$ is closed under inverses of *ld*-morphisms, which concludes the proof. □

Let us now come back to the proof of Proposition 5.3. Since $\mathcal{W}$ is a commutative positive *ld*-variety by Lemma 5.1, it suffices to prove, by Proposition 5.1, that $\mathcal{V}(a^*) = \mathcal{W}(a^*)$ for each one-letter alphabet a. But this follows from the definition of $\mathcal{W}$. □

Proposition 5.3 has an interesting consequence.

Theorem 5.1. *Every commutative positive ld-variety of languages is a positive variety of languages.*

Proof. Let $\mathcal{V}$ be a commutative positive *ld*-variety of languages and let $\mathcal{W}$ be the positive variety of languages generated by $\mathcal{V}$. We claim that $\mathcal{V} = \mathcal{W}$. Since $\mathcal{V}$ is contained in $\mathcal{W}$, Corollary 5.1 shows that it suffices to prove that $\mathcal{W}(a^*) \subseteq \mathcal{V}(a^*)$ for each one-letter alphabet a. Since inverses of morphisms commute with Boolean operations and quotients, it suffices to prove that if $\varphi : a^* \to A^*$ is a morphism and $L \in \mathcal{V}(A^*)$, then $\varphi^{-1}(L) \in \mathcal{V}(a^*)$.

Let $\varphi(a) = a_1 \cdots a_k$, where $a_1, \dots, a_k$ are letters of A. Setting $C = \{c_1, \dots, c_k\}$, where $c_1, \dots, c_k$ are distinct letters, one may write φ as $\alpha \circ \beta$ where $\beta : a^* \to C^*$ is defined by $\beta(a) = c_1 \cdots c_k$ and $\alpha : C^* \to A^*$ is defined by $\alpha(c_i) = a_i$ for $1 \leqslant i \leqslant k$.

$$\begin{tikzcd} a^* \arrow[r, "\beta"] \arrow[rr, bend right, "\varphi"'] & C^* \arrow[r, "\alpha"] & A^* \end{tikzcd}$$

Since α is length-preserving, the language $K = \alpha^{-1}(L)$ belongs to $\mathcal{V}(C^*)$. It follows by Proposition 5.3 that K is a finite union of languages of the form $L_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup L_k$ where, for $1 \leqslant i \leqslant k$, $L_i \in \mathcal{V}(c_i^*)$. Let, for $1 \leqslant i \leqslant k$, β_i be the unique length preserving morphism from a^* to c_i^*, defined by $\beta_i(a^r) = c_i^r$. We claim that

$$\beta^{-1}(L_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup L_k) = \beta_1^{-1}(L_1) \cap \cdots \cap \beta_k^{-1}(L_k) \tag{4}$$

Let R be the right hand side of (4). If $a^r \in R$, then $\beta_i(a^r) \in L_i$. Therefore $c_i^r \in L_i$ and since $\beta(a^r) = (c_1 \cdots c_k)^r$, $\beta(a^r) \in L_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup L_k$. Thus R is a subset of $\beta^{-1}(L_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup L_k)$.

If now $a^r \in \beta^{-1}(L_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup L_k)$, then $\beta(a^r) \in c_1^{n_1} \sqcup\!\sqcup \cdots \sqcup\!\sqcup c_k^{n_k}$ with $c^{n_i} \in L_i$ for $1 \leqslant i \leqslant k$. But since $\beta(a^r) = (c_1 \cdots c_k)^r$, one has $n_1 = \cdots = n_k = r$ and hence $c_i^r \in L_i$. Therefore $a^r \in \beta_i^{-1}(L_i)$ for all i and thus a^r belongs R. This proves (4).

Since $L_i \in \mathcal{V}(c_i^*)$ and β_i is length-preserving, $\beta_i^{-1}(L_i) \in \mathcal{V}(a^*)$. As K is a finite union of languages of the form $L_1 \sqcup\!\sqcup \cdots \sqcup\!\sqcup L_k$, Formula (4) shows that $\beta^{-1}(K) \in \mathcal{V}(a^*)$. Finally, since $\varphi = \alpha \circ \beta$, one gets $\varphi^{-1}(L) = \beta^{-1}(\alpha^{-1}(L)) = \beta^{-1}(K)$. Therefore $\varphi^{-1}(L) \in \mathcal{V}(a^*)$, which concludes the proof. $\square$

Theorem 5.1 has a curious interpretation on the set of natural numbers, mentioned in [11]. Setting, for each subset L of $\mathbb{N}$ and each positive integer k,

$$L - 1 = \{n \in \mathbb{N} \mid n + 1 \in L\}$$
$$L \div k = \{n \in \mathbb{N} \mid kn \in L\}$$

one gets the following result:

Proposition 5.4. *Let $\mathcal{L}$ be a lattice of finite subsets*[2] *of $\mathbb{N}$ such that if $L \in \mathcal{L}$, then $L - 1 \in \mathcal{L}$. Then for each positive integer k, $L \in \mathcal{L}$ implies $L \div k \in \mathcal{L}$.*

[2] It also works for a lattice of regular subsets of $\mathbb{N}$.

6 Operations on commutative languages

In this section, we compare the expressive power of three operations on commutative languages: product, shuffle and renaming.

6.1 Shuffle

Let us say that a positive variety of languages $\mathcal{V}$ is *closed under product over one-letter alphabets* if, for each one-letter alphabet a, $\mathcal{V}(a^*)$ is closed under product. Commutative positive varieties closed under shuffle may be described in various ways.

Proposition 6.1. *Let $\mathcal{V}$ be a commutative positive variety of languages and let $\mathbf{V}$ be the corresponding variety of ordered monoids. The following conditions are equivalent:*

(1) *$\mathcal{V}$ is closed under surjective renaming,*

(2) *$\mathcal{V}$ is closed under shuffle product,*

(3) *$\mathcal{V}$ is closed under product over one-letter alphabets,*

(4) $\mathbf{V} = \mathbf{P}^{\downarrow}\mathbf{V}$.

Proof. (1) implies (2). Let $B = A \times \{0,1\}$ and let π_0, π_1 and π be the three morphisms from B^* to A^* defined for all $a \in A$ by

$$\begin{array}{lll} \pi_0(a,0) = a & \pi_1(a,0) = 1 & \pi(a,0) = a \\ \pi_0(a,1) = 1 & \pi_1(a,1) = a & \pi(a,1) = a \end{array}$$

Let L_0 and L_1 be two languages of A^*. Since π is a surjective renaming, the formula $L_0 \sqcup\!\sqcup L_1 = \pi(\pi_0^{-1}(L_0) \cap \pi_1^{-1}(L_1))$ shows that every positive variety closed under surjective renaming is closed under shuffle product.

(2) implies (3) is trivial since on a one-letter alphabet, shuffle product and product are the same.

(3) implies (1). Let $\pi : A^* \to B^*$ be a surjective renaming. For each $b \in B$, let $\gamma_b : b^* \to a^*$ be the renaming which maps b onto a. Let L be a language of $\mathcal{V}(A^*)$. By Proposition 5.3, L is a finite union of languages of the form $\sqcup\!\sqcup_{a\in A} L_a$ where $L_a \in \mathcal{V}(a^*)$ for each $a \in A$. For each $b \in B$, let

$$K_b = \prod_{a\in\pi^{-1}(b)} \gamma_b^{-1}(L_a)$$

If $\mathcal{V}(a^*)$ is closed under product for each one-letter alphabet a, then K_b belongs to $\mathcal{V}(b^*)$. Finally, the formula $\pi(L) = \sqcup\!\sqcup_{b\in B} K_b$ shows that $\pi(L)$ belongs to $\mathcal{V}(B^*)$. Therefore $\mathcal{V}$ is closed under surjective renaming.

Finally, the equivalence of (1) and (4) follows from Corollary 4.1. □

6.2 Renaming

Let us say that a positive variety of languages contains $\{1\}$ if, for every alphabet A, $\mathcal{V}(A^*)$ contains the language $\{1\}$. The following result is a slight variation on Proposition 6.1.

Proposition 6.2. *Let $\mathcal{V}$ be a commutative positive variety of languages and let $\mathbf{V}$ be the corresponding variety of ordered monoids. The following conditions are equivalent:*

(1) *$\mathcal{V}$ is closed under renaming,*
(2) *$\mathcal{V}$ is closed under surjective renaming and contains $\{1\}$,*
(3) *$\mathcal{V}$ is closed under shuffle product and contains $\{1\}$,*
(4) *$\mathcal{V}$ is closed under product over one-letter alphabets and contains $\{1\}$,*
(5) $\mathbf{V} = \mathbf{P}_0^{\downarrow}\mathbf{V}$.

Proof. The equivalence of (2)—(4) follows directly from Proposition 6.1. If (2) holds, then $\mathcal{V}$ is closed under injective renaming by Proposition 4.2 and hence is closed under renaming by Lemma 4.1. Thus (2) implies (1).

To show that (1) implies (2), it suffices to show that if $\mathcal{V}$ is closed under renaming then it contains $\{1\}$. Let $A = \{a, b\}$ and let $\pi : A^* \to A^*$ be the renaming defined by $\pi(a) = \pi(b) = a$. Since $A^* \in \mathcal{V}(A^*)$ and $\pi(A^*) = a^*$, one has $a^* \in \mathcal{V}(A^*)$. A similar argument would show that $b^* \in \mathcal{V}(A^*)$ and thus the language $\{1\}$, which is the intersection of a^* and b^* also belongs to $\mathcal{V}(A^*)$. Consider now an alphabet B and the morphism α from B^* to A^* defined by $\alpha(c) = a$ for each $c \in B$. Then $\alpha^{-1}(\{1\}) = \{1\}$ and thus $\mathcal{V}$ contains $\{1\}$.

Finally, the equivalence of (1) and (5) follows from Corollary 4.1. □

7 Three examples

In this section, we study the positive varieties of languages generated by the languages of Examples 2.1, 2.2 and 2.3.

7.1 The language $1 + a$

Let L be the language $1 + a$, let M be its ordered syntactic monoid and let $\mathcal{V}$ be the smallest commutative positive variety such that $\mathcal{V}(a^*)$ contains L. Let $\mathbf{V}$ be the variety of finite ordered monoids corresponding to $\mathcal{V}$.

Since a positive variety of languages is closed under quotients, $\mathcal{V}(a^*)$ contains the language $a^{-1}L = 1$. It follows that $\mathcal{V}(a^*)$ contains 4 languages: $\emptyset$, 1, $1 + a$ and a^*. We claim that

$$\mathbf{V} = [\![\, xy = yx,\ x \leqslant 1 \text{ and } x^2 \leqslant x^3 \,]\!].$$

First, the two inequalities $x \leqslant 1$ and $x^2 \leqslant x^3$ hold in M. Furthermore, the inequality $x \leqslant 1$ implies the inequalities of the form $x^p \leqslant x^q$ with $p > q$ and the inequality $x^2 \leqslant x^3$ implies all the inequalities of the form $x^p \leqslant x^q$ with $2 \leqslant p < q$.

The only other nontrivial inequalities that $\mathbf{V}$ could possibly satisfy are $1 \leqslant x^q$ for $q > 0$ or $x \leqslant x^q$ for $q > 1$. However, M does not satisfy any of these inequalities.

Let $\mathcal{V}'$ be the closure of $\mathcal{V}$ under shuffle, or equivalently, under product over one-letter alphabets. Then $\mathcal{V}'(a^*)$ contains the empty language, the language a^* and all languages of the form $(1+a)^n$ with $n \geqslant 0$. By Theorem 4.1 and Proposition 6.1, $\mathcal{V}'$ corresponds to the variety of ordered monoids $\mathbf{P}^{\downarrow}\mathbf{V}$. We claim that

$$\mathbf{P}^{\downarrow}\mathbf{V} = [\![\, xy = yx \text{ and } x \leqslant 1 \,]\!].$$

Indeed, the ordered syntactic monoids of the languages of $\mathcal{V}'(a^*)$ all satisfy $xy = yx$ and $x \leqslant 1$. Conversely, if the ordered syntactic monoid of a language K of a^* satisfies $x \leqslant 1$, then $x^n \leqslant_K 1$ for every $n \geqslant 0$, and K is closed under taking subwords. If K is infinite, this forces $K = a^*$. If K is finite, it is necessarily of the form $(1+a)^n$ with $n \geqslant 0$. In both cases, K belongs to $\mathcal{V}'(a^*)$.

Finally, let $\mathbf{W}$ be the variety of ordered monoids corresponding to the closure of $\mathcal{V}$ under renaming. Since $U_1^{\downarrow} \in \mathbf{P}^{\downarrow}\mathbf{V}$, Theorem 4.1 and Formula (1) show that

$$\mathbf{W} = \mathbf{P}_0^{\downarrow}\mathbf{V} = \mathbf{P}^{\downarrow}\mathbf{V} \vee \mathbf{Sl}^{\downarrow} = \mathbf{P}^{\downarrow}\mathbf{V} = [\![\, xy = yx \text{ and } x \leqslant 1 \,]\!].$$

7.2 The language $a + a^6a^*$

Let L be the language $a + a^6a^*$, let M be its ordered syntactic monoid and let $\mathcal{V}$ be the smallest commutative positive variety such that $\mathcal{V}(a^*)$ contains L. Let $\mathbf{V}$ be the variety of finite ordered monoids corresponding to $\mathcal{V}$.

Since a positive variety of languages is closed under quotients, $\mathcal{V}(a^*)$ contains the language $a^{-1}L = 1 + a^5a^*$ and the language $L \cap a^{-1}L = a^6a^*$. It also contains the quotients of this language, which are the languages a^ja^*, for $j \leqslant 6$. Taking the union with L, $a^{-1}L$ or both, one finally concludes that $\mathcal{V}(a^*)$ contains 20 languages: $\emptyset$, a^ia^* for $0 \leqslant i \leqslant 6$, $1 + a^ia^*$ for $1 \leqslant i \leqslant 5$, $a + a^ia^*$ for $3 \leqslant i \leqslant 6$ and $1 + a + a^ia^*$ for $3 \leqslant i \leqslant 5$.

We claim that

$$\mathbf{V} = [\![xy = yx, 1 \leqslant x^5, x^2 \leqslant x^3, x^6 = x^7]\!].$$

Indeed, all defining inequalities hold in M. Since $x^6 = x^7$, the other possible inequalities satisfied by M are equivalent to an inequality of the form $x^p \leqslant x^q$ with $p < q \leqslant 6$. For $p = 0$, the only inequalities of this form satisfied by M are $1 \leqslant x^5$ and $1 \leqslant x^6$, but $1 \leqslant x^6$ is a consequence of $1 \leqslant x^5$ and $x^2 \leqslant x^3$ since $1 \leqslant x^5 = x^3x^2 \leqslant x^3x^3 = x^6$. For $p = 1$, the only inequality of this form satisfied by M is $x \leqslant x^6$, which is a consequence of $1 \leqslant x^5$. Finally, the inequality $x^2 \leqslant x^3$ implies $x^p \leqslant x^q$ for $2 \leqslant p < q \leqslant 6$.

Let $\mathcal{V}'$ be the closure of $\mathcal{V}$ under shuffle, or equivalently, under product over one-letter alphabets. We claim that $\mathcal{V}'(a^*)$ consists of the empty set and the languages of the form

$$a^n(F + a^5a^*) \tag{5}$$

where $n \geqslant 0$ and F is a subset of $(1+a)^4$. First of all, the languages of the form (5) and the empty set form a lattice closed under product, since if $0 \leqslant n \leqslant m$ and F and G are subsets of $(1+a)^4$, then

$$\begin{aligned}
a^n(F+a^5a^*) + a^m(G+a^5a^*) &= a^n(F+a^{m-n}G+a^5a^*)\\
a^n(F+a^5a^*) \cap a^m(G+a^5a^*) &= a^m\Big(\big((a^{m-n})^{-1}(F+a^5a^*)\big) \cap G\big) + a^5a^*\Big)\\
a^n(F+a^5a^*)a^m(G+a^5a^*) &= a^{n+m}(FG+a^5a^*)
\end{aligned}$$

Since $\mathcal{V}'(a^*)$ is closed under finite unions, it just remains to prove that the languages of the form $a^n(a^k+a^5a^*)$, with $n \geqslant 0$ and $0 \leqslant k \leqslant 4$ all belong to $\mathcal{V}'(a^*)$. But since the languages $a+a^6a^*$ and $1+a^{5-k}a^*$ are in $\mathcal{V}(a^*)$, this follows from the formula

$$a^n(a^k+a^5a^*) = \big(a+a^6a^*\big)^{n+k}(1+a^{5-k}a^*)$$

By Theorem 4.1 and Proposition 6.1, $\mathcal{V}'$ corresponds to the variety of ordered monoids $\mathbf{P}^{\downarrow}\mathbf{V}$. We claim that

$$\mathbf{P}^{\downarrow}\mathbf{V} = [\![\, xy = yx \text{ and } 1 \leqslant x^n \text{ for } 5 \leqslant n \leqslant 9 \,]\!].$$

Indeed, the ordered syntactic monoid of any of the languages of the form (5) satisfies all inequalities of the form $1 \leqslant x^n$ for $n \geqslant 5$, but the syntactic ordered monoid of $1+a^2a^*$ does not satisfy any inequality of the form $x^p \leqslant x^q$ with $p > q$. Moreover, the only inequalities that are not an immediate consequence of an inequality of the form $1 \leqslant x^n$ with $5 \leqslant n \leqslant 9$ are the inequalities $x^i \leqslant x^j$ with $0 \leqslant j-i \leqslant 4$. But none of these inequalities are satisfied by the ordered syntactic monoid of $a^i(1+a^5a^*)$.

Finally, Theorem 4.1 and Formula (1) show that the variety of ordered monoids corresponding to the closure of $\mathcal{V}$ under renaming is

$$\begin{aligned}
\mathbf{P}_0^{\downarrow}\mathbf{V} &= \mathbf{P}^{\downarrow}\mathbf{V} \vee \mathbf{Sl}^{\downarrow}\\
&= [\![\, xy = yx \text{ and } 1 \leqslant x^n \text{ for } 5 \leqslant n \leqslant 9 \,]\!] \vee [\![xy = yx, x^2 = x, x \leqslant 1 \,]\!].
\end{aligned}$$

We claim that $\mathbf{P}_0^{\downarrow}\mathbf{V} = \mathbf{W}$, where

$$\mathbf{W} = [\![\, xy = yx \text{ and } x \leqslant x^n \text{ for } 6 \leqslant n \leqslant 10 \,]\!].$$

First, the inequality $x \leqslant x^n$ is a consequence both of the inequality $1 \leqslant x^{n-1}$ and of the equation $x = x^2$. It follows that $\mathbf{P}_0^{\downarrow}\mathbf{V} \subseteq \mathbf{W}$. To establish the opposite inclusion, it suffices to establish the claim that any inequality of the form $x^p \leqslant x^q$ satisfied by both $\mathbf{P}^{\downarrow}\mathbf{V}$ and $\mathbf{Sl}^{\downarrow}$ is also satisfied by $\mathbf{W}$. If $p = 0$, then the inequality becomes $1 \leqslant x^q$ and it is not satisfied by $\mathbf{Sl}^{\downarrow}$ since $1 \not\leqslant 0$ in $U_1^{\downarrow}$. Moreover, for $p > 0$, the only inequalities of the form $x^p \leqslant x^q$ that are not an immediate consequence of an inequality of the form $x \leqslant x^n$ with $6 \leqslant n \leqslant 10$ are the inequalities $x^p \leqslant x^q$ with $0 \leqslant q-p \leqslant 4$. But we already observed that the ordered syntactic monoid of $a^p(1+a^5a^*)$ belongs to $\mathbf{P}^{\downarrow}\mathbf{V}$ but does not satisfy any of these inequalities, which proves the claim.

7.3 The language $a + (a^3 + a^4)(a^7)^*$

Let L be the language $a + (a^3 + a^4)(a^7)^*$, let M be its ordered syntactic monoid and let $\mathcal{V}$ be smallest commutative positive variety such that $\mathcal{V}(a^*)$ contains L. Let $\mathbf{V}$ be the variety of finite ordered monoids corresponding to $\mathcal{V}$. One has

$$\begin{aligned}
(a)^{-1}L &= 1 + (a^2 + a^3)(a^7)^* & (a^2)^{-1}L &= (a + a^2)(a^7)^* \\
(a^3)^{-1}L &= (1 + a)(a^7)^* & (a^4)^{-1}L &= (1 + a^6)(a^7)^* \\
(a^5)^{-1}L &= (a^5 + a^6)(a^7)^* & (a^6)^{-1}L &= (a^4 + a^5)(a^7)^* \\
(a^7)^{-1}L &= (a^3 + a^4)(a^7)^* & (a^8)^{-1}L &= (a^2 + a^3)(a^7)^*
\end{aligned}$$

The set of final states of the minimal automaton of L is $\{1, 3, 4\}$. The quotients of L are recognised by the same automaton by taking a different set of final states as indicated below

$$\begin{aligned}
(a)^{-1}L &\to \{0, 2, 3\} & (a^2)^{-1}L &\to \{1, 2, 8\} \\
(a^3)^{-1}L &\to \{0, 1, 7, 8\} & (a^4)^{-1}L &\to \{0, 6, 7\} \\
(a^5)^{-1}L &\to \{5, 6\} & (a^6)^{-1}L &\to \{4, 5\} \\
(a^7)^{-1}L &\to \{3, 4\} & (a^8)^{-1}L &\to \{2, 3\}
\end{aligned}$$

Observing that

$$\begin{aligned}
\{0\} &= \{0, 2, 3\} \cap \{0, 6, 7\} & \{1\} &= \{1, 3, 4\} \cap \{1, 2, 8\} \\
\{2\} &= \{0, 2, 3\} \cap \{1, 2, 8\} & \{3\} &= \{1, 3, 4\} \cap \{0, 2, 3\} \\
\{4\} &= \{3, 4\} \cap \{4, 5\} & \{5\} &= \{4, 5\} \cap \{5, 6\} \\
\{6\} &= \{5, 6\} \cap \{0, 6, 7\} & \{0, 7\} &= \{0, 6, 7\} \cap \{0, 1, 7, 8\} \\
\{1, 8\} &= \{1, 2, 8\} \cap \{0, 1, 7, 8\}
\end{aligned}$$

it follows that a language belongs to the lattice of languages generated by the quotients of L if and only if it is accepted by the minimal automaton of L equipped with a set F of final states satisfying the two conditions

$$7 \in F \implies 0 \in F \quad \text{and} \quad 8 \in F \implies 1 \in F \tag{6}$$

Now, the complement of a set F satisfying (6) also satisfies (6). It follows that the lattice of languages generated by the quotients of L is actually a Boolean algebra and consequently, $\mathcal{V}$ is a variety of languages. It also follows that

$$\mathbf{V} = [\![xy = yx, x^2 = x^9]\!].$$

Moreover, since $U_1 = \{0, 1\}$ belongs to $\mathbf{V}$, it follows that $\mathbf{PV} = \mathbf{P}_0\mathbf{V}$. By [16, Théorème 2.14], $\mathbf{PV}$ is the variety of all commutative monoids whose groups satisfy the identity $x^7 = 1$. Therefore

$$\mathbf{PV} = [\![xy = yx, x^\omega = x^{\omega+7}]\!].$$

The closure of $\mathcal{V}$ under shuffle, or equivalently, under product over one-letter alphabets, and the closure of $\mathcal{V}$ under renaming both correspond to the variety of monoids $\mathbf{PV}$.

8 Conclusion

We gave an algebraic characterization of the commutative positive varieties of languages closed under shuffle product, renaming or product over one-letter alphabets, but several questions might be worth a further study.

First, each commutative variety of ordered monoids can be described by the equality $xy = yx$ and by a set of inequalities in one variable, like $x^p \leqslant x^q$ or more generally $x^\alpha \leqslant x^\beta$ with $\alpha, \beta \in \widehat{\mathbb{N}}$. It would then be interesting to compare these varieties. We just mention a few results of this flavour, which may help in finding bases of inequalities for commutative positive varieties of languages.

Proposition 8.1. *The variety $[\![xy = yx, x \leqslant x^{n+1}]\!]$ is contained in the variety $[\![xy = yx, x \leqslant x^{m+1}]\!]$ if and only if n divides m.*

Proof. Suppose that n divides m, that is, $m = kn$ for some $k \geqslant 0$. If $x \leqslant x^{n+1}$, then $x \leqslant xx^n$ and by induction, $x \leqslant xx^{kn} = xx^m = x^{m+1}$. Thus $[\![xy = yx, x \leqslant x^{n+1}]\!]$ is contained in the variety $[\![xy = yx, x \leqslant x^{m+1}]\!]$.

Suppose now that $[\![xy = yx, x \leqslant x^{n+1}]\!]$ is contained in the variety $[\![xy = yx, x \leqslant x^{m+1}]\!]$. Then the ordered syntactic monoid of $a(a^n)^*$ satisfies the inequality $x \leqslant x^{n+1}$ and thus it also satisfies the inequality $x \leqslant x^{m+1}$. Since $a \in a(a^n)^*$, this means in particular that $a^m \in a(a^n)^*$ and thus that n divides m. $\square$

In fact, a more general result holds. For each set of natural numbers S, let

$$\mathbf{V}_S = [\![\, xy = yx, x \leqslant x^{n+1} \text{ for all } n \in S \,]\!].$$

Let $\langle S \rangle$ denote the additive submonoid of $\mathbb{N}$ generated by S. It is a well-known fact that any additive subsemigroup of $\mathbb{N}$ is finitely generated and consequently, there exists a finite set of natural numbers F_S such that $\langle S \rangle = \langle F_S \rangle$.

Proposition 8.2. *The variety $\mathbf{V}_S$ satisfies the inequality $x \leqslant x^{m+1}$ if and only if m belongs to $\langle S \rangle$.*

Proof. Let T be the set of all natural numbers n such that $\mathbf{V}_S$ satisfies the inequality $x \leqslant x^{n+1}$. First observe that T is an additive submonoid of $\mathbb{N}$. Indeed, if $\mathbf{V}_S$ satisfies the inequalities $x \leqslant xx^m$ and $x \leqslant xx^n$, then it satisfies $x \leqslant xx^m \leqslant (xx^n)x^m = x^{n+m+1}$. Now T contains S by definition and thus also $\langle S \rangle$. It follows that if m belongs to $\langle S \rangle$, then $\mathbf{V}_S$ satisfies the inequality $x \leqslant x^{m+1}$.

Suppose now that $\mathbf{V}_S$ satisfies the inequality $x \leqslant x^{m+1}$ and let

$$L_S = \{a^{n+1} \mid n \in \langle S \rangle\}.$$

Since $\langle S \rangle = \langle F_S \rangle$, one has

$$L_S = a\{a^s \mid s \in F_S\}^*$$

and thus L_S is a regular language.

We claim that the ordered syntactic monoid M of L_S satisfies an inequality of the form $x \leqslant x^{n+1}$ if and only if $n \in \langle S \rangle$. Suppose first that M satisfies $x \leqslant x^{n+1}$. Then the property $a \in L_S$ implies $a^{n+1} \in L_S$ and hence $n \in \langle S \rangle$.

Conversely, let $n \in \langle S \rangle$. We need to prove that M satisfies the inequality $x \leqslant x^{n+1}$, or equivalently, that $a^k \leqslant_{L_S} (a^k)^{n+1}$ for all $k \geqslant 0$. But for each $r \geqslant 0$, the condition $a^r a^k \in L_S$ implies $r+k-1 \in \langle S \rangle$. Since $r+k(n+1)-1 = r+k-1+kn$, one gets $r+k(n+1)-1 \in \langle S \rangle$ and hence $a^r(a^k)^{n+1} \in L_S$ as required. This concludes the proof of the claim.

In particular, since M satisfies all the inequalities $x \leqslant x^{n+1}$ for $n \in S$, M belongs to $\mathbf{V}_S$ and thus also satisfies the inequality $x \leqslant x^{m+1}$, which finally implies that m belongs to $\langle S \rangle$. □

Corollary 8.1. *Let S and T be two sets of natural numbers. Then $V_S = V_T$ if and only if $\langle S \rangle = \langle T \rangle$.*

It would also be interesting to have a systematic approach to treat examples similar to those given in Section 7. That is, find an algorithm which takes as input a monogenic ordered monoid M and outputs a set of inequalities defining respectively $\mathbf{V}$, $\mathbf{P}^{\downarrow}\mathbf{V}$ and $\mathbf{P}_0^{\downarrow}\mathbf{V}$, where $\mathbf{V}$ is the variety of ordered monoids generated by M.

Acknowledgements

We would like to thank the anonymous referees for their helpful comments.

References

[1] Almeida, Jorge. *Finite semigroups and universal algebra. Series in Algebra*, volume 3. World Scientific, Singapore, 1995.

[2] Almeida, Jorge. Finite semigroups: an introduction to a unified theory of pseudovarieties. In *Semigroups, algorithms, automata and languages (Coimbra, 2001)*, pages 3–64. World Sci. Publ., River Edge, NJ, 2002.

[3] Almeida, Jorge. Profinite semigroups and applications. In Kudryavtsev, V. B. and Rosenberg, I. G., editors, *Structural theory of automata, semigroups and universal algebra*, pages 1–45, New York, 2005. Springer.

[4] Almeida, Jorge, Cano, Antonio, Klíma, Ondrej, and Pin, Jean-Éric. Fixed points of the lower set operator. *Internat. J. Algebra Comput.*, 25(1-2):259–292, 2015.

[5] Banaschewski, Bernhard. The Birkhoff theorem for varieties of finite algebras. *Algebra Universalis*, 17(3):360–368, 1983.

[6] Berstel, Jean, Boasson, Luc, Carton, Olivier, Pin, Jean-Éric, and Restivo, Antonio. The expressive power of the shuffle product. *Information and Computation*, 208:1258–1272, 2010.

[7] Branco, Mário J.J. and Pin, Jean-Éric. Equations for the polynomial closure. In Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., and Thomas, W., editors, *ICALP 2009, Part II*, volume 5556 of *Lect. Notes Comp. Sci.*, pages 115–126, Berlin, 2009. Springer.

[8] Cano, Antonio and Pin, Jean-Éric. Upper set monoids and length preserving morphisms. *J. of Pure and Applied Algebra*, 216:1178–1183, 2012.

[9] Cano Gómez, Antonio. *Semigroupes ordonnés et opérations sur les langages rationnels.* PhD thesis, Université Paris 7 and Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, 2003.

[10] Cano Gómez, Antonio and Pin, Jean-Éric. Shuffle on positive varieties of languages. *Theoret. Comput. Sci.*, 312:433–461, 2004.

[11] Cégielski, Patrick, Grigorieff, Serge, and Guessarian, Irène. On lattices of regular sets of natural integers closed under decrementation. *Inform. Process. Lett.*, 114(4):197–202, 2014.

[12] Eilenberg, S. *Automata, Languages and Machines*, volume B. Academic Press, New York, 1976.

[13] Ésik, Zoltán and Ito, Masami. Temporal logic with cyclic counting and the degree of aperiodicity of finite automata. *Acta Cybernetica*, 16:1–28, 2003.

[14] Ésik, Zoltan and Simon, Imre. Modeling literal morphisms by shuffle. *Semigroup Forum*, 56:225–227, 1998.

[15] Kunc, Michal. Equational description of pseudovarieties of homomorphisms. *Theoret. Informatics Appl.*, 37:243–254, 2003.

[16] Perrot, Jean-François. Variétés de langages et operations. *Theoret. Comput. Sci.*, 7:197–210, 1978.

[17] Pin, Jean-Éric. A variety theorem without complementation. *Russian Mathematics (Iz. VUZ)*, 39:80–90, 1995.

[18] Pin, Jean-Éric. Equational descriptions of languages. *Int. J. Found. Comput. S.*, 23:1227–1240, 2012.

[19] Pin, Jean-Éric and Straubing, Howard. Some results on $\mathcal{C}$-varieties. *Theoret. Informatics Appl.*, 39:239–262, 2005.

[20] Pin, Jean-Éric and Weil, Pascal. A Reiterman theorem for pseudovarieties of finite first-order structures. *Algebra Universalis*, 35:577–595, 1996.

[21] Polák, Libor. Operators on classes of regular languages. In Gomes, Gracinda, Pin, Jean-Éric, and Silva, P.V., editors, *Semigroups, Algorithms, Automata and Languages*, pages 407–422. World Scientific, 2002.

[22] Reiterman, Jan. The Birkhoff theorem for finite algebras. *Algebra Universalis*, 14(1):1–10, 1982.

[23] Reutenauer, Christophe. Sur les variétés de langages et de monoïdes. In *Theoretical computer science (Fourth GI Conf., Aachen)*, volume 67 of *Lect. Notes Comp. Sci.*, pages 260–265. Springer, Berlin, 1979.

[24] Schützenberger, Marcel Paul. Une théorie algébrique du codage. *Séminaire Dubreil. Algèbre et théorie des nombres*, 9:1–24, 1955-1956. `http://eudml.org/doc/111094`.

[25] Straubing, Howard. Aperiodic homomorphisms and the concatenation product of recognizable sets. *J. Pure Appl. Algebra*, 15(3):319–327, 1979.

[26] Straubing, Howard. Recognizable sets and power sets of finite semigroups. *Semigroup Forum*, 18:331–340, 1979.

[27] Straubing, Howard. On logical descriptions of regular languages. In *LATIN 2002*, number 2286 in Lect. Notes Comp. Sci., pages 528–538, Berlin, 2002. Springer.

[28] Weil, Pascal. Profinite methods in semigroup theory. *Int. J. Alg. Comput.*, 12:137–178, 2002.

Acta Cybernetica **23** (2017) 113–139.

Varieties of Graphoids and Birkoff's Theorem for Graphs

Symeon Bozapalidis[a] and Antonios Kalampakas[b]

Abstract

The algebraic structure of graphoids is used in order to obtain the well-known Birkhoff's theorem in the framework of graphs. Namely we establish a natural bijection between the class of Σ-graphoids and the class of strong congruences over $GR(\Sigma, X)$, which is the free graphoid over the doubly ranked alphabet Σ and the set of variables X.

Keywords: hypergraphs, graphoids, graph congruences

1 Introduction

In theoretical computer science structural aspects such as syntax and semantics have been examined by methods of universal algebra. For example, in order to describe the semantics of program schemes, algebras can be used as models in which we suitably interpret all the involved syntactic symbols.

The well known Completeness Theorem (cf. [7]) states that an equation $t = t'$, over a type Γ, is valid on the models of a set of equations $\mathcal{E}$, over Γ, if and only if we can go from t to t' and vise versa through $\mathcal{E}$, this implies that semantics is equivalent to syntax. On the other hand every variety of algebras of type Γ is equationally defined.

Our aim in the present paper is to obtain the above results within the framework of graphs. This will lead to the development of a robust graph rewriting theory analogous with the well established term rewriting theory (cf. [11], [3]). The graphs we consider have edges labeled over a doubly ranked alphabet Σ and are equipped with a designated sequence of begin and end nodes. Moreover they can be combined in two basic ways: by horizontal composition and by disjoint union. The set $GR(\Sigma)$ of all such graphs is a graphoid, that is a magmoid (cf. [1, 2]) satisfying a finite set of standard equations (cf. [6]). Actually it is the free such structure over Σ (cf.

[a]Department of Mathematics, Aristotle University of Thessaloniki, 54124, Thessaloniki, Greece E-mail: bozapali@math.auth.gr

[b]Department of Mathematics and Statistics, College of Engineering, American University of the Middle East, 15453, Egaila, Kuwait E-mail: antonios.kalampakas@aum.edu.kw

DOI: 10.14232/actacyb.23.1.2017.8

[4]). This important result allows us to construct free objects within a variety of graphoids and it is a cornerstone in the present theory.

The main result of this paper is Birkhoff's Theorem for graphs stating that there is a bijection between the class of Σ-G-varieties (i.e., varieties of Σ-designated graphoids) and the class of strong congruences over $GR(\Sigma \cup X)$, where X a countable set of variables. This implies a Completeness Theorem for graphs: a graph equation $G = G'$, of type Σ, is valid on the models of a set of graph equations $\mathcal{E}$, over Σ, iff we can go from G to G' and vise versa through $\mathcal{E}$. Moreover every Σ-G-variety is equationally defined.

The paper is divided into 7 sections. In Section 2 we define the categories of semi-magmoids and magmoids and we construct the free objects within these categories. Moreover, in the second subsection, we see how the set $GR(\Sigma)$ of all graphs over a finite doubly ranked alphabet Σ can be structured into a magmoid. This magmoid is generated by the elements of Σ, viewed as graphs together with a finite set $\bar{D}$ of elementary graphs. Furthermore, we present an important theorem that provides a finite complete set of equations (E), over $\Sigma \cup \bar{D}$, axiomatizing graphs in the sense that two graphs are isomorphic, if and only if, one can be transformed into the other by using these equations.

Graphoids are defined in the first part of Section 3, they are pairs (M, D) consisting of a semi-magmoid M and a set $D \subseteq M$ such that the equations (E) are satisfied inside M. We prove that the set $GR(\Sigma)$ is actually the free graphoid generated by Σ. In addition we introduce Σ-graphoids which are graphoids (M, D) equipped with a function interpreting the elements of Σ by elements of M of the same rank. In the second subsection, by virtue of Σ-graphoids, we introduce a substitution operation on graphs i.e., substitution of graphs inside graphs and more generally substitution of graphoid elements inside graphs. This operation is associative and allows us to explicitly describe the elements of the sub-Σ-graphoid generated by a set A. In the last subsection we see that the cartesian product (resp. directed union) of a family (resp. directed family) of semi-graphoids becomes a semi-graphoid in a natural way.

Congruences on Σ-graphoids are defined in Section 4, the quotient of a Σ-graphoid by a congruence can be organized into a Σ-graphoid and congruences can be characterized by virtue of the substitution operation. The definition of strong congruences is obtained by suitably extending the notion of a congruence. Moreover, given a relation on the set of graphs we construct the smallest congruence (resp. strong congruence) containing this relation.

In the next section we introduce varieties of Σ-graphoids. A variety is class of Σ-graphoids closed under cartesian products, sub-Σ-graphoids and quotients. Any variety of graphoids is closed under directed union; additionally a Σ-graphoid belongs to a variety if and only if every finitely generated sub-Σ-graphoid of it belongs to the variety. In the same section we construct the free Σ-graphoid over a variety generated by a set X, its elements can be viewed as "graphs inside the variety".

In the sixth section we establish the main theorem of our paper. We say that a graph equation (G, G') is satisfied by a Σ-graphoid (or that the Σ-graphoid is a model of this equation) whenever the equation holds for all possible assignments of its variables in that Σ-graphoid. We denote by $Mod(G, G')$ the class of all models of (G, G') and for a set of equations $\mathcal{E}$ we denote by $Mod(\mathcal{E})$ the intersection of all the corresponding classes. This set actually constitutes a Σ-G-variety and we say that a Σ-G-variety is *equationally defined* whenever it can be obtained in this way. In the opposite direction, given a class $\mathcal{K}$ of Σ-graphoids we denote by $Eq(\mathcal{K})$ the set of all equations that are satisfied by all the elements of $\mathcal{K}$. It is proved that every Σ-G-variety $\mathcal{V}$ is equationally defined, i.e., $\mathcal{V} = Mod(Eq(\mathcal{V}))$, moreover the variety generated by a class $\mathcal{K}$ of Σ-graphoids is equal with $Mod(Eq(\mathcal{K}))$. On the other hand, if $\mathcal{R}$ is a strong congruence on graphs then $Eq(Mod(\mathcal{R})) = \mathcal{R}$. By virtue of this result we prove (the graph analog of the well known Completeness Theorem) that if an equation is satisfied by every model of a set of equations $\mathcal{E}$ then we can go from G to G' and vise versa. The Birkhoff's Theorem for graphs follows, namely we construct a bijection between the class of all Σ-G-varieties and the class of all strong graph congruences over Σ. As an interesting application of this result we can associate to every graph language L its syntactic variety $\mathcal{V}_L$, which is the variety corresponding to the syntactic strong congruence defined by L.

The relation between graph and pattern congruences is examined in Section 7. The notions of pattern substitution, pattern congruence (resp. strong pattern congruence) and quotient magmoid are presented and a characterization of pattern congruences (resp. strong pattern congruences) via substitution is given. A bijection between graph congruences (resp. strong graph congruences) and pattern congruences (resp. strong pattern congruences) containing (E) is established. Furthermore we prove that the graph congruence which is generated by the projection on graphs of a pattern relation coincides with the projection on graphs of the pattern congruence that is generated by the same relation. A similar result is also proved for the inverse of this projection.

2 Preliminaries

2.1 Semi-magmoids

Recall that a doubly ranked alphabet is a set A equipped with a function $rank : A \to \mathbb{N} \times \mathbb{N}$ ($\mathbb{N}$ the natural numbers). We write $A = (A_{m,n})$ with

$$A_{m,n} = \{a \mid a \in A,\ rank(a) = (m, n)\},$$

for all $m, n \in \mathbb{N}$.

A *semi-magmoid* is a doubly ranked set $M = (M_{m,n})$ equipped with two operations

$$\circ : M_{m,n} \times M_{n,k} \to M_{m,k}, \qquad m, n, k \geqslant 0$$

$$\Box : M_{m,n} \times M_{m',n'} \to M_{m+m',n+n'}, \qquad m, n, m', n' \geqslant 0$$

which are associative in the obvious way and satisfy the distributivity law

$$(f \circ g) \,\square\, (f' \circ g') = (f \,\square\, f') \circ (g \,\square\, g')$$

whenever all the above operations are defined.

A *magmoid* is a semi-magmoid $M = (M_{m,n})$, equipped with a sequence of constants $e_n \in M_{n,n}$ $(n \geqslant 0)$, called units, such that

$$e_m \circ f = f = f \circ e_n, \;\; e_0 \,\square\, f = f = f \,\square\, e_0$$

for all $f \in M_{m,n}$ and all $m, n \geqslant 0$, and the additional condition

$$e_m \,\square\, e_n = e_{m+n}, \qquad \text{for all } m, n \geqslant 0$$

holds (cf. [1, 2]).

Notice that, due to the last equation, the element e_n $(n \geq 2)$ is uniquely determined by e_1. From now on e_1 will be simply denoted by e.

Subsemi-magmoids and morphisms of semi-magmoids (resp. magmoids) are defined in the obvious way.

Let Σ be a doubly ranked alphabet. *We denote by $SM(\Sigma) = (SM_{m,n}(\Sigma))$ the smallest doubly ranked set satisfying the next items*:

- $\Sigma_{m,n} \subseteq SM_{m,n}(\Sigma)$ for all $m, n \geq 0$,
- if $p \in SM_{m,n}(\Sigma)$ and $q \in SM_{n,k}(\Sigma)$ then their horizontal concatenation $p\,q \in SM_{m,k}(\Sigma)$,
- if $p \in SM_{m,n}(\Sigma)$ and $p' \in SM_{m',n'}(\Sigma)$ then their vertical concatenation $\begin{matrix} p \\ p' \end{matrix} \in SM_{m+m',n+n'}(\Sigma)$.

Let $\sim = (\sim_{m,n})$ be the doubly ranked equivalence on $SM(\Sigma)$, compatible with horizontal and vertical concatenation, generated by the relations

$$\begin{matrix} p_1\, p_1' \\ \\ p_2\, p_2' \end{matrix} \sim \begin{matrix} p_1 & p_1' \\ p_2 & p_2' \end{matrix}$$

for all p_i, p_i' of suitable ranks. The quotient

$$SM(\Sigma)/\sim = (SM_{m,n}(\Sigma)/\sim_{m,n})$$

is denoted $smag(\Sigma)$ and is, by construction, a semi-magmoid. The elements of $smag_{m,n}(\Sigma)$ are called (m,n)*-patterns* or *patterns of rank* (m,n).

Convention. *In order to avoid confusion in the pattern calculus instead of* $\begin{matrix} p \\ p' \end{matrix}$ *we write* $\begin{pmatrix} p \\ p' \end{pmatrix}$. *The associativity law takes the form*

$$\begin{pmatrix} p_1 \\ \begin{pmatrix} p_2 \\ p_3 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \\ p_3 \end{pmatrix}.$$

This common pattern will be denoted

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}.$$

The distributivity law takes the form

$$\begin{pmatrix} p_1\, p_1' \\ p_2\, p_2' \end{pmatrix} = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \begin{pmatrix} p_1' \\ p_2' \end{pmatrix}.$$

Actually $smag(\Sigma)$ is the *free* semi-magmoid generated by Σ as confirms the next result.

Proposition 1. *For every semi-magmoid $M = (M_{m,n})$ and every doubly ranked function $f : \Sigma \to M$, there exists a unique morphism of semi-magmoids $\hat{f} : smag(\Sigma) \to M$ making the following triangle commutative.*

$$\begin{tikzcd} \Sigma \arrow[r, "j"] \arrow[dr, "f"'] & smag(\Sigma) \arrow[d, "\hat{f}"] \\ & M \end{tikzcd} \qquad j(x) = x,\;\; x \in \Sigma$$

Actually, $\hat{f}$ is given by the clauses,

- $\hat{f}(x) = f(x)$, *for all* $x \in \Sigma$,
- $\hat{f}(p\,q) = \hat{f}(p) \circ \hat{f}(q)$, $\hat{f}\begin{pmatrix} p \\ p' \end{pmatrix} = \hat{f}(p) \,\Box\, \hat{f}(p')$,

for all $p, q, p' \in smag(\Sigma)$ of suitable rank.

The construction of the free magmoid follows naturally. Let $(e_n)_{n\geq 0}$ be a sequence of symbols not in Σ and denote by $mag(\Sigma)$ the free semi-magmoid $smag(\Sigma \cup \{e_n \mid n \geq 0\})$ divided by the congruence generated by the relations

$$e_m\, p \approx p \approx p\, e_n, \quad \begin{pmatrix} e_0 \\ p \end{pmatrix} \approx p \approx \begin{pmatrix} p \\ e_0 \end{pmatrix}, \quad \begin{pmatrix} e_m \\ e_n \end{pmatrix} \approx e_{m+n}$$

for all $m, n \geq 0$ and all patterns p of suitable rank. Then $mag(\Sigma)$ clearly constitutes a magmoid which has a universal property analogous to the one stated in Proposition 1, i.e., $mag(\Sigma)$ is the *free* magmoid generated by Σ (cf. [4]).

2.2 Graphs

Now we introduce the magmoid of hypergraphs which will be of constant use throughout this paper. Given a finite alphabet X, we denote by X^* the set of all words over X and for every word $w \in X^*$, $|w|$ denotes its length. Formally, a *concrete* (m,n)*-graph* over a doubly ranked alphabet $\Sigma = (\Sigma_{m,n})$ is a tuple

$$G = (V, E, s, t, l, begin, end)$$

where

- V is the finite set of nodes,
- E is the finite set of hyperedges,
- $s : E \to V^*$ is the source function,
- $t : E \to V^*$ is the target function,
- $l : E \to \Sigma$ is the labelling function such that $rank(l(e)) = (|s(e)|, |t(e)|)$ for every $e \in E$,
- $begin \in V^*$ with $|begin| = m$ is the sequence of begin nodes and
- $end \in V^*$ with $|end| = n$ is the sequence of end nodes.

Notice that according to this definition vertices can be duplicated in the begin and end sequences of the graph and also at the sources and targets of an edge. For an edge e of a hypergraph G we simply write $rank(e)$ to denote $rank(l(e))$.

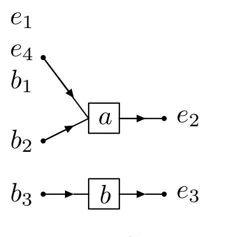
The specific sets V and E chosen to define a concrete graph G are actually irrelevant. We shall not distinguish between two isomorphic graphs. Hence we have the following definition of an abstract graph. Two concrete (m,n)-graphs $G = (V, E, s, t, l, begin, end)$ and $G' = (V', E', s', t', l', begin', end')$ over Σ are isomorphic iff there exist two bijections $h_V : V \to V'$ and $h_E : E \to E'$ commuting with source, target, labelling, *begin* and *end* in the usual way.

An *abstract* (m,n)*-graph* is defined to be the equivalence class of a concrete (m,n)-graph with respect to isomorphism. We denote by $GR_{m,n}(\Sigma)$ the set of all abstract (m,n)-graphs over Σ. Since we shall mainly be interested in abstract graphs we simply call them graphs except when it is necessary to emphasize that they are defined up to an isomorphism. Any graph $G \in GR_{m,n}(\Sigma)$ having no edges is called a *discrete* (m,n)*-graph.*

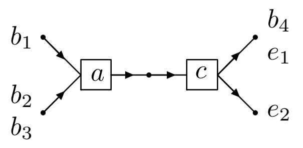
If G is an (m,n)-graph represented by $(V, E, s, t, l, begin, end)$ and H is an (n,k)-graph represented by $(V', E', s', t', l', begin', end')$ then their *product* $G \circ H$ is the (m,k)-graph represented by the concrete graph obtained by taking the disjoint union of G and H and then identifying the ith end node of G with the ith begin node of H, for every $i \in \{1, ..., n\}$; also, $begin(G \circ H) = begin(G)$ and $end(G \circ H) = end(H)$.

The *sum* $G \,\square\, H$ of arbitrary graphs G and H is their disjoint union with their sequences of begin nodes concatenated and similarly for their end nodes.

For instance let $\Sigma = \{a, b, c\}$, with $rank(a) = (2,1)$, $rank(b) = (1,1)$ and $rank(c) = (1,2)$. In the following pictures, edges are represented by boxes, nodes by dots, and the sources and targets of an edge by directed lines that enter and leave the corresponding box, respectively. The order of the sources and targets of an edge is the vertical order of the directed lines as drawn in the pictures. We display two graphs $G \in GR_{3,4}(\Sigma)$ and $H \in GR_{4,2}(\Sigma)$, where the ith begin node is indicated by b_i, and the ith end node by e_i.

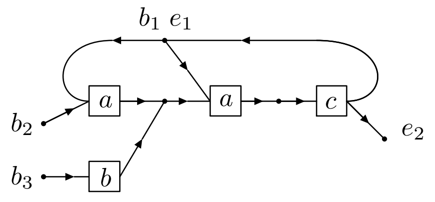


G

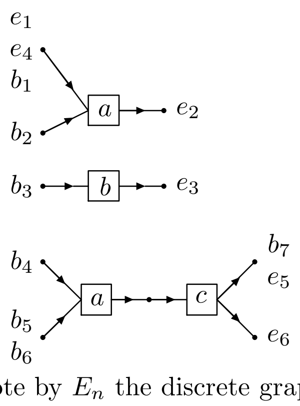


H

Then their product $G \circ H$ is the $(3, 2)$-graph



and, their sum $G \,\square\, H$ is the $(7, 6)$-graph



For every $n \in \mathbb{N}$ we denote by E_n the discrete graph of rank (n, n) with nodes $x_1, ..., x_n$ and $begin(E_n) = end(E_n) = x_1 \cdots x_n$; we write E for E_1. Note that E_0 is the empty graph.

It is straightforward to verify that $GR(\Sigma) = (GR_{m,n}(\Sigma))$ with the operations defined above is a magmoid whose units are the graphs E_n, $n \geq 0$, see Lemma 6 of [10]. The discrete graphs of $GR(\Sigma)$ form obviously a sub-magmoid $DISC$ of $GR(\Sigma)$ and the function sending each graph $G \in GR(\Sigma)$ to its underlying discrete graph is an epimorphism of magmoids

$$disc_\Sigma : GR(\Sigma) \to DISC.$$

Let us denote by $I_{p,q}$ the discrete (p,q)-graph having a single node x and whose begin and end sequences are $x \cdots x$ (p times) and $x \cdots x$ (q times) respectively. Note that $I_{1,1}$ is equal with E. Let also Π be the discrete $(2,2)$-graph having two nodes x and y and whose begin and end sequences are xy and yx, respectively. Finally, for every $\sigma \in \Sigma_{m,n}$, we denote again by σ the (m,n)-graph having only one edge and $m+n$ nodes $x_1, \ldots, x_m, y_1, \ldots, y_n$. The edge is labelled by σ, and the begin (resp. end sequence) of the graph is the sequence of sources (resp. targets) of the edge, viz. $x_1 \cdots x_m$ (resp. $y_1 \cdots y_n$).

As usual $\mathcal{S}_m$ stands for the group of all permutations of the set $\{1, 2, \ldots, m\}$. Given a permutation $\alpha \in \mathcal{S}_m$

$$\alpha = \begin{pmatrix} 1 & 2 & \ldots & m \\ \alpha(1) & \alpha(2) & \ldots & \alpha(m) \end{pmatrix}$$

the discrete graph having $\{x_1, x_2, \ldots, x_m\}$ as set of nodes, $x_{\alpha(1)} x_{\alpha(2)} \cdots x_{\alpha(m)}$ as begin sequence and $x_1 x_2 \cdots x_m$ as end sequence, is denoted Π_α and is called the *permutation graph associated with* α. Observe that Π is the graph associated with the permutation

$$\alpha = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}.$$

We denote by $\Pi_{n,1}$ the graph associated with the permutation

$$\begin{pmatrix} 1 & 2 & \ldots & n+1 \\ 2 & \ldots & n+1 & 1 \end{pmatrix}$$

interchanging the last n numbers with the first one.

For every $\sigma \in \Sigma_{m,n}$, $BF(\sigma)$ is the same as the graph σ, except that

$$begin(BF(\sigma)) = begin(\sigma) end(\sigma) = x_1 \cdots x_m y_1 \cdots y_n$$

and $end(BF(\sigma)) = \varepsilon$ (where ε denotes the empty sequence).

Next important result provides a complete set of equations axiomatizing graphs.

Theorem 1 (cf. [4]). *Let $\Sigma = \{\sigma_1, \ldots, \sigma_r\}$ and assume that in the graph $G \in GR_{m,n}(\Sigma)$ the symbol σ_i occurs λ_i times, ($1 \leq i \leq r$) then G admits a decomposition of the form*

$$\begin{aligned} G = {} & \Pi_\alpha \circ (I_{p_1,q_1} \square \cdots \square I_{p_s,q_s}) \circ \Pi_\beta \\ & \circ (E_n \square BF(\sigma_1^1) \square \cdots \square BF(\sigma_1^{\lambda_1}) \square \cdots \square BF(\sigma_r^1) \square \cdots \square BF(\sigma_r^{\lambda_r})) \end{aligned}$$

where α, β are suitable permutations and $\sigma_i^1 = \cdots = \sigma_i^{\lambda_i} = \sigma_i$, $1 \leq i \leq r$. Moreover, if

$$\begin{aligned} G' = {} & \Pi_{\alpha'} \circ (I_{p'_1,q'_1} \square \cdots \square I_{p'_t,q'_t}) \circ \Pi_{\beta'} \\ & \circ (E_n \square BF(\sigma_1^1) \square \cdots \square BF(\sigma_1^{\lambda'_1}) \square \cdots \square BF(\sigma_r^1) \square \cdots \square BF(\sigma_r^{\lambda'_r})) \end{aligned}$$

then $G = G'$ if and only if we can transform G into G' through the finite set of equations (E):

$$\Pi \circ \Pi = E \,\Box\, E, \quad (E \,\Box\, \Pi) \circ (\Pi \,\Box\, E) \circ (E \,\Box\, \Pi) = (\Pi \,\Box\, E) \circ (E \,\Box\, \Pi) \circ (\Pi \,\Box\, E)\,,$$

$$(E \,\Box\, I_{2,1}) \circ I_{2,1} = (I_{2,1} \,\Box\, E) \circ I_{2,1}, \quad (E \,\Box\, I_{0,1}) \circ I_{2,1} = E, \quad \Pi \circ I_{2,1} = I_{2,1}$$

$$(\Pi \,\Box\, E) \circ (E \,\Box\, \Pi) \circ (I_{2,1} \,\Box\, E) = (E \,\Box\, I_{2,1}) \circ \Pi, \quad (E \,\Box\, I_{0,1}) \circ \Pi = (I_{0,1} \,\Box\, E)\,,$$

$$I_{1,2} \circ (E \,\Box\, I_{1,2}) = I_{1,2} \circ (I_{1,2} \,\Box\, E)\,, \quad I_{1,2} \circ (E \,\Box\, I_{1,0}) = E, \quad I_{1,2} \circ \Pi = I_{1,2},$$

$$I_{1,2} \circ I_{2,1} = E, \quad (I_{1,2} \,\Box\, E) \circ (E \,\Box\, \Pi) \circ (\Pi \,\Box\, E) = \Pi \circ (E \,\Box\, I_{1,2})\,,$$

$$\Pi \circ (E \,\Box\, I_{1,0}) = (I_{1,0} \,\Box\, E)\,, \quad (I_{1,2} \,\Box\, E) \circ (E \,\Box\, I_{2,1}) = I_{2,1} \circ I_{1,2},$$

$$\Pi_{p,1} \circ (\sigma \,\Box\, E) = (E \,\Box\, \sigma) \circ \Pi_{q,1}, \quad \textit{where } \sigma \in \Sigma_{p,q}, \ \ p, q \geq 0.$$

Now let us introduce the alphabet $\bar{D}$, formed by the following five symbols

$$\bar{i}_{21} : 2 \to 1 \quad \bar{i}_{01} : 0 \to 1 \quad \bar{i}_{12} : 1 \to 2 \quad \bar{i}_{10} : 1 \to 0 \quad \bar{\pi} : 2 \to 2$$

where $x : m \to n$ indicates that symbol x has first rank m and second rank n, and denote by $mag(\Sigma \cup \bar{D})$ the free magmoid generated by the doubly ranked alphabet $\Sigma \cup \bar{D}$. We denote by

$$val_\Sigma : mag(\Sigma \cup \bar{D}) \to GR(\Sigma)$$

the unique magmoid morphism extending the function described by the assignments

$$\bar{i}_{21} \mapsto I_{2,1}, \quad \bar{i}_{01} \mapsto I_{0,1}, \quad \bar{i}_{12} \mapsto I_{1,2}, \quad \bar{i}_{10} \mapsto I_{1,0}, \quad \bar{\pi} \mapsto \Pi,$$

$$\sigma \mapsto \sigma, \ \text{ for all } \sigma \in \Sigma, \qquad e_n \mapsto E_n, \ \text{ for all } n \in \mathbb{N}.$$

According to the previous theorem val_Σ is a surjection and $GR(\Sigma)$ is the quotient of $mag(\Sigma \cup \bar{D})$ by the congruence generated by the relations (E):

$$\bar{\pi}\bar{\pi} \sim e_2, \quad \begin{pmatrix} e \\ \bar{\pi} \end{pmatrix} \begin{pmatrix} \bar{\pi} \\ e \end{pmatrix} \begin{pmatrix} e \\ \bar{\pi} \end{pmatrix} \sim \begin{pmatrix} \bar{\pi} \\ e \end{pmatrix} \begin{pmatrix} e \\ \bar{\pi} \end{pmatrix} \begin{pmatrix} \bar{\pi} \\ e \end{pmatrix},$$

$$\begin{pmatrix} e \\ \bar{i}_{21} \end{pmatrix} \bar{i}_{21} \sim \begin{pmatrix} \bar{i}_{21} \\ e \end{pmatrix} \bar{i}_{21}, \quad \begin{pmatrix} e \\ \bar{i}_{01} \end{pmatrix} \bar{i}_{21} \sim e,$$

$$\bar\pi\bar i_{21}\sim\bar i_{21},\quad \begin{pmatrix}e\\ \bar i_{01}\end{pmatrix}\bar\pi\sim\begin{pmatrix}\bar i_{01}\\ e\end{pmatrix},$$

$$\begin{pmatrix}\bar\pi\\ e\end{pmatrix}\begin{pmatrix}e\\ \bar\pi\end{pmatrix}\begin{pmatrix}\bar i_{21}\\ e\end{pmatrix}\sim\begin{pmatrix}e\\ \bar i_{21}\end{pmatrix}\bar\pi,$$

$$\bar i_{12}\begin{pmatrix}e\\ \bar i_{12}\end{pmatrix}\sim\bar i_{12}\begin{pmatrix}\bar i_{12}\\ e\end{pmatrix},\quad \bar i_{12}\begin{pmatrix}e\\ \bar i_{10}\end{pmatrix}\sim e,$$

$$\bar i_{12}\bar\pi\sim\bar i_{12},\quad \bar\pi\begin{pmatrix}e\\ \bar i_{10}\end{pmatrix}\sim\begin{pmatrix}\bar i_{10}\\ e\end{pmatrix},$$

$$\begin{pmatrix}\bar i_{12}\\ e\end{pmatrix}\begin{pmatrix}e\\ \bar\pi\end{pmatrix}\begin{pmatrix}\bar\pi\\ e\end{pmatrix}\sim\bar\pi\begin{pmatrix}e\\ \bar i_{12}\end{pmatrix},$$

$$\bar i_{12}\,\bar i_{21}\sim e,\quad \begin{pmatrix}\bar i_{12}\\ e\end{pmatrix}\begin{pmatrix}e\\ \bar i_{21}\end{pmatrix}\sim\bar i_{21}\,\bar i_{12},$$

$$\bar\pi_{m,1}\begin{pmatrix}\sigma\\ e\end{pmatrix}\sim\begin{pmatrix}e\\ \sigma\end{pmatrix}\bar\pi_{n,1},$$

where $\sigma\in\Sigma_{m,n}$, $m,n\geq 0$ and $\bar\pi_{n,1}$ is the pattern inductively defined by

$$\bar\pi_{1,0}=e,\quad \bar\pi_{n,1}=\begin{pmatrix}\bar\pi_{n-1,1}\\ e\end{pmatrix}\begin{pmatrix}e_{n-1}\\ \bar\pi\end{pmatrix}$$

which will represent the graph $\Pi_{n,1}$.

The next definition will be used later on. We call *size* of a pattern $p\in mag(\Sigma\cup\bar D)$ the number of symbols of $\Sigma\cup\bar D$ occurring in p. The *size* of a graph $G\in GR(\Sigma)$ is then

$$size(G)=min\{size(p)\mid p\in val_\Sigma^{-1}(G)\}.$$

3 Graphoids and their algebra

3.1 Free graphoids

The algebraic structure underneath $GR(\Sigma)$ is that of a graphoid.

More precisely, a graphoid $\mathbf{M}=(M,D)$ consists of a semi-magmoid M and a set

$$D=\{e_0,e,\pi,i_{01},i_{21},i_{10},i_{12}\},$$

where $e_0\in M_{0,0}$, $e\in M_{1,1}$, $\pi\in M_{2,2}$, $i_{01}\in M_{0,1}$, $i_{21}\in M_{2,1}$, $i_{10}\in M_{1,0}$, $i_{12}\in M_{1,2}$ such that (M,e_0,e) is a magmoid, i.e.,

$$(1)\quad e_m\circ f=f=f\circ e_n,\quad e_0\,\square\, f=f=f\,\square\, e_0,$$

where $e_n=e\,\square\cdots\square\, e$ (n-times, $n\geq 0$), $f\in M_{m,n}$, $(m,n\geq 0)$, and additionally the following equations hold

$$(2) \qquad \pi \circ \pi = e_2, \quad (\pi \,\square\, e) \circ (e \,\square\, \pi) \circ (\pi \,\square\, e) = (e \,\square\, \pi) \circ (\pi \,\square\, e) \circ (e \,\square\, \pi)$$

$$\begin{gathered}(3) \quad (e \,\square\, i_{21}) \circ i_{21} = (i_{21} \,\square\, e) \circ i_{21}, \quad (e \,\square\, i_{01}) \circ i_{21} = e, \\ \pi \circ i_{21} = i_{21}, \quad (e \,\square\, i_{01}) \circ \pi = (i_{01} \,\square\, e)\,, \\ (\pi \,\square\, e) \circ (e \,\square\, \pi) \circ (i_{21} \,\square\, e) = (e \,\square\, i_{21}) \circ \pi,\end{gathered}$$

$$\begin{gathered}(4) \quad i_{12} \circ (e \,\square\, i_{12}) = i_{12} \circ (i_{12} \,\square\, e)\,, \quad i_{12} \circ (e \,\square\, i_{10}) = e, \\ i_{12} \circ \pi = i_{12}, \quad \pi \circ (e \,\square\, i_{10}) = (i_{10} \,\square\, e)\,, \\ (i_{12} \,\square\, e) \circ (e \,\square\, \pi) \circ (\pi \,\square\, e) = \pi \circ (e \,\square\, i_{12})\,,\end{gathered}$$

$$(5) \quad i_{12} \circ i_{21} = e, \quad (i_{12} \,\square\, e) \circ (e \,\square\, i_{21}) = i_{21} \circ i_{12}.$$

$$(6) \quad \pi_{m,1} \circ (f \,\square\, e) = (e \,\square\, f) \circ \pi_{n,1}, \qquad \text{for all } f \in M_{m,n},$$

where the element $\pi_{m,1} \in M_{m,1}$ is defined by

$$\pi_{1,0} = e, \quad \pi_{m,1} = (\pi_{m-1,1} \,\square\, e) \circ (e_{m-1} \,\square\, \pi).$$

We point out that equation (6) holds in $GR(\Sigma)$ since it holds for all the letters of the alphabet Σ (cf. [4]).

Observe that $(GR(\Sigma), \{E_0, E, \Pi, I_{0,1}, I_{2,1}, I_{1,0}, I_{1,2}\})$ is a graphoid which from now on will be simply denoted by $GR(\Sigma)$.

Given graphoids (M, D) and (M', D'), a semi-magmoid morphism $H : M \to M'$ preserving D-sets, i.e., $H(e_0) = e'_0$, $H(e) = e'$, $H(\pi) = \pi'$ and $H(i_{\kappa\lambda}) = i'_{\kappa\lambda}$, is called a morphism of graphoids.

We have already discussed how the set $GR(\Sigma)$ can be structured into a graphoid; in fact it is the free graphoid generated by Σ.

Theorem 2. *The doubly ranked function $j : \Sigma \to GR(\Sigma)$, with $j(\sigma) = \sigma$, for all $\sigma \in \Sigma$, has the following universal property: for any graphoid $\mathbb{M} = (M, D)$, $D = \{e_0, e, \pi, i_{10}, i_{12}, i_{01}, i_{21}\}$ and any doubly ranked function $f : \Sigma \to M$, there exists a unique morphism of graphoids $\bar{f} : GR(\Sigma) \to \mathbb{M}$ making commutative the following triangle.*

$$\begin{tikzcd} \Sigma \arrow[r, "j"] \arrow[dr, "f"'] & GR(\Sigma) \arrow[d, "\bar{f}"] \\ & \mathbb{M} \end{tikzcd}$$

The morphism $\bar{f}$ is defined by the clauses

- $\bar{f}(\sigma) = f(\sigma)$, $\sigma \in \Sigma$,
- $\bar{f}(E_0) = e_0$, $\bar{f}(E) = e$, $\bar{f}(\Pi) = \pi$, $\bar{f}(I_{p,q}) = i_{pq}$,
- $\bar{f}(G_1 \circ G_2) = \bar{f}(G_1) \circ \bar{f}(G_2)$,
- $\bar{f}(G_1 \,\square\, G_2) = \bar{f}(G_1) \,\square\, \bar{f}(G_2)$,

for all graphs G_1, G_2 of suitable rank.

Proof. Since $mag(\Sigma \cup \bar{D})$ is the free magmoid over $\Sigma \cup \bar{D}$, f is uniquely extended into a morphism of magmoids $\hat{f} : mag(\Sigma \cup \bar{D}) \to \mathbb{M}$ making commutative the triangle:

$$\begin{tikzcd}
\Sigma \cup \bar{D} \arrow[r, "\mathbf{j}"] \arrow[dr, "\mathbf{f}"'] & mag(\Sigma \cup \bar{D}) \arrow[d, "\hat{f}"] \\
 & \mathbf{M}
\end{tikzcd}$$

where

- $\mathbf{j}(\sigma) = j(\sigma)$, $\sigma \in \Sigma$ and $\mathbf{j}(\alpha) = \alpha$, $\alpha \in \bar{D}$;
- $\mathbf{f}(\sigma) = f(\sigma)$, $\sigma \in \Sigma$ and $\mathbf{f}(\bar{i}_{21}) = i_{21}$, $\mathbf{f}(\bar{i}_{01}) = i_{01}$, $\mathbf{f}(\bar{i}_{12}) = i_{12}$, $\mathbf{f}(\bar{i}_{10}) = i_{10}$, $\mathbf{f}(\bar{\pi}) = \pi$.

Since all the relations (E) are valid in $\mathbf{M}$, the kernel of $\hat{f}$ includes the congruence $=_{(E)}$ generated by (E), and thus $\hat{f}$ induces a unique graphoid morphism

$$\bar{f} : mag(\Sigma \cup \bar{D})/ =_{(E)} \longrightarrow \mathbf{M}$$

rendering commutative the triangle

$$\begin{tikzcd}
mag(\Sigma \cup \bar{D}) \arrow[r, "g"] \arrow[dr, "\hat{f}"'] & mag(\Sigma \cup \bar{D})/ =_{(E)} \arrow[d, "\bar{f}"] \\
 & \mathbf{M}
\end{tikzcd}$$

where g is the canonical projection sending every element of $mag(\Sigma \cup \bar{D})$ to its class with respect to $=_{(E)}$. The result comes by combining the above two diagrams and Theorem 1. $\square$

A *graph homomorphism* $H : GR(\Sigma) \to GR(\Sigma')$ is just a morphism of graphoids. Hence, by virtue of the previous theorem it is completely determined by its values $H(\sigma)$, $\sigma \in \Sigma$. A graph homomorphism $H : GR(\Sigma) \to GR(\Sigma')$ is called a projection whenever $H(\Sigma) \subseteq \Sigma'$.

In the sequel, we mostly deal with Σ-graphoids, (Σ doubly ranked alphabet) which are graphoids (M, D) equipped with a function $\mu : \Sigma \to M$ interpreting the letters $\sigma \in \Sigma$ by elements of M with the same rank.

By virtue of Theorem 2 any graph G of $GR(\Sigma)$ is interpreted, in a Σ-graphoid $\mathbf{M} = (M, D, \mu)$, by the element $\bar{\mu}(G)$. In other words, we are able to make graph theory inside any Σ-graphoid.

The set $GR(\Sigma)$ is a Σ-graphoid where t is the function sending every element $\sigma \in \Sigma$ to the graph it represents.

If $\mathbf{M} = (M, D, \mu)$ and $\mathbf{M}' = (M', D', \mu')$ are two Σ-graphoids then any graphoid morphism $H : (M, D) \to (M', D')$ commuting with the μ-assignments, i.e., rendering commutative the triangle

$$\begin{tikzcd} & \Sigma \arrow[dl, "\mu"'] \arrow[dr, "\mu'"] & \\ M \arrow[rr, "H"'] & & M' \end{tikzcd}$$

is called a Σ-graphoid morphism.

Graphs whose labels of edges are variables will be frequently used. From now on $X_p = \{x_1, \dots, x_p\}$ is a set of doubly ranked variables with $rank(x_i) = (\alpha_i, \beta_i)$, $1 \le i \le p$.

Convention. *The Σ-graphoid $GR(\Sigma \cup X_p)$ will be denoted by $GR(\Sigma, X_p)$.*

Theorem 3. *The Σ-graphoid $GR(\Sigma, X)$ is free over X, i.e., for any Σ-graphoid $\mathbf{M} = (M, D, \mu)$ and any function $f : X \to M$ there is a unique Σ-graphoid morphism $\bar{f} : GR(\Sigma, X) \to \mathbf{M}$ such that the next diagram commutes*

$$\begin{tikzcd} X \arrow[r, "j"] \arrow[dr, "f"'] & GR(\Sigma, X) \arrow[d, "\bar{f}"] \\ & \mathbf{M} \end{tikzcd} \qquad \begin{array}{c} \bar{f}(\sigma) = \mu(\sigma), \\ j(x) = x, \quad x \in X. \end{array}$$

3.2 The substitution operation

The well known edge replacement operation on graphs (cf. [9], [8]) can be defined with the help of Theorem 3 in an elegant way. Let $X = \{x_1, x_2, \dots\}$ with $rank(x_i) = (\alpha_i, \beta_i)$, $i = 1, 2, \dots$ and $X_p = \{x_1, \dots, x_p\}$. For

$$G_i \in GR_{\alpha_i, \beta_i}(\Sigma, X_p).$$

The function

$$f : X_p \to GR(\Sigma, X_k), \qquad f(x_i) = G_i,$$

is then uniquely extended into a Σ-graphoid morphism

$$\bar{f} : GR(\Sigma, X_p) \to GR(\Sigma, X_k).$$

For all $G \in GR_{\alpha,\beta}(\Sigma, X_p)$, the graph $\bar{f}(G)$ is denoted by $G[G_1, \dots, G_p]$ (it is the graph obtained by simultaneously replacing x_i by G_i inside G, $i = 1, \dots, p$).

Proposition 2. *It holds*

$$G[G_1, \dots, G_p][G'_1, \dots, G'_k] = G[G_1[G'_1, \dots, G'_k], \dots, G_p[G'_1, \dots, G'_k]]$$

for all graphs of suitable rank.

Proof. We define the functions

$$f : X_p \to GR(\Sigma, X_k), \quad g : X_k \to GR(\Sigma, X_s)$$

by setting

$$f(x_i) = G_i \ \ (1 \le i \le p), \quad g(x_j) = G'_j \ \ (1 \le j \le k),$$

respectively. For uniqueness reasons invoked by Theorem 3 we get the equality

$$\overline{\bar{g} \circ f} = \bar{g} \circ \bar{f}.$$

Since $(\bar{g} \circ f)(x_i) = \bar{g}(f(x_i)) = \bar{g}(G_i) = G_i[G'_1, \dots, G'_k]$ we get

$$\begin{aligned} G[G_1, \dots, G_p][G'_1, \dots, G'_k] &= \bar{g}(G[G_1, \dots, G_p]) \\ &= \bar{g}(\bar{f}(G)) = (\bar{g} \circ \bar{f})(G) = \overline{\bar{g} \circ f}(G) \\ &= G[G_1[G'_1, \dots, G'_k], \dots, G_p[G'_1, \dots, G'_k]] \end{aligned}$$

as wanted. □

Let $\xi_{m,n}$ be a new symbol with rank (m, n) and denote by $FR^{\alpha,\beta}_{m,n}(\Sigma, X)$ the subset of $GR_{\alpha,\beta}(\Sigma, X \cup \{\xi_{m,n}\})$ with just one occurrence of $\xi_{m,n}$; its elements are called frames with exterior rank (α, β) and interior rank (m, n). The set $FR^{\alpha,\beta}_{m,n}(\Sigma, X)$ acts on $GR_{m,n}(\Sigma, X)$ via substitution at $\xi_{m,n}$: for $F \in FR^{\alpha,\beta}_{m,n}(\Sigma, X)$ and $G \in GR_{m,n}(\Sigma, X)$,

$$F \cdot G = F[G/\xi_{m,n}].$$

The substitution of graphs inside graphs can be extended into a substitution of graphoid elements inside graphs. This will be achieved be means of the universal property described in Theorem 3. Let $a_1, \dots, a_p$ be elements of a Σ-graphoid $\mathbf{M} = (M, D, \mu)$

$$a_i \in M_{\alpha_i,\beta_i}, \quad i = 1, \dots, p$$

and consider the Σ-graphoid morphism $\bar{h} : GR(\Sigma, X_p) \to \mathbf{M}$ determined by the assignments

$$h(x_1) = a_1, \dots, h(x_p) = a_p.$$

For any graph $G \in GR(\Sigma, X_p)$ we denote by $G[a_1, \dots, a_p]$ the element $\bar{h}(G)$. Notice that from Theorem 3 it holds $\bar{h}(\sigma) = \mu(\sigma)$. The so defined mixed substitution operation has the nice properties of graph substitution. More precisely,

Proposition 3 (Associativity law for mixed substitution)**.** *It holds*

$$G[G_1,\dots,G_p][a_1,\dots,a_k] = G[G_1[a_1,\dots,a_k],\dots,G_p[a_1,\dots,a_k]],$$

where $G \in GR(\Sigma, X_p)$, $G_i \in GR(\Sigma, X_k)$ *and* $a_i \in M_{\alpha_i,\beta_i}$, $1 \le i \le k$.

Let $\mathbf{M} = (M, D, \mu)$ be a Σ-graphoid. A *sub-*Σ*-graphoid* of $\mathbf{M}$ is a subset $S \subseteq M$ such that

sg_1) $D \cup \mu(\Sigma) \subseteq S$,

sg_2) S is closed under the $\circ$- and the $\Box$-operation.

Thus S becomes a Σ-graphoid with operations the restriction on S of the corresponding operations of M.

Lemma 1. *Let* S *be a sub-*Σ*-graphoid of* $\mathbf{M} = (M, D, \mu)$*. Then*

$$a_1,\dots,a_p \in S \text{ and } G \in GR(\Sigma, X_p) \text{ implies } G[a_1,\dots,a_p] \in S.$$

Proof. By induction on the size of G. The assertion is true if $G \in \Sigma \cup X_p \cup D$. Next let G be a graph of size > 1; then either $G = G_1 \circ G_2$ or $G = G_1 \Box G_2$. In the first case we have

$$G[a_1,\dots,a_p] = (G_1 \circ G_2)(a_1,\dots,a_p) = G_1(a_1,\dots,a_p) \circ G_2(a_1,\dots,a_p) \in S,$$

because $G_1(a_1,\dots,a_p)$, $G_2(a_1,\dots,a_p)$ belong by induction to S, and S is a sub-Σ-graphoid of $\mathbf{M}$. The case $G = G_1 \Box G_2$ is treated in a similar way. $\Box$

Let $\mathbf{M} = (M, D, \mu)$ be a Σ-graphoid and $A \subseteq M$. The sub-Σ-graphoid of $\mathbf{M}$ generated by A is the smallest sub-Σ-graphoid of $\mathbf{M}$ containing A. It is denoted by $< A >_{\mathbf{M}}$. Next result gives us information about the form of the elements of $< A >_{\mathbf{M}}$.

Proposition 4. *It holds*

$$< A >_{\mathbf{M}} = \{G[a_1,\dots,a_p] \mid G \in GR(\Sigma, X_p), a_1,\dots,a_p \in A \text{ and } p \ge 0\}.$$

Proof. We consider the doubly ranked set $\tilde{A} = (A_{m,n})$, where

$$\tilde{A}_{m,n} = \{G[a_1,\dots,a_p] \mid G \in GR_{m,n}(\Sigma, X_p), a_1,\dots,a_p \in A \text{ and } p \ge 0\}.$$

By construction $D \cup \mu(\Sigma) \subseteq \tilde{A}$. On the other hand, let

$$G[a_1,\dots,a_p] \in \tilde{A}_{m,n} \text{ and } G'[a'_1,\dots,a'_q] \in \tilde{A}_{n,k}$$

with $G \in GR_{m,n}(\Sigma, X_p)$, $G' \in GR_{n,k}(\Sigma, X_q)$, respectively. We set

$$H = G \circ (G'[x_{p+1}/x_1,\dots,x_{p+q}/x_q]) \in GR_{m,k}(\Sigma, X_{p+q})$$

then

$$G[a_1,\dots,a_p] \circ G'[a'_1,\dots,a'_q] = H[a_1,\dots,a_p,a'_1,\dots,a'_q] \in \tilde{A}_{m,k}$$

and thus $\tilde{A}$ is closed under $\circ$-product. Closure under $\Box$-product can be proved analogously. Therefore, $\tilde{A}$ is a sub-Σ-graphoid of $\mathbf{M}$ including A. Next let S be a sub-graphoid of $\mathbf{M}$, such that $A \subseteq S$, according to the previous proposition

$$a_1, \ldots, a_p \in A \text{ and } G \in GR(\Sigma, X_p)$$

implies $G[a_1, \ldots, a_p] \in S$, i.e., $\tilde{A} \subseteq S$. We deduce that $\tilde{A} =< A >_{\mathbf{M}}$. □

4 Graph congruences

A notion of great importance in graph theory is that of a congruence. Actually there are two kinds of graph congruence: the ordinary and the strong graph congruences which correspond respectively to the ordinary and the fully stable tree congruences (cf. [12]).

Let $\mathbf{M} = (M, D, \mu)$ be a Σ-graphoid and assume that $\sim_{m,n}$ is an equivalence on $M_{m,n}$ $(m, n \geq 0)$ compatible with $\circ$- and $\Box$-product:

$$a \sim_{m,n} a' \text{ and } b \sim_{n,k} b', \qquad \text{implies} \qquad a \circ b \sim_{m,k} a' \circ b',$$

$$a \sim_{m,n} a' \text{ and } b \sim_{r,s} b', \qquad \text{implies} \qquad a \,\Box\, b \sim_{m+r,n+s} a' \,\Box\, b'.$$

Then we say that $\sim = (\sim_{m,n})$ is a congruence on $\mathbf{M}$.

The quotient sets $(M_{m,n}/ \sim_{m,n})$ are organized into a semi-magmoid $M/\sim$ by defining the corresponding operations in the natural way

$$[a]\Diamond[b] = [a\Diamond b], \qquad \Diamond = \circ, \Box,$$

where $[a]$ stands for the $\sim$-class of the element a. Actually $M/\sim$ becomes a Σ-graphoid through the set

$$D/\sim = \{[e_0], [e], [i_{12}], [i_{10}], [i_{21}], [i_{01}], [\pi]\}$$

and the function

$$\mu/\sim : \Sigma \to M/\sim, \quad (\mu/\sim)(\sigma) = [\mu(\sigma)], \quad \sigma \in \Sigma.$$

We use the notation $\mathbf{M}/\sim$ for the so obtained Σ-graphoid. Clearly the function sending every element a of M into $[a]$ is a Σ-graphoid morphism from $\mathbf{M}$ to $\mathbf{M}/\sim$.

Congruences can be characterized through the substitution operation.

Proposition 5. *A family of equivalences $\sim = (\sim_{m,n})$ is a congruence on $\mathbf{M} = (M, D, \mu)$ if and only if $a_1 \sim a'_1, \ldots, a_p \sim a'_p$ and $G \in GR(\Sigma, X_p)$ implies*

$$G[a_1, \ldots, a_p] \sim G[a'_1, \ldots, a'_p].$$

Proof. $(\Rightarrow)$ We establish the announced implication by using induction on the size of the graph G. Clearly we have nothing to prove if $G \in \Sigma \cup X_p \cup D$. Now any graph G of positive size can be written either as $G = G_1 \circ G_2$ or $G = G_1 \,\square\, G_2$ with $size(G_1)$, $size(G_2) < size(G)$. Then

$$\begin{aligned} G[a_1, \ldots, a_p] &= (G_1 \circ G_2)[a_1, \ldots, a_p] \\ &= G_1[a_1, \ldots, a_p] \circ G_2[a_1, \ldots, a_p] \sim G_1[a'_1, \ldots, a'_p] \circ G_2[a'_1, \ldots, a'_p] \\ &= G[a'_1, \ldots, a'_p]. \end{aligned}$$

The $\square$-case is treated analogously.

($\Leftarrow$) The converse is easy to prove: we only have to take $G = \xi_1 \Diamond \xi_2$ with $\Diamond = \circ, \square$. $\square$

Corollary 1. *The equivalence $\sim$ on $GR(\Sigma, X)$ is a congruence if for all graphs $G_i, G'_i \in GR(\Sigma, X)$, $1 \leq i \leq p$, and $G \in GR(\Sigma, X_p)$ we have*

$$G_1 \sim G'_1, \ldots, G_p \sim G'_p \quad \textit{implies} \quad G[G_1, \ldots, G_p] \sim G[G'_1, \ldots, G'_p].$$

Congruences on $GR(\Sigma, X)$ can be characterized through frame action.

Proposition 6. *The equivalence $\sim = (\sim_{m,n})$ is a congruence on $GR(\Sigma, X)$ if and only if for all $G, G' \in GR_{m,n}(\Sigma, X)$ and all frames $F \in FR^{r,s}_{m,n}(\Sigma, X)$ we have*

$$G \sim_{m,n} G' \textit{ implies } F \cdot G \sim_{r,s} F \cdot G'.$$

Proof. ($\Rightarrow$) One direction comes immediately from the previous corollary:

$$F \cdot G = F[G/\xi_{m,n}] \sim_{r,s} F[G'/\xi_{m,n}] = F \cdot G'.$$

($\Leftarrow$) For the converse, let $G_i, G'_i \in GR_{r_i,s_i}(\Sigma, X)$, and $G \in GR_{r,s}(\Sigma, X_p)$ with $rank(x_i) = (r_i, s_i)$, $1 \leq i \leq p$. We may assume that the occurrences of the same variable x_i into G can be linearly ordered. For this we only have to decompose G into a normal form as in Theorem 1:

$$\begin{aligned} G = \;& \Pi_\alpha \circ (I_{p_1,q_1} \square \cdots \square I_{p_t,q_t}) \circ \Pi_\beta \\ & \circ (E_s \,\square\, BF(x_1^1) \,\square \cdots \square\, BF(x_1^{\mu_1}) \,\square \cdots \square\, BF(x_p^1) \,\square \cdots \square\, BF(x_p^{\mu_p}) \\ & \square\, BF(\sigma_1^1) \,\square \cdots \square\, BF(\sigma_1^{\lambda_1}) \,\square \cdots \square\, BF(\sigma_u^1) \,\square \cdots \square\, BF(\sigma_u^{\lambda_u})) \end{aligned}$$

where $x_i^1 = \cdots = x_i^{\mu_i} = x_i$ $(1 \leq i \leq p)$ and $\sigma_j^1 = \cdots = \sigma_j^{\lambda_j} = \sigma_j$, $\sigma_j \in \Sigma$.

We introduce the frames

$$\begin{aligned} F_i^{(j)} = G[& G'_1/x_1, \ldots, G'_{i-1}/x_{i-1}, G'_i/x_i^1, \ldots, G'_i/x_i^{j-1}, \xi_{r_i,s_i}/x_i^j, \\ & G_i/x_i^{j+1}, \ldots, G_i/x_i^{\lambda_i}, G_{i+1}/x_{i+1}, \ldots, G_p/x_p] \end{aligned}$$

then we have

$$\begin{aligned} G[G_1, \ldots, G_p] &= F_1^1 \cdot G_1 \sim_{r_1,s_1} F_1^1 \cdot G'_1 \\ &= F_1^2 \cdot G_1 \sim_{r_1,s_1} F_1^2 \cdot G'_1 \sim_{r_1,s_1} \cdots \sim_{r_1,s_1} F_1^{\lambda_1} \cdot G_1 \sim_{r_1,s_1} F_1^{\lambda_1} \cdot G'_1 \\ &= G[G'_1, G_2, \ldots, G_p] \sim \cdots \sim G[G'_1, G'_2, \ldots, G'_p] \end{aligned}$$

and so, according to Corollary 1 the equivalence $\sim$ is a congruence. $\square$

This result allows us to characterize the congruence generated by a relation on $GR(\Sigma, X)$.

Let $R \subseteq GR(\Sigma, X) \times GR(\Sigma, X)$, i.e., $R_{m,n} \subseteq GR_{m,n}(\Sigma, X) \times GR_{m,n}(\Sigma, X)$ for all $m, n \geq 0$. For all $G, G' \in GR_{\alpha,\beta}(\Sigma, X)$ we set $G \sim_{R,\alpha,\beta} G'$ if there exist $(H, H') \in R_{m,n}$ and $F \in FR^{\alpha,\beta}_{m,n}(\Sigma, X)$ so that

$$G = F \cdot H,\ G' = F \cdot H' \text{ or } G = F \cdot H',\ G' = F \cdot H.$$

The relation $\sim^*_R$ is clearly an equivalence relation on $GR(\Sigma, X)$ which by construction contains R. To show that $\sim^*_R$ is a congruence it suffices to show that

$$G \sim_{R,m,n} G' \text{ and } F \in FR^{\alpha,\beta}_{m,n}(\Sigma, X), \text{ implies } F \cdot G \sim_{R,\alpha,\beta} F \cdot G'.$$

Indeed there exist a pair $(H, H') \in R_{r,s}$ and $F' \in FR^{m,n}_{r,s}(\Sigma, X)$ so that either

$$G = F' \cdot H,\ G' = F' \cdot H' \text{ or } G = F' \cdot H',\ G' = F' \cdot H.$$

Hence either,

$$F \cdot G = (F \cdot F') \cdot H,\ F \cdot G' = (F \cdot F') \cdot H'$$

or

$$F \cdot G = (F \cdot F') \cdot H',\ F \cdot G' = (F \cdot F') \cdot H.$$

By the definition of $\sim_R$:

$$F \cdot G \sim_{R,\alpha,\beta} F \cdot G',$$

as wanted.

Furthermore, let $\sim$ be a congruence on $GR(\Sigma, X)$ such that $R \subseteq \sim$. Since $\sim$ is reflexive and transitive, in order to show that $\sim^*_R \subseteq \sim$ it suffices to show the inclusion $\sim_R \subseteq \sim$. For this let $G \sim_{R,\alpha,\beta} G'$ then for some $(H, H') \in R_{m,n}$ and $F \in FR^{\alpha,\beta}_{m,n}(\Sigma, X)$, we have either $G = F \cdot H$, $G' = F \cdot H'$ or $G = F \cdot H'$, $G' = F \cdot H$. But, according to Proposition 6

$$H \sim_{m,n} H' \quad \text{implies} \quad G = F \cdot H \sim_{\alpha,\beta} F \cdot H' = G' \quad \text{implies} \quad G \sim_{\alpha,\beta} G'.$$

In other words $\sim^*_R$ is the smallest congruence on $GR(\Sigma, X)$ containing R.

We summarize.

Theorem 4. *Given $R \subseteq GR(\Sigma, X) \times GR(\Sigma, X)$, $\sim^*_R$ is the congruence generated by R.*

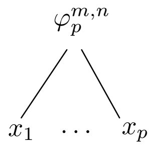
5 Strong Graph Congruences

Let $X = \{x_1, x_2, \dots\}$, with $rank(x_i) = (\alpha_i, \beta_i)$, $i \geq 1$. An equivalence $\sim$ on $GR(\Sigma, X)$ is said to be a strong congruence if for all $p \geq 1$, $G, G', G_i, G'_i \in GR(\Sigma, X_p)$, $1 \leq i \leq p$, it holds

$$G \sim G' \text{ and } G_1 \sim G'_1, \dots, G_p \sim G'_p \quad \text{implies} \quad G[G_1, \dots, G_p] \sim G'[G'_1, \dots, G'_p].$$

Since the intersection of any family of such congruences has the same property we may speak of the strong congruence generated by a relation $\mathcal{E} \subseteq GR(\Sigma, X) \times GR(\Sigma, X)$. It is the intersection of all strong congruences including $\mathcal{E}$ and is denoted by $< \mathcal{E} >$.

In order to get a more treatable form of $< \mathcal{E} >$ we introduce the p-ranked symbols



$rank(x_i) = (m_i, n_i),\ 1 \le i \le p \quad (m, n, p \ge 1)$

and we consider the set $REDEX(\Sigma)^{\alpha,\beta}_{m,n}$ of all pairs $\pi = (F, \phi_p^{m,n}(G_1, \dots, G_p))$, where $F \in FR^{\alpha,\beta}_{m,n}(\Sigma, X_p)$ and $G_i \in GR_{m_i,n_i}(\Sigma, X_p)$, $1 \le i \le p$. For every $\pi \in REDEX(\Sigma)^{\alpha,\beta}_{m,n}$ and $\pi' \in REDEX(\Sigma)^{m,n}_{r,s}$:

$$\pi = (F, \varphi_p^{m,n}(G_1, \dots, G_p)), \qquad \pi' = (F', \varphi_p^{r,s}(G'_1, \dots, G'_p))$$

we define the product

$$\pi \cdot \pi' = (F \cdot F'[G_1/x_1, \dots, G_p/x_p], \varphi_p^{r,s}(G'_1[G_1, \dots, G_p], \dots, G'_p[G_1, \dots, G_p])).$$

The sets $REDEX(\Sigma)^{\alpha,\beta}_{m,n}$ are organized into a category whose object set is $\mathbb{N} \times \mathbb{N}$ and whose composition is given by the above formula. The identity element in $REDEX(\Sigma)^{m,n}_{m,n}$ is the pair $\varepsilon^{m,n} = (\xi_{m,n}, \varphi_p^{m,n}(x_1, \dots, x_p))$. There result actions

$$REDEX(\Sigma)^{\alpha,\beta}_{m,n} \times GR_{m,n}(\Sigma, X_p) \to GR_{\alpha,\beta}(\Sigma, X_p), \quad \alpha, \beta, m, n \ge 0,$$

defined as follows: for $\pi = (F, \varphi_p^{m,n}(G_1, \dots, G_p))$ and $G \in GR_{m,n}(\Sigma, X_p)$ we set

$$(act) \qquad \pi \cdot G = F \cdot G[G_1, \dots, G_p].$$

The formulas $(\pi \cdot \pi') \cdot G = \pi \cdot (\pi' \cdot G)$, $\varepsilon^{m,n} \cdot G = G$ folow.

Proposition 7. *The equivalence $\sim = (\sim_{m,n})$ in $GR(\Sigma, X_p)$ is a strong congruence iff it is compatible with the above actions.*

Proof. ($\Rightarrow$) Suppose that $G \sim_{m,n} G'$ and

$$\pi = (F, \varphi_p^{m,n}(G_1, \dots, G_p)) \in REDEX(\Sigma)^{\alpha,\beta}_{m,n},$$

then we have $G[G_1, \dots, G_p] \sim_{m,n} G'[G_1, \dots, G_p]$ and thus by Proposition 6 we get

$$F \cdot G[G_1, \dots, G_p] \sim_{\alpha,\beta} F \cdot G'[G_1, \dots, G_p]$$

that is $\pi \cdot G \sim_{\alpha,\beta} \pi \cdot G'$.

($\Leftarrow$) Conversely, assume that

$$G \sim_{m,n} G' \quad \text{and} \quad G_i \sim_{m_i,n_i} G'_i, \quad 1 \le i \le p$$

and choose
$$\pi = (F, \varphi_p^{m,n}(x_1, \dots, x_p)), \quad F \in FR_{m,n}^{\alpha,\beta}(\Sigma, X_p)$$
then we have
$$\pi \cdot G \sim_{\alpha,\beta} \pi \cdot G', \text{ i.e., } F \cdot G \sim_{\alpha,\beta} F \cdot G'$$
and thus $\sim$ is a congruence (see Proposition 6). Consequently, we have
$$(\sigma) \qquad G[G_1, \dots, G_p] \sim_{m,n} G[G_1', \dots, G_p'],$$
choosing this time $\pi = (\xi_{m,n}, \varphi_p^{m,n}(G_1', \dots, G_p'))$ we obtain $\pi \cdot G \sim_{m,n} \pi \cdot G'$ or
$$(\tau) \qquad G[G_1', \dots, G_p'] \sim_{m,n} G'[G_1', \dots, G_p'].$$
Combining (σ) and (τ) above we find
$$G[G_1, \dots, G_p] \sim_{m,n} G'[G_1', \dots, G_p']$$
i.e., $\sim$ is a strong congruence. □

We are now ready to describe the strong congruence generated by a set $\mathcal{E} \subseteq GR(\Sigma, X) \times GR(\Sigma, X)$. For $H, H' \in GR_{\alpha,\beta}(\Sigma, X_p)$, we write $H \underset{\mathcal{E}}{\leftrightarrow} H'$ if there exist $\pi \in REDEX(\Sigma)_{m,n}^{\alpha,\beta}$ and $(G, G') \in \mathcal{E}$, $G, G' \in GR_{m,n}(\Sigma, X_p)$ so that either
$$H = \pi \cdot G, \quad H' = \pi \cdot G' \quad \text{or} \quad H = \pi \cdot G', \quad H' = \pi \cdot G.$$
We denote by $\underset{\mathcal{E}}{\overset{*}{\leftrightarrow}}$, the reflexive and transitive closure of $\underset{\mathcal{E}}{\leftrightarrow}$, i.e., $H \underset{\mathcal{E}}{\overset{*}{\leftrightarrow}} H'$ if
$$H = H_0 \underset{\mathcal{E}}{\leftrightarrow} H_1 \underset{\mathcal{E}}{\leftrightarrow} \cdots \underset{\mathcal{E}}{\leftrightarrow} H_{k-1} \underset{\mathcal{E}}{\leftrightarrow} H_k = H',$$
for some $k \geq 0$. The $\underset{\mathcal{E}}{\overset{*}{\leftrightarrow}}$ is by construction an equivalence relation.

If $H \underset{\mathcal{E}}{\leftrightarrow} H'$, then by definition,
$$H = \pi \cdot G, \quad H' = \pi \cdot G' \quad \text{or} \quad H = \pi \cdot G', \quad H' = \pi \cdot G,$$
for some $\pi \in REDEX(\Sigma)_{m,n}^{\alpha,\beta}$ and $(G, G') \in \mathcal{E}$ and so for every $\bar{\pi} \in REDEX(\Sigma)_{\alpha,\beta}^{\gamma,\delta}$ we shall have
$$\bar{\pi} \cdot H = (\bar{\pi} \cdot \pi) \cdot G, \quad \bar{\pi} \cdot H' = (\bar{\pi} \cdot \pi) \cdot G'$$
or
$$\bar{\pi} \cdot H = (\bar{\pi} \cdot \pi) \cdot G', \quad \bar{\pi} \cdot H' = (\bar{\pi} \cdot \pi) \cdot H$$
and thus $\bar{\pi} \cdot H \underset{\mathcal{E}}{\leftrightarrow} \bar{\pi} \cdot H'$.

It turns out that $\underset{\mathcal{E}}{\overset{*}{\leftrightarrow}}$ is compatible with (act) and so it is a strong congruence by virtue of the previous proposition. Now if $\sim$ is a strong congruence on $GR(\Sigma, X)$

including $\mathcal{E}$ we shall show that $\overset{*}{\underset{\mathcal{E}}{\leftrightarrow}} \subseteq \sim$. Since $\sim$ is reflexive and transitive it suffices to show that $\underset{\mathcal{E}}{\leftrightarrow} \subseteq \sim$. For this let $H \underset{\mathcal{E}}{\leftrightarrow} H'$, i.e.,

$$H = \pi \cdot G, \quad H' = \pi \cdot G' \quad \text{or} \quad H = \pi \cdot G', \quad H' = \pi \cdot G,$$

for $\pi \in REDEX(\Sigma)^{\alpha,\beta}_{m,n}$ and $(G, G') \in \mathcal{E}$. As $\mathcal{E} \subseteq \sim$ we shall have $G \sim_{m,n} G'$ and so $\pi \cdot G \sim_{\alpha,\beta} \pi \cdot G'$. Hence, $H \sim_{\alpha,\beta} H'$. In other words, $\overset{*}{\underset{\mathcal{E}}{\leftrightarrow}}$ is the smallest strong congruence including $\mathcal{E}$. We state

Theorem 5. *Given a relation $\mathcal{E} \subseteq GR(\Sigma, X) \times GR(\Sigma, X)$, the $\overset{*}{\underset{\mathcal{E}}{\leftrightarrow}}$ is the strong congruence generated by $\mathcal{E}$.*

6 Pattern Congruences

In this section we discuss congruences on patterns in connection with congruences on graphs.

Pattern substitution is obtained in a natural way. Let $X = \{x_1, x_2, \dots\}$, $rank(x_i) = (m_i, n_i)$, $i \geq 1$ and set $X_k = \{x_1, \dots, x_k\}$. For $p \in mag_{\alpha,\beta}(\Sigma \cup X_k)$ and $p_i \in mag_{m_i,n_i}(\Sigma)$, $1 \leq i \leq k$, the pattern $p[p_1, \dots, p_k]$ is by definition the image of p via the magmoid morphism $mag(\Sigma \cup X_k) \to mag(\Sigma)$ defined by the assignments

$$x_1 \mapsto p_1, \dots, x_k \mapsto p_k, \quad \sigma \mapsto \sigma \ \ (\sigma \in \Sigma).$$

The set $Fr(\Sigma)^{\alpha,\beta}_{m,n}$ of *pattern frames* is the subset of $mag_{\alpha,\beta}(\Sigma \cup \xi_{m,n})$ consisting of all patterns with just one occurrence of the symbol $\xi_{m,n}$, $rank(\xi_{m,n}) = (m, n)$. Again $Fr(\Sigma)^{\alpha,\beta}_{m,n}$ acts on $mag_{m,n}(\Sigma)$ via substitution

$$f \cdot p = f[p/\xi_{m,n}] \quad \text{for } f \in Fr(\Sigma)^{\alpha,\beta}_{m,n}, \ \ p \in mag_{m,n}(\Sigma).$$

Given an equivalence relation $\mathcal{S}_{m,n}$ on $mag_{m,n}(\Sigma)$, we say that $\mathcal{S} = (\mathcal{S}_{m,n})$ is a *congruence* on $mag(\Sigma)$ whenever we have compatibility with horizontal and vertical pattern concatenation, i.e.,

$$p_1 \equiv p'_1(\mathcal{S}_{m,n}) \text{ and } p_2 \equiv p'_2(\mathcal{S}_{n,k}) \text{ imply } p_1 p_2 \equiv p'_1 p'_2(\mathcal{S}_{m,k})$$

and

$$p_1 \equiv p'_1(\mathcal{S}_{m,n}) \text{ and } p_2 \equiv p'_2(\mathcal{S}_{r,s}) \text{ imply } \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \equiv \begin{pmatrix} p'_1 \\ p'_2 \end{pmatrix} (\mathcal{S}_{m+r,n+s}).$$

Of course the quotients $mag_{m,n}(\Sigma)/\mathcal{S}_{m,n}$ are organized, in the obvious way, into a magmoid denoted by $mag(\Sigma)/\mathcal{S}$.

Proposition 8. *Given an equivalence $\mathcal{S} \subseteq mag(\Sigma) \times mag(\Sigma)$ next conditions are equivalent:*

i) *$\mathcal{S}$ is a congruence;*

ii) *$\mathcal{S}$ is compatible with frame action, i.e.,*

$$p \equiv p'(\mathcal{S}_{m,n}) \textit{ and } f \in Fr(\Sigma)^{\alpha,\beta}_{m,n} \textit{ imply } f \cdot p \equiv f \cdot p'(\mathcal{S}_{\alpha,\beta});$$

iii) *$\mathcal{S}$ is compatible with substitution, i.e.,*

$$p_i \equiv p'_i(\mathcal{S}_{m_i,n_i}),\ 1 \leq i \leq k,\ \textit{ imply } p[p_1,\ldots,p_k] \equiv p[p'_1,\ldots,p'_k],$$

for all $p \in mag_{m,n}(\Sigma \cup X_k)$.

We now return to the standard magmoid morphism

$$val_\Sigma : mag(\Sigma \cup \bar{D}) \to GR(\Sigma)$$

whose kernel, denoted by $\sim_\Sigma$,

$$p_1 \sim_\Sigma p_2 \ \ if \ \ val_\Sigma(p_1) = val_\Sigma(p_2)$$

coincides with the congruence generated by the set of relations (E) (see Subsection 2.2). Given any pattern congruence $\mathcal{S} \subseteq mag(\Sigma \cup \bar{D}) \times mag(\Sigma \cup \bar{D})$ containing (E), its projection $val_\Sigma(\mathcal{S})$ defined by

$$G_1 \equiv G_2(val_\Sigma(\mathcal{S})) \quad \text{if} \quad G_i = val_\Sigma(p_i),\ 1 \leq i \leq 2,\ p_1 \equiv p_2(\mathcal{S})$$

is a graph congruence.

Conversely, for any congruence $\mathcal{R} \subseteq GR(\Sigma) \times GR(\Sigma)$ its inverse image $val_\Sigma^{-1}(\mathcal{R})$ defined by

$$p_1 \equiv p_2(val_\Sigma^{-1}(\mathcal{R})) \quad \text{iff} \quad val_\Sigma(p_1) \equiv val_\Sigma(p_2)(\mathcal{R})$$

is a congruence on $mag(\Sigma \cup \bar{D})$ containing (E). Therefore

Proposition 9. *The mappings*

$$\mathcal{S} \mapsto val_\Sigma(\mathcal{S}) \textit{ and } \mathcal{R} \mapsto val_\Sigma^{-1}(\mathcal{R})$$

establish a bijection between the congruences on $GR(\Sigma)$ and the congruences on $mag(\Sigma \cup \bar{D})$ including (E).

By working as in Section 4 we can show that the congruence generated by the relation $S \subseteq mag(\Sigma) \times mag(\Sigma)$ is the reflexive and transitive closure of $\sim_S$ with

$$p_1 \sim_S p_2 \quad \text{iff} \quad p_i = f \cdot q_i, \quad (1 \leq i \leq 2),$$

$f \in Fr(\Sigma)$ and either $(q_1, q_2) \in S$ or $(q_2, q_1) \in S$.

We have the next remarkable result.

Proposition 10. *It holds*

$$val_\Sigma(\overset{*}{\sim}_{S\cup(E)}) = \overset{*}{\sim}_{val_\Sigma(S)}, \quad S \subseteq mag(\Sigma \cup \bar{D}) \times mag(\Sigma \cup \bar{D})$$

that is the graph congruence generated by $val_\Sigma(S)$ coincides with the projection, via val_Σ, of the congruence generated by $S \cup (E)$.

Proof. By construction $val_\Sigma(\overset{*}{\sim}_{S\cup(E)})$ is a congruence on $GR(\Sigma)$ containing $val_\Sigma(S)$. Now, if $\mathcal{R}$ is a congruence with $\mathcal{R} \supseteq val_\Sigma(S)$, then $val_\Sigma^{-1}(\mathcal{R})$ is a congruence on $mag(\Sigma \cup \bar{D})$ containing S and so

$$val_\Sigma^{-1}(\mathcal{R}) \supseteq \overset{*}{\sim}_{S\cup(E)} .$$

Projecting by val_Σ we get

$$\mathcal{R} = val_\Sigma(val_\Sigma^{-1}(\mathcal{R})) \supseteq val_\Sigma(\overset{*}{\sim}_{S\cup(E)}),$$

i.e., $val_\Sigma(\overset{*}{\sim}_{S\cup(E)})$ is the smallest congruence containing $val_\Sigma(S)$. Hence the result. □

Proposition 11. *If $R \subseteq GR(\Sigma) \times GR(\Sigma)$ is a relation then*

$$val_\Sigma^{-1}(\overset{*}{\sim}_R) = \overset{*}{\sim}_{val_\Sigma^{-1}(R)\cup(E)} .$$

Proof. By construction $val_\Sigma^{-1}(\overset{*}{\sim}_R)$ is a congruence including $val_\Sigma^{-1}(R) \cup (E)$ while if $\mathcal{S}$ is a congruence on $mag(\Sigma \cup \bar{D})$ with

$$\mathcal{S} \supseteq val_\Sigma^{-1}(R) \cup (E),$$

then its projection $val_\Sigma(S)$ is a congruence on $GR(\Sigma)$ such that

$$val_\Sigma(val_\Sigma^{-1}(R \cup (E))) = val_\Sigma(val_\Sigma^{-1}(R)) \cup val_\Sigma((E)) = R.$$

Thus $\overset{*}{\sim}_R \subseteq val_\Sigma(S)$ and so

$$val_\Sigma(\overset{*}{\sim}_R) \subseteq val_\Sigma^{-1}(val_\Sigma(S)) = S,$$

i.e., $val_\Sigma^{-1}(\overset{*}{\sim}_R)$ is the congruence generated by $val_\Sigma^{-1}(R) \cup (E)$, as wanted. □

In the sequel we discuss strong pattern congruences. An equivalence $\mathcal{S} = (\mathcal{S}_{m,n})$ on $mag(\Sigma \cup X)$ is called a *strong congruence* if for every $k \geq 0$ and $p, p' \in mag_{\alpha,\beta}(\Sigma \cup X_k)$, $p_i, p'_i \in mag_{m_i,n_i}(\Sigma \cup X)$, $1 \leq i \leq k$, we have

$$p \equiv p'(\mathcal{S}_{\alpha,\beta}) \quad \text{and} \quad p_i \equiv p'_i(\mathcal{S}_{m_i,n_i}), \ \ 1 \leq i \leq k$$

imply

$$p[p_1, \ldots, p_k] \equiv p'[p'_1, \ldots, p'_k](\mathcal{S}_{\alpha,\beta}).$$

Here we also can achieve an action characterization. For this we introduce the set of redexes $Redex(\Sigma)_{m,n}^{\alpha,\beta}$ consisting of all pairs

$$(f, \varphi_k^{m,n}(p_1, \dots, p_k)), \quad f \in Fr(\Sigma)_{m,n}^{\alpha,\beta},\ p_i \in mag_{m_i,n_i}(\Sigma \cup X),\ 1 \leq i \leq k.$$

where $\varphi_k^{m,n}$ are k-ranked symbols as in Section 4.

These sets are organized into a category whose object set is $\mathbb{N} \times \mathbb{N}$ and whose composition is defined by

$$\begin{aligned}(f, \varphi_k^{m,n}(p_1, \dots, p_k)) \cdot (f', \varphi_k^{r,s}(p'_1, \dots, p'_k))\\ = (f \cdot f'[p_1, \dots, p_k], \varphi_k^{r,s}(p'_1[p_1, \dots, p_k], \dots, p'_k[p_1, \dots, p_k])).\end{aligned}$$

There results a canonical action

$$Redex(\Sigma)_{m,n}^{\alpha,\beta} \times mag_{m,n}(\Sigma \cup X) \to mag_{\alpha,\beta}(\Sigma \cup X)$$

if for every $\pi = (f, \varphi_k^{m,n}(p_1, \dots, p_k))$ and $p \in mag_{m,n}(\Sigma \cup X_k)$ we set

$$\pi \cdot p = (f \cdot p[p_1, \dots, p_k])$$

Proposition 12. *The equivalence $\mathcal{S} \subseteq mag(\Sigma \cup X) \times mag(\Sigma \cup X)$ is a strong congruence if and only if it is compatible with the above action i.e.,*

$$p \equiv p'(\mathcal{S}_{m,n}) \text{ and } \pi \in Cont(\Sigma)_{m,n}^{\alpha,\beta}$$

implies

$$\pi \cdot p \equiv \pi \cdot p'(\mathcal{S}_{\alpha,\beta}).$$

Theorem 6. *The strong congruence generated by the relation*

$$S \subseteq mag(\Sigma \cup X) \times mag(\Sigma \cup X)$$

is $\overset{*}{\leftrightarrow}_S$, the reflexive and transitive closure of $\leftrightarrow_S$ defined by

$$p_1 \leftrightarrow_S p_i \quad \textit{iff} \quad p_i = \pi \cdot q_i \ \ (1 \leq i \leq 2)$$

with $\pi \in Cont(\Sigma)$ and either $(q_1, q_2) \in S$ or $(q_2, q_1) \in S$.

Let $val_{\Sigma,X} : mag(\Sigma \cup \bar{D} \cup X) \to GR(\Sigma \cup X)$ be the magmoid morphism extending val_Σ by setting $val_{\Sigma,X}(x) = x$, for all $x \in X$.

Theorem 7. *The mapping $\mathcal{R} \mapsto val_{\Sigma,X}^{-1}(\mathcal{R})$ and $S \mapsto val_{\Sigma,X}(S)$ establish a bijection between the strong congruences $\mathcal{R}$ of $GR(\Sigma \cup X)$ and the strong congruences $\mathcal{S}$ on $mag(\Sigma \cup \bar{D} \cup X)$ including the set E_X obtained by adding to (E) the set of relations*

$$\bar{\pi}_{m,1} \begin{pmatrix} x \\ e \end{pmatrix} \sim \begin{pmatrix} e \\ x \end{pmatrix} \bar{\pi}_{n,1}, \quad x \in X.$$

Moreover, given relations

$$S \subseteq mag(\Sigma \cup \bar{D} \cup X)^2, \quad R \subseteq GR(\Sigma \cup X)^2$$

it holds

$$val_{\Sigma,X}^{-1}(\overset{*}{\leftrightarrow}_R) = \overset{*}{\leftrightarrow}_{val_{\Sigma,X}^{-1}(R) \cup E_X},$$

and

$$val_{\Sigma,X}(\overset{*}{\leftrightarrow}_{S \cup E_X}) = \overset{*}{\leftrightarrow}_{val_{\Sigma,X}(S)}\ .$$

7 Birkhoff's Variety Theorem for Graphs

The important Birkhoff's variety theorem is valid for graphs due to the universal characterization of $GR(\Sigma)$ displayed in Theorem 3. We must point out that most of the results of this section can be obtained by suitably adapting the proof argument of the corresponding results for trees (cf. [12]). Hence we will only present, without proofs, the main theorems in this direction. Let us begin with a classical definition.

A class $\mathcal{V}$ of Σ-graphoids forms a variety of Σ-graphoids (Σ-G-variety in short) if it is closed under isomorphism, cartesian products, sub-Σ-graphoids and quotients. Of course cartesian products are defined in the classical categorical manner.

Theorem 8. *Let $\mathcal{V}$ be a Σ-G-variety. For every set X the (doubly ranked) function*

$$j_X : X \to GR(\mathcal{V}, X), \qquad j_X(x) = x, \ \ x \in X$$

has the following universal property: for every $\mathbf{M} \in \mathcal{V}$ and every (doubly ranked) function $g : X \to \mathbf{M}$, there is a unique Σ-graphoid morphism $\tilde{g} : GR(\mathcal{V}, X) \to \mathbf{M}$ rendering commutative the triangle

$$\begin{tikzcd} X \arrow[r, "j_X"] \arrow[dr, "g"'] & GR(\mathcal{V}, X) \arrow[d, "\tilde{g}"] \\ & \mathbf{M} \end{tikzcd}$$

Remark 1. The previous result means that $GR(\mathcal{V}, X)$ is the free Σ-graphoid over $\mathcal{V}$ generated by X and its elements can be considered as "graphs inside $\mathcal{V}$".

In the case that $X = \emptyset$, the Σ-graphoid $GR(\mathcal{V}) = GR(\mathcal{V}, \emptyset)$ has the characteristic property: for all $\mathbf{M} \in \mathcal{V}$ there is a unique Σ-graphoid morphism $h_{\mathcal{V},\mathbf{M}} : GR(\mathcal{V}) \to \mathbf{M}$.

Theorem 9. *Every Σ-G-variety $\mathcal{V}$ is generated by the Σ-graphoids $GR(\mathcal{V}, X_n)$, $n \geq 0$, i.e.,*

$$\mathcal{V} = V(\{GR(\mathcal{V}, X_n) \mid n \geq 0\}).$$

Next we discuss equationality of Σ-G-varieties.

A graph equation over the doubly ranked alphabet Σ is just a pair (G, G') where $G, G' \in GR(\Sigma, X_n)$, for some $n \geq 0$. Frequently a graph equation (G, G') will be denoted by $G = G'$. We say that the equation (G, G') is satisfied in the Σ-graphoid $\mathbf{M}$ (or that $\mathbf{M}$ is a model of (G, G')) whenever

$$G[a_1, \ldots, a_n] = G'[a_1, \ldots, a_n], \quad \text{for all } a_1, \ldots, a_n \in \mathbf{M}.$$

This fact is denoted by $\mathbf{M} \models (G, G')$.

For a class $\mathcal{K}$ of Σ-graphoids, we write $\mathcal{K} \models (G, G')$ if $\mathbf{M} \models (G, G')$, for all $\mathbf{M} \in \mathcal{K}$. By $Mod(G, G')$ we denote the class of all models of (G, G') and for a set of equations $\mathcal{E} \subseteq GR(\Sigma, X) \times GR(\Sigma, X)$, we set

$$Mod(\mathcal{E}) = \bigcap_{(G,G') \in \mathcal{E}} Mod(G, G').$$

Clearly $Mod(\mathcal{E})$ is a Σ-G-variety. We say that a Σ-G-variety is equationally defined whenever $\mathcal{V} = Mod(\mathcal{E})$ for some $\mathcal{E}$.

Given a class $\mathcal{K}$ of Σ-graphoids, $Eq(\mathcal{K})$ is the set of all equations (G, G') with $\mathcal{K} \models (G, G')$. It is easy to see that $Eq(\mathcal{K})$ is a strong congruence on $GR(\Sigma, X)$.

Applying Theorem 8 we get that for any Σ-G-variety $\mathcal{V}$ we have

$$Eq(\mathcal{V}) = \sim_{\mathcal{V},X} .$$

Proposition 13. *For every Σ-G-variety $\mathcal{V}$, it holds*

$$\mathcal{V} = Mod(Eq(\mathcal{V})).$$

Consequently every Σ-G-variety is equationally defined.

Theorem 10 (Birkhoff)**.** *The correspondences*

$$\mathcal{R} \mapsto Mod(\mathcal{R}), \quad \mathcal{V} \mapsto Eq(\mathcal{V})$$

define a bijection between the class of all Σ-G-varieties and the class of all strong congruences on $GR(\Sigma, X)$.

Theorem 11 (Completness)**.** *For any set $\mathcal{E}$ of equations in $GR(\Sigma, X)$, we have*

$$Mod(\mathcal{E}) \models (G, G') \text{ if and only if } G \overset{*}{\underset{\mathcal{E}}{\leftrightarrow}} G',$$

i.e., if the equation (G, G') is satisfied by every model of $\mathcal{E}$ then we can go from G to G' by means of the equations of $\mathcal{E}$ and vise versa.

References

[1] A. Arnold and M. Dauchet, *Théorie des magmoides. I.* RAIRO Inform. Théor. 12 (3), 235-257 (1978).

[2] A. Arnold and M. Dauchet, *Théorie des magmoides. II.* RAIRO Inform. Théor. 13 (2), 135-154 (1979).

[3] F. Baader and T. Nipkow, *Term Rewriting and All That*, Cambridge University Press, (1999).

[4] S. Bozapalidis and A. Kalampakas, *An axiomatization of graphs* Acta Informatica 41, 19 - 61 (2004).

[5] S. Bozapalidis and A. Kalampakas, *Recognizability of graph and pattern languages.* Acta Informatica 42, 553 - 581 (2006).

[6] S. Bozapalidis and A. Kalampakas, *Graph Automata*, Theoretical Computer Science 393, 147 - 165 (2008).

[7] P.M. Cohn, *Universal Algebra*, D. Reidel Publishing, Dordrecht, Netherlands, (1981)

[8] F. Drewes, H.-J. Kreowski and A. Habel, *Hyperedge Replacement, Graph Grammars.* Handbook of Graph Grammars, 95-162, (1997).

[9] J. Engelfriet, *Context-free graph grammars.* In G. Rozenberg and A. Salomaa, eds, Handbook of Formal Languages. Vol. III: Beyond Words, chapter 3, pages 125-213. Springer, 1997.

[10] J. Engelfriet, J.J. Vereijken, *Context-free graph grammars and concatenation of graphs.* Acta Informatica 34, 773-803, (1997).

[11] J.W. Klop, *Term rewriting systems*, in S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors, Handbook of Logic in Computer Science, volume 2, pages 1116. Oxford University Press, 1992.

[12] W. Wechler, *Universal Algebra for Computer Scientists.* Grzegorz Rozenberg, Arto Salomaa, W. Brauer (Eds), Eatcs Monographs on Theoretical Computer 25, Springer-Verlag New York, (1992).

Acta Cybernetica **23** (2017) 141–157.

Ambiguity, Nondeterminism and State Complexity of Finite Automata

Yo-Sub Han[a], Arto Salomaa[b], and Kai Salomaa[c]

Abstract

The degree of ambiguity counts the number of accepting computations of a nondeterministic finite automaton (NFA) on a given input. Alternatively, the nondeterminism of an NFA can be measured by counting the amount of guessing in a single computation or the number of leaves of the computation tree on a given input. This paper surveys work on the degree of ambiguity and on various nondeterminism measures for finite automata. In particular, we focus on state complexity comparisons between NFAs with quantified ambiguity or nondeterminism.

Keywords: finite automata, nondeterminism, degree of ambiguity, state complexity

Dedicated to the memory of Zoltán Ésik (1951–2016).

1 Introduction

Finite automata are a fundamental model of computation that has been systematically studied since the 1950's. At the same time many important questions on finite automata and regular languages remain open [7, 18, 52]. The last decades have seen much work on the descriptional complexity, or state complexity, of regular languages [10, 13, 15, 16, 17, 28]. The state complexity (respectively, nondeterministic state complexity) of a regular language L is the optimal size of a deterministic finite automaton (DFA) (respectively, a nondeterministic finite automaton (NFA)) recognizing L. The effect of a regularity preserving operation on the minimal DFA (or alternatively a minimal NFA) is called the state complexity of the operation. The state complexity of basic operations on regular languages was considered first by Maslov [34] and further references can be found in the survey by Gao et al. [9].

[a]Department of Computer Science, Yonsei University, 50, Yonsei-Ro, Seodaemum-Gu, Seoul 120-749, Republic of Korea, E-mail: `emmous@yonsei.ac.kr`

[b]Department of Mathematics and Statistics and Turku Centre for Computer Science, University of Turku, 20014 Turku, Finland, E-mail: `salomaaenator@gmail.com`

[c]School of Computing, Queen's University, Kingston, Ontario K7L 2N8, Canada, E-mail: `ksalomaa@cs.queensu.ca`

DOI: 10.14232/actacyb.23.1.2017.9

Yu and co-authors have considered also the state complexity of combined operations and in a sequence of papers culminating with [6] have determined the precise worst-case state complexity of all combinations of two basic language operations. Establishing the precise state complexity of combined language operations is often quite involved, and for general combinations of operations that include marked concatenation and intersection the question is even undecidable [45]. Ésik et al. [8] have introduced techniques to estimate the state complexity of combined operations.

Ambiguity is a fundamental concept in grammar derivations. The ambiguity of regular expressions and finite state machines was first systematically considered by Book et al. [3]. A regular expression is unambiguous if it denotes each string in at most one way. A nondeterministic finite automaton (NFA) is unambiguous if each string has at most one accepting computation. Book et al. [3] show that the Glushkov automaton construction preserves ambiguities of a regular expression. A more restrictive notion of one-unambiguity was introduced by Brüggemann-Klein and Wood [4]: every regular language can be denoted by an unambiguous regular expression but not, in general, by a one-unambiguous regular expression.

The degree of ambiguity of an NFA A on a string w is the number of accepting computations of A on w. The degree of ambiguity of A is the maximal degree of ambiguity of A on any input string, if the maximum exists, and in this case A is said to be finitely ambiguous. Otherwise the degree of ambiguity of A can be measured as a function of the length of the inputs. Ravikumar and Ibarra [42] have first studied systematically the size trade-offs between the unambiguous, finitely ambiguous, polynomially ambiguous and exponentially ambiguous NFAs. The celebrated separation result of Leung [30] establishes that there exist (exponentially ambiguous) n-state NFAs such that any equivalent polynomially ambiguous NFA needs $2^n - 1$ states. Hromkovič et al. [19, 20] have used powerful techniques from communications complexity for state complexity separations for NFAs with different degree of ambiguity.

The degree of ambiguity is defined in terms of the number of accepting computations, and does not directly limit the amount of nondeterminism, or the amount of guessing, used by an automaton. In an unambiguous NFA, even though an accepting computation is unique, the computation may include any number of nondeterministic steps – unambiguity implies just that at any nondeterministic step at most one choice can lead to acceptance. In order to develop a quantitative understanding of the power of nondeterminism, one can directly measure the number of nondeterministic steps used by an NFA.

Nondeterminism measures for Turing machine computations were originally considered by Kintala and Fischer [25]. Kintala and Wotschke [26] first quantified the amount of nondeterminism in a finite automaton computation and showed, roughly speaking, that there is a significant difference in the determinization size blow-up between NFAs allowing different finite numbers of nondeterministic choices in a computation (where the number of nondetermistic steps is at most the logarithm of the number of states). The hierarchy result has been refined in the spectrum result of Goldstine et al. [11] that will be discussed in section 4.3.

Commonly used nondeterminism measures count the number of nondeterministic steps (or the amount of guessing in bits of information) on a best accepting computation [11], or the number of leaves of the entire computation tree [19, 38]. Further variants limit the amount of nondeterminism on a worst computation [19, 39]. Some interesting relationships between the degree of ambiguity and various nondeterminism measures have been established by Goldstine et al. [12] and Hromkovič et al. [19].

This paper surveys work on the growth rates of the degree of ambiguity and the various nondeterminism measures, and on algorithms to determine the growth rate for a given NFA. In particular, we focus on state completity comparisons between NFAs having different degrees of ambiguity or allowing different amounts of nondeterminism. Strong separation results are known for succinctness comparisons between NFAs of different ambiguity growth rates (finite, polynomial or exponential). However, in the case of limited nondeterminism, practically all existing work on state complexity is restricted to comparisons between different finite amounts of nondeterminism, that is, the amount of nondeterminism on any input is at most a given constant. State complexity of NFAs with limited nondeterminism that is measured as a function of input length is a topic for future study.

First we fix some notation in section 2. Work on the degree of ambiguity is described in section 3 and section 4 deals with the various nondeterminism measures for NFAs.

2 Definitions

Here we recall and introduce some basic notation and definitions. More information on finite automata and regular languages can be found e.g. in [44, 47, 51]. General background on degrees of ambiguity and limited nondeterminism for finite automata can be found in [10, 13, 16, 41].

The set of positive integers is $\mathbb{N}$ and the cardinality of a finite set F is $|F|$. The set of strings over a finite alphabet Σ is Σ^* and ε is the empty string. A *bounded language* is a subset of $a_1^* a_2^* \cdots a_k^*$, where a_i, $1 \leq i \leq k$, are (not necessarily distinct) elements of the alphabet Σ.

A *nondeterministic finite automaton* (NFA) is a tuple $A = (Q, \Sigma, \delta, q_0, F)$ where Q is the finite set of states, Σ is the input alphabet, $\delta : Q \times \Sigma \to 2^Q$ is the transition function, $q_0 \in Q$ is the initial state and $F \subseteq Q$ is the set of final states. The transition function δ is in the usual way extended as a function $Q \times \Sigma^* \to 2^Q$ and the language recognized by A is $L(A) = \{w \in \Sigma^* \mid \delta(q_0, w) \cap F \neq \emptyset\}$. If $|\delta(q, b)| \leq 1$ for all $q \in Q$ and $b \in \Sigma$, the automaton A is a *deterministic finite automaton* (DFA). Note that we allow DFAs to have undefined transitions.

It is well known that DFA's and NFA's both recognize the class of regular languages. For a regular language L, the state complexity of L (respectively, the nondeterministic state complexity of L) is the number of states of the state minimal DFA (respectively, of a state minimal NFA[1]) recognizing L.

[1]An NFA with the smallest number of states recognizing a language L need not be unique.

Consider an NFA $A = (Q, \Sigma, \delta, q_0, F)$. The *branching* of a transition from state $q \in Q$ on input symbol $b \in \Sigma$ is $|\delta(q, b)|$. A *computation* of A on a string $w = b_1 b_2 \cdots b_k$, $b_i \in \Sigma$, $i = 1, \ldots, k$, $k \geq 0$, is a sequence of states $(p_1, \ldots, p_\ell)$, where $p_1 \in \delta(q_0, b_1)$, $p_{j+1} \in \delta(p_j, b_{j+1})$, $j = 1, \ldots \ell - 1$, and either $\ell = k$, or, $\ell < k$ and $\delta(p_\ell, b_{\ell+1}) = \emptyset$.

The sequence of states $(p_1, \ldots, p_\ell)$ is a *complete computation* on $b_1 b_2 \cdots b_k$ if $\ell = k$ and an *accepting computation* is a complete computation that ends in an accepting state of F. The set of all computations (respectively, all accepting computations) of A on the string w is denoted $\mathrm{comp}_A(w)$ (respectively, $\mathrm{comp}_A^{\mathrm{acc}}(w)$).

Intuivively, a computation of A on a string w is a sequence of states that A reaches when started with the initial state and the symbols of w are read one by one. A complete computation ends with a state reached after consuming all symbols of w. A computation may also end with a state where the transition on the next symbol of w is undefined.

3 Degree of ambiguity

Book et al. [3] first considered systematically the ambiguity of regular expressions and NFAs, and the relationship between these notions. A regular expression is unambiguous if it denotes each string in at most one way. A more restrictive notion of 1-unambiguity, or 1-determinism, was introduced by Brüggemann-Klein and Wood [4]. A regular expression is 1-unambiguous if its position automaton is deterministic. Every regular language has an unambiguous regular expression but the 1-unambiguous expressions define a strict subclass of regular languages [4, 14].

An NFA is unambiguous if any string has at most one accepting computation. Formally, the degree of ambiguity of an NFA A on a string w, $\mathrm{da}_A(w)$, is the number of accepting computations of A on w. The *degree of ambiguity* of A on strings of length m is defined as

$$\mathrm{da}_A(m) = \max\{\mathrm{da}_A(w) \mid w \in \Sigma^m\}.$$

Strictly speaking, we use the symbol da_A to denote two different functions: it denotes a function $\Sigma^* \to \mathbb{N}$ and a function $\mathbb{N} \to \mathbb{N}$.

The degree of ambiguity of A is said to be finite (or bounded) if the values $\mathrm{da}_A(m)$, $m \in \mathbb{N}$ are bounded, and in this case we denote

$$\mathrm{da}_A^{\mathrm{sup}} = \sup_{m \in \mathbb{N}} \mathrm{da}_A(m).$$

The NFA A is *unambiguous* if $\mathrm{da}_A^{\mathrm{sup}} = 1$. Clearly every DFA is unambiguous.

Following Ravikumar and Ibarra [42], with respect to the degree of ambiguity we consider five different classes of NFAs: DFAs, unambiguous NFAs (UFA), finitely ambiguous NFAs (FNFA), polynomially ambiguous NFAs (PNFA) and general (potentially exponentially ambiguous) NFAs. An NFA A is strictly polynomially ambiguous if A is not finitely ambiguous and there is a polynomial $p(\cdot)$ such that $\mathrm{da}_A(m) \leq p(m)$ for all $m \in \mathbb{N}$. The polynomial degree of growth of A is the minimal degree of a polynomial $p'(m)$ that upper bounds the function $\mathrm{da}_A(m)$. An NFA A is strictly exponentially ambiguous if it is not polynomially ambiguous.

It is known that for a fixed $k \in \mathbb{N}$ the equivalence of FNFAs with degree of ambiguity k can be tested in polynomial time [27, 49]. This is significant because, as we will see, the determinization of even UFAs can cause an exponential size blow-up. Also, different variants of the minimization problem for UFAs remain intractable, see [16] for references.

The syntactic definition of an NFA A does not directly tell us what is the degree of ambiguity of A. It was shown by Mandel and Simon [33] and Reutenauer [43], and by others independently, that it is decidable whether a given NFA is finitely ambiguous or polynomially ambiguous. Reutenauer [43] also gave an algorithm to compute the polynomial degree of growth of an NFA.

Building on charaterizations by Ibarra and Ravikumar [21] and Reutenauer [43], Weber and Seidl [50] gave a simpler structural characterization of finitely ambiguous and polynomial ambiguous NFAs that yields a polynomial time algorithm for the corresponding decision problems. The characterization implies also that, for an NFA with unbounded ambiguity, the degree of ambiguity must grow at least linearly.

Theorem 1 (Weber and Seidl [50]). *It can be decided in polynomial time whether a given NFA A is finitely ambiguous, strictly polynomially ambiguous or strictly exponentially ambiguous. Furthermore, the polynomial degree of growth of A can be computed in polynomial time.*

For the question of determining the exact finite degree of ambiguity, the complexity depends essentially on whether the finite degree of ambiguity is a constant or considered part of the input. For a fixed k, it can be tested in polynomial time whether the degree of ambiguity of an NFA is greater than k [49], but when k is part of the input the complexity is essentially worse.

Theorem 2 (Chan and Ibarra [5]). *For a given NFA A and $k \in \mathbb{N}$, testing whether the degree of ambiguity of A is at least k is PSPACE-complete.*

A relevant question is also how large can be the degree of ambiguity of an n-state FNFA. A double exponential upper bound was given already by Mandel and Simon [33] and this was impoved to $2^{\Theta(n^3)}$ by Reutenauer [43]. The bound was further improved by Weber and Seidl [50] who also show that for some subclasses of NFAs the maximal finite degree of ambiguity is exactly $2^{\Theta(n)}$.

Theorem 3 (Weber and Seidl [50]). *The degree of ambiguity of an n-state FNFA is at most $5^{\frac{n}{2}} \cdot n^n$.*

3.1 Ambiguity and state complexity

Clearly every regular language can be recognized by an unambiguous NFA, but the succinctness of the description depends significantly on the degree of ambiguity. Schmidt [46] first developed methods to prove lower bounds for the size of UFAs and also showed that the determinization of UFAs causes, in the worst case, an exponential size blow-up. The lower bound was improved by different authors and the precise worst case size blow-up was determined by Leung [32]. Leiss [29] constructed n-state UFAs with multiple initial states where any equivalent DFA needs $2^n - 1$ states and Leung [32] gave a construction for the same exponential size blow-up using UFAs with only one initial state.

Theorem 4 (Leiss [29], Leung [32])**.** *For each $n \in \mathbb{N}$, there exists an UFA with n states such that the minimal equivalent DFA has $2^n - 1$ states. For each $n \in \mathbb{N}$, there exists an FNFA with n states such that any equivalent UFA has $2^n - 1$ states.*

Note that because our definition allows DFAs to be incomplete the above bound for the UFA determinization differs by one from the bound stated in [32]. Input-driven pushdown automata (IDPDA) define a subclass of deterministic context-free languages that retains many of the desirable properties of the regular languages. In particular, an n-state nondeterministic IDPDA has an equivalent deterministic machine with $2^{\Theta(n^2)}$ states [1]. Recently, Okhotin and Salomaa [36] have shown that, analogously with Theorem 4, determinizing an unambiguous IDPDA and converting a general nondeterministic IDPDA to an unambiguous one both cause, in the worst case, the same $2^{\Theta(n^2)}$ size blow-up.

For state complexity comparisons between NFAs with different growth rates of ambiguity we use the following terminology. Consider classes X and Y of devices (the classes we consider are DFAs, UFAs, FNFAs, PNFAs and general NFAs, possibly with additional restictions). We say that class Y is *(super-polynomially) separated* from class X, if there exists a collection of languages L_n, $n \in \mathbb{N}$, such that L_n is recognized by a device from class Y having n states, but for any polynomial $p(n)$ and for sufficiently large values of n, a device from class X for L_n must have more than $p(n)$ states. This means, roughly speaking, that simulation of devices of class Y by devices of class X, in the worst case, causes a super-polynomial size blow-up.

Ravikumar and Ibarra [42] first considered systematically succinctness comparisons between FNFAs, PNFAs and general NFAs. In particular, they established the following result for NFAs accepting bounded languages.

Theorem 5 (Ravikumar and Ibarra [42])**.** *Any NFA accepting a bounded language can be converted to an FNFA with at most polynomial size blow-up. The class of FNFAs (respectively, the class of UFAs) recognizing a bounded language is super-polynomially separated from the corresponding class of UFAs (respectively, of DFAs).*

The descriptional complexity comparison between the classes FNFA, PNFA and NFA recognizing general regular languages was left open in [42]. Although in the case of bounded languages, NFAs of exponential ambiguity can be simulated by PNFAs and FNFAs of polynomial size, it was conjectured that for general regular languages the classes are super-polynomially separated. Leung [30] and Hromkovič et al. [19] have established that general NFAs can be super-polynomially more succinct than PNFAs.

Theorem 6 (Leung [30], Hromkovič et al. [19]). *The class of NFAs is super-polynomially separated from the class of PNFAs.*

The communication complexity techniques used by Hromkovič et al. [19] to prove Theorem 6 yield a substantially simplified proof. However, their proof does not give the optimal size blow-up $2^n - 1$ for the NFA–to–PNFA transformation that is obtained in the original ad hoc proof where Leung [30] shows that any PNFA for the family of languages $L_n = (0 + (01^*)^{n-1}0)^*$, $n \geq 1$, cannot be smaller than an incomplete DFA. It is easy to give an n-state NFA of exponential ambiguity that recognizes L_n.

Ravikumar and Ibarra [42] also conjectured that polynomially ambiguous NFAs can be significantly more succinct than finitely ambiguous NFAs. This question remained open for over 20 years. After about 10 years Hromkovič et al. [19] gave a partial result showing that there exist $(n+2)$-state PNFAs (with linear degree of ambiguity) such that any equivalent FNFA with degree of ambiguity k must have at least $2^{\frac{n-2}{k}} - 2$ states. The question was solved affirmatively by Hromkovič and Schnitger [20] using the powerful communication complexity techniques.

Theorem 7 (Hromkovič and Schnitger [20]). *For $n \in \mathbb{N}$ there exists PNFA A with number of states polynomial in n such that any FNFA recognizing the language $L(A)$ has at least $2^{\Omega(n^{\frac{1}{3}})}$ states.*

Theorem 7 is obtained as a special case of the more technical statement given next in Theorem 8 by setting there the parameter k to be one and, in fact, the degree of ambiguity of the PNFA A is only linear. The general result by Hromkovič and Schnitger [20] gives a super-polynomial succinctness separation between NFAs with degree of ambiguity $O(m^k)$ and $O(m^{k-1})$, $k \in \mathbb{N}$.

Theorem 8 (Hromkovič and Schnitger [20]). *Let r and $t = (r/k^2)^{\frac{1}{3}}$ be positive integers. There exist languages $L_{r,k}$ having an NFA with degree of ambiguity $O(m^k)$ and $k \cdot \text{poly}(r)$ states such that any NFA for $L_{r,k}$ with degree of ambiguity $o(m^k)$ has at least $2^{\Omega(r^{(\frac{1}{3}}/k^{\frac{5}{3}})}$ states.*

Theorem 7 and Theorem 8 give a super-polynomial separation, respectively, between PNFAs and FNFAs and between NFAs having different polynomial degree of growth for ambiguity. The statement of Theorem 8 defines the languages $L_{r,k}$ only for restricted values of the subindices, but for the separation result it is sufficient that $L_{r,k}$ exists for infinitely many values of r and k. However, the lower bounds are not of the order $2^{\Theta(n)}$ as is known in the separation of general NFAs and PNFAs [30]. In fact, Hromkovič and Schnitger [20] suspect that the lower bound of Theorem 8 may not be optimal even for the languages used in the lower bound construction.

To conclude this section, we mention that Okhotin [35] has studied the state complexity of determinization of unary UFAs and Jirasek et al. [22] recently studied the state complexity of operations on UFAs.

4 Limited nondeterminism

Nondeterminism measures can be based on the amount of nondeterminism used in a best accepting computation of an NFA A on a given string w, on the amount of nondeterminism in a worst computation of A on w or on the size of the computation tree of A on w [10, 11, 19, 38].

In the following, $A = (Q, \Sigma, \delta, q_0, F)$ is always an NFA. Consider a string $w = b_1 b_2 \cdots b_k$, $b_i \in \Sigma$, $i = 1, \ldots, k$, and a computation of A on w,

$$C = (p_1, \ldots, p_\ell), \;\; p_i \in Q, 1 \leq i \leq \ell \leq k.$$

Recall that $\ell < k$ is possible only if $\delta(p_\ell, b_{\ell+1}) = \emptyset$, that is, a computation reads the entire string w unless it encounters an undefined transition.

The *guessing* of the computation C, $\gamma_A(C)$ [11], is

$$\gamma_A(C) = \log_2 |\delta(q_0, b_1)| + \sum_{i=1}^{\ell-1} \log_2 |\delta(p_i, b_{i+1})|.$$

The branching of the first step of the computation C is $|\delta(q_0, b_1)|$, and after the first step the state is p_1. The branching of the second step is then $|\delta(p_1, b_2)|$, and the branching of the ith step is $|\delta(p_{i-1}, b_i)|$, $3 \leq i \leq \ell - 1$. Thus, intuitively, $\gamma_A(C)$ represents the amount of guessing, in bits of information, that occurs during the computation C. If A is a DFA, the amount of guessing in any computation of A is zero.

The *branching of the computation C*, $\beta_A(C)$ [11], is defined as the product of the branchings of the individual transitions of C, or in other words, $\beta_A(C) = 2^{\gamma_A(C)}$.

The amount of guessing an NFA uses on a string can be defined either as a best case or a worst case measure. The *guessing of a string* $w \in L(A)$ [11] is the amount of guessing of the best accepting computation:

$$\gamma_A(w) = \min\{\ \gamma_A(C) \mid C \in \text{comp}_A^{\text{acc}}(w)\ \},$$

and the *maximum guessing* of A on a string $w \in \Sigma^*$ [39] is

$$\gamma_A^{\max}(w) = \max\{\ \gamma_A(C) \mid C \in \text{comp}_A(w)\ \}.$$

Note that the best case measure is defined as the amount of guessing on the best accepting computation while the maximum guessing considers all, not necessarily complete, computations. Instead of counting the amount of guessing in bits of information, Hromkovič et al. [19] use the *advice measure* that counts the number of nondeterministic steps on the worst computation on a given input and Leung [31] uses a corresponding best case measure. These measures are within a multiplicative constant (depending only on the NFA A) of the $\gamma_A^{\max}$ and γ_A measures, respectively.

The *branching* (respectively, the *trace*) of A on the string w is then $\beta_A(w) = 2^{\gamma_A(w)}$ (respectively, $\tau_A(w) = 2^{\gamma_A^{\max}(w)}$ [39, 41]).

The total amount of nondeterminism used by A in all computations on a string w is represented by the number of leaves of the computation tree of A on w. The number of leaves is the same as the number of computations of A on w, $|\mathrm{comp}_A(w)|$, and this value is called the *tree width* of A on w, $\mathrm{tw}_A(w)$. The tree width measure is called 'leaf size' in [19].

Similarly as we did with the degree of ambiguity, the tree width, the (maximum) guessing, the branching and the trace of an NFA A defines a function on naturals by taking the maximum value of the measure on strings of length m ($m \in \mathbb{N}$). If χ is any of tw, γ, $\gamma^{\max}$, β, or τ then $\chi_A : \mathbb{N} \to \mathbb{N}$ is defined as

$$\chi_A(m) = \max\{\ \chi_A(w) \mid w \in \Sigma^m\ \},\quad m \in \mathbb{N}.$$

We say that the χ-function of A is *finite* (or *bounded*) if the value $\chi_A^{\sup} =^{\mathrm{def}} \sup_{m\in\mathbb{N}} \chi_A(m)$ is finite.

Hromkovič et al. [19] have characterized the possible growth rates of the tree width of an NFA. As for degree of ambiguity, the tree width of an NFA cannot be unbounded and sublinear.

Theorem 9 (Hromkovič et al. [19]). *For any NFA A, the function $\mathrm{tw}_A(m)$ is either bounded by a constant, or between linear and polynomial in m, or otherwise in $2^{\Theta(m)}$.*

The above characterization can be effectively decided. An NFA A has unbounded tree width if and only if some cycle of A contains a nondeterministic transition and this observation yields a simple polynomial time algorithm to test whether $\mathrm{tw}_A(m)$ is bounded [38]. On the other hand, there is no efficient algorithm to determine whether the guessing of an NFA is bounded.

Theorem 10 (Leung [31]). *For a given NFA A, it is PSPACE-complete to decide whether $\gamma_A(m)$ is bounded.*

Interestingly it is known that the guessing of an NFA may be unbounded and grow sublinearly.

Theorem 11 (Simon [48], Goldstine et al. [12]). *For each $k \in \mathbb{N}$, there is an NFA A such that $\gamma_A(m) = \Theta(\sqrt[k]{m})$.*

Due to the exponential correspondence between the guessing and branching measures, Theorem 11 implies that, for each $k \in \mathbb{N}$, there exists an NFA A such that $\beta_A(m) = 2^{\Theta(\sqrt[k]{m})}$. It is not known whether the branching of an NFA can be polynomially bounded but infinite [39].

Open 1. *If NFA A has unbounded branching does this imply that the growth rate of $\beta_A(m)$ must be superpolynomial?*

It is known that, for a unary NFA A, $\beta_A(m)$ is always either bounded or in $2^{\Theta(m)}$ [41] and the possible growth rates of a variant of the branching measure considered in [37] are similarly restricted. For the worst-case branching measure trace, Palioudakis et al. [39] have shown that, for an n-state NFA A, $\tau_A(m)$ is either bounded or $\tau_A(m) \geq 2^{\lfloor \frac{m}{n} \rfloor}$.

4.1 NFAs with large finite nondeterminism

Similarly as in the case of degree of ambiguity [50], for an n-state NFA A with bounded guessing (respectively, bounded tree width) we can ask how large can the guessing (respectively, the tree width) of A be. Leung [31] has shown that an n-state NFA with limited nondeterminism in any computation can make at most $2^n - 2$ nondeterministic transitions and has constructed a family of NFAs with bounded nondeterminism that is considerably larger than the number of states.

Theorem 12 (Leung [31]). *If A is an n-state NFA with bounded guessing, then $\gamma_A(m) = O(2^n)$. There exist n-state NFAs B_n, $n \in \mathbb{N}$ such that $\gamma_{B_n}^{\sup} = 2^{\frac{n}{3}} - 2$.*

A limitation of the above result is that in the NFAs B_n are defined over a growing alphabet and a large number of the nondeterministic moves are redundant. It remains open whether there exist n-state NFAs A with bounded guessing that is larger than n and where the language $L(A)$ cannot be recognized by an NFA of same size and less nondeterminism [31]. The notion of "less nondeterminism" could be formalized analogously as is done below with the notion of optimality in the case of tree width.

Hromkovič et al. [19] observed that the tree width of an n-state NFA, if bounded, is at most n^n. Palioudakis et al. [38] improved this bound and, furthermore, gave a construction of n-state NFAs with all possible values of bounded tree width that do not have "redundant" nondeterminism.

The notion of avoiding redundant nondeterminism is formalized as follows. A finite tree width NFA A with n states is said to have *optimal tree width* if $L(A)$ cannot be recognized by any NFA B with n_1 states where $n_1 \leq n$ and $\mathrm{tw}_B^{\sup} \leq \mathrm{tw}_A^{\sup}$ and at least one of the inequalities is strict.

Theorem 13 (Palioudakis et al. [38]). *The tree width of an n-state finite tree width NFA is at most 2^{n-2}. For every $n \geq 2$ and $1 \leq k \leq 2^{n-2}$ there exists an n-state NFA over a binary alphabet having optimal tree width k.*

Note that the above bound is less than the upper bound for the finite ambiguity of an n-state NFA (from Theorem 3). Naturally, for any NFA A and string w, the degree of ambiguity of A on w is at most the tree width of A on w (and usually much smaller than the tree width). However, an upper bound for the finite tree width of an n-state NFA does not imply a corresponding bound for the degree of ambiguity because an NFA may have finite ambiguity and unbounded tree width.

4.2 Comparing nondeterminism measures and ambiguity

Directly from the definitions it follows that if an NFA A has finite tree width, then the guessing (and branching) of A is also finite, but the converse implication does not need to hold. The tree width of A is finite if and only if the trace of A is finite.

Proposition 1 (Palioudakis et al. [39]). *If A is an NFA with finite tree width, then*

$$\mathrm{tw}_A^{\sup} \leq \tau_A^{\sup} \leq 2^{\mathrm{tw}_A^{\sup} - 1}.$$

It is known that the above inequalities cannot be improved in general, that is, there are NFAs for which either of the inequalities of Proposition 1 becomes and equality [39].

Hromkovič et al. [19] have established relationships between the tree width, maximum guessing and degree of ambiguity in a minimal NFA. They use the name 'leaf size' for tree width and instead of maximum guessing they use an "advice" measure that is within a constant factor of maximum guessing. The advice of an NFA A on a string w counts the largest number of nondeterministic steps in any computation of A on w.

Theorem 14 (Hromkovič et al. [19])**.** *If A is a minimal NFA, then for all $m \in \mathbb{N}$,*

$$\max(\gamma_A^{\max}(m), \mathrm{da}_A(m)) \leq \mathrm{tw}_A(m) = O(\mathrm{da}_A(m) \cdot \gamma_A^{\max}(m)).$$

Goldstine et al. [12] have established a subtle relationship between ambiguity and guessing for NFAs where all states are final. They define the ambiguity of a string w as the number of complete computations on w. To avoid confusion, we call the number of complete computations of an NFA A on a string w the *complete ambiguity*[2] of A on w. Note that if all states of A are final, the complete ambiguity of A coincides with the degree of ambiguity as defined in section 3 and if A has no undefined transitions then the complete ambiguity of A coincides with the tree width of A.

By definition, the guessing function $\gamma_A(m)$ of an NFA grows at most linearly. If the guessing is bounded or grows linearly, then the complete ambiguity may be either bounded or unbounded but, in the intermediate case, where the guessing is unbounded but sublinear, then ambiguity must always be unbounded. Recall from Theorem 11 that there exist NFAs with unbounded and sublinear growth rate of the guessing function.

Theorem 15 (Goldstine et al. [12])**.** *Let A be an NFA. If $\gamma_A(m)$ is non-constant and sublinear, then the complete ambiguity of A must be unbounded. On the other hand, if $\gamma_A(m)$ is in $O(1)$ or in $\Theta(m)$, then the complete ambiguity may be either bounded or unbounded.*

4.3 Limited nondeterminism and state complexity

An important descriptional complexity question is the succinctness comparison of NFAs employing different amounts of nondeterminism and, in particular, the determinization size blow-up of NFAs with limited nondeterminism. Goldstine et al. [11] have shown that converting a general NFA to an NFA with finite branching involves, in the worst case, an exponential size blow-up.

Theorem 16 (Goldstine et al. [11])**.** *For each $n \in \mathbb{N}$, there exists an n-state NFA A such that any finite branching NFA recognizing the language $L(A)$ needs at least 2^{n-1} states.*

[2]Keeler [24] calls this the string path width of A on w.

Also, Goldstine et al. [11] have shown that there exist regular languages for which different finite amounts of nondeterminism yield incremental savings in the number of states. The following two theorems give the "spectrum" result of [11] stated in a slightly simplified form.

Theorem 17 (Goldstine et al. [11])**.** *Let A be a minimal DFA of size $2^n - 1$, $n \geq 2$. Then*

(i) for $2 \leq k \leq \frac{n}{\log_2 n}$, the optimal size of an NFA with branching k for $L(A)$ is at least $2^{\frac{n}{k}}$, and,

(ii) for $k \geq \frac{n}{\log_2 n}$, an NFA with branching k for $L(A)$ has size at least n.

Furthermore, they show that the bounds of Theorem 17 are close to best possible:

Theorem 18 (Goldstine et al. [11])**.** *For $n \geq 2$ there exists a minimal NFA A_n with $n + 1$ states such that if we denote by $\sigma_n[k]$ the optimal size of an NFA with branching k recognizing $L(A_n)$ then the following relations hold:*

(i) $\sigma_n[1] = 2^n$, and, $2^{\frac{n}{k}} \leq \sigma_n[k] < 2k \cdot 2^{\frac{n}{k}}$ when $2 \leq k < \frac{n}{\log_2 n}$,

(ii) $n + 1 \leq \sigma_n[k] < 2k \cdot 2^{\frac{n}{k}}$ when $\frac{n}{\log_2 n} \leq k < n$,

(iii) $n + 1 \leq \sigma_n[k] < 4n$, when $k \geq n$.

Recall that tree width is more restrictive than branching in the sense that an NFA with finite tree width necessarily has finite branching, but the converse implication does not hold, in general. Contrasting the result of Theorem 16, every finite tree width NFA has an equivalent DFA of polynomial size.

Theorem 19 (Palioudakis et al. [38])**.** *For an NFA A with n states having tree width at most $k \leq n - 1$, the language $L(A)$ has a DFA of size $1 + \sum_{j=1}^{k} \binom{n-1}{j}$. Furthermore, for every $1 \leq k \leq n - 1$, there exists an n state NFA $A_{n,k}$ with tree width k over a binary alphabet such that the minimal DFA for $L(A_{n,k})$ has $1 + \sum_{j=1}^{k} \binom{n-1}{j}$ states.*

Palioudakis et al. [38] gives also an upper bound $1+\sum_{i=1}^{k-\ell+1} \binom{n-1}{i}$ for converting an n state NFA with tree width k to an NFA with tree width $2 \leq \ell < k$, but a corresponding lower bound is missing. Also no spectrum result for tree width analogous to the spectrum result for branching (Theorem 18) is known. That is, there is no result that yields good bounds, for a given sequence of languages, for the succinctness of NFAs over a range of different tree width values.

Deterministic finite automata with multiple initial states (MDFA) can be viewed as a restricted type of automata with limited nondeterminism: the only nondeterminism consists of the choice of the initial state. With an elegant construction based on modular arithmetic, Kappes [23] has given an efficient simulation of an NFA with finite branching by an MDFA.

Theorem 20 (Kappes [23]). *An NFA with n states and branching k ($k \in \mathbb{N}$) can be simulated by an MDFA with $k \cdot n$ states and k initial states.*

The bound as stated in [23] is $k \cdot n + 1$ and the construction produces an MDFA with a dead state. Above in Theorem 20 we allow the possibility that an MDFA can have undefined transitions (following the definition of [40]). Palioudakis et al. [40] have established an almost matching lower bound by showing that for infinitely many values $n, k \in \mathbb{N}$ there exists an n-state NFA with branching k such that any equivalent MDFA needs at least $\frac{k}{1+\log k} \cdot n$ states.

To conclude we mention that limiting the nondeterminism of an NFA is not sufficient to make the minimization problem tractable. It is well known that minimization of general NFAs is PSPACE-complete. Björklund and Martens [2] have shown that minimization remains NP-hard, roughly speaking, for all finite automaton models that extend the class of DFAs. The hardness result is for the class of δNFAs which are a very restricted subclass of tree width two NFAs [2].

5 Conclusion and open problems

Descriptional complexity comparison of nondeterministic finite automata of different degrees of ambiguity and employing different amounts of nondeterminism is a foundational question in automata theory. The spectrum result of Goldstine et al. [11] (Theorems 17 and 18) establishes the existence of a sequence of languages for which different finite amounts of branching allow incremental savings in the number of states, and the succinctness comparisons are approximately the best possible.

On the other hand, very little is known about the state complexity of NFAs where the amount of nondeterminism is unbounded and measured as a function of input length. While there is a super-polynomial separation between the size of finitely ambiguous, polynomially ambiguous, and general NFAs, no succinctness comparisons between NFAs of different unbounded branching or unbounded tree width are known. This can be a topic for future research. In particular, it would be interesting to know whether the powerful communication complexity techniques used by Hromkovič et al. [19, 20] for succinctness comparisons of NFAs with different degrees of ambiguity can be used to establish good lower bounds and separation results for the state complexity of NFAs where the branching (or the tree width) is measured as a function of input length.

A further topic of interest could be the succinctness comparison of NFAs of given degree of ambiguity and NFAs of given branching (or tree width). Only a few tentative results are known [38] in this direction.

Acknowledgements

Han was supported by the Basic Science Research Program through NRF funded by MEST (2015R1D1A1A01060097) and the Yonsei University Future-leading Research Initiative of 2016. K. Salomaa was supported by the Natural Sciences and Engineering Research Council of Canada Grant OGP0147224.

References

[1] Alur, R. and Madhusudan, P. Adding nesting structure to words. *Journal of the ACM,* 56(3), 2009.

[2] Björklund, H. and Martens, W. The tractability frontier for NFA minimization. *J. Comput. System Sci.,* 78: 198–210, 2012.

[3] Book, R., Even, S., Greibach, S. and Ott, G. Ambiguity in graphs and expressions. *IEEE Transactions on Computers,* c-20(2):149–153, 1971.

[4] Brüggemann-Klein, A. and Wood, D. One-unambiguous regular languages. *Information and Computation,* 140(2):229–253, 1998.

[5] Chan, T.-H. and Ibarra, O. On the finite-valuedness problem for sequential machines. *Theoret. Comput. Sci.,* 23: 95–101, 1983.

[6] Cui, B., Gao, Y., Kari, L., Yu, S. State complexity of combined operations with two basic operations. Theoret. Comput. Sci., 437: 82–102, 2012.

[7] Ellul, K., Krawetz, B., Shallit, J. and Wang, M.-W. Regular expressions: New results and open problems. *Journal of Automata, Languages and Combinatorics,* 10: 407–437, 2005.

[8] Ésik, Z., Gao, Y, Liu, G., Yu, S. Estimation of state complexity of combined operations. *Theoret. Comput. Sci.,* 410: 3272–3280, 2009.

[9] Gao, Y., Moreira, N., Reis, R., Yu, S. A review on state complexity of individual operations. To appear in *Computer Science Review.* (Available at `www.dcc.fc.up.pt/dcc/Pubs/TReports/TR11/dcc-2011-08.pdf`)

[10] Goldstine, J., Kappes, M., Kintala, C.M.R., Leung, H., Malcher, A. and Wotschke, D. Descriptional complexity of machines with limited resources. *J. Universal Computer Science,* 8(2): 193–234, 2002.

[11] Goldstine, J., Kintala, C.M.R. and Wotschke, D. On measuring nondeterminism in regular languages. *Inform. Comput.,* 86: 179–194, 1990.

[12] Goldstine, J., Leung, H. and Wotschke, D. On the relation between ambiguity and nondeterminism in finite automata. *Inform. Comput.,* 100: 261–270, 1992.

[13] Gruber, H., Holzer, M. From finite automata to regular expressions and back — A summary of descriptional complexity. *Intern. J. Foundations Comput. Sci.,* 26: 1009–1040, 2015.

[14] Han, Y.-S. and Wood, D. Generalizations of 1-deterministic regular languages. *Information and Computation,* 206(9-1): 1117–1125, 2008.

[15] Holzer, M. and Kutrib, M. Nondeterministic finite automata – Recent results on the descriptional and computational complexity. *Intern. J. Foundations Comput. Sci.,* **20**(4): 563–580, 2009.

[16] Holzer, M. and Kutrib M. Descriptional complexity of (un)ambiguous finite state machines and pushdown automata. In: *Reachability Problems 2010,* Lect. Notes Comput. Sci. 6227, pp. 1–23, 2010

[17] Holzer, M. and Kutrib, M. Descriptional and computational complexity of finite automata — A survey. *Information and Computation,* 209: 456–470, 2011.

[18] Hromkovič, J. Descriptional complexity of finite automata: Concepts and open problems. *Journal of Automata, Languages and Combinatorics,* 7(4): 519–531, 2002.

[19] Hromkovič, J., Seibert, S., Karhumäki, J., Klauck, H. and Schnitger, G. Communication complexity method for measuring nondeterminism in finite automata. *Inform. Comput.,* 172: 202–217, 2002.

[20] Hromkovič, J. and Schnitger, G. Ambiguity and communication. *Theory Comput. Syst.,* 48: 517–534, 2011.

[21] Ibarra, O. and Ravikumar, B. On sparseness, ambiguity and other decision problems for acceptors and transducers. In *Proc. of STACS'86,* Lect. Notes Comput. Sci. 210, Springer, pp. 171–179, 1986.

[22] Jirásek, J., Jirásková, G., Šebej, J. Operations on unambiguous finite automata. In *Developments in Language Theory, DLT,* Lect. Notes Comput. Sci. 9840, Springer, pp. 243–255, 2016.

[23] Kappes, M. Descriptional complexity of deterministic finite automata with multiple intial states. *J. Automata, Languages and Combinatorics,* 5: 269–278, 2000.

[24] Keeler, C. *New measures for finite automaton complexity and subregular language hierarchies.* MSc thesis, Queen's University, 2016.

[25] Kintala, C.M.R. and Fischer, P.C. Refining nondeterminism in relativized polynomial time computations. *SIAM J. Comput.,* 9(1): 46–53, 1980.

[26] Kintala, C.M.R and Wotschke, D. Amounts of nondeterminism in finite automata. *Acta Informatica,* 13(2): 199–204, 1980.

[27] Kuich, W. Finite automata and ambiguity. Rept. 253 of the IIG. Techinische Universität Graz, 1988.

[28] Kutrib, M. and Pighizzini, G. Recent trends in descriptional complexity of formal languages. *Bulletin of the EATCS,* 111: 70–86, 2013.

[29] Leiss, E. Succinct representation of regular languages by Boolean automata. *Theoret. Comput. Sci.,* 13: 323–330, 1981.

[30] Leung, H. Separating exponentially ambiguous finite automata from polynomially ambiguous finite automata. *SIAM Journal of Computing,* 27(4): 1073–1082, 1998.

[31] Leung, H. On finite automata with limited nondeterminism. *Acta Informatica,* 35: 595–624, 1998.

[32] Leung, H. Descriptional complexity of NFA of different ambiguity. *Intern. J. Foundations Comput. Sci.,* 16(5): 975–984, 2005.

[33] Mandel, A. and Simon, I. On finite semi-groups of matrices. *Theoret. Comput. Sci.,* 5: 183–204, 1977.

[34] Maslov, A.N. Estimates of the number of states of finite automata. *Soviet Math. Dokl.,* 11: 1373–1375, 1970.

[35] Okhotin, A. Unambiguous finite automata over a unary alphabet. *Inform. Comput.,* 212: 15–36, 2012.

[36] Okhotin, A., Salomaa, K. Descriptional complexity of unambiguous input-driven pushdown automata. *Theoret. Comput. Sci.,* 566: 1–11, 2015.

[37] Palioudakis, A., Han, Y.-S. and Salomaa, K. Growth rate of minimum branching. Submitted for publication, February 2017.

[38] Palioudakis, A., Salomaa, K. and Akl, S.G. State complexity of finite tree width NFAs. *J. Automata, Languages and Combinatorics,* 17(2–4): 245–264, 2012.

[39] Palioudakis, A., Salomaa, K. and Akl, S.G.: Comparisons between measures of nondeterminism for finite automata. In *Proc. of DCFS'13,* Lect. Notes Comput. Sci. 8031, Springer, pp. 217–228, 2013.

[40] Palioudakis, A., Salomaa, K. and Akl, S.G. Lower bound for converting an NFA with finite nondeterminism into an MDFA. *J. Automata, Languages and Combinatorics,* 19: 251–264, 2014.

[41] Palioudakis, A., Salomaa, K. and Akl, S.G. Quantifying nondeterminism in finite automata. *Annals of the University of Bucharest,* No. 2, pp. 89–100, 2015.

[42] Ravikumar, B. and Ibarra, O.H. Relating the type of ambiguity of finite automata to the succinctness of their representation. *SIAM Journal of Computing,* 18(6): 1263–1282, 1989.

[43] Reutenauer, C. Propriétés arithmétiques et topologiques de séries rationnelles en variables non commutatives. *Thése troisiéme cycle,* Université Paris VI, 1977.

[44] Rozenberg, G., Salomaa, A. (Eds.) *Handbook of Formal Languages,* Vol. I–III, Springer-Verlag, 1997.

[45] Salomaa, A., Salomaa, K., and Yu, S. Undecidability of state complexity. *Internat. J. Comput. Math.,* 90: 1310–1320, 2013.

[46] Schmidt, E.M. *Succinctness of descriptions of context-free, regular and finite languages.* PhD thesis, Cornell University, Ithaca, NY, 1978

[47] Shallit, J. *A Second Course in Formal Languages and Automata Theory,* Cambridge University Press, 2009.

[48] Simon, I. The nondeterministic complexity of a finite automaton. In: Lothaire, M. (ed.) *Mots - mélanges offerts a M.P. Schützenberger,* Paris, Hermes, pp. 384–400, 1990.

[49] Stearns, R. and Hunt III, H. On the equivalence and containment problems for unambiguous regular expressions, regular grammars and finite automata. *SIAM J. Comput.,* 14: 598–611, 1985.

[50] Weber, A. and Seidl, H. On the degree of ambiguity of finite automata. *Theoret. Comput. Sci.,* 88: 325–349, 1991.

[51] Yu, S. Regular languages, in [44], Vol. I, pp. 41–110, 1997.

[52] Yu, S. State complexity of regular languages. *Journal of Automata, Languages and Combinatorics,* 6(2): 221–234, 2001.

Acta Cybernetica **23** (2017) 159–174.

On DR Tree Automata, Unary Algebras and Syntactic Path Monoids

Magnus Steinby[a]

To the memory of Zoltán Ésik

Abstract

We consider deterministic root-to-frontier (DR) tree recognizers and the tree languages recognized by them from an algebraic point of view. We make use of a correspondence between DR algebras and unary algebras shown by Z. Ésik (1986). We also study a question raised by F. Gécseg (2007) that concerns the definability of families of DR-recognizable tree languages by syntactic path monoids. We show how the families of DR-recognizable tree languages path-definable by a variety of finite monoids (or semigroups) can be derived from varieties of string languages. In particular, the three path-definable families of Gécseg and B. Imreh (2002, 2004) are obtained this way.

Keywords: deterministic root-to-frontier tree automaton, tree language, unary algebra, syntactic path monoid, variety of finite monoids, variety of languages

1 Introduction

The tree languages recognized by deterministic root-to-frontier (top-down) tree recognizers form a proper subfamily *DRec* of the family *Rec* of all recognizable tree languages. The members of *DRec*, the DR-recognizable tree languages, are characterized by the fact that they are completely determined by the labeled paths appearing in their trees (cf. [11, 15, 16, 20]). Any path from the root of a tree to one of its leaves is represented as a word over the so-called path alphabet. Each symbol of this alphabet indicates both the label of a node of a tree and the direction taken at that node. If we group together the paths leading to a leaf labeled with a given symbol x of the leaf alphabet X, then all the paths appearing in the trees of a given tree language T form a family $\langle T_x \rangle_{x \in X}$ of languages over the path alphabet, and if T is DR-recognizable, it is completely determined by these languages T_x. This implies that the DR-recognizable tree languages resemble string languages more than general recognizable tree languages do. In particular, while few known

[a]Department of Mathematics and Statistics, University of Turku, FI-20014 Turku, Finland, E-mail: steinby@utu.fi

DOI: 10.14232/actacyb.23.1.2017.10

families of recognizable tree languages can be characterized by syntactic monoids or semigroups, Gécseg and Imreh [7, 8] could characterize three subfamilies of $DRec$, those of DR nilpotent, DR monotone and DR definite tree languages, in terms of the syntactic path monoids or semigroups introduced in [12]. We shall show that there exist many more such examples: any $*$- or $+$-variety of string languages, as defined by Eilenberg [2], yields a subfamily of $DRec$ that can be characterized by syntactic path monoids or semigroups.

If we regard the path alphabet as a unary ranked alphabet, then the path set of a tree language T becomes a unary tree language $\delta(T)$ that carries the same information as the family $\langle T_x \rangle_{x \in X}$. A DR recognizer may be seen as a finite DR algebra equipped with an initial state and a final state assignment. In [3] Zoltán Ésik associated with each DR algebra a unary algebra over the path alphabet, and noted that using this association one may apply ideas of standard general algebra to DR algebras. We complete the bijection between the two types of algebras by the converse transformation from unary algebras to DR algebras. The usefulness of this bijection derives from the fact that it preserves subalgebras, homomorphisms, congruences and direct products. In particular, we may refer to varieties of finite unary algebras when considering varieties of finite DR algebras. By extending this correspondence to tree recognizers, we study the connections between DR-recognizable tree languages and their unary path languages.

We shall recall or introduce all the special concepts used here. The basic universal algebra needed can be found in the first two chapters of [1], for example. For tree automata and tree languages, the reader may consult [11] and for the theory of varieties of (string) languages the books [2] and [17].

This paper is dedicated to the memory of Zoltán Ésik whom I learned know already in the 1970s. He has made many important contributions in several areas of theoretical computer science, and all his work is characterized by mathematical elegance and precision. It is a pleasure to acknowledge the inspiration I got from one of his, probably less well known, papers.

2 Preliminaries

For any integer $n > 0$, let $[n] = \{1, \ldots, n\}$. Let A be any set. For any $i \in [n]$, let $\pi_i : A^n \to A, (a_1, \ldots, a_n) \mapsto a_i$ be the i^{th} projection map. The power-set of A is denoted by $\wp(A)$. If $\varphi : A \to B$ is a mapping, the image $\varphi(a)$ of an element $a \in A$ may be denoted also by $a\varphi$. Especially homomorphisms will be written this way as right operators. For any equivalence $\theta \in \mathrm{Eq}(A)$ on A, we write $a\theta$ for the θ-class of an element $a \in A$, and $A/\theta := \{a\theta \mid a \in A\}$ is the quotient set. For any alphabet X, the set of (finite) words over X is denoted by X^* and the empty word by ε.

Let Σ be a *ranked alphabet*, i.e., a finite set of operation symbols, which does not contain nullary symbols. For each $m \geq 1$, Σ_m denotes the set of m-ary symbols in Σ. The *rank type* of Σ is the set $r(\Sigma) := \{m \mid \Sigma_m \neq \emptyset\}$. In what follows, Σ is always a ranked alphabet of rank type R and X is an ordinary finite non-empty alphabet, called a *leaf alphabet*, disjoint from Σ. The set $T_\Sigma(X)$ of ΣX-*trees* is the least set such that $X \subseteq T_\Sigma(X)$, and $f(t_1,\dots,t_m) \in T_\Sigma(X)$, for all $m \in R$, $f \in \Sigma_m$ and $t_1,\dots,t_m \in T_\Sigma(X)$. A ΣX*-tree language* is any subset of $T_\Sigma(X)$. Often we speak about *trees* and *tree languages* without specifying the alphabets.

A Σ*-algebra* $\mathcal{A}$ consists of a nonempty set A and a Σ-indexed family of operations on A such that if $f \in \Sigma_m$ $(m \in R)$, then $f^{\mathcal{A}} : A^m \to A$ is an m-ary operation. We write $\mathcal{A} = (A,\Sigma)$, and call $\mathcal{A}$ *finite* if A is a finite set. Subalgebras, homomorphisms, congruences etc. are defined as usual. For any class $\mathbf{K}$ of Σ-algebras, let $S(\mathbf{K})$ consist of all algebras isomorphic to a subalgebra of a member of $\mathbf{K}$, $H(\mathbf{K})$ be the class of all images of members of $\mathbf{K}$, and $P_f(\mathbf{K})$ be the class of algebras isomorphic to a finite direct product of members of $\mathbf{K}$. A class $\mathbf{K}$ of finite Σ-algebras is a *variety of finite* Σ*-algebras* (a Σ-VFA for short) if $S(\mathbf{K})$, $H(\mathbf{K})$, $P_f(\mathbf{K}) \subseteq \mathbf{K}$. The Σ*-VFA generated* by a class $\mathbf{K}$ of finite Σ-algebras is denoted by $V_f(\mathbf{K})$.

The ΣX*-term algebra* $\mathcal{T}_\Sigma(X) = (T_\Sigma(X),\Sigma)$ is defined by setting, for all $m \in R, f \in \Sigma_m$ and $t_1,\dots,t_m \in T_\Sigma(X)$, $f^{\mathcal{T}_\Sigma(X)}(t_1,\dots,t_m) = f(t_1,\dots,t_m)$.

A ΣX*-recognizer* $\mathbf{A} = (\mathcal{A},\alpha,F)$ consists of a finite Σ-algebra $\mathcal{A} = (A,\Sigma)$, an *initial assignment* $\alpha : X \to A$, and a set $F \subseteq A$ of *final states*. The ΣX-tree language *recognized* by $\mathbf{A}$ is the set $T(\mathbf{A}) := \{t \in T_\Sigma(X) \mid t\alpha_{\mathcal{A}} \in F\}$ where $\alpha_{\mathcal{A}} : \mathcal{T}_\Sigma(X) \to \mathcal{A}$ is the homomorphic extension of α. A ΣX-tree language is called *recognizable*, or *regular*, if it is recognized by a ΣX-recognizer. Let $Rec(\Sigma, X)$ denote the set of all recognizable ΣX-tree languages.

If $\Sigma = \Sigma_1$, we call Σ a *unary alphabet* and Σ-algebras are *unary algebras.* A unary alphabet Σ may also be treated as an ordinary alphabet and we write Σ-terms as expressions ξu, where ξ is a variable and $u \in \Sigma^*$. The *term functions* induced by such terms ξu in a Σ-algebra $\mathcal{C} = (C,\Sigma)$ are mappings $u^{\mathcal{C}} : C \to C, c \mapsto cu^{\mathcal{C}}$, defined by $c\varepsilon^{\mathcal{C}} = c$ and $cu^{\mathcal{C}} = g^{\mathcal{C}}(cv^{\mathcal{C}})$ for $u = vg$ $(v \in \Sigma^*,\ g \in \Sigma)$.

On the other hand, we may view an ordinary (finite) alphabet Z as a unary ranked alphabet and define Z*-automata* as unary algebras $\mathcal{A} = (A,Z)$ in which each letter $z \in Z$ induces a unary operation $z^{\mathcal{A}} : A \to A$, $a \mapsto az^{\mathcal{A}}$. For any word $w = z_1 \dots z_k$ in Z^* $(k \geq 0, z_1,\dots,z_k \in Z)$, the operation $w^{\mathcal{A}} : A \to A$ is the composition of $z_1^{\mathcal{A}},\dots,z_k^{\mathcal{A}}$. A Z*-recognizer* is now a system $\mathbf{A} = (\mathcal{A}, a_0, F)$, where $\mathcal{A} = (A,Z)$ is a finite Z-algebra, $a_0 \in A$ is the initial state, and $F \subseteq A$ is the set of final states. The *language recognized* by $\mathbf{A}$ is the set $L(\mathbf{A}) := \{w \in Z^* \mid a_0 w^{\mathcal{A}} \in F\}$. A language $L \subseteq Z^*$ is *regular* if it is recognized by a Z-recognizer.

Often we will say that a string or tree language is *recognized by an algebra* $\mathcal{A}$ if it is recognized by a recognizer of an appropriate kind based on $\mathcal{A}$.

3 DR algebras and unary algebras

In what follows, the frequently recurring phrase *deterministic root-to-frontier* is abbreviated to DR. The following basic algebraic notions for DR algebras were defined by Virágh [20], but most of them are obtained from those defined for DR tree recognizers in [10]. To simplify the notation, we extend mappings and equivalence relations to m-tuples: if $\mathbf{a} = (a_1,\dots,a_m) \in A^m$, then for any map $\varphi : A \to B$, let $\mathbf{a}\bar{\varphi} := (a_1\varphi,\dots,a_m\varphi)$, and for any $\theta \in \mathrm{Eq}(A)$, let $\mathbf{a}\bar{\theta} := (a_1\theta,\dots,a_m\theta)$.

A (*finite*) *DR* Σ*-algebra* consists of a nonempty (finite) set A and a Σ-indexed family of *root-to-frontier operations* $f_{\mathcal{A}} : A \longrightarrow A^m \quad (f \in \Sigma)$, where the arity m is that of $f (\in \Sigma_m)$. Again we write simply $\mathcal{A} = (A, \Sigma)$.

Let $\mathcal{A} = (A, \Sigma)$ and $\mathcal{B} = (B, \Sigma)$ be any DR Σ-algebras. Then $\mathcal{A}$ is a *subalgebra* of $\mathcal{B}$ if $A \subseteq B$ and $f_{\mathcal{A}}(a) = f_{\mathcal{B}}(a)$ for all $f \in \Sigma$ and $a \in A$. A mapping $\varphi : A \to B$ is a *homomorphism* from $\mathcal{A}$ to $\mathcal{B}$, and we write $\varphi : \mathcal{A} \to \mathcal{B}$, if $f_{\mathcal{A}}(a)\bar{\varphi} = f_{\mathcal{B}}(a\varphi)$ for all $f \in \Sigma$ and $a \in A$. If φ is also bijective, it is an *isomorphism*, and we write $\mathcal{A} \cong \mathcal{B}$ if $\mathcal{A}$ and $\mathcal{B}$ are isomorphic. The *direct product* of $\mathcal{A}$ and $\mathcal{B}$ is the DR Σ-algebra $\mathcal{A} \times \mathcal{B} = (A \times B, \Sigma)$ such that for all $m \in R$, $f \in \Sigma_m$ and $(a, b) \in A \times B$, if $f_{\mathcal{A}}(a) = (a_1, \ldots, a_m)$ and $f_{\mathcal{A}}(b) = (b_1, \ldots, b_m)$, then $f_{\mathcal{A} \times \mathcal{B}}((a, b)) = ((a_1, b_1), \ldots, (a_m, b_m))$. The general finite direct product $\mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ $(n \geq 0)$ is defined the same way.

A *congruence* on $\mathcal{A}$ is an equivalence θ on A such that for any $a, a' \in A$ and $f \in \Sigma$, if $a\theta a'$, then $f_{\mathcal{A}}(a)\bar{\theta} = f_{\mathcal{A}}(a')\bar{\theta}$. Let $\mathrm{Con}(\mathcal{A})$ denote the set of all congruences on $\mathcal{A}$. For any $\theta \in \mathrm{Con}(\mathcal{A})$, the *quotient DR algebra* $\mathcal{A}/\theta = (A/\theta, \Sigma)$ is defined by $f_{\mathcal{A}/\theta}(a\theta) := f_{\mathcal{A}}(a)\bar{\theta}$ for all $a \in A$ and $f \in \Sigma$.

All the usual facts about subalgebras, homomorphisms, congruences, etc. hold for DR algebras, too. For example, the kernel of any homomorphism $\varphi : \mathcal{A} \to \mathcal{B}$ is a congruence on $\mathcal{A}$, and $\mathcal{A}/\ker\varphi \cong \mathcal{B}$ if φ is surjective.

The tree languages recognized by deterministic root-to-frontier recognizers are characterized by the labeled paths appearing in their trees. The paths are described using the *path alphabet* $\widehat{\Sigma} := \bigcup_{m \in R} \Sigma_m \times [m]$. A pair $(f, i) \in \widehat{\Sigma}$ is written simply as f_i. We regard $\widehat{\Sigma}$ either as a unary ranked alphabet or as an ordinary alphabet.

Following Ésik [3] we associate with any DR Σ-algebra $\mathcal{A} = (A, \Sigma)$ a unary algebra $\mathcal{A}^u = (A, \widehat{\Sigma})$ such that $f_i^{\mathcal{A}^u}(a) = f_{\mathcal{A}}(a)\pi_i$ for all $a \in A$, $m \in R$, $f \in \Sigma_m$ and $i \in [m]$. Let us also introduce a converse transformation: for any $\widehat{\Sigma}$-algebra $\mathcal{C} = (C, \widehat{\Sigma})$, let $\mathcal{C}^d = (C, \Sigma)$ be the DR Σ-algebra with $f_{\mathcal{C}^d}(c) = (f_1^{\mathcal{C}}(c), \ldots, f_m^{\mathcal{C}}(c))$ for all $c \in C$, $m \in R$ and $f \in \Sigma_m$. Since $\mathcal{A}^{ud} = \mathcal{A}$ for any DR Σ-algebra $\mathcal{A}$ and $\mathcal{C}^{du} = \mathcal{C}$ for any $\widehat{\Sigma}$-algebra $\mathcal{C}$, there is a natural bijective correspondence between DR Σ-algebras and $\widehat{\Sigma}$-algebras.

Lemma 3.1. *Let $\mathcal{A} = (A, \Sigma)$ and $\mathcal{B} = (B, \Sigma)$ be any DR Σ-algebras.*

(a) *$\mathcal{A}$ is a subalgebra of $\mathcal{B}$ if and only if $\mathcal{A}^u$ is a subalgebra of $\mathcal{B}^u$.*

(b) *A mapping $\varphi : A \to B$ is a homomorphism from $\mathcal{A}$ to $\mathcal{B}$ if and only if it is a homomorphism from $\mathcal{A}^u$ to $\mathcal{B}^u$.*

(c) $(\mathcal{A} \times \mathcal{B})^u = \mathcal{A}^u \times \mathcal{B}^u$.

(d) $\mathrm{Con}(\mathcal{A}) = \mathrm{Con}(\mathcal{A}^u)$.

(e) $(\mathcal{A}/\theta)^u = \mathcal{A}^u/\theta$ *for any* $\theta \in \mathrm{Con}(\mathcal{A})$.

Proof. All five statements follow directly from the appropriate definitions, and (c) was noted already in [3]. Let us verify (e) as an example.

Firstly, $(\mathcal{A}/\theta)^u$ and $\mathcal{A}^u/\theta$ are $\widehat{\Sigma}$-algebras with the same set A/θ of elements. Moreover, for any $a \in A$, $m \in R$, $f \in \Sigma_m$ and $i \in [m]$,

$$f_i^{(\mathcal{A}/\theta)^u}(a\theta) = f_{\mathcal{A}/\theta}(a\theta)\pi_i = f_{\mathcal{A}}(a)\bar{\theta}\pi_i = f_{\mathcal{A}}(a)\pi_i\theta = f_i^{\mathcal{A}^u}(a)\theta = f_i^{\mathcal{A}^u/\theta}(a\theta),$$

so also their operations are the same. □

4 DR tree recognizers and unary recognizers

Let us now extend the correspondence between DR algebras and unary algebras to recognizers. A *DR* ΣX*-recognizer* $\mathbf{A} = (\mathcal{A}, a_0, \alpha)$ consists of a finite DR Σ-algebra $\mathcal{A} = (A, \Sigma)$, an *initial state* $a_0 \in A$, and a *final state assignment* $\alpha : X \to \wp(A)$. To accept or reject an input tree $t \in T_\Sigma(X)$, $\mathbf{A}$ starts at the root of t in state a_0, and if it has reached a node ν of t labeled with $f \in \Sigma_m$ in state a and $f_\mathcal{A}(a) = (a_1, \ldots, a_m)$, then it continues its working at the i^{th} immediate successor node of ν in state a_i ($i \in [m]$). The tree is accepted if $\mathbf{A}$ reaches each leaf in a state $a\,(\in A)$ matching the label $x\,(\in X)$ of that leaf, i.e., $a \in \alpha(x)$. For a formal definition, we extend α to a mapping $\widetilde{\alpha} : T_\Sigma(X) \to \wp(A)$ by setting $\widetilde{\alpha}(x) = \alpha(x)$ for each $x \in X$, and $\widetilde{\alpha}(t) = \{a \in A \mid f_\mathcal{A}(a) \in \widetilde{\alpha}(t_1) \times \ldots \times \widetilde{\alpha}(t_m)\}$ for $t = f(t_1, \ldots, t_m)$. Then $T(\mathbf{A}) := \{t \in T_\Sigma(X) \mid a_0 \in \widetilde{\alpha}(t)\}$ is the tree language *recognized* by $\mathbf{A}$, and the ΣX-tree language $T(\mathbf{A})$ is said to be *DR-recognizable.* Let $DRec(\Sigma, X)$ denote the set of DR-recognizable ΣX-tree languages. Two DR ΣX-recognizers $\mathbf{A}$ and $\mathbf{B}$ are *equivalent* if $T(\mathbf{A}) = T(\mathbf{B})$.

The set $\delta(t) \subseteq T_{\widehat{\Sigma}}(X)$ of *paths* in a ΣX-tree t is defined by $\delta(x) = \{x\}$ for $x \in X$, and $\delta(t) = f_1\delta(t_1) \cup \ldots \cup f_m\delta(t_m)$ for $t = f(t_1, \ldots, t_m)$. Thus $\delta(t)$ is a set of unary trees in Polish form. The *path language* of a ΣX-tree language T is the set $\delta(T) := \bigcup\{\delta(t) \mid t \in T\}$. The *path closure* $\Delta(T) := \delta^{-1}(\delta(T))$ of $T \subseteq T_\Sigma(X)$ consists of all ΣX-trees t such that $\delta(t) \subseteq \delta(T)$, and T is *path closed* if $T = \Delta(T)$. Quite generally, for any $U \subseteq T_{\widehat{\Sigma}}(X)$, the set $\delta^{-1}(U) := \{t \in T_\Sigma(X) \mid \delta(t) \subseteq U\}$ is path-closed. As shown in [16], a regular tree language is DR-recognizable if and only if it is path closed. For properties of the operators δ and Δ, cf. [15, 20].

Remark 4.1. Let Σ be unary. Then $\widehat{\Sigma} = \{f_1 \mid f \in \Sigma\}$ and we may use Σ itself as the path alphabet. Furthermore, we may regard any DR Σ-algebra $\mathcal{A} = (A, \Sigma)$ also as a Σ-algebra by identifying any 1-tuple (a) with the element $a(\in A)$, but as a DR ΣX-recognizer and as a ΣX-recognizer $\mathcal{A}$ reads the input trees in opposite directions. Nevertheless, it is clear that $DRec(\Sigma, X) = Rec(\Sigma, X)$ for every X.

Let us now regard $\widehat{\Sigma}$ as a usual alphabet. For each $x \in X$, the set of x*-paths* in a ΣX-tree t is $g_x(t) := \{u \in \widehat{\Sigma}^* \mid ux \in \delta(t)\}$. For any $T \subseteq T_\Sigma(X)$ and $x \in X$, let T_x denote the set $\bigcup\{g_x(t) \mid t \in T\}$ of x-paths appearing in T. Obviously, $\delta(T)$ can be recovered from the family $\langle T_x \rangle_{x \in X}$.

Next we recall a few notions from [10, 11]. Let $\mathbf{A} = (\mathcal{A}, a_0, \alpha)$ be a DR ΣX-recognizer and $\mathcal{A} = (A, \Sigma)$. For any $a \in A$, let $\mathbf{A}_a := (\mathcal{A}, a, \alpha)$. A state a is a 0*-state* if $T(\mathbf{A}_a) = \emptyset$, and it is *reachable* if $a_0 \Rightarrow_\mathcal{A}^* a$ for the reflexive transitive closure $\Rightarrow_\mathcal{A}^*$ of the relation $\Rightarrow_\mathcal{A} \subseteq A \times A$, where for any $a, b \in A$, $a \Rightarrow_\mathcal{A} b$ if and only if $b = f_\mathcal{A}(a)\pi_i$ for some $m \in R$, $f \in \Sigma_m$ and $i \in [m]$. The recognizer $\mathbf{A}$ is *normalized*, if for all $m \in R$, $f \in \Sigma_m$ and $a \in A$, either every component in $f_\mathcal{A}(a) = (a_1, \ldots, a_m)$ is a 0-state or no a_i is a 0-state, *reduced* if $T(\mathbf{A}_a) = T(\mathbf{A}_b)$ implies $a = b$ $(a, b \in A)$, *connected* if all of its states are reachable, and it is *minimal* if it is connected and reduced. In [10] it was shown that any DR ΣX-recognizer can be converted into an equivalent normalized minimal DR ΣX-recognizer, and this is also minimal with respect to the number of states and unique up to isomorphism (the isomorphism of DR ΣX-recognizers is defined in the natural way, cf. [10]).

Let us associate with any DR ΣX-recognizer $\mathbf{A} = (\mathcal{A}, a_0, \alpha)$ the DR $\widehat{\Sigma}X$-recognizer $\mathbf{A}^u = (\mathcal{A}^u, a_0, \alpha)$, and with any DR $\widehat{\Sigma}X$-recognizer $\mathbf{C} = (\mathcal{C}, c_0, \gamma)$ the DR ΣX-recognizer $\mathbf{C}^d = (\mathcal{C}^d, c_0, \gamma)$. Obviously, $\mathbf{A}^{ud} = \mathbf{A}$ and $\mathbf{C}^{du} = \mathbf{C}$.

Proposition 4.1. *$T(\mathbf{A}) = \delta^{-1}(T(\mathbf{A}^u))$ for any DR ΣX-tree recognizer $\mathbf{A}$.*

Proof. Let $\mathbf{A} = (\mathcal{A}, a_0, \alpha)$ with $\mathcal{A} = (A, \Sigma)$. We show by induction on t that for all $t \in T_\Sigma(X)$ and $a \in A$, $t \in T(\mathbf{A}_a)$ if and only if $t \in \delta^{-1}(T(\mathbf{A}^u_a))$. The case $t \in X$ is obvious, so let $t = f(t_1, \ldots, t_m)$. If $f_\mathcal{A}(a) = (a_1, \ldots, a_m)$, then

$$\begin{aligned} t \in T(\mathbf{A}_a) \quad & \text{iff} \quad t_1 \in T(\mathbf{A}_{a_1}), \ldots, t_m \in T(\mathbf{A}_{a_m}) \\ & \text{iff} \quad \delta(t_1) \subseteq T(\mathbf{A}^u_{a_1}), \ldots, \delta(t_m) \subseteq T(\mathbf{A}^u_{a_m}) \\ & \text{iff} \quad f_1\delta(t_1), \ldots, f_m\delta(t_m) \subseteq T(\mathbf{A}^u_a) \\ & \text{iff} \quad \delta(t) \subseteq T(\mathbf{A}^u_a) \\ & \text{iff} \quad t \in \delta^{-1}(T(\mathbf{A}^u_a)). \end{aligned}$$

Hence, $T(\mathbf{A}_a) = \delta^{-1}(T(\mathbf{A}^u_a))$ and, in particular, $T(\mathbf{A}) = \delta^{-1}(T(\mathbf{A}^u))$. □

Corollary 4.1. *$T(\mathbf{C}^d) = \delta^{-1}(T(\mathbf{C}))$ for any DR $\widehat{\Sigma}X$-recognizer $\mathbf{C}$.*

Proof. $T(\mathbf{C}^d) = \delta^{-1}(T(\mathbf{C}^{du})) = \delta^{-1}(T(\mathbf{C}))$ by Proposition 4.1 and $\mathbf{C}^{du} = \mathbf{C}$. □

Proposition 4.2. *$T(\mathbf{A}^u) = \delta(T(\mathbf{A}))$ for any normalized DR ΣX-recognizer $\mathbf{A}$.*

Proof. Let $\mathbf{A} = (\mathcal{A}, a_0, \alpha)$ with $\mathcal{A} = (A, \Sigma)$. The inclusion $\delta(T(\mathbf{A})) \subseteq T(\mathbf{A}^u)$ follows from Proposition 4.1 and the fact that $\delta(\delta^{-1}(U)) \subseteq U$ for any $U \subseteq T_{\widehat{\Sigma}}(X)$.

For the converse inclusion we need the assumption that $\mathbf{A}$ is normalized. It is enough to show that for all $a \in A$ and $r \in T_{\widehat{\Sigma}}(X)$, if $r \in T(\mathbf{A}^u_a)$, then $r \in \delta(t)$ for some $t \in T(\mathbf{A}_a)$. This we do by induction on r. For $r \in X$, we may let $t := r$. Next, let $r = f_i s$ for some $f \in \Sigma_m$, $m \in R$, $i \in [m]$ and $s \in T_{\widehat{\Sigma}}(X)$, and assume that the claim holds for s. If $f_\mathcal{A}(a) = (a_1, \ldots, a_m)$, then $s \in T(\mathbf{A}^u_{a_i})$ implies that $s \in \delta(t_i)$ for some $t_i \in T(\mathbf{A}_{a_i})$. Since a_i is not a 0-state, there is for every $j \in [m], j \neq i$ a tree $t_j \in T(\mathbf{A}_{a_j})$. Clearly, $t := f(t_1, \ldots, t_m) \in T(\mathbf{A}_a)$ and $r \in \delta(t)$. □

The following fact appears, in a different form, already in [16].

Corollary 4.2. *If $T \in DRec(\Sigma, X)$, then $\delta(T) \in DRec(\widehat{\Sigma}, X)$.*

Proposition 4.3. *A normalized DR ΣX-recognizer $\mathbf{A}$ is minimal if and only if $\mathbf{A}^u$ is a minimal DR $\widehat{\Sigma}X$-recognizer of $\delta(T(\mathbf{A}))$.*

Proof. Let $\mathbf{A} = (\mathcal{A}, a_0, \alpha)$ with $\mathcal{A} = (A, \Sigma)$. Consider any two states $a, b \in A$. If $T(\mathbf{A}_a) = T(\mathbf{A}_b)$, then $T(\mathbf{A}^u_a) = \delta(T(\mathbf{A}_a)) = \delta(T(\mathbf{A}_b)) = T(\mathbf{A}^u_b)$ by Proposition 4.2. On the other hand, if $T(\mathbf{A}^u_a) = T(\mathbf{A}^u_b)$, then Proposition 4.2 yields $\delta(T(\mathbf{A}_a)) =$

$\delta(T(\mathbf{A}_b))$. Since $T(\mathbf{A}_a)$ and $T(\mathbf{A}_b)$ are path-closed, this means that $T(\mathbf{A}_a) = T(\mathbf{A}_b)$. Hence, $\mathbf{A}$ is reduced if and only if $\mathbf{A}^u$ is reduced.

It is obvious that the reachability relations $\Rightarrow^*_{\mathcal{A}}$ and $\Rightarrow^*_{\mathcal{A}^u}$ of $\mathbf{A}$ and $\mathbf{A}^u$ are identical. Hence, $\mathbf{A}$ is connected if and only if $\mathbf{A}^u$ is connected. □

Next we show how a DR Σ-algebra recognizing a ΣX-tree language T yields a DR $\widehat{\Sigma}$-algebra recognizing the $\widehat{\Sigma}$-languages T_x $(x \in X)$, and how a DR Σ-algebra recognizing T is obtained from DR $\widehat{\Sigma}$-algebras recognizing the $\widehat{\Sigma}$-languages T_x.

Proposition 4.4. *Let T be a DR-recognizable ΣX-tree language.*

(a) *If a finite DR Σ-algebra $\mathcal{A} = (A, \Sigma)$ recognizes T, then $\mathcal{A}^u = (A, \widehat{\Sigma})$ recognizes every language T_x $(x \in X)$.*

(b) *For each $x \in X$, let $\mathcal{A}_x = (A_x, \widehat{\Sigma})$ be a finite $\widehat{\Sigma}$-algebra that recognizes T_x. Then the direct product $\prod(\mathcal{A}_x^d \mid x \in X)$ recognizes T.*

Proof. Let $\mathbf{A} = (\mathcal{A}, a_0, \alpha)$ be a DR ΣX-recognizer of T. It is easy to see that, for each $x \in X$, the $\widehat{\Sigma}$-recognizer $\mathbf{A}_x = (\mathcal{A}^u, a_0, \alpha(x))$ recognizes T_x.

To prove (b), consider for each $x \in X$ a $\widehat{\Sigma}$-recognizer $\mathbf{A}_x = (\mathcal{A}_x, a_{x0}, F_x)$ of T_x. The direct product $\mathcal{A} := \prod_{x\in X} \mathcal{A}_x^d$ simulates the computation of $\mathcal{A}_x$ by its x-component along every path of a given tree $t \in T_\Sigma(X)$. Hence, started in state $(a_{x0})_{x\in X}$, $\mathcal{A}$ should accept t if and only if it reaches, for each $y \in X$, every y-labeled leaf in a state $(a_x)_{x\in X}$ such that $a_y \in F_y$. This means that $T = T(\mathbf{A})$ for $\mathbf{A} = (\mathcal{A}, (a_{x0})_{x\in X}, \alpha)$ if we define α by $\alpha(y) = \prod_{x\in X} G_y(x)$, where $G_y(y) = F_y$ and $G_y(x) = A_x$ for all $x \in X, x \neq y$. □

5 Definability by syntactic monoids

Let us first recall (cf. [2, 17]) that the *syntactic congruence* of a string language $L \subseteq Z^*$ is the relation θ_L on Z^* defined by

$$u\,\theta_L\,v \quad \text{iff} \quad (\forall w, w' \in Z^*)(wuw' \in L \leftrightarrow wvw' \in L),$$

and that the *syntactic monoid* of L is the quotient monoid $\mathrm{M}(L) := Z^*/\theta_L$.

Next we define the syntactic monoids and syntactic path monoids of tree languages introduced in [19] and [12], respectively.

Let ξ be a symbol that does not appear in our alphabets Σ or X. A ΣX*-context* is a $\Sigma(X \cup \{\xi\})$-tree in which ξ occurs exactly once. The set of all ΣX-contexts is denoted by $C_\Sigma(X)$. If $p, q \in C_\Sigma(X)$ and $t \in T_\Sigma(X)$, then $p \cdot q = q(p)$ and $t \cdot q = q(t)$ are the ΣX-context and the ΣX-tree obtained by replacing the ξ in q by p or t, respectively. Then $C_\Sigma(X)$ forms for the product $p \cdot q$ a monoid in which ξ is the identity element. If Σ is unary, no X-symbols appear in ΣX-contexts, and hence we write C_Σ for $C_\Sigma(X)$.

The *syntactic monoid congruence* μ_T of a ΣX-tree language T is the relation on $C_\Sigma(X)$ is defined by

$$p\,\mu_T\,q \quad \text{iff} \quad (\forall t \in T_\Sigma(X))(\forall r \in C_\Sigma(X))(t \cdot p \cdot r \in T \;\leftrightarrow\; t \cdot q \cdot r \in T),$$

and the *syntactic monoid* of T is the quotient monoid $\mathrm{SM}(T) := C_\Sigma(X)/\mu_T$. The *syntactic path congruence* $\widehat{\mu}_T$ is the relation on $C_{\widehat{\Sigma}}$ defined by

$$p\,\widehat{\mu}_T\,q \quad \text{iff} \quad (\forall s \in T_{\widehat{\Sigma}}(X))(r \in C_{\widehat{\Sigma}})(s \cdot p \cdot r \in \delta(T) \;\leftrightarrow\; s \cdot q \cdot r \in \delta(T)),$$

and the *syntactic path monoid* of T is the quotient monoid $\mathrm{PM}(T) := C_{\widehat{\Sigma}}/\widehat{\mu}_T$.

In [19] it was shown that T is regular if and only if $\mathrm{SM}(T)$ is finite, and in [12] that a path closed T is DR-recognizable if and only if $\mathrm{PM}(T)$ is finite.

In [12] $\mathrm{PM}(T)$ was defined as the quotient $\widehat{\Sigma}^*/\theta_T$ where θ_T is the intersection of the congruences θ_{T_x} $(x \in X)$. It is easy to see that $C_{\widehat{\Sigma}}/\widehat{\mu}_T \cong \widehat{\Sigma}^*/\theta_T$, and hence the next lemma follows from the fact that $\theta_T = \bigcap\{\theta_{T_x} \mid x \in X\}$. To see this, combine Theorem II.6.2 and Lemma II.8.2 of [1]. In [5] the corresponding fact about transition monoids was used.

Lemma 5.1. *For any $T \in DRec(\Sigma, X)$, $\mathrm{PM}(T)$ is a subdirect product of the monoids $\mathrm{M}(T_x)$ $(x \in X)$.*

If Σ is unary and we use Σ itself as the path alphabet, then $T_{\widehat{\Sigma}}(X) = T_\Sigma(X)$ and $C_{\widehat{\Sigma}} = C_\Sigma$. Moreover, $\delta(U) = U$ for any $U \subseteq T_\Sigma(X)$, and $\widehat{\mu}_U$ and μ_U become identical. Hence, $\mathrm{PM}(U) \cong \mathrm{SM}(U)$ for any unary tree language U. Similarly, for any ranked alphabet Σ, we may use $\widehat{\Sigma}$ as its own path alphabet, and then $\delta(\delta(T)) = \delta(T)$ for any $T \subseteq T_\Sigma(X)$, which implies $\widehat{\mu}_T = \widehat{\mu}_{\delta(T)}$. By combining these observations, we obtain the following result.

Proposition 5.1. $\mathrm{PM}(T) \cong \mathrm{PM}(\delta(T)) \cong \mathrm{SM}(\delta(T))$ *for any ΣX-tree language T.*

In [5] Gécseg poses the following question. Assume that some property $\mathcal{P}$ of regular string languages is determined by a class $\mathbf{M}$ of finite monoids in the sense that a language has property $\mathcal{P}$ if and only if its syntactic monoid is in $\mathbf{M}$. Under what conditions can we conclude that the 'corresponding' property of regular tree languages is similarly determined by $\mathbf{M}$? In [6] the question is also considered for DR tree languages in terms of syntactic path monoids. Let us describe the result of [6] concerning the DR-case.

A class $\mathbf{M}$ of finite monoids is said to be *closed under subdirect products* if any subdirect product of a finite family of members of $\mathbf{M}$ also belongs to $\mathbf{M}$, and it is *closed under subdirect factors* if whenever a subdirect product of a finite family of monoids belongs to $\mathbf{M}$, then all the factors are in $\mathbf{M}$, too[1]. A property $\mathcal{P}$ of DR tree languages is *path-defined* by $\mathbf{M}$ if a DR-recognizable tree language T has property $\mathcal{P}$ if and only if $\mathrm{PM}(T) \in \mathbf{M}$. Somewhat reformulated, Theorem 11 of [6] reads as follows.

Proposition 5.2. (*F. Gécseg 2011*) *Let $\mathcal{P}$ be a property of tree languages that is also defined for string languages, and let $\mathbf{M}$ be a class of finite monoids. Assume that the following three conditions are satisfied.*

(1) *A DR-recognizable ΣX-tree language T has property $\mathcal{P}$ if and only if T_x has property $\mathcal{P}$ for every $x \in X$.*

[1]In [5, 6] this is required of subdirect products of two factors only, but the stronger form is actually used.

(2) *For string languages the property $\mathcal{P}$ is defined by* $\mathbf{M}$.

(3) $\mathbf{M}$ *is closed under subdirect products and subdirect factors.*

Then $\mathcal{P}$ is path-defined for DR-recognizable tree languages by $\mathbf{M}$.

Condition (3) is explained by Lemma 5.1. If $\mathbf{M}$ is a *variety of finite monoids* (VFM), i.e., if $S(\mathbf{M}), H(\mathbf{M}), P_f(\mathbf{M}) \subseteq \mathbf{M}$, condition (3) is always satisfied. Hence, it is redundant when we consider properties that define varieties of string languages (cf. [2, 17]), and this concerns many of the best-known families of regular languages. In what follows, we replace "properties" by families of tree or string languages.

A *family of tree languages (FTL)* $\mathcal{V}$ assigns to all pairs Σ, X a set $\mathcal{V}(\Sigma, X)$ of ΣX-tree languages. We write $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ with the understanding that Σ and X range over the appropriate alphabets. If Σ is allowed to range over the unary ranked alphabets only, then $\mathcal{V}$ is a *family of unary tree languages (FUTL)*. From any FTL $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ we get a FUTL $\mathcal{V}^u = \{\mathcal{V}^u(\Sigma, X)\}$ by restricting the range of Σ to unary alphabets. We call $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ a *DR family of tree languages (DR-FTL)* if $\mathcal{V}(\Sigma, X) \subseteq \mathit{DRec}(\Sigma, X)$ for all Σ and X. Similarly, a FUTL $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ is a *DR-FUTL* if $\mathcal{V}(\Sigma, X) \subseteq \mathit{DRec}(\Sigma, X)$ for every unary Σ and every X.

Let $\mathbf{M}$ be a class of finite monoids. For any Σ and X, let

$$\mathbf{M}^p(\Sigma, X) := \{T \in \mathit{DRec}(\Sigma, X) \mid \mathrm{PM}(T) \in \mathbf{M}\}.$$

Then $\mathbf{M}^p = \{\mathbf{M}^p(\Sigma, X)\}$ is the *DR-FTL path-defined* by $\mathbf{M}$, and $\mathbf{M}^u := (\mathbf{M}^p)^u$ is the *DR-FUTL path-defined* by $\mathbf{M}$.

Note that owing to Proposition 5.1, the third condition could be dropped in the following variant of Proposition 5.2.

Proposition 5.3. *Let $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ be a DR-FTL, and let $\mathbf{M}$ be a class of finite monoids. If*

(1) $\mathcal{V}^u = \mathbf{M}^u$, *and*

(2) $T \in \mathcal{V}(\Sigma, X)$ *if and only if* $\delta(T) \in \mathcal{V}^u(\widehat{\Sigma}, X)$ *for all* Σ, X *and* $T \subseteq T_\Sigma(X)$,

then $\mathcal{V} = \mathbf{M}^p$.

Proof. Consider any Σ and X. For every $T \in \mathit{DRec}(\Sigma, X)$,

$$\begin{aligned} T \in \mathbf{M}^p(\Sigma, X) \quad & \text{iff} \quad \mathrm{PM}(T) \in \mathbf{M} \quad \text{iff} \quad \mathrm{PM}(\delta(T)) \in \mathbf{M} \\ & \text{iff} \quad \delta(T) \in \mathbf{M}^u(\widehat{\Sigma}, X) \quad \text{iff} \quad \delta(T) \in \mathcal{V}^u(\widehat{\Sigma}, X) \\ & \text{iff} \quad T \in \mathcal{V}(\Sigma, X), \end{aligned}$$

where we used the definition of $\mathbf{M}^p$, Proposition 5.1, the definition of $\mathbf{M}^u$, assumption (1), and finally assumption (2). Hence $\mathcal{V} = \mathbf{M}^p$. □

Let us now show a way to get all DR-FTLs path-defined by a VFM. For this, recall that Eilenberg [2] defines a $*$-*variety* as a family of languages with certain closure properties and shows that every $*$-variety $\mathcal{L} = \{\mathcal{L}(Z)\}$ is defined by a unique

VFM $\mathbf{M}$ in the sense that for any $L \subseteq Z^*$, $L \in \mathcal{L}(Z)$ if and only if $\mathrm{M}(L) \in \mathbf{M}$. For any $*$-variety $\mathcal{L}$, let $\mathcal{V}_\mathcal{L} = \{\mathcal{V}_\mathcal{L}(\Sigma, X)\}$ be the FTL where

$$\mathcal{V}_\mathcal{L}(\Sigma, X) = \{T \in DRec(\Sigma, X) \mid (\forall x \in X)\, T_x \in \mathcal{L}(\widehat{\Sigma})\},$$

for all Σ and X.

Proposition 5.4. *If $\mathcal{L} = \{\mathcal{L}(Z)\}$ is a $*$-variety and $\mathbf{M}$ is the corresponding VFM, then $\mathcal{V}_\mathcal{L} = \{\mathcal{V}_\mathcal{L}(\Sigma, X)\}$ is a DR-FTL path-defined by $\mathbf{M}$.*

Proof. Let $T \in DRec(\Sigma, X)$. Since $\mathrm{PM}(T)$ is a subdirect product of the monoids $\mathrm{M}(T_x)$ $(x \in X)$ and $\mathbf{M}$ is a VFM, we conclude that $\mathrm{PM}(T) \in \mathbf{M}$ if and only if $\mathrm{M}(T_x) \in \mathbf{M}$ for every $x \in X$. Because $T \in \mathcal{V}_\mathcal{L}(\Sigma, X)$ if and only if $T_x \in \mathcal{L}(\widehat{\Sigma})$ for every $x \in X$, and $T_x \in \mathcal{L}(\widehat{\Sigma})$ if and only if $\mathrm{M}(T_x) \in \mathbf{M}$ $(x \in X)$, this implies that $T \in \mathcal{V}_\mathcal{L}(\Sigma, X)$ if and only if $\mathrm{PM}(T) \in \mathbf{M}$. Hence, $\mathcal{V}_\mathcal{L} = \mathbf{M}^p$. □

Next we show that every DR-FTL path-defined by a VFM is obtained this way.

Proposition 5.5. *If a DR-FTL $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ is path-defined by a VFM $\mathbf{M}$ and $\mathcal{L} = \{\mathcal{L}(Z)\}$ is the $*$-variety defined by $\mathbf{M}$, then $\mathcal{V} = \mathcal{V}_\mathcal{L}$.*

Proof. Consider any Σ and X. For every $T \in DRec(\Sigma, X)$,

$$\begin{aligned} T \in \mathcal{V}(\Sigma, X) \quad &\text{iff} \quad \mathrm{PM}(T) \in \mathbf{M} \quad \text{iff} \quad \mathrm{PM}(T_x) \in \mathbf{M} \text{ for every } x \in X \\ &\text{iff} \quad T_x \in \mathcal{L}(\widehat{\Sigma}) \text{ for every } x \in X \quad \text{iff} \quad T \in \mathcal{V}_\mathcal{L}(\Sigma, X), \end{aligned}$$

where we used the assumptions of the proposition and Lemma 5.1. □

Many families of regular languages are characterized by syntactic semigroups rather than by syntactic monoids, and in [6] Gécseg gives the corresponing versions of his results. Let us modify our Propositions 5.3, 5.4 and 5.5 in the same manner.

Firstly, all languages to be considered are ε-free, and $*$-varieties are replaced by *$+$-varieties* and VFMs by *varieties of finite semigroups* (VFSs). Also, instead of the syntactic monoid $\mathrm{M}(L)$ of a language $L \subseteq Z^+ (:= Z^* \setminus \{\varepsilon\})$ we use its *syntactic semigroup* $\mathrm{S}(T)$. For these notions, cf. [2] or [17]. Furthermore, the syntactic path monoid $\mathrm{PM}(T)$ of a ΣX-tree language is replaced by the *syntactic path semigroup* $\mathrm{PS}(T) := C^+_{\widehat{\Sigma}}/\widehat{\sigma}_T$, where $C^+_{\widehat{\Sigma}} := C_{\widehat{\Sigma}} \setminus \{\xi\}$ and $\widehat{\sigma}_T$ is $\widehat{\mu}_T$ restricted to $C^+_{\widehat{\Sigma}}$.

Let $T^+_\Sigma(X) := T_\Sigma(X) \setminus X$, and let us call a ΣX-tree language T *ε-free* if $T \subseteq T^+_\Sigma(X)$. Obviously T is ε-free if and only if every T_x $(x \in X)$ is ε-free. Furthermore, we call a DR-FTL or a DR-FUTL $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ *ε-free* if every tree language in $\mathcal{V}$ is ε-free. The *ε-free DR-FTL* $\mathbf{S}^p = \{\mathbf{S}^p(\Sigma, X)\}$ *path-defined* by a class of finite semigroups $\mathbf{S}$ is defined by

$$\mathbf{S}^p(\Sigma, X) := \{T \in DRec(\Sigma, X) \mid T \subseteq T^+_\Sigma(X),\ \mathrm{PS}(T) \in \mathbf{S}\},$$

and the *ε-free DR-FUTL path-defined* by $\mathbf{S}$ is $\mathbf{S}^u := (\mathbf{S}^p)^u$. Finally, for any $+$-variety $\mathcal{L} = \{\mathcal{L}(Z)\}$, let $\mathcal{V}_\mathcal{L} = \{\mathcal{V}_\mathcal{L}(\Sigma, X)\}$ be the ε-free DR-FTL defined by $\mathcal{V}_\mathcal{L}(\Sigma, X) := \{T \in DRec(\Sigma, X) \mid T \subseteq T^+_\Sigma(X),\ (\forall x \in X)\, T_x \in \mathcal{L}(\widehat{\Sigma})\}$.

We may now state the following variants of Propositions 5.3, 5.4 and 5.5.

Proposition 5.6. *Let $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ be an ε-free DR-FTL, and let* $\mathbf{S}$ *be a class of finite semigroups. If (1) $\mathcal{V}^u = \mathbf{S}^u$, and (2) $T \in \mathcal{V}(\Sigma, X)$ if and only if $\delta(T) \in \mathcal{V}(\widehat{\Sigma}, X)$ for all Σ, X and $T \subseteq T_\Sigma^+(X)$, then $\mathcal{V} = \mathbf{S}^p$.*

Proposition 5.7. *If $\mathcal{L} = \{\mathcal{L}(Z)\}$ is a $+$-variety and* $\mathbf{S}$ *is the corresponding VFS, then $\mathcal{V}_\mathcal{L} = \{\mathcal{V}_\mathcal{L}(\Sigma, X)\}$ is an ε-free DR-FTL path-defined by* $\mathbf{S}$.

Proposition 5.8. *If an ε-free DR-FTL $\mathcal{V} = \{\mathcal{V}(\Sigma, X)\}$ is path-defined by a VFS* $\mathbf{S}$ *and $\mathcal{L} = \{\mathcal{L}(Z)\}$ is the $+$-variety defined by* $\mathbf{S}$*, then $\mathcal{V} = \mathcal{V}_\mathcal{L}$.*

6 Some examples

Let us now apply the above results to the families of DR-recognizable tree languages considered by Gécseg and Imreh [5, 6, 7, 8]. In [7] they studied monotone string and tree automata and languages. Since monotonicity is basically a property of the underlying algebras, we begin by defining monotone algebras. A Z-algebra $\mathcal{A} = (A, Z)$ is *monotone* if there is a partial order $\leq$ on A such that $a \leq az^\mathcal{A}$ for all $a \in A$ and $z \in Z$. A language is *monotone* if it is recognized by a finite monotone algebra. Let $Mon = \{Mon(Z)\}$ be the family monotone languages.

A DR Σ-algebra $\mathcal{A}$ is *monotone* if $\mathcal{A}^u$ is monotone. A ΣX-tree language is *DR monotone* if it is recognized by a finite monotone DR Σ-algebra. Let $DMon = \{DMon(\Sigma, X)\}$ be the DR-FTL of DR monotone tree languages. These definitions are equivalent to the ones of Gécseg and Imreh [7] and the next lemma is an immediate consequence of their corresponding results.

Lemma 6.1. *A DR Σ-algebra $\mathcal{A} = (A, \Sigma)$ is monotone if and only if the reachability relation $\Rightarrow_\mathcal{A}^*$ is a partial order on A. Moreover, if $\mathcal{A}$ is monotone for some partial order $\leq$, then $a \Rightarrow_\mathcal{A}^* b$ implies $a \leq b$ $(a, b \in A)$.*

Let $\mathbf{M}_{cld}$ be the class of the finite monoids in which all *right-unit submonoids* are *closed under divisors*. These notions, introduced in [7], can be defined by saying that a finite monoid M belongs to $\mathbf{M}_{cld}$ if $s(r_1r_2) = s$ implies $sr_1 = s$ for all $s, r_1, r_2 \in M$; each $s \in M$ has its right-unit submonoid $\mathrm{RU}(s) := \{r \in M \mid sr = s\}$. In [7] it was shown that that a regular string language is monotone if and only if its syntactic monoid is in $\mathbf{M}_{cld}$, and in [5] Proposition 5.2 was used for showing that the DR monotone tree languages are path-defined by $\mathbf{M}_{cld}$. That $\mathbf{M}_{cld}$ is closed under subdirect products and factors is also implied by the following fact.

Proposition 6.1. $\mathbf{M}_{cld}$ *is a VFM.*

Proof. It is clear that $S(\mathbf{M}_{cld}), P_f(\mathbf{M}_{cld}) \subseteq \mathbf{M}_{cld}$. To prove $H(\mathbf{M}_{cld}) \subseteq \mathbf{M}_{cld}$, let $\varphi : M \to M'$ be an epimorphism and assume that $M \in \mathbf{M}_{cld}$. If $M' \notin \mathbf{M}_{cld}$, there are elements $a', b', c' \in M'$ such that $a'b'c' = a'$, but $a'b' \neq a'$. Let $a, b, c \in M$ be elements for which $a\varphi = a'$, $b\varphi = b'$ and $c\varphi = c'$. Since M is finite, there are numbers $k \geq 0, m \geq 1$ such that $(bc)^{k+m} = (bc)^k$. Let $d := a(bc)^k$. Then $d \cdot (b \cdot (c(bc)^{m-1})) = a(bc)^{k+m} = d$ but $d \cdot b \neq d$ because $(d \cdot b)\varphi = a'b'$ while $d\varphi = a'$. This would mean that $b \notin \mathrm{RU}(d)$ although $b \cdot (c(bc)^{m-1}) \in \mathrm{RU}(d)$. Hence, $M' \in \mathbf{M}_{cld}$ must hold. □

Since Mon is defined by $\mathbf{M}_{cld}$, it follows from Eilenberg's Variety Theorem that $Mon = \{Mon(Z)\}$ is a $*$-variety.

Proposition 6.2. *$DMon = \mathcal{V}_{Mon}$.*

Proof. Let $T \in DRec(\Sigma, X)$. If $T \in DMon(\Sigma, X)$, then T is recognized by a finite monotone DR Σ-algebra $\mathcal{A}$. By Proposition 4.4, every T_x $(x \in X)$ is recognized by the monotone $\widehat{\Sigma}$-algebra $\mathcal{A}^u$. Hence, $T \in \mathcal{V}_{Mon}(\Sigma, X)$.

Conversely, if $T_x \in Mon(\widehat{\Sigma})$ for every $x \in X$, then each T_x is recognized by a monotone $\widehat{\Sigma}$-algebra $\mathcal{A}_x$. By Proposition 4.4 T is recognized by the direct product of the algebras $\mathcal{A}_x^d$, and this DR algebra is monotone (cf. Proposition 7.2 below). □

By Proposition 5.4, Theorem 22 of [7] follows from Proposition 6.2.

Corollary 6.1. *$DMon$ is path-defined by the VFM $\mathbf{M}_{cld}$.*

As a second example we consider nilpotent DR algebras and tree languages. Since nilpotent string languages are characterized by their syntactic semigroups, we shall use Proposition 5.7. The finite and co-finite ε-free languages form a $+$-variety $Nil = \{Nil(Z)\}$ which corresponds to the VFS **Nil** of finite nilpotent semigroups: a semigroup S is *nilpotent* if it has a zero-element 0 and there is a number $k \geq 1$ such that $s_1 \cdot \ldots \cdot s_k = 0$ for all $s_1, \ldots, s_k \in S$ (cf. [2, 17]). A Z-algebra $\mathcal{A} = (A, Z)$ is *nilpotent* if there exist an element $\tilde{a} \in A$ and a number $k \geq 0$ such that $av^{\mathcal{A}} = \tilde{a}$ for all $a \in A$ and every $v \in Z^*$ of length k. It is easy to see that the ε-free languages recognized by these algebras are exactly the members of Nil (cf. [9], p. 125).

Let us call a DR Σ-algebra $\mathcal{A}$ *nilpotent* if the $\widehat{\Sigma}$-algebra $\mathcal{A}^u$ is nilpotent. It is clear that this definition is equivalent to the one of [8]. A ΣX-tree language is *DR nilpotent* if it is recognized by a finite nilpotent DR Σ-algebra. Let $DNil = \{DNil(\Sigma, X)\}$ be the FTL of ε-free DR nilpotent tree languages.

Proposition 6.3. *$DNil = \mathcal{V}_{Nil}$.*

Proof. The proposition follows from Proposition 4.4 and the facts mentioned above. Firstly, if $T \subseteq T_\Sigma^+(X)$ is recognized by a finite nilpotent DR Σ-algebra $\mathcal{A}$, then every T_x is recognized by the finite nilpotent $\widehat{\Sigma}$-algebra $\mathcal{A}^u$, and therefore belongs to $Nil(\widehat{\Sigma})$. On the other hand, if each T_x is recognized by a finite nilpotent $\widehat{\Sigma}$-algebra $\mathcal{A}_x$, then T is recognized by the finite nilpotent DR Σ-algebra $\prod(\mathcal{A}_x^d \mid x \in X)$. □

Proposition 5.7 yields the following result proved in [8].

Corollary 6.2. *$DNil$ is path-defined by the VFS* **Nil**.

As our third example we discuss the DR definite tree languages considered in [6, 8]. Let us first recall that a Z-algebra $\mathcal{A} = (A, Z)$ is *definite* if there is a $k \geq 0$ such that $av^{\mathcal{A}} = bv^{\mathcal{A}}$ for all $a, b \in A$ and every $v \in Z^*$ of length k. The languages recognized by definite algebras are also called *definite*, and they are the languages of the form $E \cup Z^*F$ where, for some $k \geq 0$, E is a set of words of length $< k$ and F is a set of words of length k (cf. [9], for example). The ε-free definite languages form a +-variety $Def = \{Def(Z)\}$ characterized by the VFS $\mathbf{D}$ of all finite semigroups S such that $Se = \{e\}$ for every idempotent $e \in S$ (cf. [2]).

Let us call a DR Σ-algebra $\mathcal{A}$ *definite* if the $\widehat{\Sigma}$-algebra $\mathcal{A}^u$ is definite, and say that a ΣX-tree language is *DR definite* if it is recognized by a finite definite DR Σ-algebra. Again, these definitions are equivalent to those of [8]. Let $DDef = \{DDef(\Sigma, X)\}$ be the DR-FTL of all ε-free DR definite tree languages.

It follows directly from our definitions and Proposition 4.4 that $DDef = \mathcal{V}_{Def}$. Hence, Proposition 5.7 yields the following result proved in [8].

Proposition 6.4. *$DDef$ is path-defined by the VFS $\mathbf{D}$.*

Thus all three families of DR-recognizable tree languages shown in [6, 8] to be path-definable by a class of finite monoids or semigroups are obtained from a $*$- or a $+$-variety using Proposition 5.4 or Proposition 5.7. By Propositions 5.5 and 5.8 this can be expected once we know that the corresponding family of string languages is a $*$- or a $+$-variety. Indeed, any $*$- and $+$-variety will yield a DR-FTL or an ε-free DR-FTL path-definable by a VFM or a VFS, respectively.

7 Varieties of finite DR algebras

In this section we discuss varieties of finite DR Σ-algebras for an arbitrarily fixed ranked alphabet Σ. The class operators S, H, P_f and V_f are defined for DR Σ-algebras in the natural way, and we call a class $\mathbf{K}$ of finite DR Σ-algebras a *variety of finite DR Σ-algebras* (DR Σ-VFA) if $S(\mathbf{K})$, $H(\mathbf{K})$, $P_f(\mathbf{K}) \subseteq \mathbf{K}$.

In [3] Ésik defines $\mathbf{K}^u := \{\mathcal{A}^u \mid \mathcal{A} \in \mathbf{K}\}$ for any class $\mathbf{K}$ of DR Σ-algebras. He noted that $\mathbf{K}$ is a variety of DR Σ-algebras if and only if $\mathbf{K}^u$ is a variety of $\widehat{\Sigma}$-algebras, and that notions from the theory of varieties of algebras may therefore be extended to DR algebras. We shall apply the operation $\mathbf{K} \mapsto \mathbf{K}^u$ to classes of finite DR Σ-algebras, and we also introduce the converse operation that associates with each class $\mathbf{U}$ of finite $\widehat{\Sigma}$-algebras the class $\mathbf{U}^d := \{\mathcal{C}^d \mid \mathcal{C} \in \mathbf{U}\}$ of finite DR Σ-algebras. Obviously, $\mathbf{K}^{ud} = \mathbf{K}$ and $\mathbf{U}^{du} = \mathbf{U}$.

Lemma 7.1. *Let $\mathbf{K}$ be a class of finite DR Σ-algebras and $\mathbf{U}$ be a class of finite $\widehat{\Sigma}$-algebras. Then*

(a) *$H(\mathbf{K})^u = H(\mathbf{K}^u)$, $S(\mathbf{K})^u = S(\mathbf{K}^u)$ and $P_f(\mathbf{K})^u = P_f(\mathbf{K}^u)$, and*

(b) *$H(\mathbf{U})^d = H(\mathbf{U}^d)$, $S(\mathbf{U})^d = S(\mathbf{U}^d)$ and $P_f(\mathbf{U})^d = P_f(\mathbf{U}^d)$.*

Hence, $\mathbf{K}$ is a DR Σ-VFA if and only if $\mathbf{K}^u$ is a $\widehat{\Sigma}$-VFA, and $\mathbf{U}$ is a $\widehat{\Sigma}$-VFA if and only if $\mathbf{U}^d$ is a DR Σ-VFA.

Proof. The equalities in (a) and (b) are immediate consequences of Lemma 3.1, and the remaining claims follow from them. □

An easy modification of Tarski's well-known HSP-theorem (cf. [1], p. 61) yields $V_f(\mathbf{K}) = HSP_f(\mathbf{K})$ for any ranked alphabet Σ and any class $\mathbf{K}$ of finite Σ-algebras (cf. [18], for example). Let us derive the corresponding representation for finite DR Σ-algebras.

Proposition 7.1. *$V_f(\mathbf{K}) = HSP_f(\mathbf{K})$ for any class $\mathbf{K}$ of finite DR Σ-algebras.*

Proof. Clearly, $\mathbf{K} \subseteq HSP_f(\mathbf{K})$. As a special case of the fact noted above, we get $V_f(\mathbf{K}^u) = HSP_f(\mathbf{K}^u)$. This means, in particular, that $HSP_f(\mathbf{K}^u)$ is a $\widehat{\Sigma}$-VFA. Since $HSP_f(\mathbf{K})^u = HSP_f(\mathbf{K}^u)$, this implies by Lemma 7.1 that $HSP_f(\mathbf{K})$ is a DR Σ-VFA. If $\mathbf{L}$ is a DR Σ-VFA with $\mathbf{K} \subseteq \mathbf{L}$, then $HSP_f(\mathbf{K}) \subseteq HSP_f(\mathbf{L}) = \mathbf{L}$. Hence, $HSP_f(\mathbf{K})$ is the DR Σ-VFA generated by $\mathbf{K}$. □

Lemma 7.1 and Proposition 7.1 yield the following fact.

Corollary 7.1. *$V_f(\mathbf{K}) = V_f(\mathbf{K}^u)^d$ for any class $\mathbf{K}$ of finite DR Σ-algebras.*

Let $\mathbf{DMon}_\Sigma$, $\mathbf{DNil}_\Sigma$ and $\mathbf{DDef}_\Sigma$ denote the classes of all finite monotone, nilpotent and definite DR Σ-algebras, respectively.

Proposition 7.2. *$\mathbf{DMon}_\Sigma$, $\mathbf{DNil}_\Sigma$ and $\mathbf{DDef}_\Sigma$ are DR Σ-VFAs.*

Proof. By Lemma 7.1, it suffices to verify that the corresponding classes $\mathbf{DMon}_\Sigma^u$, $\mathbf{DNil}_\Sigma^u$ and $\mathbf{DDef}_\Sigma^u$ of unary algebras are $\widehat{\Sigma}$-VFAs.

Gécseg and Imreh [7] proved that all finite direct products and homomorphic images of monotone finite automata are monotone. These results apply directly to unary algebras, and hence $P_f(\mathbf{DMon}_\Sigma^u) \subseteq \mathbf{DMon}_\Sigma^u$ and $H(\mathbf{DMon}_\Sigma^u) \subseteq \mathbf{DMon}_\Sigma^u$. Since it is clear that subalgebras of monotone algebras are monotone, we may conclude that $\mathbf{DMon}_\Sigma^u$ is a $\widehat{\Sigma}$-VFA.

In [18] it was noted that for any ranked alphabet Σ, the finite nilpotent Σ-algebras form an Σ-VFA, and in [4] Ésik proved the corresponding fact about finite definite Σ-algebras. Hence, also $\mathbf{DNil}_\Sigma^u$ and $\mathbf{DDef}_\Sigma^u$ are $\widehat{\Sigma}$-VFAs. □

Let us now consider equational definitions of DR Σ-VFAs. The terms appearing in $\widehat{\Sigma}$-identities are written as expressions ξu, where ξ is a variable and $u \in \widehat{\Sigma}^*$. There are two types of $\widehat{\Sigma}$-identities, the *regular identities* $\xi u \approx \xi v$ and the *irregular identities* $\xi u \approx \xi' v$, in which ξ and ξ' are two distinct variables. A $\widehat{\Sigma}$-algebra $\mathcal{C} = (C, \widehat{\Sigma})$ *satisfies* $\xi u \approx \xi v$ if $u^{\mathcal{C}} = v^{\mathcal{C}}$, and it satisfies $\xi u \approx \xi' v$, where $\xi \neq \xi'$, if $cu^{\mathcal{C}} = dv^{\mathcal{C}}$ for all $c, d \in C$. Furthermore, $\mathcal{C}$ *ultimately satisfies* an ω-sequence

$$E = \langle \ell_0 \approx r_0, \ell_1 \approx r_1, \ell_2 \approx r_2, \ldots \rangle$$

of $\widehat{\Sigma}$-identities if there exists an $n_0 \geq 0$ such that $\mathcal{C}$ satisfies $\ell_n \approx r_n$ for every $n \geq n_0$. The class E^u of finite $\widehat{\Sigma}$-algebras *ultimately defined* by E consists of the finite $\widehat{\Sigma}$-algebras ultimately satisfying E. By a well-known theorem by Eilenberg and Schützenberger (cf. [2, 17]), a class $\mathbf{K}$ of finite $\widehat{\Sigma}$-algebras is a $\widehat{\Sigma}$-VFA if and only if $\mathbf{K} = E^u$ for some ω-sequence E of $\widehat{\Sigma}$-identities.

Following Ésik [3] we say that a *DR Σ-algebra $\mathcal{A}$ satisfies* a $\widehat{\Sigma}$-identity $\ell \approx r$ if $\mathcal{A}^u$ satisfies $\ell \approx r$. Naturally, $\mathcal{A}$ *ultimately satisfies* an ω-sequence E of $\widehat{\Sigma}$-identities if $\mathcal{A}^u$ ultimately satisfies E. The class of finite DR Σ-algebras ultimate satisfying E is denoted by E^d. Thus E^u is a class of $\widehat{\Sigma}$-algebras and $E^{ud} := (E^u)^d$ is the corresponding class of DR Σ-algebras. Similarly, $E^{du} := (E^d)^u$ is the class of $\widehat{\Sigma}$-algebras corresponding to the class E^d of DR Σ-algebras. Since $E^{ud} = E^d$ and

$E^{du} = E^u$, the next proposition follows from the Eilenberg-Schützenberger theorem and Lemma 7.1.

Proposition 7.3. *A class* $\mathbf{K}$ *of finite DR* Σ*-algebras is a DR* Σ*-VFA if and only if* $\mathbf{K} = E^d$ *for some* ω*-sequence* E *of* $\widehat{\Sigma}$*-identities.*

8 Concluding remarks

We have considered some aspects of DR tree recognizers and DR-recognizable tree languages. Our algebraic approach is in a large part based on the connection between DR algebras and unary algebras put forward by Ésik [3]. It was used for describing the relationship between a DR-recognizable tree language and its path language as well as in the discussion of varieties of finite DR algebras.

In Section 5 we showed that any $*$- or $+$-variety defines a family of DR-recognizable tree languages path-definable by a variety of finite monoids or a variety of finite semigroups, respectively. Hence there are many families of DR-recognizable tree languages that could be investigated similarly as the families $DMon$, $DNil$ and $DDef$ were studied in the papers [7, 8, 13, 14]. On a more general level, one should describe the common properties of all such families. In particular, it is conceivable that they are characterized by some closure properties. It is also natural to consider the families of DR-recognizable tree languages that correspond to a varieties of finite DR algebras. Such questions belong to a variety theory of DR-recognizable tree languages still to be developed.

References

[1] S. Burris and H.P. Sankappanavar, *A Course in Universal Algebra*, Springer-Verlag, New York, 1981.

[2] S. Eilenberg, *Automata, Languages, and Machines.* Vol. B, Academic Press, New York 1976.

[3] Z. Ésik, Varieties and general products of top-down algebras, *Acta Cybernetica* **7** (1986), 293-298.

[4] Z. Ésik, Definite tree automata and their cascade composition, *Publicationes Mathematicae Debrecen* **48** 3–4 (1996), 243–261.

[5] F. Gécseg, Classes of tree languages determined by classes of monoids, *International Journal of Foundations of Computer Science* **18** (2007), 1237-1246.

[6] F. Gécseg, Classes of tree languages and DR tree languages given by classes of semigroups, *Acta Cybernetica* **20** (2011), 253-267.

[7] F. Gécseg and B. Imreh, On monotone automata and monotone languages, *Journal of Automata, Languages and Combinatorics* **7** (2002), 71-82.

[8] F. Gécseg and B. Imreh, On definite and nilpotent DR tree languages, *Journal of Automata, Languages and Combinatorics* **9** (2004), 55-60.

[9] F. Gécseg and I. Peák: *Algebraic theory of automata*, Akadémiai Kiadó, Budapest 1972.

[10] F. Gécseg and M. Steinby, Minimal ascending tree automata, *Acta Cybernetica* **4** (1978), 37-44.

[11] F. Gécseg and M. Steinby, *Tree Automata*, Akadémiai Kiadó, Budapest, 1984. 2. ed. downloadable from arXiv.org as arXiv:1509.06233, September 2015.

[12] F. Gécseg and M. Steinby, Minimal recognizers and syntactic monoids of DR tree languages, *Words, Semigroups and Transductions* (eds. M. Ito, G. Paun and S. Yu), World Scientific, Singapore 2001, 155-167.

[13] Gy. Gyurica, On monotone languages and their characterization by regular expressions, *Acta Cybernetica* **18** (2007), 117-134.

[14] Gy. Gyurica, On nilpotent languages and their characterization by regular expressions, *Acta Cybernetica* **19** (2009), 231-244.

[15] E. Jurvanen, *On Tree Languages Defined by Deterministic Root-to-frontier Recognizers*, Dissertation, Department of Mathematics, University of Turku, Turku 1995.

[16] M. Magidor and G. Moran, *Finite Automata over Finite Trees*, Technical Report 30, Department of Mathematics, Hebrew University, Jerusalem 1969.

[17] J.E. Pin, *Varieties of Formal Languages*, North Oxford Academic Publishers, London 1986.

[18] M. Steinby, A theory of tree language varieties, *Tree Automata and Languages* (eds. M. Nivat and A. Podelski), North-Holland, Amsterdam 1992, 57-81.

[19] W. Thomas, Logical aspects in the study of tree languages, *9th Colloquium on Trees in Algebra and Programming* (ed. B. Courcelle), Proc. 9th CAAP, Bordeaux, 1984, Cambridge University Press, London, 1984, pp. 31–49.

[20] J. Virágh, Deterministic tree automata I, *Acta Cybernetica* **5** (1980), 33-42; II, *ibid* **6** (1983), 291-301.

Acta Cybernetica **23** (2017) 175–189.

Variations of the Morse-Hedlund Theorem for k-Abelian Equivalence[*]

Juhani Karhumäki[a], Aleksi Saarela[a], and Luca Q. Zamboni[b]

Abstract

In this paper we investigate local-to-global phenomena for a new family of complexity functions of infinite words indexed by $k \geq 0$. Two finite words u and v are said to be k-abelian equivalent if for all words x of length less than or equal to k, the number of occurrences of x in u is equal to the number of occurrences of x in v. This defines a family of equivalence relations, bridging the gap between the usual notion of abelian equivalence (when $k = 1$) and equality (when $k = \infty$). Given an infinite word w, we consider the associated complexity function which counts the number of k-abelian equivalence classes of factors of w of length n. As a whole, these complexity functions have a number of common features: Each gives a characterization of periodicity in the context of bi-infinite words, and each can be used to characterize Sturmian words in the framework of aperiodic one-sided infinite words. Nevertheless, they also exhibit a number of striking differences, the study of which is one of the main topics of our paper.

1 Introduction

A fundamental problem in both mathematics and computer science is to describe local constraints which imply global regularities. A splendid example of this phenomena may be found in the framework of combinatorics on words. In their seminal papers [19, 20], G. A. Hedlund and M. Morse proved that a bi-infinite word w is periodic if and only if for some positive integer n, the word w contains at most n distinct factors of length n. In other words, it describes the exact borderline between periodicity and aperiodicity of words in terms of the *factor complexity function* which counts the number of distinct factors of each length n. An analogous result was established some thirty years later by E. Coven and G. A. Hedlund in the framework of abelian equivalence. They show that a bi-infinite word is periodic if

[*]Partially supported by the Academy of Finland grants 257857 and 137991 (FiDiPro), by ANR grant *SUBTILE*, and by the Vilho, Yrjö and Kalle Väisälä Foundation.

[a]Department of Mathematics and Statistics, University of Turku, 20014 Turku, Finland, E-mail: `{karhumak,amsaar}@utu.fi`

[b]Institut Camille Jordan, Université Lyon 1, 43 boulevard du 11 novembre 1918, 69622 Villeurbanne Cedex, France, E-mail: `lupastis@gmail.com`

DOI: 10.14232/actacyb.23.1.2017.11

and only if for some positive integer n all factors of w are abelian equivalent. Thus once again it is possible to distinguish between periodic and aperiodic words on a local level by counting the number of abelian equivalence classes of factors of length n.

In this paper we study the local-to-global behavior for a new family of complexity functions $\mathcal{P}_w^k$ of infinite words indexed by $k \in \mathbb{Z}_+ \cup \{\infty\}$ where $\mathbb{Z}_+ = \{1, 2, 3, \ldots\}$ denotes the set of positive integers. Let $k \in \mathbb{Z}_+ \cup \{\infty\}$ and A be a finite non-empty set. Two finite words $u, v \in A^*$ are said to be *k-abelian equivalent* if for all $x \in A^*$ of length at most k, the number of occurrences of x in u is equal to the number of occurrences of x in v. This defines a family of equivalence relations $\sim_k$ on A^*, bridging the gap between the usual notion of abelian equivalence (when $k = 1$) and equality (when $k = \infty$). Abelian equivalence of words has long been a subject of great interest (see, for instance, Erdős's problem, [5, 6, 7, 9, 17, 22, 23, 24, 26]). Although the notion of k-abelian equivalence is quite new, there are already a number of papers on the topic [11, 12, 13, 14, 15, 18].

Given an infinite word $w \in A^\omega$, we consider the associated complexity function $\mathcal{P}_w^k : \mathbb{Z}_+ \to \mathbb{Z}_+$ which counts the number of k-abelian equivalence classes of factors of w of length n. Thus $\mathcal{P}_w^\infty$ corresponds to the usual factor complexity (sometimes called subword complexity in the literature) while $\mathcal{P}_w^1$ corresponds to abelian complexity. As it turns out, each intermediate complexity function $\mathcal{P}_w^k$ can be used to detect periodicity of words. As a starting point of our research, we list two classical results on factor and abelian complexity in connection with periodicity, and their k-abelian counterparts proved by the authors in [15]. We note that in each case, the first two items are included in the third.

Theorem 1. *Let w be a bi-infinite word over a finite alphabet. Then the following properties hold:*

- *(M. Morse, G. A. Hedlund, [19]) The word w is periodic if and only if $\mathcal{P}_w^\infty(n) < n + 1$ for some $n \geq 1$.*
- *(E. M. Coven, G. A. Hedlund, [6]) The word w is periodic if and only if $\mathcal{P}_w^1(n) < 2$ for some $n \geq 1$.*
- *The word w is periodic if and only if $\mathcal{P}_w^k(n) < \min\{n + 1, 2k\}$ for some $k \in \mathbb{Z}_+ \cup \{\infty\}$ and $n \geq 1$.*

Also, each complexity provides a characterization for an important class of binary words, the so-called *Sturmian* words:

Theorem 2. *Let w be an aperiodic one-sided infinite word. Then the following properties hold:*

- *(M. Morse, G. A. Hedlund, [20]). The word w is Sturmian if and only if $\mathcal{P}_w^\infty(n) = n + 1$ for all $n \geq 1$.*
- *(E. M. Coven, G. A. Hedlund, [6]). The word w is Sturmian if and only if $\mathcal{P}_w^1(n) = 2$ for all $n \geq 1$.*

- *The word w is Sturmian if and only if $\mathcal{P}_w^k(n) = \min\{n+1, 2k\}$ for all $k \in \mathbb{Z}_+ \cup \{\infty\}$ and $n \geq 1$.*

However, in other respects, these various complexities exhibit radically different behaviors. For instance, in the context of one-sided infinite words, the first item in Theorem 1 gives rise to a characterization of ultimately periodic words, while for the other two, the result holds in only one direction: If $\mathcal{P}_w^k(n) < \min\{n+1, 2k\}$ for some $k \in \mathbb{Z}_+$ and $n \geq 1$ then w is ultimately periodic, but not conversely (see [15]). For instance in the simplest case when $k = 1$, it is easy to see that if w is the ultimately periodic word 01^ω, then for each positive integer n there are precisely two abelian classes of factors of w of length n. However, the same is true of the (aperiodic) *infinite Fibonacci word* $w = 0100101001001\cdots$ defined as the fixed point of the morphism $0 \mapsto 01$, $1 \mapsto 0$. Analogously, in Theorem 2 the first item holds without the added assumption that w be aperiodic, while the other two items do not. Another striking difference between them is in their rate of growth. Consider for instance the binary Champernowne word $\mathcal{C} = 011011100101110111\cdots$ obtained by concatenating the binary representation of the consecutive natural numbers. Let w denote the morphic image of $\mathcal{C}$ under the Thue–Morse morphism $0 \mapsto 01$, $1 \mapsto 10$. Then while $\mathcal{P}_w^\infty(n)$ has exponential growth, it can be shown that $\mathcal{P}_w^1(n) \leq 3$ for all n. Yet another fundamental disparity concerns the difference $\mathcal{P}_w^k(n+1) - \mathcal{P}_w^k(n)$. For factor complexity, one always has $\mathcal{P}_w^\infty(n+1) - \mathcal{P}_w^\infty(n) \geq 0$, while for general k this inequality is far from being true.

A primary objective in this paper is to study the asymptotic lower and upper complexities defined by

$$\mathcal{L}_w^k(n) = \min_{m \geq n} \mathcal{P}_w^k(m) \qquad \text{and} \qquad \mathcal{U}_w^k(n) = \max_{m \leq n} \mathcal{P}_w^k(m).$$

Surprisingly these quantities can deviate from one another quite drastically. Indeed, one of our main results is to compute these values for the famous Thue–Morse word. We show that the upper limit is logarithmic, while the lower limit is just constant, in fact at most 8 in the case $k = 2$. This is quite unexpected considering the Thue–Morse word is both pure morphic and abelian periodic (of period 2). If we however allow more general words, then we obtain much stronger evidence of the non-existence of gaps in low k-abelian complexity classes. We construct uniformly recurrent infinite words having arbitrarily low upper limit and just constant lower limit. The concept of k-abelian complexity also leads to many interesting open questions. We conclude the paper in Section 6 by mentioning some of these problems.

This is an extended version of an article that was presented at the 18th conference on Developments in Language Theory [16].

2 Preliminaries

Let Σ be a finite non-empty set called the *alphabet*. The set of all finite words over Σ is denoted by Σ^* and the set of all (right) infinite words is denoted by Σ^ω. The set of positive integers is denoted by $\mathbb{Z}_+$. A function $f : \mathbb{Z}_+ \to \mathbb{R}$ is *increasing* if $f(m) \leq f(n)$ for all $m < n$, and *strictly increasing* if $f(m) < f(n)$ for all $m < n$.

Let $w \in \Sigma^\omega$. The word w is *periodic* if there is $u \in \Sigma^*$ such that $w = u^\omega$, and *ultimately periodic* if there are $u, v \in \Sigma^*$ such that $w = vu^\omega$. If w is not ultimately periodic, then it is *aperiodic*. Let $u = a_0 \cdots a_{m-1}$ and $a_0, \dots, a_{m-1} \in \Sigma$. The *prefix* of length n of u is $\mathrm{pref}_n(u) = a_0 \cdots a_{n-1}$ and the *suffix* of length n of u is $\mathrm{suff}_n(u) = a_{m-n} \cdots a_{m-1}$. If $0 \leq i \leq m$, then the notation $\mathrm{rfact}_n^i(u) = a_i \cdots a_{i+n-1}$ is used. The length of a word u is denoted by $|u|$ and the number of occurrences of another word x as a factor of u by $|u|_x$. As a trivial boundary case, $|u|_\varepsilon = |u| + 1$. Two words $u, v \in \Sigma^*$ are *abelian equivalent* if $|u|_a = |v|_a$ for all $a \in \Sigma$.

Let $k \in \mathbb{Z}_+$. Two words $u, v \in \Sigma^*$ are *k-abelian equivalent* if $|u|_x = |v|_x$ for all words x of length at most k. k-abelian equivalence is denoted by $\sim_k$. If the length of u and v is at least $k-1$, then $u \sim_k v$ if and only if $|u|_x = |v|_x$ for all words x of length k and $\mathrm{pref}_{k-1}(u) = \mathrm{pref}_{k-1}(v)$ and $\mathrm{suff}_{k-1}(u) = \mathrm{suff}_{k-1}(v)$. This gives an alternative definition for k-abelian equivalence. A proof can be found in [15].

Let $w \in \Sigma^\omega$. The set of factors of w of length n is denoted by $\mathcal{F}_w(n)$. The *factor complexity* of w is the function $\mathcal{P}_w^\infty : \mathbb{Z}_+ \to \mathbb{Z}_+$ defined by

$$\mathcal{P}_w^\infty(n) = \#\mathcal{F}_w(n),$$

where $\#$ is used to denote the cardinality of a set. Let $k \in \mathbb{Z}_+$. The *k-abelian complexity* of w is the function $\mathcal{P}_w^k : \mathbb{Z}_+ \to \mathbb{Z}_+$ defined by

$$\mathcal{P}_w^k(n) = \#(\mathcal{F}_w(n) / \sim_k).$$

Factor complexity functions are always increasing, and even strictly increasing for aperiodic words. For k-abelian complexity this is not true. This is why we define *upper k-abelian complexity* $\mathcal{U}_w^k$ and *lower k-abelian complexity* $\mathcal{L}_w^k$:

$$\mathcal{U}_w^k(n) = \max_{m \leq n} \mathcal{P}_w^k(m) \qquad \text{and} \qquad \mathcal{L}_w^k(n) = \min_{m \geq n} \mathcal{P}_w^k(m).$$

These two functions can be significantly different. For example, if w is the Thue–Morse word and $k \geq 2$, then $\mathcal{U}_w^k(n) = \Theta(\log n)$ and $\mathcal{L}_w^k(n) = \Theta(1)$. This will be proved in Section 4.

When using Θ-notation, the parameter k and the size of the alphabet are assumed to be fixed, so the implied constants of the Θ-notation can depend on them.

The abelian complexity of a binary word $w \in \{0,1\}^\omega$ can be determined by using the formula (see [24])

$$\mathcal{P}_w^1(n) = \max\{|u|_1 \mid u \in \mathcal{F}_n(w)\} - \min\{|u|_1 \mid u \in \mathcal{F}_n(w)\} + 1. \tag{1}$$

For $k \in \mathbb{Z}_+ \cup \{\infty\}$, we define

$$q^k : \mathbb{Z}_+ \to \mathbb{Z}_+, q^k(n) = \min\{n+1, 2k\}.$$

The significance of this function is that if w is Sturmian, then $\mathcal{P}_w^k = q^k$. This is further discussed in Section 3.

There are large classes of words for which the k-abelian complexities are of the same order for many values of k. This is shown in the next two lemmas. Thus when analyzing the growth rate of the k-abelian complexity of a word, it may be sufficient to analyze the abelian or 2-abelian complexity.

Lemma 1. *Let $w \in \{0,1\}^\omega$ be such that every factor of w of length k contains at most one occurrence of 1. Then $\mathcal{P}_w^k(n) = \Theta(\mathcal{P}_w^1(n))$.*

Proof. Clearly $\mathcal{P}_w^k(n) \geq \mathcal{P}_w^1(n)$. Let u be a factor of w of length n. Let $x = 0^i 1 0^{k-i-1}$. Every factor of w of length k except 0^k is of this form, because every factor of w of length k contains at most one occurrence of 1. For the same reason, $|u|_x = |u|_1 - a$, where $a \in \{0,1,2\}$ depending on $\mathrm{pref}_{k-1}(u)$ and $\mathrm{suff}_{k-1}(u)$. It follows that the k-abelian equivalence class of u is determined by $\mathrm{pref}_{k-1}(u)$, $\mathrm{suff}_{k-1}(u)$, and $|u|_1$. The number of possible pairs $(\mathrm{pref}_{k-1}(u), \mathrm{suff}_{k-1}(u))$ is at most k^2, and the number of possible values for $|u|_1$ is $\mathcal{P}_w^1(n)$, so $\mathcal{P}_w^k(n) \leq k^2 \mathcal{P}_w^1(n)$. □

Lemma 2. *Let $k, m \geq 2$ and let w be a fixed point of an m-uniform morphism h. Let i be such that $m^i \geq k-1$. Then $\mathcal{P}_w^k(m^i(n+1)) = O(\mathcal{P}_w^2(n))$.*

Proof. Every factor of w of length $m^i(n+1)$ can be written as $ph^i(u)q$, where u is a factor of w of length n and $|pq| = m^i$. The k-abelian equivalence class of $ph^i(u)q$ is determined by p, q, and the 2-abelian equivalence class of u. The number of possible pairs (p,q) is $O(1)$, and the number of possible values for the 2-abelian equivalence class of u is $\mathcal{P}_w^2(n)$. The claim follows. □

In particular, Lemma 2 can be applied to the Thue–Morse word to analyze its k-abelian complexity once the behavior of its 2-abelian complexity is known.

It has been shown that there are many words for which the k-abelian and $(k+1)$-abelian complexities are similar, but there are also many words for which they are very different. For example, there are words having bounded k-abelian complexity but linear $(k+1)$-abelian complexity. These words can even be assumed to be k-abelian periodic, meaning that they are of the form $u_1 u_2 \cdots$, where $u_1, u_2, \ldots$ are k-abelian equivalent. This is shown in the next lemma.

Lemma 3. *For every $k \geq 1$, there is a k-abelian periodic word w such that $\mathcal{P}_w^{k+1}(n) = \Theta(n)$.*

Proof. Let $W \in \{0,1\}^\omega$ be a word with linear abelian complexity (e.g., the Champernowne word) and let h be the morphism defined by

$$h(0) = 0^{k+1} 1 0^{k-1} 1, \qquad h(1) = 0^k 1 0^k 1.$$

Then the word $w = h(W)$ is k-abelian periodic of period $2k+2$. If $u, v \in \{0,1\}^*$ are not abelian equivalent, then $h(u)$ and $h(v)$ are not $(k+1)$-abelian equivalent because the factor $1 0^{k-1} 1$ appears only inside $h(0)$. On the other hand, if $u, v \in \{0,1\}^*$ are abelian equivalent and $p, q \in \{0,1\}^*$, then $ph(u)q$ and $ph(v)q$ are $(k+1)$-abelian equivalent. It follows that

$$\mathcal{P}_w^{k+1}((2k+2)n) = \Theta(\mathcal{P}_W^1(n)) = \Theta(n). \tag{2}$$

We know that

$$\mathcal{P}_w^{k+1}(n+1) \leq 2\mathcal{P}_w^{k+1}(n) \tag{3}$$

for all n (this would work for all words w if 2 would be replaced by the size of the alphabet). Every n can be written as $(2k+2)n' + r$, where $0 \leq r < 2k+2$, so from (2) and (3) it follows that

$$\mathcal{P}_w^{k+1}(n) = \mathcal{P}_w^{k+1}((2k+2)n' + r) \leq 2^r \mathcal{P}_w^{k+1}((2k+2)n') = \Theta(n') = \Theta(n).$$

Similarly, every n can be written as $(2k+2)n' - r$, where $0 \leq r < 2k+2$, so from (2) and (3) it follows that

$$\mathcal{P}_w^{k+1}(n) = \mathcal{P}_w^{k+1}((2k+2)n' - r) \geq 2^{-r} \mathcal{P}_w^{k+1}((2k+2)n') = \Theta(n') = \Theta(n).$$

The claim follows. □

3 Minimal k-Abelian Complexities

In this section classes of words with small k-abelian complexity are studied. Some well-known results about factor complexity are compared to results on k-abelian complexity proved in [15]. It should be expected that ultimately periodic words have low complexity, and this is indeed true for k-abelian complexity, although the k-abelian complexity of some ultimately periodic words is higher that the k-abelian complexity of some aperiodic words. For many complexity measures, Sturmian words have the lowest complexity among aperiodic words. This is also true for k-abelian complexity.

We recall the famous theorem of Morse and Hedlund [19] characterizing ultimately periodic words in terms of factor complexity. This theorem can be generalized for k-abelian complexity: If $\mathcal{P}_w^k(n) < q^k(n)$ for some n, then w is ultimately periodic, and if w is ultimately periodic, then $\mathcal{P}_w^\infty(n)$ is bounded. This was proved in [15].

If k is finite, then this generalization does not give a characterization of ultimately periodic words, because the function q^k is bounded. In fact, it is impossible to characterize ultimately periodic words in terms of k-abelian complexity. For example, the word $0^{2k-1}1^\omega$ has the same k-abelian complexity as every Sturmian word. On the other hand, for every ultimately periodic word w there is a finite k such that $\mathcal{P}_w^k(n) < q^k(n)$ for all sufficiently large n.

The theorem of Morse and Hedlund has a couple of immediate consequences. The words w with $\mathcal{P}_w^\infty(n) = n + 1$ for all n are, by definition, Sturmian words. Thus the following classification is obtained:

- w is ultimately periodic $\Leftrightarrow$ $\mathcal{P}_w^\infty$ is bounded.

- w is Sturmian $\Leftrightarrow \mathcal{P}_w^\infty(n) = n + 1$ for all n.
- w is aperiodic and not Sturmian $\Leftrightarrow \mathcal{P}_w^\infty(n) \geq n+1$ for all n and $\mathcal{P}_w^\infty(n) > n+1$ for some n.

This can be generalized for k-abelian complexity if the equivalences are replaced with implications:

- w is ultimately periodic $\Rightarrow \mathcal{P}_w^k$ is bounded.
- w is Sturmian $\Rightarrow \mathcal{P}_w^k = q^k$.
- w is aperiodic and not Sturmian $\Rightarrow \mathcal{P}_w^k(n) \geq q^k(n)$ for all n and $\mathcal{P}_w^k(n) > q^k(n)$ for some n.

For $k = 1$ this follows from the theorem of Coven and Hedlund [6]. For $k \geq 2$ it follows from a theorem in [15].

The above result means that one similarity between factor complexity and k-abelian complexity is that Sturmian words have the lowest complexity among aperiodic words. Another similarity between them is that ultimately periodic words have bounded complexity, and the largest values can be arbitrarily high: For every n, there is a finite word u having every possible factor of length n. Then $\mathcal{P}_{u^\omega}^k(n)$ is as high as it can be for any word, i.e., the number of k-abelian equivalence classes of words of length n.

Another direct consequence of the theorem of Morse and Hedlund is that there is a gap between constant complexity and the complexity of Sturmian words. For k-abelian complexity there cannot be a gap between bounded complexities and q^k, because the function q^k itself is bounded. However, the question whether there is a gap above bounded complexity is more difficult. The answer is that there is no such gap, even if only uniformly recurrent words are considered. This is proved in Section 5.

4 k-Abelian Complexity of the Thue–Morse Word

In this section the k-abelian complexity of the Thue–Morse word is analyzed. Before that, the abelian complexity of a closely related word is determined.

Let σ be the morphism defined by $\sigma(0) = 01, \sigma(1) = 00$. Let

$$S = 0100010101000100010001010100010 1\cdots$$

be the *period-doubling word*, which is the fixed point of σ; see, e.g., [8].

The abelian complexity of S is completely determined by the recurrence relations in the next lemma and by the first value $\mathcal{P}_S^1(1) = 2$. These relations were proved independently in [3]. It is an easy consequence that the abelian complexity of S is 2-regular (2-regular sequences were defined in [2]). The 2-abelian complexity of the Thue–Morse word has been conjectured to be 2-regular [25], and this is proved in [10] and [21].

Lemma 4. *For $n \geq 1$,*

$$\mathcal{P}_S^1(2n) = \mathcal{P}_S^1(n) \qquad \text{and} \qquad \mathcal{P}_S^1(4n \pm 1) = \mathcal{P}_S^1(n) + 1.$$

Proof. Let

$$p_n = \min\{|u|_1 \mid u \in \mathcal{F}_n(S)\} \qquad \text{and} \qquad q_n = \max\{|u|_1 \mid u \in \mathcal{F}_n(S)\}.$$

Let $\overline{0} = 1$ and $\overline{1} = 0$. For $a \in \{0,1\}$, $\sigma(a) = 0\overline{a}$ and $\sigma^2(a) = 010a$. Because

$$\mathcal{F}_{2n}(S) = \{\sigma(u) \mid u \in \mathcal{F}_n(S)\} \cup \{\overline{a}\sigma(u)0 \mid au \in \mathcal{F}_n(S)\},$$

it can be seen that $p_{2n} = n - q_n$ and $q_{2n} = n - p_n$. Because

$$\begin{aligned}\mathcal{F}_{4n-1}(S) = &\left\{\sigma^2(u)010 \mid u \in \mathcal{F}_{n-1}(S)\right\} \cup \left\{10a\sigma^2(u) \mid au \in \mathcal{F}_n(S)\right\} \cup \\ &\left\{0a\sigma^2(u)0 \mid au \in \mathcal{F}_n(S)\right\} \cup \left\{a\sigma^2(u)01 \mid au \in \mathcal{F}_n(S)\right\},\end{aligned}$$

it can be seen that

$$\begin{aligned}p_{4n-1} &= \min\{p_{n-1} + n, p_n + n, p_n + n - 1, p_n + n\} = p_n + n - 1,\\ q_{4n-1} &= \max\{q_{n-1} + n, q_n + n, q_n + n - 1, q_n + n\} = q_n + n.\end{aligned}$$

Because

$$\begin{aligned}\mathcal{F}_{4n+1}(S) = &\left\{\sigma^2(u)0 \mid u \in \mathcal{F}_n(S)\right\} \cup \left\{10a\sigma^2(u)01 \mid au \in \mathcal{F}_n(S)\right\} \cup \\ &\left\{0a\sigma^2(u)010 \mid au \in \mathcal{F}_n(S)\right\} \cup \left\{a\sigma^2(u) \mid au \in \mathcal{F}_{n+1}(S)\right\}\end{aligned}$$

it can be seen that

$$\begin{aligned}p_{4n+1} &= \min\{p_n + n, p_n + n + 1, p_n + n, p_{n+1} + n - 1\} = p_n + n,\\ q_{4n+1} &= \max\{q_n + n, q_n + n + 1, q_n + n, q_{n+1} + n - 1\} = q_n + n + 1.\end{aligned}$$

The claim follows because $\mathcal{P}_S^1(n) = q_n - p_n + 1$ for all n by (1). □

Theorem 3. *For $n \geq 1$ and $m \geq 0$,*

$$\mathcal{P}_S^1(n) = O(\log n), \quad \mathcal{P}_S^1((2 \cdot 4^m + 1)/3) = m + 2, \quad \mathcal{P}_S^1(2^m) = 2.$$

Proof. Follows from Lemma 4 by induction. □

The abelian complexity of S has a logarithmic upper bound and a constant lower bound. These bounds are the best possible increasing bounds.

Corollary 1. *$\mathcal{U}_S^1(n) = \Theta(\log n)$ and $\mathcal{L}_S^1(n) = 2$.*

Let τ be the Thue–Morse morphism defined by $\tau(0) = 01, \tau(1) = 10$. Let

$$T = 0110100110010110100101100110100 1 \cdots$$

be the Thue–Morse word, which is a fixed point of τ. The first values of $\mathcal{P}_T^2$ are

$$2, 4, 6, 8, 6, 8, 10, 8, 6, 8, 8, 10, 10, 10, 8, 8, 6, 8, 10, 10.$$

The 2-abelian equivalence of factors of T can be determined with the help of the following lemma.

Lemma 5. *Words $u, v \in \{0,1\}^*$ are 2-abelian equivalent if and only if*

$$|u| = |v|, \qquad |u|_{00} = |v|_{00}, \qquad |u|_{11} = |v|_{11}, \qquad \textit{and} \qquad \mathrm{pref}_1(u) = \mathrm{pref}_1(v).$$

Proof. The "only if" direction follows immediately from the alternative definition of 2-abelian equivalence. For the other direction, it follows from the assumptions that $|u|_{01} + |u|_{10} = |v|_{01} + |v|_{10}$. In any word $w \in \{0,1\}^*$, the numbers $|w|_{01}$ and $|w|_{10}$ can differ by at most one. If $|w|_{01} + |w|_{10}$ is even, then $|w|_{01} = |w|_{10}$. If it is odd and $\mathrm{pref}_1(w) = 0$, then $|w|_{01} = |w|_{10} + 1$. If it is odd and $\mathrm{pref}_1(w) = 1$, then $|w|_{01} + 1 = |w|_{10}$. This means that $|u|_{01} = |v|_{01}$ and $|u|_{10} = |v|_{10}$ and u and v are 2-abelian equivalent. □

The following lemma states that if u is a factor of T, then the numbers $|u|_{00}$ and $|u|_{11}$ can differ by at most one.

Lemma 6. *In the image of any word under τ, between any two occurrences of* 00 *there is an occurrence of* 11 *and vice versa.*

Proof. 00 can only occur in the middle of $\tau(10)$, and 11 can only occur in the middle of $\tau(01)$. The claim follows because 10's and 01's alternate in all binary words. □

Let u be a factor of T. If $|u|$ and $|u|_{00} + |u|_{11}$ are given, then there are at most 4 possibilities for the 2-abelian equivalence class of u. This is stated in a more precise way in the next lemma. First we define a function ϕ as follows. If $w = a_1 \cdots a_n$, then $\phi(w) = b_1 \cdots b_{n-1}$, where $b_i = 0$ if $a_i a_{i+1} \in \{01, 10\}$ and $b_i = 1$ if $a_i a_{i+1} \in \{00, 11\}$. If $w = a_1 a_2 \cdots$ is an infinite word, then $\phi(w) = b_1 b_2 \cdots$ is defined in an analogous way.

Lemma 7. *Let $u_1, \dots, u_n$ be factors of T. Let $\phi(u_1), \dots, \phi(u_n)$ be abelian equivalent and $|\phi(u_1)|_1 = m$. If m is even, then $u_1, \dots, u_n$ are in at most 2 different 2-abelian equivalence classes, and if m is odd, then $u_1, \dots, u_n$ are in at most 4 different 2-abelian equivalence classes.*

Proof. We have $|u_i|_{00} + |u_i|_{11} = |\phi(u_i)|_1 = m$ for all i. By Lemma 6, we have $\{|u_i|_{00}, |u_i|_{11}\} = \{\lfloor m/2 \rfloor, \lceil m/2 \rceil\}$. If m is even, there are at most two different possible values for the triples $(|u_i|_{00}, |u_i|_{11}, \mathrm{pref}_1(u_i))$, and if m is odd, there are at most four different possible values. The claim follows from Lemma 5. □

Now it can be proved that the 2-abelian complexity of T is of the same order as the abelian complexity of $\phi(T)$. It is known that $\phi(T)$ is actually the period-doubling word S [1].

Lemma 8. *For $n \geq 2$,*

$$\mathcal{P}_S^1(n-1) \leq \mathcal{P}_T^2(n) \leq 3\mathcal{P}_S^1(n-1) + \begin{cases} 0 & \textit{if } \mathcal{P}_S^1(n-1) \textit{ is even} \\ 1 & \textit{if } \mathcal{P}_S^1(n-1) \textit{ is odd.} \end{cases}$$

Proof. If the factors of T of length n are $u_1, \ldots, u_m$, then the factors of $\phi(T)$ of length $n-1$ are $\phi(u_1), \ldots, \phi(u_m)$. If u_i and u_j are 2-abelian equivalent, then $\phi(u_i)$ and $\phi(u_j)$ are abelian equivalent, so the first inequality follows. The second inequality follows from Lemma 7, because the number of different values $|\phi(u_i)|_1$ is $\mathcal{P}_S^1(n-1)$, and at least $\lfloor \mathcal{P}_S^1(n-1)/2 \rfloor$ of these different values are even. □

Theorem 4. *For $n \geq 1$ and $m \geq 0$,*

$$\mathcal{P}_T^2(n) = O(\log n), \quad \mathcal{P}_T^2((2 \cdot 4^m + 4)/3) = \Theta(m), \quad \mathcal{P}_T^2(2^m + 1) \leq 6.$$

Proof. Follows from Lemma 8 and Theorem 3. □

With the help of Lemma 2, we see that the k-abelian complexity of T behaves in a similar way as the abelian complexity of S.

Corollary 2. *Let $k \geq 2$. Then $\mathcal{U}_T^k(n) = \Theta(\log n)$ and $\mathcal{L}_T^k(n) = \Theta(1)$.*

5 Arbitrarily Slowly Growing k-Abelian Complexities

In this section we study whether there is a gap above bounded k-abelian complexity. This question can be formalized in several different ways:

1. Does there exist an increasing unbounded function $f : \mathbb{Z}_+ \to \mathbb{Z}_+$ such that for every infinite word w, either $\mathcal{P}_w^k$ is bounded or $\mathcal{P}_w^k = \Omega(f)$?

2. Does there exist an increasing unbounded function $f : \mathbb{Z}_+ \to \mathbb{Z}_+$ such that for every infinite word w, either $\mathcal{P}_w^k$ is bounded or $\mathcal{P}_w^k \neq O(f)$?

3. Does there exist an increasing unbounded function $f : \mathbb{Z}_+ \to \mathbb{Z}_+$ such that for every infinite word w, either $\liminf \mathcal{P}_w^k < \infty$ or $\mathcal{P}_w^k \neq O(f)$?

The first question has already been answered negatively in Section 4. The answers to the second and third question are also negative. In the case of the second question, we prove this by a uniformly recurrent construction, and in the case of the third question, we prove this by a recurrent construction.

First, consider the second question. Let $n_1, n_2, \ldots$ be a sequence of integers greater than 1. Let $m_j = \prod_{i=1}^{j} n_i$ for $j = 0, 1, 2, \ldots$. Let $a_i = 0$ if the greatest j such that $m_j | i$ is even and $a_i = 1$ otherwise. Let $U = a_1 a_2 a_3 \cdots$. The idea is that the faster the sequence $n_1, n_2, \ldots$ grows, the slower the k-abelian complexity of the word U grows.

The word U could also be described by a Toeplitz-type construction: Start with the word $(0^{n_1 - 1} \diamond)^\omega$, then replace the $\diamond$'s by the letters of $(1^{n_2 - 1} \diamond)^\omega$, then replace the remaining $\diamond$'s by the letters of $(0^{n_3 - 1} \diamond)^\omega$, then replace the remaining $\diamond$'s by the letters of $(1^{n_4 - 1} \diamond)^\omega$, and keep repeating this procedure so that U is obtained as a limit. It follows from the construction that $U \in (\mathrm{pref}_{m_j - 1}(U)\{0, 1\})^\omega$ for all j.

Lemma 9. *The word U is uniformly recurrent.*

Proof. For every factor u of U, there is a j such that u is a factor of $\mathrm{pref}_{m_j-1}(U)$. Because $U \in \{\mathrm{pref}_{m_j-1}(U)0, \mathrm{pref}_{m_j-1}(U)1\}^\omega$, every factor of U of length $2m_j - 2$ contains u. $\square$

Lemma 10. *For every $n \geq 2$, let n' be such that $m_{n'-1} < n \leq m_{n'}$. Then*

$$\mathcal{P}_U^1(n) \leq n' + 1.$$

For all $J \geq 1$, if $n = 2\sum_{j=1}^{J}(m_{2j} - m_{2j-1})$, then

$$\mathcal{P}_U^1(n) \geq \frac{n'+1}{2}.$$

For all $j \geq 1$,

$$\mathcal{P}_U^1(m_j) = 2.$$

Proof. Formula (1) will be used repeatedly in this proof. Another important simple fact is that if a, b, c are integers and c divides a, then $\lfloor (a+b)/c \rfloor = a/c + \lfloor b/c \rfloor$.

For all $n \geq 1$,

$$|\mathrm{pref}_n(U)|_1 = \sum_{i=1}^{\infty} (-1)^{i+1} \left\lfloor \frac{n}{m_i} \right\rfloor,$$

and for all $n \geq 1$ and $l \geq 0$,

$$|\mathrm{rfact}_n^l(U)|_1 = |\mathrm{pref}_{n+l}(U)|_1 - |\mathrm{pref}_l(U)|_1 = \sum_{i=1}^{\infty} (-1)^{i+1} \left(\left\lfloor \frac{n+l}{m_i} \right\rfloor - \left\lfloor \frac{l}{m_i} \right\rfloor \right).$$

For all i,

$$\left\lfloor \frac{(n+l)}{m_i} \right\rfloor - \left\lfloor \frac{l}{m_i} \right\rfloor \in \left\{ \left\lfloor \frac{n}{m_i} \right\rfloor, \left\lceil \frac{n}{m_i} \right\rceil \right\}.$$

Moreover, for every n and l there is an i' such that, for $i \geq n'$,

$$\left\lfloor \frac{n+l}{m_i} \right\rfloor - \left\lfloor \frac{l}{m_i} \right\rfloor = \begin{cases} 1 & \text{if } n' \leq i < i' \\ 0 & \text{if } i \geq i' \end{cases},$$

so

$$\sum_{i=n'}^{\infty} (-1)^{i+1} \left(\left\lfloor \frac{n+l}{m_i} \right\rfloor - \left\lfloor \frac{l}{m_i} \right\rfloor \right) \in \left\{ 0, (-1)^{n'+1} \right\}.$$

Thus there are at most $n' + 1$ possible values for $|\mathrm{rfact}_n^l(U)|_1$ and $\mathcal{P}_U^1(n) \leq n' + 1$.

Consider the second claim. Let $n = 2\sum_{j=1}^{J}(m_{2j} - m_{2j-1})$. The sequence (m_j) is increasing and, moreover, $m_{j+1} \geq 2m_j$ for all j, so by standard estimates for alternating sums,

$$m_{2J} \leq 2(m_{2J} - m_{2J-1}) < n < 2m_{2J} \leq m_{2J+1}.$$

Thus $n' = 2J + 1$. Let $l = m_{2J+1} - n/2$. Then

$$|\mathrm{rfact}_n^l(U)|_1 - |\mathrm{pref}_n(U)|_1 = \sum_{i=1}^{\infty}(-1)^{i+1}\left(\left\lfloor\frac{n+l}{m_i}\right\rfloor - \left\lfloor\frac{l}{m_i}\right\rfloor - \left\lfloor\frac{n}{m_i}\right\rfloor\right)$$

and for $i \leq 2J$ (recall that $m_i | m_j$ when $j \geq i$)

$$\begin{aligned}
&\left\lfloor\frac{(n+l)}{m_i}\right\rfloor - \left\lfloor\frac{l}{m_i}\right\rfloor - \left\lfloor\frac{n}{m_i}\right\rfloor \\
=&\frac{m_{2J+1} + \sum_{(i+1)/2\leq j\leq J}(m_{2j} - m_{2j-1})}{m_i} + \left\lfloor\frac{\sum_{1\leq j<(i+1)/2}(m_{2j} - m_{2j-1})}{m_i}\right\rfloor \\
&-\frac{m_{2J+1} - \sum_{(i+1)/2\leq j\leq J}(m_{2j} - m_{2j-1})}{m_i} - \left\lfloor-\frac{\sum_{1\leq j<(i+1)/2}(m_{2j} - m_{2j-1})}{m_i}\right\rfloor \\
&-\frac{2\sum_{(i+1)/2\leq j\leq J}(m_{2j} - m_{2j-1})}{m_i} - \left\lfloor\frac{2\sum_{1\leq j<(i+1)/2}(m_{2j} - m_{2j-1})}{m_i}\right\rfloor \\
=&\left\lfloor\frac{s}{m_i}\right\rfloor - \left\lfloor-\frac{s}{m_i}\right\rfloor - \left\lfloor\frac{2s}{m_i}\right\rfloor,
\end{aligned}$$

where $s = \sum_{1\leq j<(i+1)/2}(m_{2j} - m_{2j-1})$. If i is even, then $m_i/2 \leq s < m_i$, and if i is odd and $i > 1$, then $m_{i-1}/2 \leq s < m_{i-1}$. Thus

$$\left\lfloor\frac{s}{m_i}\right\rfloor - \left\lfloor-\frac{s}{m_i}\right\rfloor - \left\lfloor\frac{2s}{m_i}\right\rfloor = \begin{cases} 0 & \text{if } i \text{ is even or } i = 1 \\ 1 & \text{if } i \text{ is odd and } i > 1 \end{cases}$$

and

$$\begin{aligned}
\mathcal{P}_U^1(n) &\geq |\mathrm{rfact}_n^l(U)|_1 - |\mathrm{pref}_n(U)|_1 + 1 \\
&= \sum_{i'=2}^{J}(-1)^{(2i'-1)+1} + \sum_{i=2J+1}^{\infty}(-1)^{i+1}\left(\left\lfloor\frac{n+l}{m_i}\right\rfloor - \left\lfloor\frac{l}{m_i}\right\rfloor - \left\lfloor\frac{n}{m_i}\right\rfloor\right) + 1 \\
&= J + 1 = \frac{n'+1}{2}.
\end{aligned}$$

Consider the third claim. Because $U \in \{\mathrm{pref}_{m_j-1}(U)0, \mathrm{pref}_{m_j-1}(U)1\}^\omega$, every factor of U of length m_j is abelian equivalent to either the word $\mathrm{pref}_{m_j-1}(U)0$ or the word $\mathrm{pref}_{m_j-1}(U)1$. Thus $\mathcal{P}_U^1(m_j) \leq 2$. Both $\mathrm{pref}_{m_j-1}(U)0$ and $\mathrm{pref}_{m_j-1}(U)1$ are factors of U, so $\mathcal{P}_U^1(m_j) = 2$. □

If $n_i = 2$ for all i, then the word U is the period-doubling word S. Thus Lemma 10 gives an alternative proof for Corollary 1.

Theorem 5. *For every increasing unbounded function $f : \mathbb{Z}_+ \to \mathbb{Z}_+$, there is a uniformly recurrent word $w \in \{0,1\}^\omega$ such that $\mathcal{P}_w^k(n) = O(f(n))$ but $\mathcal{P}_w^k(n)$ is not bounded.*

Proof. Follows from Lemmas 1, 9 and 10. □

Consider the third question. Let $m_0, m_1, \dots$ be a sequence of positive integers. Let $v_0 = 0^{m_0}1$ and $v_n = v_{n-1}v_{n-1}0^{m_i}$ for $n \geq 1$. Let V be the limit of the sequence $v_0, v_1, v_2, \dots$. Again, the idea is that the faster the sequence $m_0, m_1, \dots$ grows, the slower the k-abelian complexity of the word V grows.

Lemma 11. *The word V is recurrent and* $\liminf \mathcal{P}_V^1(n) = \infty$.

Proof. Every factor of V is a factor of v_n for some n, and v_nv_n is a prefix of V, so every factor appears at least twice in V. Thus V is recurrent.

The word V has factors 0^i for all i, so by (1), $\mathcal{P}_V^1$ is increasing. Moreover, the word V has factors with arbitrarily many 1's, so $\liminf \mathcal{P}_V^1(n) = \infty$. □

Lemma 12. *For every $n \geq m_0 + 2$, let n' be such that $|v_{n'-1}| < n \leq |v_{n'}|$. Then*

$$\mathcal{P}_V^1(n) \leq 2^{n'} + 1.$$

Proof. The word V has factors 0^i for all i, so by (1),

$$\mathcal{P}_V^1(n) = \max\{|v|_1 \mid v \in \mathcal{F}_n(V)\} + 1.$$

Because $V \in (\{v_{n'}\} \cup 0^*)^\omega$,

$$\max\{|v|_1 \mid v \in \mathcal{F}_n(V)\} \leq |v_{n'}|_1 = 2^{n'}.$$

The claim follows. □

Theorem 6. *For every increasing unbounded function $f : \mathbb{Z}_+ \to \mathbb{Z}_+$, there is a recurrent word $w \in \{0,1\}^\omega$ such that $\mathcal{P}_w^k(n) = O(f(n))$ but* $\liminf \mathcal{P}_w^k(n) = \infty$.

Proof. Follows from Lemmas 1, 11 and 12. □

6 Conclusion

In this paper we have investigated some generalizations of the results of Morse and Hedlund and those of Coven and Hedlund for k-abelian complexity. We have pointed out many similarities but also many differences. We have studied the k-abelian complexity of the Thue–Morse word and proved that there are uniformly recurrent words with arbitrarily slowly growing k-abelian complexities.

There are many open questions and possible directions for future work. Inspired by Lemma 3, the relations of k-abelian complexities for different values of k could be studied. In fact, several questions related to this idea were answered in [4]. Another interesting topic would be the k-abelian complexities of morphic words. For example, for a morphic (or pure morphic) word w, how slowly can $\mathcal{U}_w^k(n)$ grow without being bounded? Can it grow slower than logarithmically? More generally, can the possible k-abelian complexities of some subclass of morphic words be classified?

References

[1] Allouche, Jean-Paul, Arnold, André, Berstel, Jean, Brlek, Srečko, Jockusch, William, Plouffe, Simon, and Sagan, Bruce E. A relative of the Thue-Morse sequence. *Discrete Math.*, 139(1–3):455–461, 1995.

[2] Allouche, Jean-Paul and Shallit, Jeffrey. The ring of k-regular sequences. *Theoret. Comput. Sci.*, 98(2):163–197, 1992.

[3] Blanchet-Sadri, Francine, Currie, James, Rampersad, Narad, and Fox, Nathan. Abelian complexity of fixed point of morphism $0 \mapsto 012, 1 \mapsto 02, 2 \mapsto 1$. *Integers*, 14:A11, 2014.

[4] Cassaigne, Julien, Karhumäki, Juhani, and Saarela, Aleksi. On growth and fluctuation of k-abelian complexity. In *Proceedings of the 10th CSR*, volume 9139 of *LNCS*, pages 109–122. Springer, 2015.

[5] Cassaigne, Julien, Richomme, Gwénaël, Saari, Kalle, and Zamboni, Luca Q. Avoiding Abelian powers in binary words with bounded Abelian complexity. *Internat. J. Found. Comput. Sci.*, 22(4):905–920, 2011.

[6] Coven, Ethan M. and Hedlund, Gustav A. Sequences with minimal block growth. *Math. Systems Theory*, 7:138–153, 1973.

[7] Currie, James and Rampersad, Narad. Recurrent words with constant Abelian complexity. *Adv. in Appl. Math.*, 47(1):116–124, 2011.

[8] Damanik, David. Local symmetries in the period-doubling sequence. *Discrete Appl. Math.*, 100(1–2):115–121, 2000.

[9] Dekking, Michel. Strongly nonrepetitive sequences and progression-free sets. *J. Combin. Theory Ser. A*, 27(2):181–185, 1979.

[10] Greinecker, Florian. On the 2-abelian complexity of the Thue–Morse word. *Theoret. Comput. Sci.*, 593:88–105, 2015.

[11] Huova, Mari and Karhumäki, Juhani. Observations and problems on k-abelian avoidability. In *Combinatorial and Algorithmic Aspects of Sequence Processing (Dagstuhl Seminar 11081)*, pages 2215–2219, 2011.

[12] Huova, Mari, Karhumäki, Juhani, and Saarela, Aleksi. Problems in between words and abelian words: k-abelian avoidability. *Theoret. Comput. Sci.*, 454:172–177, 2012.

[13] Huova, Mari, Karhumäki, Juhani, Saarela, Aleksi, and Saari, Kalle. Local squares, periodicity and finite automata. In Calude, Cristian, Rozenberg, Grzegorz, and Salomaa, Arto, editors, *Rainbow of Computer Science*, volume 6570 of *LNCS*, pages 90–101. Springer, 2011.

[14] Karhumäki, Juhani, Puzynina, Svetlana, and Saarela, Aleksi. Fine and Wilf's theorem for k-abelian periods. *Internat. J. Found. Comput. Sci.*, 24(7):1135–1152, 2013.

[15] Karhumäki, Juhani, Saarela, Aleksi, and Zamboni, Luca Q. On a generalization of Abelian equivalence and complexity of infinite words. *J. Combin. Theory Ser. A*, 120(8):2189–2206, 2013.

[16] Karhumäki, Juhani, Saarela, Aleksi, and Zamboni, Luca Q. Variations of the Morse-Hedlund theorem for k-abelian equivalence. In *Proceedings of the 18th DLT*, volume 8633 of *LNCS*, pages 203–214. Springer, 2014.

[17] Keränen, Veikko. Abelian squares are avoidable on 4 letters. In *Proceedings of the 19th ICALP*, volume 623 of *LNCS*, pages 41–52. Springer, 1992.

[18] Mercaş, Robert and Saarela, Aleksi. 3-abelian cubes are avoidable on binary alphabets. In *Proceedings of the 17th DLT*, volume 7907 of *LNCS*, pages 374–383. Springer, 2013.

[19] Morse, Marston and Hedlund, Gustav A. Symbolic dynamics. *Amer. J. Math.*, 60(4):815–866, 1938.

[20] Morse, Marston and Hedlund, Gustav A. Symbolic dynamics II: Sturmian trajectories. *Amer. J. Math.*, 62(1):1–42, 1940.

[21] Parreau, Aline, Rigo, Michel, Rowland, Eric, and Vandomme, Elise. A new approach to the 2-regularity of the l-abelian complexity of 2-automatic sequences. *Electron. J. Combin.*, 22(1):P1.27, 2015.

[22] Puzynina, Svetlana and Zamboni, Luca Q. Abelian returns in Sturmian words. *J. Combin. Theory Ser. A*, 120(2):390–408, 2013.

[23] Richomme, Gwénaël, Saari, Kalle, and Zamboni, Luca Q. Balance and Abelian complexity of the Tribonacci word. *Adv. in Appl. Math.*, 45(2):212–231, 2010.

[24] Richomme, Gwénaël, Saari, Kalle, and Zamboni, Luca Q. Abelian complexity of minimal subshifts. *J. Lond. Math. Soc. (2)*, 83(1):79–95, 2011.

[25] Rigo, Michel and Vandomme, Elise. 2-abelian complexity of the Thue–Morse sequence, 2012. http://hdl.handle.net/2268/135841.

[26] Saarela, Aleksi. Ultimately constant abelian complexity of infinite words. *J. Autom. Lang. Comb.*, 14(3–4):255–258, 2009.

Acta Cybernetica **23** (2017) 191–201.

Initial Algebra for a System of Right-Linear Functors[*]

Anna Labella[a] and Rocco De Nicola[b]

Abstract

In 2003 we showed that right-linear systems of equations over regular expressions, when interpreted in a category of trees, have a solution whenever they enjoy a specific property that we called *hierarchicity* and that is instrumental to avoid critical mutual recursive definitions. In this note, we prove that a right-linear system of polynomial endofunctors on a cocartesian monoidal closed category which enjoys parameterized *left list arithmeticity*, has an initial algebra, provided it satisfies a property similar to hierarchicity.

Keywords: regular expressions, monoidal categories, system of functors

1 Introduction

Our paper [4] acknowledges that "*the ideas that led to the work stemmed from discussions with Zoltán Ésik*"; as a homage to Zoltán here we generalise the results of [4] to a much larger setting. There we defined the class of the linear systems whose solution is expressible as a tuple of nondeterministic regular expressions [3] when they are interpreted as trees of actions rather than as sets of action sequences. We exactly characterized those systems that have a regular expression as a "canonical" solution, and showed that any regular expression can be obtained as a canonical solution of a system of the defined class.

The key ingredient for obtaining the wanted solution was our restriction to "hierarchical" equations that were instrumental to avoid critical mutual recursive definitions. Indeed, if we model variables as nodes of graphs and their dependences as directed arcs, we required that whenever a variable y depends on x, (x is at the beginning of a loop that contains y) we have that y never occurs in other loops originated by other variables different from x.

Thus, in [4] we proved that a right-linear system of equations, interpreted in a category of trees has a solution whenever it is hierarchical. In this short note

[*]This work is dedicated to Zoltán Ésik whose unexpected and untimely death left us shattered and without words.

[a]Dipartimento di Informatica, Sapienza Università di Roma, E-mail: labella@di.uniroma1.it
[b]IMT School for Advanced Studies Lucca, E-mail: rocco.denicola@imtlucca.it

DOI: 10.14232/actacyb.23.1.2017.12

we prove that a right-linear system of polynomial endofunctors on a cocartesian monoidal closed category which enjoys parameterized *left list arithmeticity*, has an initial algebra, provided it satisfies a property similar again to a hierarchicity condition. We could thus say that the “solution” for the system provided here is canonical in a strict sense.

2 Initial algebras and *llist*-arithmeticity

In order to introduce an initial algebra for a linear polynomial endofunctor expressed in terms of (canonical) sum $+$ and a possibly non commutative tensor product $\otimes$, we have to consider a notion of recursive object which generalizes Cockett definition [2] of $rec(U, V)$, where the canonical product $\times$ played the role of multiplication. As a matter of fact, we still ask for an initial algebra for an endofunctor $U \otimes (-) + V : C \to C$ in a monoidal category $(C, \otimes, I)$, but we have to be aware of a non commutative situation. We chose to have the *left composition*, because our result is particularly meaningful for categories which are monoidal (right) closed whose objects have an elegant representation (see Proposition 1).

Definition 1. *Given a cocartesian monoidal category $(C, \otimes, I)$, we call U^*V the initial algebra of the functor $U \otimes (-) + V$, if it does exist. In that case there is a morphism $U \otimes (U^*V) + V \to U^*V$ canonical w.r.t. any other $U \otimes (-) + V$-algebra. This means that, U^*V is equipped with two morphisms ρ_0, ρ_1 such that, given another object X with two similar morphisms x_0, x_1, there is a unique morphism λ making the following diagram commute.*

$$\begin{tikzcd}
V \arrow[r, "\rho_0"'] \arrow[dr, "x_0"'] & U^*V \arrow[d, "\lambda"] & U \otimes U^*V \arrow[l, "\rho_1"'] \arrow[d, "U \otimes \lambda"] \\
 & X & U \otimes X \arrow[l, "x_1"]
\end{tikzcd}$$

In case C is a partial order, U^*V is the minimal solution of the corresponding inequation $U \otimes X + V \leq X$. But, in any case, being U^*V an initial algebra, we have that $U \otimes U^*V + V \simeq U^*V$ ($U \otimes U^*V + V = U^*V$, in the case of partial order), i.e. it is an initial fixed point.

When $\otimes$ is the canonical product, we do get the well known definition of $rec(U, V)$ provided by Cockett [2], i.e., the V-parameterized $list(U)$, that becomes $list(U)$ when $V \simeq \mathbf{1}$ is the terminal object. Since the constant value of the tensor product we consider is on the left and the tensor product is non-commutative, we will talk about left lists, that we will refer as *llist* and as *parameterized llist*.

For a generic tensor product, $\otimes$, we have that the initial algebra of $U \otimes (-) + V$ is U^*V that we call parameterized *llist*(U); in case $V \simeq I$ the initial algebra is U^*I that we call *llist*(U).

One can easily prove (see Adámek theorem in [1]) that in a monoidal cocartesian category, which has colimits for every countable chain, there is an initial algebra for all the functors above. Such initial algebras can be obtained as initial fixed points, i.e., as colimits of the chain built starting from the initial object 0 and then repeatedly applying the functor.

Proposition 1. *In a monoidal cocartesian, chain cocomplete category C, semidistributive on the right, in the sense of [5], we have that:*

1. *There is a canonical morphism $U^*V \to U^*I \otimes V$*
2. *If tensor product distributes on the right w.r.t. chain colimits, e.g. it has a right adjoint, then we have $U^*V \simeq U^*I \otimes V$*

Proof.

1. It suffices to prove that $U^*I \otimes V$ is a fixed point of the same functor as U^*V. From this, the existence of the required canonical morphism would follow because U^*V is the initial fixed point. To prove that $U^*I \otimes V$ is a fixed point, let us apply the functor $U \otimes (-) + V$ to $U^*I \otimes V$. By using the associativity law and the right distributivity law, we get the following series of isomorphisms:

$$U \otimes (U^*I \otimes V) + V \simeq (U \otimes U^*I) \otimes V + I \otimes V \simeq (I + U \otimes U^*I) \otimes V \simeq U^*I \otimes V.$$

2. If the tensor product preserves chain colimits, it preserves also fixed points. In particular it is true in case C is monoidal (right)-closed.

□

If we write U^* instead of U^*I, Proposition 1 allows us to interchangeably use $U^* \otimes V$ and U^*V when working with monoidal closed categories.

By relying on Proposition 1 we have that if C has $llist(U)$, it has also parameterized $llist(U)$. In analogy with the case of categories with cartesian product where a category having (parameterized) $list$s is called $list$-arithmetic [1], we will call our category $left - list$-arithmetic or $llist$-arithmetic when it has (parameterized) $llist$s.

Proposition 2. *Given a cocartesian monoidal right closed category C which has initial algebra for the functor $U \otimes (-) + I$, it has initial algebra for all the functors $U \otimes (-) + V$.*

Proof. The proof follows from Proposition 1. If U^* is an initial algebra for functor $U \otimes (-) + I$, $U^* \otimes V$ is an initial algebra for functor $U \otimes (-) + V$. □

We can now consider three instances of $llist$-arithmetic categories that build on A^*, the free monoid generated by an alphabet A:

[1]This name is related to the fact that, when a *list*-arithmetic category is also a pretopos, it is possible to develop arithmetic in it and we speak of an arithmetic universe in the sense of Joyal [7].

$P(A^*)$ the algebra of sets of words on A, a monoidal category w.r.t. concatenation whose morphisms are inclusions. Here parameterized *llist*(U,V), i.e. U^*V, is the binary Kleene star U^*V and as a consequence of Proposition 1 we have that it is reducible to the unary star because $P(A^*)$ is a monoidal right closed category (the derivation operation is right adjoint to concatenation). In this case, the tensor product distributes over sums on both sides.

$Set|A^*$ the topos of A^*-labelled sets, where the (non-commutative) tensor product is obtained from the concatenation in A^*. By taking a (commutative) monoid M, we could obtain from $Set|M$ a (commutative) monoidal structure.

$Tree(A)$ is generalization of $P(A^*)$. Structured sets of computations are organised as a category of generalised trees built over a (complete) meet-semilattice monoid generated from A. The tensor product $\otimes$ is provided by the concatenation of trees allowed by the concatenation of A^*. This concatenation is non commutative and only right-distributive w.r.t. sums [5], but also right closed. The category $Tree(A)$ has initial algebra for functors $s \otimes (-) + t$, i.e it is *llist*-arithmetic with the *llist* s^*t given by iteration of a tree s, followed every time by a copy of t [4][2].

3 Right-linear hierarchical systems of functors

It is a result of classical theory of regular languages [8] that we can consider a grammar on an alphabet A as a continuous operator from $P(A^*)^n$ to $P(A^*)^n$ consisting of a system of n linear equations in n variables. This system can be "solved" by repeatedly applying the rule

$$X = U^*V \text{ implies } X = U \otimes X + V \qquad (* - rule)$$

In this way, we obtain a minimal fixed point for the operator associated with the grammar. In the present categorical context, we could say this rule is a direct consequence the *llist*-arithmeticity of the considered structure.

In [4], we extended this result to the category $Tree(A)$, but, due to the fact that only a right side distributivity of tensor product w.r.t. sum holds, we had to restrict the class of solvable systems by considering only so-called right-linear hierarchical systems (*rlhs*) that allowed us to avoid critical mutual recursive definitions. Formulated according to the current terminology the result of [4] is described by the following proposition.

Proposition 3. *In the category $Tree(A)$, the $* - rule$ provides a solution for hierarchical (see below) finite right-linear systems of polynomial equations.* $\square$

Now we do generalize this result again and show that *llist*-arithmeticity in a cocartesian right-distributive monoidal category C all finite right-linear hierarchical systems of functors have an initial algebras.

[2]Actually, $Tree(A)$ is a coherent *llist*-arithmetic category, but not a pre-topos because not all its monos are regular.

When such a category C contains as its objects the elements of an alphabet A, some of the objects of C can be rendered as regular expressions generated by means of the following BNF starting from the elments a of an alphabet A.

$$E ::= \; 0 \mid I \mid a \mid E+E \mid E\otimes E \mid E^* \qquad \text{where } a \text{ is in } A.$$

In such a grammar, 0 denotes the initial object of our category, I denotes the unit for $\otimes$, that is the tensor product of C. Moreover $+$ stands for the coproduct of C and * denotes the *llist*-constructor.

Our result will be formulated by relying on such terminology. Indeed, if we suppose that C is cocartesian monoidal closed and elements of A are its objects, then the interpretation of *llist*s will allow the construction of parameterized *llist*s, as described in Proposition 2.

Our aim is to prove that, by relying on the following rule

$$U\otimes X+V\rightarrow X \;\; \textit{implies} \;\; U^*V\rightarrow X \qquad\qquad (\textit{initiality}-\textit{rule})$$

that guarantees that if there is a morphism $U\otimes X+V\rightarrow X$ then there is a canonical morphism $U^*V\rightarrow X$, it is possible to find an initial algebra for every right-linear hierarchical system of functors on regular expressions.

Summing up, we will extend the result proved in [4] for $Tree(A)$ to a category C with the properties mentioned above. To this aim we have to formulate it in terms of functors instead of equations of linear functions in order to prove that the obtained solution is canonical because it is the initial algebra of the system of functors.

We need to provide some definitions.

Definition 2.

- *Given a category C interpreting regular expressions, a functor $F: C^n\rightarrow C$ of the form $\sum_{1\leq i\leq n} U_i\otimes X_i+V$ is called* right-linear *polynomial functor in n variables.*

- *A right-linear polynomial functor is called* simple *when all U_i and V do not contain the $()^*$ operator.*

- *A right-linear polynomial system of functors of dimension n is a n-tuple*

$$\Phi=<F_1,\ldots,F_n>: C^n\rightarrow C^n$$

of right-linear polynomial functors in n variables.

Given a functor $F: C\times D\rightarrow C$, for every object d of D, we can consider the endofunctor $F((-),d): C\rightarrow C$.

Moreover, if F is a functor with n-argument $\sum_{1\leq i\leq n} U_i\otimes X_i+V: C^n\rightarrow C$, we can write it as $U_1\otimes X_1+\sum_{2\leq i\leq n} U_i\otimes X_i+V: C\times C^{n-1}\rightarrow C$.

Then, taking d as $\sum_{2\leq i\leq n} U_i\otimes X_i+V$, *llist*-arithmeticity implies that the functor $U_1\otimes(-)+d: C\rightarrow C$ has an initial algebra $U_1^*\otimes d$ for every d. Obviously, the same can be done for every index i.

Let us recall now Bekič theorem about initial algebras of functors [6] that is important because it means that the simultaneous construction of an initial algebra for a system of n operators in n variables can be replaced by recursively constructing of initial algebras for one operator at a time.

Bekič theorem
Given two functors $F : C \times D \to C$ and $G : C \times D \to D$, let $(F^{\mu}(d), \chi_d)$ be an initial $F((-), d)$-algebra for each object d of D and suppose that there exists an initial algebra, say $< \xi, \zeta >$ with $\xi : F^{\mu}(\beta) \to \alpha$, $\zeta : G(\alpha, \beta) \to \beta$, of the functor $< F^{\mu} \circ pr_D, G >: C \times D \to C \times D$, where the first component is obtained by composing the projection $pr_D : C \times D \to D$ with the functor constructing the D-parameterized initial algebra $F^{\mu} : D \to C$; then the pair $< \chi_{\beta}, G(\xi, \beta) \bullet \zeta >$ where

- $\chi_{\beta} : F(F^{\mu}(\beta), \beta) \to F^{\mu}(\beta)$
- $G(\xi, \beta) \bullet \zeta : G(F^{\mu}(\beta), \beta) \to \beta$

is an initial algebra of the functor $< F, G >: C \times D \to C \times D$.

$\square$

To understand the impact of this theorem in our context, let us consider a simple case where F and G are two right-linear polynomial functors over two variables[3] and C coincides with D:

$$\begin{cases} F \equiv ax + by \\ G \equiv a'x + b'y + c' \end{cases}$$

We take the initial algebra a^*by (not depending on x) associated with the first functor, when it is considered as $F((-), by)$, and then we substitute this value in the expression of G to obtain $a'a^*by + b'y + c'$. We can get from this an initial algebra for the pair $< F, G >$, i.e. for the system. Indeed, using distributivity on the right, we get $(a'a^*b + b')y + c'$ and thus, thanks to the initiality rule, we get $(a'a^*b + b')^*c'$ as the second component of the initial algebra for the functor system above.

It is worth noting that this has been possible only because there was no constant term in the definition of F. Indeed, the slightly different system

$$\begin{cases} F' \equiv ax + by + c \\ G \equiv a'x + b'y + c' \end{cases}$$

is not solvable using the same machinery. In fact, in this case, we would obtain $a^*(by + c)$ as initial algebra for $F'((-), by + c)$, and once this is substituted in G we would get $a'a^*(by + c) + b'y + c'$, but then, due to the lack of left distributivity, the initiality rule cannot be used.

[3]In the sequel we will often omit the symbol $\otimes$ using juxtaposition to replace it and use small letters for variables.

In general, for n functors, we have that, if F is F_1, then G is $< F_2, \ldots, F_n >$ and the initial algebras can be inductively obtained by performing appropriate substitutions. We fix a variable, say x_1, in the expression of F_1 and consider the parameterization of F_1 w.r.t. the sum of all the monomials not containing x_1. We calculate the parameterized initial algebra and substitute this value everywhere for x_1. Please notice that a constant functor has this constant with its identity as an initial algebra.

Summing up, in our case the first requirement of Bekič's theorem is always satisfied because our category is *llist*-arithmetic; but we have to impose additional conditions on the system of functors in order to meet the second requirement.

Given the system $\Phi \equiv < F_1, \ldots, F_n >$ with variables $x_1 \ldots, x_n$, let us now define a different indexing for both, functors and variables. This will allow us to introduce a (partial) order on the set of variables in Φ in such a way that we can exclude their mutual interference when we build the initial algebra step by step. The partial ordering, $\leq$, is obtained by using a string of natural numbers as index for every variable while guaranteeing that two different variables do not have the same index. For any two indexed variables x_s and x_t, we will write $x_s \leq x_t$ if t is a prefix of s.

We will consider *hierarchical* (*rlhs*) any system for which it is possible to introduce an indexing that satisfies a number of conditions that we will introduce below.

Definition 3. *Let $x_s \leq x_t$, we say that*

- x_s *is* ruled by x_t *if x_t appears in F_s.*
- x_s *is* recursive *if x_s appears in the expression of F_s;*
- x_s *is* strictly recursive *if it is recursive and is not ruled by any other variable.*

Definition 4. Right Linear Hierarchical Systems - rlhs
A system of right-linear functors Φ whose variables are ordered by $\leq$ is hierarchical *if it is possible to associate, as index to every functor, a common prefix of the indexes of the variables appearing in its expression (either of the same length or at most one number longer), with the only possible exception for one of the variables, in case this rules on all the others. Moreover, the indexing has to guarantee that:*

1. *the ordering of the indexes of the functors is tree-shaped;*
2. *a variable can be ruled by at most another one;*
3. *If x_i appears in F_i (it is recursive), then alternatively, either F_i does contain a constant different from* 0*, or it does contain a variable ruling on x_i;*
4. *If x_i does not appear in F_i, then all variables that are immediately smaller than x_i (one number more in their index) appear in F_i and, some of them can have a common ruling variable while the others are strictly recursive in the functor corresponding to them.*

If we look at the examples above, we have that we can index the first system as

$$\begin{cases} F_0 \equiv ax_0 + bx_\epsilon \\ F_\epsilon \equiv a'x_0 + b'x_\epsilon + c' \end{cases}$$

to obtain a *rlhs*.

Due to the presence of a constant in both the expressions, we cannot provide a similar ordering for the second system. Indeed, none of the two functors can satisfy condition 3. of Definition 4.

$$\begin{cases} F_0 \equiv ax_0 + bx_\epsilon + c \\ F_\epsilon \equiv a'x_0 + b'x_\epsilon + c' \end{cases}$$

We give now a more complex example of a right-linear system that can be indexed in such a way that it is *rlhs*. Please notice that, in the example below, we provide directly the indexed equational system. The original one can be recovered by giving different names to the variables and the functors with different indexes. After presenting the indexed system we also outline the procedure to obtain its initial algebra.

Example 1. Consider the following system

$$\Phi :< F_{211}, F_{212}, F_{21}, F_{22}, F_{11}, F_1, F_2, F_\epsilon >$$

where we have indexed functors and, accordingly, their variables.

$F_{211} \equiv cx_{211} + x_2$

$F_{212} \equiv ax_{212} + bx_2$

$F_{21} \equiv x_{211} + x_{212}$

$F_{22} \equiv cx_{22} + I$

$F_{11} \equiv ax_{11} + b$

$F_1 \equiv x_{11} + c$

$F_2 \equiv x_{21} + x_{22}$

$F_\epsilon \equiv ax_1 + x_2 + a$

All variables under x_{21} depend on the ruling variable x_2 (which becomes recursive when substitutions are made into F_2), and the corresponding functors do not contain any constant. We will start from F_{211} and F_{212}, that are leaf functors according to $\leq$. The parametrized initial algebra for F_{211} is c^*x_2 while the one for F_{212} is a^*bx_2. Since F_{211} and F_{212} have no constant and x_{211} and x_{212} have the same ruling variable, we can substitute c^*x_2 and a^*bx_2 in F_{21} to obtain $c^*x_2 + a^*bx_2 = (c^* + a^*b)x_2$ thanks to right distributivity. We can now consider F_{22}. Its initial algebra is the constant c^*, thus in F_2 we can replace x_{21} and x_{22} with

$(c^* + a^*b)x_2$ and c^*, which yields $(c^* + a^*b)x_2 + c^*$ that has $(c^* + a^*b)^*c^*$ as initial algebra. We consider now F_{11}, whose initial algebra is a^*b, that we substitute in F_1 to obtain $(a^*b) + c$ as initial algebra; finally once we substitute all variables in F_ϵ with the corresponding initial algebras we get the constant $(a(a^*b + c) + (c^* + a^*b)^*c^* + a$. Which is the basis for obtaining the full solution by means of appropriate substitutions.

Theorem 1. *In a right semidistributive category C where we have a parameterized initial algebra for linear polynomial functors (parameterized llists), all right-linear hierarchical systems of functors, with chosen indexing, have a family of regular expressions as their initial algebra.*

Proof. Given a hierarchical right-linear functors system, we can find an initial algebra for it by repeatedly using the $(initiality - rule)$ above, and by relying on Bekič theorem. This theorem provides an initial algebra for a system of functors in presence of parameterized initial algebras; to take advantage of it we need to show that the restricted set of hierarchical systems satisfy its two conditions. The first condition, i.e. the existence of parameterized initial algebras for a chosen functor in a recursive variable holds by hypothesis, the second one corresponds to the fact that the reduced system (with fewer functors) obtained after substituting the initial algebra has still an initial algebra. We then proceed by induction on the length of indexes starting from the longest ones. This is possible because, by exploiting right distributivity, we can take out a (ruling) variable as a common factor from terms containing it. This is due to our definition of *rlhs*.

Let us start by considering a functor F_i which has maximal index. The expression corresponding to F_i cannot contain variables with indexes longer than i. Thus, the expression may contain x_i and at most a single variable, say x_t ruling on it (t is a strict prefix of i), moreover when such ruling variable is present the expression does not contain any constants.

Let us consider the two cases separately:

1. In case x_i is strictly recursive, we obtain as initial algebra of F_i a constant term which might be 0 in case the expression contains only 0 as a constant.

2. In case the expression of F_i contains a variable x_t ruling on x_i, the initiality rule gives a parameterized (w.r.t. x_t) initial algebra.

When substituting these terms in F_s with s an immediate prefix of i ($i = s\,n$), we obtain a sum of constants and of terms all containing the same variable x_t (due to condition 4. in Definition 4, no other term with another ruling variable can be present in F_s). We can then take x_t as a common factor. Now we distinguish two cases, if $t = s$ we can proceed as above because all variables with an index longer than t have been eliminated. If t is instead a strict prefix of s, we can operate further substitutions until we reach functor F_t possibly having other terms with the same variable and nomore variables with a longer index. In this way we have in any case reduced the system to a smaller one, still *rlhs*, producing an initial algebra at every step. At this point we can apply the procedure again. □

It could seem that a very particular kind of linear functors system is taken into account, but we can prove that any regular expression, when interpreted in C is the initial algebra of some finite linear hierarchical system of functors.

Theorem 2. *Given a cocartesian monoidal semidistributive category C, where we can interpret regular expressions in such a way that the $()^*$ operator is the llist-operator corresponding to the tensor product $\otimes$, every regular expression E can be obtained as the initial algebra of the root component of a n-tuple of right-linear hierarchical system of simple polynomial functors $< F_1, \ldots, F_n >: C^n \to C^n$.* $\square$

In order to prove item 1. of Theorem 2 we need to transform every regular expression in normal form and we will show that any normal form can be first associated with a system of simple quadratic polynomial functors and that the system can be associated with a *rlhs* of simple linear polynomial functors. The way we obtain such a *rlhs* guarantees that the original regular expression E is the component in the initial algebra of the generated *rlhs* associated to the root functor.

Here we omit the details of the proof, it proceeds along the same lines of the corresponding one in [4] while referring to functors rather than to equations. In particular we need to use normal forms similar to that of Definition 1. in [4] and functor systems associated to them like in Definition 5. of [4]. From the functor systems we will do obtain system of quadratic functors (like in Proposition 1. in [4]) which we transform into linear ones (Proposition 2. in [4]). The fact that the regular expression E is the initial algebra of the root component of the system will descend from the construction, while the verification of the hierarchicity of the system is now almost immediate by the chosen indexing strategy.

4 Conclusions

A classical result of the theory of regular languages [8] states that we can obtain solutions of systems of linear equations over regular expressions interpreted as languages variables.

In [4] we showed that right-linear systems of equations over regular expressions, when interpreted in a category of trees, have a solution whenever they enjoy a specific property that we called *hierarchicity.*

Here, we have completed the generalisation by considering cocartesian non commutative monoidal categories where the tensor product preserves colimits and a property similar to hierarchicity is satisfied. The key requirement for this kind of categories was the presence of an iteration operator thought of as initial algebra of a linear polynomial functor. The existence of such initial algebra is a form of a one-side list arithmeticity. Now list arithmeticity is a key ingredient to develop arithmetics in a pretopos [7]: this fact could suggest further investigations about a connection between results in (possibly non deterministic) language theory and in an arithmetic based on a one-side natural number object.

References

[1] Adámek, J., Koubek, V. Least Fixed Point of a Functor. *Journal of Computer and System Sciences*, 19: 163–178, 1979.

[2] Cockett, J.R.B. *list*-arithmetic distributive categories: locoi. *Journal of Pure and Applied Algebra*, 66: 1–29, 1990.

[3] Corradini, F., De Nicola, R., Labella, A. Models of Nondeterministic Regular Expressions. *Journal of Computer and System Science*, 59: 412–449, 1999.

[4] De Nicola, R., Labella, A. Nondeterministic regular expressions as solutions of equational systems, *Theoretical Computer Science*, 302: 179–189, 2003.

[5] Labella, A. Categories with sums and right distributive tensor product. *Journal of Pure and Applied Algebra*, 178 (3): 273–296, 2003.

[6] Lehmann, D. J., Smyth, M. B. Algebraic specification of data types: a synthetic approach. *Mathematical Systems Theory*, 14 (2): 97-139, 1981.

[7] Maietti, M.E. Joyal's arithmetic universe as list-arithmetic pretopos, *Theory and Applications of Categories*, 24 (3): 39–83, 2010.

[8] Moll, R.N., Arbib, M.A., Kfoury, A.J. *An Introduction to Formal Language Theory* Springer-Verlag, Berlin, 1987.

Acta Cybernetica **23** (2017) 203–228.

An Algebraic Approach to Energy Problems I *-Continuous Kleene ω-Algebras[‡]

Zoltán Ésik[a], Uli Fahrenberg[b], Axel Legay[c], and Karin Quaas[d]

Abstract

Energy problems are important in the formal analysis of embedded or autonomous systems. With the purpose of unifying a number of approaches to energy problems found in the literature, we introduce energy automata. These are finite automata whose edges are labeled with energy functions that define how energy levels evolve during transitions.

Motivated by this application and in order to compute with energy functions, we introduce a new algebraic structure of *-continuous Kleene ω-algebras. These involve a *-continuous Kleene algebra with a *-continuous action on a semimodule and an infinite product operation that is also *-continuous.

We define both a finitary and a non-finitary version of *-continuous Kleene ω-algebras. We then establish some of their properties, including a characterization of the free finitary *-continuous Kleene ω-algebras. We also show that every *-continuous Kleene ω-algebra gives rise to an iteration semiring-semimodule pair.

Keywords: Energy problem, Kleene algebra, *-continuity, *-continuous Kleene ω-algebra

1 Introduction

Energy problems are concerned with the question whether a given system admits infinite schedules during which (1) certain tasks can be repeatedly accomplished and (2) the system never runs out of energy (or other specified resources). These are important in areas such as embedded systems or autonomous systems and,

[‡]This research was supported by grant no. K 108448 from the National Foundation of Hungary for Scientific Research (OTKA), by ANR MALTHY, grant no. ANR-13-INSE-0003 from the French National Research Foundation, and by Deutsche Forschungsgemeinschaft (DFG), projects QU 316/1-1 and QU 316/1-2.

[a]University of Szeged, Hungary (deceased)

[b]École polytechnique, Palaiseau, France. Most of this work was carried out while this author was still employed at Inria Rennes.

[c]Inria Rennes, France

[d]Universität Leipzig, Germany

DOI: 10.14232/actacyb.23.1.2017.13

starting with [4], have attracted some attention in recent years, for example in [20, 27, 3, 5, 28, 7, 6, 23, 9].

With the purpose of generalizing some of the above approaches, we have in [14, 21] introduced *energy automata*. These are finite automata whose transitions are labeled with *energy functions* which specify how energy values change from one system state to another. Using the theory of semiring-weighted automata [10], we have shown in [14] that energy problems in such automata can be solved in a simple static way which only involves manipulations of energy functions.

In order to put the work of [14] on a more solid theoretical footing and with an eye to future generalizations, we have recently introduced a new algebraic structure of **-continuous Kleene ω-algebras* [12, 13].

A continuous (or complete) Kleene algebra is a Kleene algebra in which all suprema exist and are preserved by products. These have nice algebraic properties, but not all Kleene algebras are continuous, for example the semiring of regular languages over some alphabet. Hence a theory of *-continuous Kleene algebras has been developed to cover this and other interesting cases [25].

For infinite behaviors, complete semiring-semimodule pairs involving an infinite product operation have been developed [19]. Motivated by some examples of structures which are not complete in this sense, for example the energy functions of the preceding section, we generalize the notion of *-continuous Kleene algebra to one of *-continuous Kleene ω-algebra. These are idempotent semiring-semimodule pairs which are not necessarily complete, but have enough suprema in order to develop a fixed-point theory and solve weighted Büchi automata (*i.e.,* to compute infinitary power series).

We will define both a finitary and a non-finitary version of *-continuous Kleene ω-algebras. We then establish several properties of *-continuous Kleene ω-algebras, including the existence of the suprema of certain subsets related to regular ω-languages. Then we will use these results in our characterization of the free finitary *-continuous Kleene ω-algebras. We also show that each *-continuous Kleene ω-algebra gives rise to an iteration semiring-semimodule pair.

Structure of the Paper This is the first in a series of two papers which deal with energy problems and their algebraic foundation. In the present paper, we motivate the introduction of our new algebraic structures by two sections on energy automata (Section 2) and on the algebraic structure of energy functions (Section 3). We then pass to introduce continuous Kleene ω-algebras in Section 4 and to expose the free continuous Kleene ω-algebras in Section 5.

In Section 6 we generalize continuous Kleene ω-algebras to our central notion of *-continuous Kleene ω-algebras and finitary *-continuous Kleene ω-algebras. Section 7 exposes the free finitary *-continuous Kleene ω-algebras; the question whether general free *-continuous Kleene ω-algebras exist is left open.

The penultimate Section 8 shows that every *-continuous Kleene ω-algebra is an iteration semiring-semimodule pair, hence techniques from matrix semiring-semimodule pairs apply. This will be important in the second paper of the series. In Section 9 we concern ourselves with least and greatest fixed points and introduce a notion of Kleene ω-algebra, analogous to the concept of Kleene algebra for least fixed points.

In the second paper of the series [15], we show that one can use matrix operations to solve reachability and Büchi acceptance in weighted automata over *-continuous Kleene ω-algebras, and that energy functions form a *-continuous Kleene ω-algebra. This will allows us to connect the algebraic structures developed in the present paper back to their motivating energy problems.

Acknowledgment The origin of this work is a joint short paper [21] between the last three authors which was presented at the 2012 International Workshop on Weighted Automata: Theory and Applications. After the presentation, the presenter was approached by Zoltán Ésik, who told him that the proper setting for energy problems should be idempotent semiring-semimodule pairs. This initiated a long-lasting collaboration, including several mutual visits, which eventually led to the work presented in this paper and its follow-up [15].

We are deeply indebted to our colleague and friend Zoltán Ésik who taught us all we know about semiring-semimodule pairs and *-continuity. Unfortunately Zoltán could not see this work completed, so any errors are the responsibility of the last three authors.

In honor of Zoltán Ésik, we propose to give the name "*Ésik algebra*" to *-continuous Kleene ω-algebras.

2 Energy Automata

The transition labels on the energy automata which we consider in this paper will be functions which model transformations of energy levels between system states. Such transformations have the (natural) properties that below a certain energy level, the transition might be disabled (not enough energy is available to perform the transition), and an increase in input energy always yields at least the same increase in output energy. Thus the following definition.

Definition 1. *An* energy function *is a partial function* $f : \mathbb{R}_{\geq 0} \rightharpoonup \mathbb{R}_{\geq 0}$ *which is defined on a closed interval* $[l_f, \infty[$ *or on an open interval* $]l_f, \infty[$*, for some lower bound* $l_f \geq 0$*, and such that for all* $x \leq y$ *for which* f *is defined,*

$$yf \geq xf + y - x\,. \tag{$*$}$$

The class of all energy functions is denoted by $\mathcal{F}$*.*

We will write composition and application of energy functions in diagrammatical order, from left to right. Hence we write $f; g$, or simply fg, for the composition $g \circ f$ and $x; f$ or xf for function application $f(x)$. This is because we will be concerned with *algebras* of energy functions, in which function composition is multiplication, and where it is customary to write multiplication in diagrammatical order.

Thus energy functions are strictly increasing, and in points where they are differentiable, the derivative is at least 1. The inverse functions to energy functions

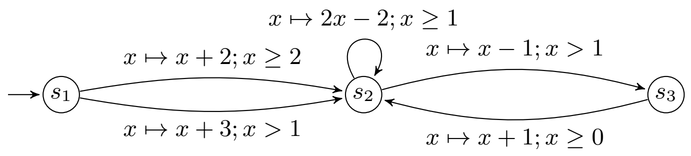

$x \mapsto 2x-2; x \geq 1$
$x \mapsto x+2; x \geq 2$
$x \mapsto x-1; x > 1$
s_1
s_2
s_3
$x \mapsto x+3; x > 1$
$x \mapsto x+1; x \geq 0$

Figure 1: A simple energy automaton.

exist, but are generally not energy functions. Energy functions can be composed, where it is understood that for a composition fg, the interval of definition is $\{x \in \mathbb{R}_{\geq 0} \mid xf \text{ and } xfg \text{ defined}\}$. The following lemma shows an important property of energy functions which we will use repeatedly later, mostly without mention of the lemma.

Lemma 1. *Let $f \in \mathcal{F}$ and $x \in \mathbb{R}_{\geq 0}$.*

- *If $xf < x$, then there is $N \geq 0$ such that xf^n is undefined for all $n \geq N$.*
- *If $xf = x$, then $xf^n = x$ for all $n \geq 0$.*
- *If $xf > x$, then for all $P \in \mathbb{R}$ there is $N \geq 0$ such that $xf^n \geq P$ for all $n \geq N$.*

Proof. In the first case, we have $x - xf = M > 0$. Using $(*)$, we see that $xf^{n+1} \leq xf^n - M$ for all $n \geq 0$ for which xf^{n+1} is defined. Hence the sequence $(xf^n)_{n \geq 0}$ decreases without bound, so that there must be $N \geq 0$ such that xf^N is undefined, and then so is xf^n for any $n > N$.

The second case is trivial. In the third case, we have $xf - x = M > 0$. Again using $(*)$, we see that $xf^{n+1} > xf^n + M$ for all $n \geq 0$. Hence the sequence $(xf^n)_{n \geq 0}$ increases without bound, so that for any $P \in \mathbb{R}$ there must be $N \geq 0$ for which $xf^N \geq P$, and then $xf^n \geq xf^N \geq P$ for all $n \geq N$. □

Example 1. The following example shows that property $(*)$ is not only sufficient for Lemma 1, but in a sense also necessary: Let $\alpha \in \mathbb{R}$ with $0 < \alpha < 1$ and $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be the function $xf = 1 + \alpha x$. Then $yf = xf + \alpha(y - x)$ for all $x \leq y$, so $(*)$ "almost" holds. But $xf^n = \sum_{i=0}^{n-1} \alpha^i + \alpha^n x$ for all $n \in \mathbb{N}$, hence $\lim_{n \to \infty} xf^n = \frac{1}{1-\alpha} < \infty$.

Definition 2. *An* energy automaton (S, s_0, T, F) *consists of a finite set S of states, with initial state $s_0 \in S$, a finite set $T \subseteq S \times \mathcal{F} \times S$ of transitions labeled with energy functions, and a subset $F \subseteq S$ of acceptance states.*

Example 2. Figure 1 shows a simple energy automaton. Here we have used inequalities to give the definition intervals of energy functions, so that for example, the function labeling the loop at s_2 is given by $f(x) = 2x - 2$ for $x \geq 1$ and undefined for $x < 1$.

A finite *path* in an energy automaton is a finite sequence of transitions $\pi = (s_0, f_1, s_1), (s_1, f_2, s_2), \dots, (s_{n-1}, f_n, s_n)$. We use f_π to denote the combined energy function $f_\pi = f_1 f_2 \cdots f_n$ of such a finite path. We will also use infinite paths, but note that these generally do not allow for combined energy functions.

A *global state* of an energy automaton is a pair $q = (s, x)$ with $s \in S$ and $x \in \mathbb{R}_{\geq 0}$. A transition between global states is of the form $((s, x), f, (s', x'))$ such that $(s, f, s') \in T$ and $x' = f(x)$. A (finite or infinite) *run* of (S, T) is a path in the graph of global states and transitions.

We are ready to state the decision problems with which our main concern will lie. As the input to a decision problem must be in some way finitely representable, we will state them for subclasses $\mathcal{F}' \subseteq \mathcal{F}$ of *computable* energy functions; an $\mathcal{F}'$-automaton is an energy automaton (S, s_0, T, F) with $T \subseteq S \times \mathcal{F}' \times S$. Note that we give no technical meaning to the term "computable" here; we simply need to take care that the input be finitely representable.

Problem 1 (State reachability)**.** Given an $\mathcal{F}'$-automaton $A = (S, s_0, T, F)$ and a computable initial energy $x_0 \in \mathbb{R}_{\geq 0}$: does there exist a finite run of A from (s_0, x_0) which ends in a state in F?

Problem 2 (Coverability)**.** Given an $\mathcal{F}'$-automaton $A = (S, s_0, T, F)$, a computable initial energy $x_0 \in \mathbb{R}_{\geq 0}$ and a computable function $z : F \to \mathbb{R}_{\geq 0}$: does there exist a finite run of A from (s_0, x_0) which ends in a global state (s, x) such that $s \in F$ and $x \geq sz$?

Problem 3 (Büchi acceptance)**.** Given an $\mathcal{F}'$-automaton $A = (S, s_0, T, F)$ and a computable initial energy $x_0 \in \mathbb{R}_{\geq 0}$: does there exist an infinite run of A from (s_0, x_0) which visits F infinitely often?

As customary, a run such as in the statements above is said to be *accepting.* The special case of Problem 3 with $F = S$ is the question whether there *exists an infinite run* in the given energy automaton. This is what is usually referred to as *energy problems* in the literature; our extension to general Büchi conditions has not been treated before.

3 The Algebra of Energy Functions

Let $[0, \infty]_\perp = \{\perp\} \cup [0, \infty]$ denote the complete lattice of non-negative real numbers together with extra elements $\perp$ and ∞, with the standard order on $\mathbb{R}_{\geq 0}$ extended by $\perp < x < \infty$ for all $x \in \mathbb{R}_{\geq 0}$. Also, $\perp + x = \perp - x = \perp$ for all $x \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ and $\infty + x = \infty - x = \infty$ for all $x \in \mathbb{R}_{\geq 0}$.

Definition 3. *An* extended energy function *is a mapping $f : [0, \infty]_\perp \to [0, \infty]_\perp$, for which $\perp f = \perp$ and $yf \geq xf + y - x$ for all $x \leq y$. Moreover, $\infty f = \infty$, unless $xf = \perp$ for all $x \in [0, \infty]_\perp$. The class of all extended energy functions is denoted $\mathcal{E}$.*

This means, in particular, that $xf = \perp$ implies $yf = \perp$ for all $y \leq x$, and $xf = \infty$ implies $yf = \infty$ for all $y \geq x$. Hence, except for the extension to ∞, these functions are indeed the same as the energy functions from Definition 1. More precisely, every energy function $f : \mathbb{R}_{\geq 0} \rightharpoonup \mathbb{R}_{\geq 0}$ as of Definition 1 gives rise to an extended energy function $\tilde{f} : [0, \infty]_\perp \to [0, \infty]_\perp$ given by $\perp \tilde{f} = \perp$, $x\tilde{f} = \perp$ if xf is undefined, $x\tilde{f} = xf$ otherwise for $x \in \mathbb{R}_{\geq 0}$, and $\infty\tilde{f} = \infty$.

Composition of extended energy functions is defined as before, but needs no more special consideration about its definition interval.

We define a partial order on $\mathcal{E}$, by $f \leq g$ iff $xf \leq xg$ for all $x \in [0, \infty]_\perp$. We will need three special energy functions, $\perp\!\!\!\perp$, id and $\top\!\!\!\top$; these are given by $x\perp\!\!\!\perp = \perp$, $x; \mathsf{id} = x$ for $x \in [0, \infty]_\perp$, and $\perp\top\!\!\!\top = \perp$, $x\top\!\!\!\top = \infty$ for $x \in [0, \infty]$.

Lemma 2. *With the ordering $\leq$, $\mathcal{E}$ is a complete lattice with bottom element $\perp\!\!\!\perp$ and top element $\top\!\!\!\top$. The supremum on $\mathcal{E}$ is pointwise, i.e., $x(\sup_{i\in I} f_i) = \sup_{i\in I} xf_i$ for any set I, all $f_i \in \mathcal{E}$ and $x \in [0, \infty]_\perp$. Also, $h(\sup_{i\in I} f_i) = \sup_{i\in I}(hf_i)$ for all $h \in \mathcal{E}$.*

Proof. The pointwise supremum of any set of extended energy functions is an extended energy function. Indeed, if f_i, $i \in I$ are extended energy functions and $x < y$ in $\mathbb{R}_{\geq 0}$, then $yf_i \geq xf_i + y - x$ for all i. It follows that $\sup_{i\in I} yf_i \geq \sup_{i\in I} xf_i + y - x$. Also, since $\perp f_i = \perp$ for all $i \in I$, $\sup_{i\in I} \perp f_i = \perp$. Finally, if there is some i such that $\infty f_i = \infty$, then $\sup_{i\in I} \infty f_i = \infty$. Otherwise each function f_i is constant with value $\perp$.

The fact that $h(\sup_{i\in I} f_i) = \sup_{i\in I} hf_i$ is now clear, since the supremum is taken pointwise: For all x, $x(h(\sup_{i\in I} f_i)) = (xh)(\sup_{i\in I} f_i) = \sup_{i\in I} xhf_i = x(\sup_i hf_i)$. □

We denote binary suprema using the symbol $\vee$; hence $f \vee g$, for $f, g \in \mathcal{E}$, is the function $x(f \vee g) = \max(xf, xg)$.

Recall that an *idempotent semiring* [1, 22] $S = (S, \vee, \cdot, \perp, 1)$ consists of a commutative idempotent monoid $(S, \vee, \perp)$ and a monoid $(S, \cdot, 1)$ such that the distributive laws

$$x(y \vee z) = xy \vee xz$$
$$(y \vee z)x = yx \vee zx$$

and the zero laws

$$\perp \cdot x = \perp = x \cdot \perp$$

hold for all $x, y, z \in S$. It follows that the product operation distributes over all finite sums.

Each idempotent semiring S is partially ordered by its *natural order* relation $x \leq y$ iff $x \vee y = y$, and then sum and product preserve the partial order and $\perp$ is the least element. Moreover, for all $x, y \in S$, $x \vee y$ is the least upper bound of the set $\{x, y\}$.

Lemma 3. $(\mathcal{E}, \vee, ;, \perp\!\!\!\perp, \mathsf{id})$ *is an idempotent semiring with natural order $\leq$.*

Proof. It is clear that $(\mathcal{E}, \vee, \perp\!\!\!\perp)$ is a commutative idempotent monoid and that $(\mathcal{E}, ;, \mathsf{id})$ is a monoid. $\leq$ is the natural order on $\mathcal{E}$ because $\vee$ is given pointwise. It is also clear that $\perp\!\!\!\perp f = f \perp\!\!\!\perp = \perp\!\!\!\perp$ for all $f \in \mathcal{E}$.

To show distributivity, we have already shown that $x(h(f \vee g)) = x(hf \vee hg)$ in the proof of Lemma 2; using monotonicity of h, we also have

$$x((f \vee g)h) = x(f \vee g)h = (xf \vee xg)h = xfg \vee xgh = x(fh \vee gh)\,.$$

The proof is complete. $\square$

We will show in the second paper [15] of this series that $\mathcal{E}$ in fact forms a **-continuous Kleene algebra* [25], which will allow us to solve energy problems algebraically.

4 Continuous Kleene Algebras and Continuous Kleene ω-Algebras

We have already recalled the notion of idempotent semiring in the last section. A *homomorphism* of idempotent semirings $(S, \vee, \cdot, \perp, 1)$, $(S', \vee', \cdot', \perp', 1')$ is a function $h : S \to S'$ which respects the constants and operations, *i.e.,* such that $h(\perp) = \perp'$, $h(1) = 1'$, $h(x \vee y) = h(x) \vee' h(y)$, and $h(x \cdot y) = h(x) \cdot' h(y)$ for all $x, y \in S$.

A Kleene algebra [25] is an idempotent semiring $S = (S, \vee, \cdot, \perp, 1)$ equipped with a star operation ${}^* : S \to S$ such that for all $x, y \in S$, yx^* is the least solution of the fixed point equation $z = zx \vee y$ and x^*y is the least solution of the fixed point equation $z = xz \vee y$ with respect to the natural order. A Kleene algebra *homomorphism* is a semiring homomorphism h which respects the star: $h(x^*) = (h(x))^*$ for all $x \in S$.

Examples of Kleene algebras include the language semiring $P(A^*)$ over an alphabet A, whose elements are the subsets of the set A^* of all finite words over A, and whose operations are set union and concatenation, with the languages $\emptyset$ and $\{\varepsilon\}$ serving as $\perp$ and 1. Here, ε denotes the empty word. The star operation is the usual Kleene star: $X^* = \bigcup_{n \geq 0} X^n = \{u_1 \dots u_n : u_1, \dots, u_n \in X,\ n \geq 0\}$.

Another example is the Kleene algebra $P(A \times A)$ of binary relations over any set A, whose operations are union and relational composition (written in diagrammatic order), and where the empty relation $\emptyset$ and the identity relation id serve as the constants $\perp$ and 1. The star operation is the formation of the reflexive-transitive closure, so that $R^* = \bigcup_{n \geq 0} R^n$ for all $R \in P(A \times A)$.

The above examples are in fact *continuous Kleene algebras*, *i.e.,* idempotent semirings S such that equipped with the natural order, they are complete lattices (hence all suprema exist), and the product operation preserves arbitrary suprema in either argument:

$$y(\bigvee X) = \bigvee yX \quad \text{and} \quad (\bigvee X)y = \bigvee Xy$$

for all $X \subseteq S$ and $y \in S$. The star operation is given by $x^* = \bigvee_{n \geq 0} x^n$, so that x^* is the supremum of the set $\{x^n : n \geq 0\}$ of all powers of x.

Homomorphisms of continuous Kleene algebras S, S' are homomorphisms of idempotent semirings $h : S \to S'$ which respect arbitrary suprema: $h(\bigvee X) = \bigvee h(X) = \bigvee\{h(x) \mid x \in X\}$ for all $X \subseteq S$. To distinguish these from semiring homomorphisms, they are sometimes called *continuous homomorphisms*, but we will not do this here.

A larger class of models is given by the **-continuous Kleene algebras* [25]. By the definition of a *-continuous Kleene algebra $S = (S, \vee, \cdot, \perp, 1)$, all suprema of sets of the form $\{x^n \mid n \geq 0\}$ are required to exist, where x is any element of S, and x^* is given by this supremum. Moreover, product preserves such suprema in both arguments:

$$y(\bigvee_{n\geq 0} x^n) = \bigvee_{n\geq 0} yx^n \quad \text{and} \quad (\bigvee_{n\geq 0} x^n)y = \bigvee_{n\geq 0} x^n y\,.$$

Every *-continuous Kleene algebra is a Kleene algebra. For any alphabet A, the collection $R(A^*)$ of all regular languages over A is an example of a *-continuous Kleene algebra which is not continuous. There exist Kleene algebras which are not *-continuous, see [25]. For non-idempotent extensions of the notions of continuous Kleene algebras, *-continuous Kleene algebras and Kleene algebras, we refer to [17, 16]. *Homomorphisms* of *-continuous Kleene algebras are the Kleene algebra homomorphisms.

Recall that an *idempotent semiring-semimodule pair* [19, 2] (S, V) consists of an idempotent semiring $S = (S, \vee, \cdot, \perp, 1)$ and a commutative idempotent monoid $V = (V, \vee, \perp)$ which is equipped with a left S-action $S \times V \to V$, $(x, v) \mapsto xv$, satisfying

$$\begin{aligned} (x \vee x')v &= xv \vee x'v & \qquad x(v \vee v') &= xv \vee xv' \\ (xx')v &= x(x'v) & \perp v &= \perp \\ x\perp &= \perp & 1v &= v \end{aligned}$$

for all $x, x' \in S$ and $v \in V$. In that case, we also call V a *(left) S-semimodule*.

A *homomorphism* of semiring-semimodule pairs (S, V) and (S', V') is a pair $h = (h_S, h_V)$ of functions $h_S : S \to S'$ and $h_V : V \to V'$ such that h_S is a semiring homomorphism, h_V is a monoid homomorphism, and h respects the action, *i.e.*, $h_V(xv) = h_S(x)h_V(v)$ for all $x \in S$ and $v \in V$.

Definition 4. *A* continuous Kleene ω-algebra *is an idempotent semiring-semimodule pair (S, V) in which S is a continuous Kleene algebra, V is a complete lattice with the natural order, and the action preserves all suprema in either argument. Additionally, there is an infinite product operation which is compatible with the action and associative in the sense that the following hold:*

1. *For all $x_0, x_1, \ldots \in S$, $\prod_{n\geq 0} x_n = x_0 \prod_{n\geq 0} x_{n+1}$.*
2. *Let $x_0, x_1, \ldots \in S$ and $0 = n_0 \leq n_1 \cdots$ be a sequence which increases without a bound. Let $y_k = x_{n_k} \cdots x_{n_{k+1}-1}$ for all $k \geq 0$. Then $\prod_{n\geq 0} x_n = \prod_{k\geq 0} y_k$.*

Moreover, the infinite product operation preserves all suprema:

3. $\prod_{n\geq 0}(\bigvee X_n) = \bigvee\{\prod_{n\geq 0} x_n : x_n \in X_n,\ n \geq 0\}$*, for all* $X_0, X_1, \ldots \subseteq S$*.*

The above notion of continuous Kleene ω-algebra may be seen as a special case of the not necessarily idempotent complete semiring-semimodule pairs of [19]. A *homomorphism* of continuous Kleene ω-algebras is a semiring-semimodule homomorphism $h = (h_S, h_V)$ such that h_S is a homomorphism of continuous Kleene algebras, h_V preserves all suprema, and h respects infinite products: for all $x_0, x_1, \ldots \in S$, $h_V(\prod_{n\geq 0} x_n) = \prod_{n\geq 0} h_S(x_n)$.

One of our aims in this paper is to provide an extension of the notion of continuous Kleene ω-algebras to **-continuous Kleene* ω*-algebras*, which are semiring-semimodule pairs (S, V) consisting of a *-continuous Kleene algebra S acting on a necessarily idempotent semimodule V, such that the action preserves certain suprema in its first argument, and which are equipped with an infinite product operation satisfying the above compatibility and associativity conditions and some weaker forms of the last axiom.

5 Free Continuous Kleene ω-Algebras

In this section, we offer descriptions of the free continuous Kleene ω-algebras and the free continuous Kleene ω-algebras satisfying the identity $1^\omega = \perp$. We recall the following folklore result.

Theorem 1. *For each set A, the language semiring* $(P(A^*), \vee, \cdot, \perp, 1)$ *is the free continuous Kleene algebra on A.*

In more detail, if S is a continuous Kleene algebra and $h : A \to S$ is any function, then there is a unique homomorphism $h^\sharp : P(A^*) \to S$ of continuous Kleene algebras which extends h.

In view of Theorem 1, it is not surprising that the free continuous Kleene ω-algebras can be described using languages of finite and infinite words. Suppose that A is a set. Let A^ω denote the set of all ω-words over A and $A^\infty = A^* \cup A^\omega$. Let $P(A^*)$ denote the language semiring over A and $P(A^\infty)$ the semimodule of all subsets of A^∞ equipped with the action of $P(A^*)$ defined by $XY = \{xy : x \in X, y \in Y\}$ for all $X \subseteq A^*$ and $Y \subseteq A^\infty$. We also define an infinite product by $\prod_{n\geq 0} X_n = \{u_0u_1 \ldots : u_n \in X_n\}$. It is clear that $(P(A^*), P(A^\infty))$ is a continuous Kleene ω-algebra.

Theorem 2. *For each set A,* $(P(A^*), P(A^\infty))$ *is the free continuous Kleene* ω*-algebra on A.*

Proof. Suppose that (S, V) is any continuous Kleene ω-algebra an let $h : A \to S$ be a mapping. We want to show that there is a unique extension of h to a homomorphism $(h_S^\sharp, h_V^\sharp)$ from $(P(A^*), P(A^\infty))$ to (S, V).

For each $u = a_0 \ldots a_{n-1}$ in A^*, define $h_S(u) = h(a_0) \cdots h(a_{n-1})$ and $h_V(u) = h(a_0) \cdots h(a_{n-1})1^\omega = \prod_{k\geq 0} b_k$, where $b_k = a_k$ for all $k < n$ and $b_k = 1$ for all

$k \geq n$. When $u = a_0 a_1 \ldots \in A^\omega$, define $h_V(u) = \prod_{k \geq 0} h(a_k)$. Note that we have $h_S(uv) = h_S(u)h_S(v)$ for all $u, v \in A^*$ and $h_S(\varepsilon) = 1$. Also, $h_V(uv) = h_S(u)h_V(v)$ for all $u \in A^*$ and $v \in A^\infty$. Thus, $h_V(XY) = h_S(X)h_V(Y)$ for all $X \subseteq A^*$ and $Y \subseteq A^\infty$. Moreover, for all $u_0, u_1, \ldots$ in A^*, if $u_i \neq \varepsilon$ for infinitely many i, then $h_V(u_0 u_1 \ldots) = \prod_{k \geq 0} h_S(u_k)$. If on the other hand, $u_k = \varepsilon$ for all $k \geq n$, then $h_V(u_0 u_1 \ldots) = h_S(u_0) \cdots h_S(u_{n-1})1^\omega$. In either case, if $X_0, X_1, \ldots \subseteq A^*$, then $h_V(\prod_{n \geq 0} X_n) = \prod_{n \geq 0} h_S(X_n)$.

Suppose now that $X \subseteq A^*$ and $Y \subseteq A^\infty$. We define $h_S^\sharp(X) = \bigvee h_S(X)$ and $h_V^\sharp(Y) = \bigvee h_V(Y)$. It is well-known that $h_S^\sharp$ is a continuous semiring morphism $P(A^*) \to S$. Also, $h_V^\sharp$ preserves arbitrary suprema, since $h_V^\sharp(\bigcup_{i \in I} Y_i) = \bigvee h_V(\bigcup_{i \in I} Y_i) = \bigvee \bigcup_{i \in I} h_V(Y_i) = \bigvee_{i \in I} \bigvee h_V(Y_i) = \bigvee_{i \in I} h_V^\sharp(Y_i)$.

We prove that the action is preserved. Let $X \subseteq A^*$ and $Y \subseteq A^\infty$. Then $h_V^\sharp(XY) = \bigvee h_V(XY) = \bigvee h_S(X)h_V(Y) = \bigvee h_S(X) \bigvee h_V(Y) = h_S^\sharp(X)h_V^\sharp(Y)$.

Finally, we prove that the infinite product is preserved. Let $X_0, X_1, \ldots \subseteq A^*$. Then $h_V^\sharp(\prod_{n \geq 0} X_n) = \bigvee h_V(\prod_{n \geq 0} X_n) = \bigvee \prod_{n \geq 0} h_S(X_n) = \prod_{n \geq 0} \bigvee h_S(X_n) = \prod_{n \geq 0} h_S^\sharp(X_n)$.

It is clear that h_S extends h, and that (h_S, h_V) is unique. □

Consider now $(P(A^*), P(A^\omega))$ with infinite product defined, as a restriction of the above infinite product, by $\prod_{n \geq 0} X_n = \{u_0 u_1 \ldots \in A^\omega : u_n \in X_n,\ n \geq 0\}$. It is also a continuous Kleene ω-algebra. Moreover, it satisfies $1^\omega = \bot$.

Lemma 4. *$(P(A^*), P(A^\omega))$ is a quotient of $(P(A^*), P(A^\infty))$ under the homomorphism (φ_S, φ_V) such that φ_S is the identity on $P(A^*)$ and φ_V maps $Y \subseteq A^\infty$ to $Y \cap A^\omega$.*

Proof. Suppose that $Y_i \subseteq A^\infty$ for all $i \in I$. It holds that $\varphi_V(\bigcup_{i \in I} Y_i) = A^\omega \cap \bigcup_{i \in I} Y_i = \bigcup_{i \in I}(A^\omega \cap Y_i) = \bigcup_{i \in I} \varphi_V(Y_i)$.

Let $X \subseteq A^*$ and $Y \subseteq A^\infty$. Then $h_V(XY) = XY \cap A^\omega = X(Y \cap A^\omega) = \varphi_S(X)\varphi_V(Y)$.

Finally, let $X_0, X_1, \ldots \subseteq A^*$. Then $h_V(\prod_{n \geq 0} X_n) = \{u_0 u_1 \ldots \in A^\omega : u_n \in X_n\} = \prod_{n \geq 0} h_S(X_n)$. □

Lemma 5. *Suppose that (S, V) is a continuous Kleene ω-algebra satisfying $1^\omega = \bot$. Let (h_S, h_V) be a homomorphism $(P(A^*), P(A^\infty)) \to (S, V)$. Then (h_S, h_V) factors through (φ_S, φ_V).*

Proof. Define $h'_S = h_S$ and $h'_V : P(A^\omega) \to V$ by $h'_V(Y) = h_V(Y)$, for all $Y \subseteq A^\omega$. Then clearly $h_S = h'_S \circ \varphi_S$. Moreover, $h_V = h'_V \circ \varphi_V$, since for all $Y \subseteq A^\infty$, $h'_V(\varphi_V(Y)) = h_V(Y \cap A^\omega) = h_V(Y \cap A^\omega) \vee h_S(Y \cap A^*)1^\omega = h_V(Y \cap A^\omega) \vee h_V((Y \cap A^*)1^\omega) = h_V((Y \cap A^\omega) \cup (Y \cap A^*)1^\omega) = h_V(Y)$.

Since (φ_S, φ_V) and (h_S, h_V) are homomorphisms, so is (h'_S, h'_V). It is clear that h'_V preserves all suprema. □

Theorem 3. *For each set A, $(P(A^*), P(A^\omega))$ is the free continuous Kleene ω-algebra on A satisfying $1^\omega = \bot$.*

Proof. Suppose that (S, V) is a continuous Kleene ω-algebra satisfying $1^\omega = \perp$. Let $h : A \to S$. By Theorem 2, there is a unique homomorphism $(h_S, h_V) : (P(A^*), P(A^\infty)) \to (S, V)$ extending h. By Lemma 5, h_S and h_V factor as $h_S = h'_S \circ \varphi_S$ and $h_V = h'_V \circ \varphi_V$, where (h'_S, h'_V) is a homomorphism $(P(A^*), P(A^\omega)) \to (S, V)$. This homomorphism (h'_S, h'_V) is the required extension of h to a homomorphism $(P(A^*), P(A^\omega)) \to (S, V)$. Since the factorization is unique, so is this extension. □

6 *-Continuous Kleene ω-Algebras

In this section, we define **-continuous Kleene ω-algebras* and *finitary *-continuous Kleene ω-algebras* as an extension of the *-continuous Kleene algebras of [24]. We establish several basic properties of these structures, including the existence of the suprema of certain subsets corresponding to regular ω-languages.

Definition 5. *A* generalized *-continuous Kleene algebra *is a semiring-semimodule pair (S, V) in which S is a *-continuous Kleene algebra (hence S and V are idempotent), subject to the usual laws of unitary action as well as the following axiom*

Ax0: For all $x, y \in S$ and $v \in V$, $xy^*v = \bigvee_{n\geq 0} xy^n v$.

Definition 6. *A* *-continuous Kleene ω-algebra *is a generalized *-continuous Kleene algebra (S, V) together with an infinite product operation $S^\omega \to V$ which maps every ω-word $x_0x_1\ldots$ over S to an element $\prod_{n\geq 0} x_n$ of V, subject to the following axioms:*

Ax1: For all $x_0, x_1, \ldots \in S$, $\prod_{n\geq 0} x_n = x_0 \prod_{n\geq 0} x_{n+1}$.

Ax2: Let $x_0, x_1, \ldots \in S$ and $0 = n_0 \leq n_1 \cdots$ be a sequence which increases without a bound. Let $y_k = x_{n_k} \cdots x_{n_{k+1}-1}$ for all $k \geq 0$. Then $\prod_{n\geq 0} x_n = \prod_{k\geq 0} y_k$.

Ax3: For all $x_0, x_1, \ldots$ and y, z in S, $\prod_{n\geq 0}(x_n(y \vee z)) = \bigvee_{x'_n\in\{y,z\}} \prod_{n\geq 0} x_n x'_n$.

Ax4: For all $x, y_0, y_1, \ldots \in S$, $\prod_{n\geq 0} x^* y_n = \bigvee_{k_n\geq 0} \prod_{n\geq 0} x^{k_n} y_n$.

The first two axioms are the same as for continuous Kleene ω-algebras. The last two are weaker forms of the complete preservation of suprema of continuous Kleene ω-algebras. It is clear that every continuous Kleene ω-algebra is *-continuous.

A *homomorphism* of *-continuous Kleene ω-algebras is a semiring-semimodule homomorphism $h = (h_S, h_V) : (S, V) \to (S', V')$ such that h_S is a *-continuous Kleene algebra homomorphism and h respects infinite products: for all $x_0, x_1, \ldots \in S$, $h_V(\prod_{n\geq 0} x_n) = \prod_{n\geq 0} h_S(x_n)$.

Some of our results will also hold for weaker structures. We define a *finitary *-continuous Kleene ω-algebra* as a structure (S, V) as above, equipped with a star operation and an infinite product $\prod_{n\geq 0} x_n$ restricted to *finitary ω-words* over S, *i.e.,* to sequences $x_0, x_1, \ldots$ such that there is a finite subset F of S such that each x_n is a finite product of elements of F. (Note that F is not fixed and may depend on the sequence $x_0, x_1, \dots$) It is required that the axioms hold whenever the involved ω-words are finitary.

The above axioms have a number of consequences. For example, if $x_0, x_1, \ldots \in S$ and $x_i = \bot$ for some i, then $\prod_{n \geq 0} x_n = \bot$. Indeed, if $x_i = \bot$, then $\prod_{n \geq 0} x_n = x_0 \cdots x_i \prod_{n \geq i+1} x_n = \bot \prod_{n \geq i+1} x_n = \bot$. By Ax1 and Ax2, each *-continuous Kleene ω-algebra is an ω-semigroup [26].

Suppose that (S, V) is a *-continuous Kleene ω-algebra. For each word $w \in S^*$ there is a corresponding element $\overline{w}$ of S which is the product of the letters of w in the semiring S. Similarly, when $w \in S^*V$, there is an element $\overline{w}$ of V corresponding to w, and when $X \subseteq S^*$ or $X \subseteq S^*V$, then we can associate with X the set $\overline{X} = \{\overline{w} : w \in X\}$, which is a subset of S or V. Below we will denote $\overline{w}$ and $\overline{X}$ by just w and X, respectively.

The following two lemmas are well-known and follow from the fact that the semirings of regular languages are the free *-continuous Kleene algebras [24] (and also the free Kleene algebras [25]).

Lemma 6. *Suppose that S is a *-continuous Kleene algebra. If $R \subseteq S^*$ is regular, then $\bigvee R$ exists. Moreover, for all $x, y \in S$, $x(\bigvee R)y = \bigvee xRy$.*

Lemma 7. *Let S be a *-continuous Kleene algebra. Suppose that R, R_1 and R_2 are regular subsets of S^*. Then*

$$\begin{aligned} \bigvee(R_1 \cup R_2) &= \bigvee R_1 \vee \bigvee R_2 \\ \bigvee(R_1 R_2) &= (\bigvee R_1)(\bigvee R_2) \\ \bigvee(R^*) &= (\bigvee R)^*. \end{aligned}$$

In a similar way, we can prove:

Lemma 8. *Let (S, V) be a generalized *-continuous Kleene algebra. If $R \subseteq S^*$ is regular, $x \in S$ and $v \in V$, then $x(\bigvee R)v = \bigvee xRv$.*

Proof. Suppose that $R = \emptyset$. Then $x(\bigvee R)v = \bot = \bigvee xRv$. If R is a singleton set $\{y\}$, then $x(\bigvee R)v = xyv = \bigvee xRv$. Suppose now that $R = R_1 \cup R_2$ or $R = R_1 R_2$, where R_1, R_2 are regular, and suppose that our claim holds for R_1 and R_2. Then, if $R = R_1 \cup R_2$,

$$\begin{aligned} x(\bigvee R)v &= x(\bigvee R_1 \vee \bigvee R_2)v \quad \text{(by Lemma 7)} \\ &= x(\bigvee R_1)v \vee x(\bigvee R_2)v \\ &= \bigvee xR_1v \vee \bigvee xR_2v \\ &= \bigvee x(R_1 \cup R_2)v = \bigvee xRv, \end{aligned}$$

where the third equality uses the induction hypothesis. If $R = R_1R_2$, then

$$\begin{aligned}
x(\bigvee R)v &= x(\bigvee R_1)(\bigvee R_2)v \quad \text{(by Lemma 7)}\\
&= \bigvee(xR_1(\bigvee R_2)v)\\
&= \bigvee\{y(\bigvee R_2)v : y \in xR_1\}\\
&= \bigvee\{\bigvee yR_2v : y \in xR_1\}\\
&= \bigvee xR_1R_2v = \bigvee xRv,
\end{aligned}$$

where the second equality uses the induction hypothesis for R_1 and the fourth the one for R_2. Suppose last that $R = R_0^*$, where R_0 is regular and our claim holds for R_0. Then, using the previous case, it follows by induction that

$$x(\bigvee R_0^n)v = \bigvee xR_0^nv \tag{1}$$

for all $n \geq 0$. Using this and Ax0, it follows now that

$$\begin{aligned}
x(\bigvee R)v = x(\bigvee R_0^*)y &= x(\bigvee_{n\geq 0} \bigvee R_0^n)v\\
&= x(\bigvee_{n\geq 0} (\bigvee R_0)^n)v \quad \text{(by Lemma 7)}\\
&= \bigvee_{n\geq 0} x(\bigvee R_0)^n v \quad \text{(by}\mathsf{Ax0}\text{)}\\
&= \bigvee_{n\geq 0} x(\bigvee R_0^n)v \quad \text{(by Lemma 7)}\\
&= \bigvee_{n\geq 0} \bigvee xR_0^nv \quad \text{(by (1))}\\
&= \bigvee xR_0^*v = \bigvee xRv.
\end{aligned}$$

The proof is complete. □

Lemma 9. *Let (S, V) be a *-continuous Kleene ω-algebra. Suppose that the languages $R_0, R_1, \ldots \subseteq S^*$ are regular and that $\mathcal{R} = \{R_0, R_1, \ldots\}$ is a finite set. Moreover, let $x_0, x_1, \ldots \in S$. Then*

$$\prod_{n\geq 0} x_n(\bigvee R_n) = \bigvee \prod_{n\geq 0} x_nR_n.$$

Proof. If one of the R_i is empty, our claim is clear since both sides are equal to $\perp$, so we suppose they are all nonempty.

Below we will suppose that each regular language comes with a fixed decomposition having a minimal number of operations needed to obtain the language from the empty set and singleton sets. For a regular language R, let $|R|$ denote the minimum number of operations needed to construct it. When $\mathcal{R}$ is a finite set of regular languages, let $\mathcal{R}_{\mathrm{ns}}$ denote the set of non-singleton languages in it. Let $|\mathcal{R}| = \sum_{R\in\mathcal{R}_{\mathrm{ns}}} 3^{|R|}$. Our definition ensures that if $\mathcal{R} = \{R, R_1, \dots, R_n\}$ and $R = R' \cup R''$ or $R = R'R''$ according to the fixed minimal decomposition of R, and if $\mathcal{R}' = \{R', R'', R_1, \dots, R_n\}$, then $|\mathcal{R}'| < |\mathcal{R}|$. Similarly, if $R = R_0^*$ by the fixed minimal decomposition and $\mathcal{R}' = \{R_0, R_1, \dots, R_n\}$, then $|\mathcal{R}'| < |\mathcal{R}|$.

We will argue by induction on $|\mathcal{R}|$.

When $|\mathcal{R}| = 0$, then $\mathcal{R}$ consists of singleton languages and our claim follows from $\mathsf{Ax3}$. Suppose that $|\mathcal{R}| > 0$. Let R be a non-singleton language appearing in $\mathcal{R}$. If R appears only a finite number of times among the R_n, then there is some m such that R_n is different from R for all $n \geq m$. Then,

$$\begin{aligned}
\prod_{n\geq 0} x_n(\bigvee R_n) &= \prod_{i<m} x_i(\bigvee R_i) \prod_{n\geq m} x_n(\bigvee R_n) \quad \text{(by } \mathsf{Ax1}\text{)}\\
&= (\bigvee x_0R_0 \cdots x_{n-1}R_{n-1}) \prod_{n\geq m} x_n(\bigvee R_n) \quad \text{(by Lemma 7)}\\
&= \bigvee(x_0R_0 \cdots x_{n-1}R_{n-1} \prod_{n\geq m} x_n(\bigvee R_n)) \quad \text{(by Lemma 8)}\\
&= \bigvee\{y \prod_{n\geq m} x_n(\bigvee R_n) : y \in x_0R_0 \cdots x_{n-1}R_{n-1}\}\\
&= \bigvee\{\bigvee y \prod_{n\geq m} x_nR_n : y \in x_0R_0 \cdots x_{n-1}R_{n-1}\}\\
&= \bigvee \prod_{n\geq 0} x_nR_n,
\end{aligned}$$

where the passage from the 4th line to the 5th uses induction hypothesis and $\mathsf{Ax1}$.

Suppose now that R appears an infinite number of times among the R_n. Let $R_{i_1}, R_{i_2}, \dots$ be all the occurrences of R among the R_n. Define

$$\begin{aligned}
y_0 &= x_0(\bigvee R_0) \cdots (\bigvee R_{i_1-1})x_{i_1}\\
y_j &= x_{i_j+1}(\bigvee R_{i_j+1}) \cdots (\bigvee R_{i_{j+1}-1})x_{i_{j+1}},
\end{aligned}$$

for $j \geq 1$. Similarly, define

$$\begin{aligned}
Y_0 &= x_0R_0 \cdots R_{i_1-1}x_{i_1}\\
Y_j &= x_{i_j+1}R_{i_j+1} \cdots R_{i_{j+1}-1}x_{i_{j+1}},
\end{aligned}$$

for all $j \geq 1$. It follows from Lemma 7 that

$$y_j = \bigvee Y_j$$

for all $j \geq 0$. Then

$$\prod_{n\geq 0} x_n(\bigvee R_n) = \prod_{j\geq 0} y_j(\bigvee R), \tag{2}$$

by Ax2, and

$$\prod_{n\geq 0} x_n R_n = \prod_{j\geq 0} Y_j R.$$

If $R = R' \cup R''$, then:

$$\begin{aligned}
\prod_{n\geq 0} x_n(\bigvee R_n) &= \prod_{j\geq 0} y_j(\bigvee(R' \cup R'')) \quad \text{(by (2))}\\
&= \prod_{j\geq 0} y_j(\bigvee R' \vee \bigvee R'') \quad \text{(by Lemma 7)}\\
&= \bigvee_{z_j \in \{\bigvee R', \bigvee R''\}} \prod_{j\geq 0} y_j z_j \quad \text{(by Ax3)}\\
&= \bigvee_{z_j \in \{\bigvee R', \bigvee R''\}} \bigvee \prod_{j\geq 0} Y_j z_j\\
&= \bigvee_{Z_j \in \{R', R''\}} \bigvee \prod_{j\geq 0} Y_j Z_j\\
&= \bigvee \prod_{n\geq 0} x_n(R' \cup R'') = \bigvee \prod_{n\geq 0} x_n R,
\end{aligned}$$

where the 4th and 5th equalities hold by the induction hypothesis and Ax2.

Suppose now that $R = R'R''$. Then, applying the induction hypothesis almost directly,

$$\begin{aligned}
\prod_{n\geq 0} x_n(\bigvee R_n) &= \prod_{j\geq 0} y_j(\bigvee R'R'')\\
&= \prod_{j\geq 0} y_j(\bigvee R')(\bigvee R'') \quad \text{(by Lemma 7)}\\
&= \bigvee \prod_{j\geq 0} Y_j(\bigvee R')(\bigvee R'')\\
&= \bigvee \prod_{j\geq 0} Y_j R'R''\\
&= \bigvee \prod_{n\geq 0} x_n R'R'' = \bigvee \prod_{n\geq 0} x_n R,
\end{aligned}$$

where the third and fourth equalities come from the induction hypothesis and Ax2.

The last case to consider is when $R = T^*$, where T is regular. We argue as follows:

$$\begin{aligned}
\prod_{n\geq 0} x_n(\bigvee R_n) &= \prod_{j\geq 0} y_j(\bigvee T^*)\\
&= \prod_{j\geq 0} y_j(\bigvee T)^* \quad \text{(by Lemma 7)}\\
&= \bigvee_{k_0,k_1,\dots} \prod_{j\geq 0} y_j(\bigvee T)^{k_j} \quad \text{(by Ax1 and Ax4)}\\
&= \bigvee_{k_0,k_1,\dots} \bigvee \prod_{j\geq 0} Y_j(\bigvee T)^{k_j}\\
&= \bigvee_{k_0,k_1,\dots} \bigvee \prod_{j\geq 0} Y_j T^{k_j}\\
&= \bigvee_{j\geq 0} Y_j T^* = \bigvee_{j\geq 0} Y_j R_j = \bigvee_{n\geq 0} x_n R_n,
\end{aligned}$$

where the 4th and 5th equalities follow from the induction hypothesis and Ax2. The proof is complete. □

By the same proof, we have the following version of Lemma 9 for the finitary case:

Lemma 10. *Let (S, V) be a finitary *-continuous Kleene ω-algebra. Suppose that the languages $R_0, R_1, \ldots \subseteq S^*$ are regular and that $\mathcal{R} = \{R_0, R_1, \ldots\}$ is a finite set. Moreover, let $x_0, x_1, \ldots$ be a finitary sequence of elements of S. Then*

$$\prod_{n\geq 0} x_n(\bigvee R_n) = \bigvee \prod_{n\geq 0} x_n R_n.$$

Note that each sequence $x_0, y_0, x_1, y_1, \ldots$ with $y_n \in R_n$ is finitary.

Corollary 1. *Let (S, V) be a finitary *-continuous Kleene ω-algebra. Suppose that $R_0, R_1, \ldots \subseteq S^*$ are regular and that $\mathcal{R} = \{R_0, R_1, \ldots\}$ is a finite set. Then $\bigvee \prod_{n\geq 0} R_n$ exists and is equal to $\prod_{n\geq 0} \bigvee R_n$.*

Using our earlier convention that ω-words $v = x_0 x_1 \ldots \in S^\omega$ over S determine elements $\prod_{n\geq 0} x_n$ of V and subsets $X \subseteq S^\omega$ determine subsets of V, Lemma 9 may be rephrased as follows.

For any *-continuous Kleene ω-algebra (S, V), $x_0, x_1, \ldots \in S$ and regular sets $R_0, R_1, \ldots \subseteq S^*$ for which $\mathcal{R} = \{R_0, R_1, \ldots\}$ is a finite set, it holds that

$$\prod_{n\geq 0} x_n(\bigvee R_n) = \bigvee X,$$

where $X \subseteq S^\omega$ is the set of all ω-words $x_0 y_0 x_1 y_1 \ldots$ with $y_i \in R_i$ for all $i \geq 0$, *i.e.*, $X = x_0 R_0 x_1 R_1 \ldots$

Similarly, Corollary 1 asserts that if a subset of V corresponds to an infinite product over a finite collection of ordinary regular languages in S^*, then the supremum of this set exists.

In any (finitary or non-finitary) *-continuous Kleene ω-algebra (S, V), we define an *ω-power operation* $S \to V$ by $x^\omega = \prod_{n \geq 0} x$ for all $x \in S$. From the axioms we immediately have:

Corollary 2. *Suppose that (S, V) is a (finitary or non-finitary) *-continuous Kleene ω-algebra. Then the following hold for all $x, y \in S$:*

$$\begin{aligned} x^\omega &= xx^\omega \\ (xy)^\omega &= x(yx)^\omega \\ x^\omega &= (x^n)^\omega, \quad n \geq 2. \end{aligned}$$

Thus, each *-continuous Kleene ω-algebra gives rise to a Wilke algebra [29].

Lemma 11. *Let (S, V) be a (finitary or non-finitary) *-continuous Kleene ω-algebra. Suppose that $R \subseteq S^\omega$ is ω-regular. Then $\bigvee R$ exists in V.*

Proof. It is well-known that R can be written as a finite union of sets of the form $R_0(R_1)^\omega$ where $R_0, R_1 \subseteq S^*$ are regular, moreover, R_1 does not contain the empty word. It suffices to show that $\bigvee R_0(R_1)^\omega$ exists. But this holds by Corollary 1. $\square$

Lemma 12. *Let (S, V) be a (finitary or non-finitary) *-continuous Kleene ω-algebra. For all ω-regular sets $R_1, R_2 \subseteq S^\omega$ and regular sets $R \subseteq S^*$ it holds that*

$$\begin{aligned} \bigvee(R_1 \cup R_2) &= \bigvee R_1 \vee \bigvee R_2 \\ \bigvee(RR_1) &= (\bigvee R)(\bigvee R_1). \end{aligned}$$

And if R does not contain the empty word, then

$$\bigvee R^\omega = (\bigvee R)^\omega.$$

Proof. The first claim is clear. The second follows from Lemma 8. For the last, see the proof of Lemma 11. $\square$

7 Free Finitary *-Continuous Kleene ω-Algebras

Recall that for a set A, $R(A^*)$ denotes the collection of all regular languages in A^*. It is well-known that $R(A^*)$, equipped with the usual operations, is a *-continuous Kleene algebra on A. Actually, $R(A^*)$ is characterized up to isomorphism by the following universal property.

Theorem 4 ([25]). *For each set A, $R(A^*)$ is the free *-continuous Kleene algebra on A.*

Thus, if S is any *-continuous Kleene algebra and $h : A \to S$ is any mapping from any set A into S, then h has a unique extension to a *-continuous Kleene algebra homomorphism $h^\sharp : R(A^*) \to S$.

Now let $R'(A^\infty)$ denote the collection of all subsets of A^∞ which are finite unions of finitary infinite products of regular languages, that is, finite unions of sets of the form $\prod_{n\geq 0} R_n$, where each $R_n \subseteq A^*$ is regular, and the set $\{R_0, R_1, \ldots\}$ is finite. Note that $R'(A^\infty)$ contains the empty set and is closed under finite unions. Moreover, when $Y \in R'(A^\infty)$ and $u = a_0a_1 \ldots \in Y \cap A^\omega$, then the alphabet of u is finite, *i.e.,* the set $\{a_n : n \geq 0\}$ is finite. Also, $R'(A^\infty)$ is closed under the action of $R(A^*)$ inherited from $(P(A^*), P(A^\infty))$. The infinite product of a sequence of regular languages in $R(A^*)$ is not necessarily contained in $R'(A^\infty)$, but by definition $R'(A^\infty)$ contains all infinite products of finitary sequences over $R(A^*)$.

Example 3. Let $A = \{a, b\}$ and consider the set $X = \{aba^2b \ldots a^nb \ldots\} \in P(A^\infty)$ containing a single ω-word. X can be written as an infinite product of subsets of A^*, but it cannot be written as an infinite product $R_0R_1 \ldots$ of regular languages in A^* such that the set $\{R_0, R_1, \ldots\}$ is finite. Hence $X \notin R'(A^\infty)$.

Theorem 5. *For each set A, $(R(A^*), R'(A^\infty))$ is the free finitary *-continuous Kleene ω-algebra on A.*

Proof. Our proof is modeled after the proof of Theorem 2. First, it is clear from the fact that $(P(A^*), P(A^\infty))$ is a continuous Kleene ω-algebra, and that $R(A^*)$ is a *-continuous semiring, that $(R(A^*), R'(A^\infty))$ is indeed a finitary *-continuous Kleene ω-algebra.

Suppose that (S, V) is any finitary *-continuous Kleene ω-algebra and let $h : A \to S$ be a mapping. For each $u = a_0 \ldots a_{n-1}$ in A^*, let $h_S(u) = h(a_0) \cdots h(a_{n-1})$ and $h_V(u) = h(a_0) \cdots h(a_{n-1})1^\omega = \prod_{k\geq 0} b_k$, where $b_k = a_k$ for all $k < n$ and $b_k = 1$ for all $k \geq n$. When $u = a_0a_1 \ldots \in A^\omega$ whose alphabet is finite, define $h_V(u) = \prod_{k\geq 0} h(a_k)$. This infinite product exists in $R'(A^\infty)$.

Note that we have $h_S(uv) = h_S(u)h_S(v)$ for all $u, v \in A^*$, and $h_S(\varepsilon) = 1$. And if $u \in A^*$ and $v \in A^\infty$ such that the alphabet of v is finite, then $h_V(uv) = h_S(u)h_V(v)$. Also, $h_V(XY) = h_S(X)h_V(Y)$ for all $X \subseteq A^*$ in $R(A^*)$ and $Y \subseteq A^\infty$ in $R'(A^\infty)$.

Moreover, for all $u_0, u_1, \ldots$ in A^*, if $u_i \neq \varepsilon$ for infinitely many i, such that the alphabet of $u_0u_1 \ldots$ is finite, then $h_V(u_0u_1 \ldots) = \prod_{k\geq 0} h_S(u_k)$. If on the other hand, $u_k = \varepsilon$ for all $k \geq n$, then $h_V(u_0u_1 \ldots) = h_S(u_0) \cdots h_S(u_{n-1})1^\omega$. In either case, if $X_0, X_1, \ldots \subseteq A^*$ are regular and form a finitary sequence, then the sequence $h_S(X_0), h_S(X_1), \ldots$ is also finitary as is each infinite word in $\prod_{n\geq 0} X_n$, and $h_V(\prod_{n\geq 0} X_n) = \prod_{n\geq 0} h_S(X_n)$.

Suppose now that $X \subseteq A^*$ is regular and $Y \subseteq A^\infty$ is in $R'(A^\infty)$. We define $h_S^\sharp(X) = \bigvee h_S(X)$ and $h_V^\sharp(Y) = \bigvee h_V(Y)$. It is well-known that $h_S^\sharp$ is a *-continuous Kleene algebra morphism $R(A^*) \to S$. Also, $h_V^\sharp$ preserves finite suprema, since when I is finite, $h_V^\sharp(\bigcup_{i\in I} Y_i) = \bigvee h_V(\bigcup_{i\in I} Y_i) = \bigvee \bigcup_{i\in I} h_V(Y_i) = \bigvee_{i\in I} \bigvee h_V(Y_i) = \bigvee_{i\in I} h_V^\sharp(Y_i)$.

We prove that the action is preserved. Let $X \in R(A^*)$ and $Y \in R'(A^\infty)$. Then $h_V^\sharp(XY) = \bigvee h_V(XY) = \bigvee h_S(X)h_V(Y) = \bigvee h_S(X) \bigvee h_V(Y) = h_S^\sharp(X)h_V^\sharp(Y)$.

Finally, we prove that infinite product of finitary sequences is preserved. Let $X_0, X_1, \ldots$ be a finitary sequence of regular languages in $R(A^*)$. Then, using Corollary 1, $h_V^\sharp(\prod_{n\geq 0} X_n) = \bigvee h_V(\prod_{n\geq 0} X_n) = \bigvee \prod_{n\geq 0} h_S(X_n) = \prod_{n\geq 0} \bigvee h_S(X_n) = \prod_{n\geq 0} h_S^\sharp(X_n)$.

It is clear that h_S extends h, and that (h_S, h_V) is unique. □

Consider now $(R(A^*), R'(A^\omega))$ equipped with the infinite product operation $\prod_{n\geq 0} X_n = \{u_0 u_1 \in A^\omega : u_n \in X_n,\ n \geq 0\}$, defined on finitary sequences $X_0, X_1, \ldots$ of languages in $R(A^*)$.

Lemma 13. *Suppose that (S, V) is a finitary *-continuous Kleene ω-algebra satisfying $1^\omega = \bot$. Let (h_S, h_V) be a homomorphism $(R(A^*), R'(A^\infty)) \to (S, V)$. Then (h_S, h_V) factors through (φ_S, φ_V).*

Proof. Similar to the proof of Lemma 5. □

Theorem 6. *For each set A, $(R(A^*), R'(A^\omega))$ is the free finitary *-continuous Kleene ω-algebra satisfying $1^\omega = \bot$ on A.*

Proof. This follows from Theorem 5 using Lemma 13. □

8 *-Continuous Kleene ω-Algebras Are Iteration Semiring-Semimodule Pairs

In this section, we will show that every (finitary or non-finitary) *-continuous Kleene ω-algebra is an iteration semiring-semimodule pair.

Some definitions are in order. Suppose that $S = (S, \vee, \cdot, \bot, 1)$ is an idempotent semiring. Following [2], we call S a *Conway semiring* if S is equipped with a star operation $^* : S \to S$ satisfying, for all $x, y \in S$,

$$(x \vee y)^* = (x^* y)^* x^*$$
$$(xy)^* = 1 \vee x(yx)^* y\,.$$

(Note that in [2], also non-idempotent Conway semirings have been considered, but we stick to the idempotent case here.)

It is known [2] that if S is a Conway semiring, then for each $n \geq 1$, so is the semiring $S^{n\times n}$ of all $n \times n$-matrices over S with the usual sum and product operations and the star operation defined by induction on n so that if $n > 1$ and $M = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$, where a and d are square matrices of dimension $< n$, then

$$M^* = \begin{pmatrix} (a \vee bd^*c)^* & (a \vee bd^*c)^* bd^* \\ (d \vee ca^*b)^* ca^* & (d \vee ca^*b)^* \end{pmatrix}.$$

The above definition does not depend on how M is split into submatrices.

Suppose that S is a Conway semiring and $G = \{g_1, \ldots, g_n\}$ is a finite group of order n. For each $x_{g_1}, \ldots, x_{g_n} \in S$, consider the $n \times n$ matrix $M_G = M_G(x_{g_1}, \ldots, x_{g_n})$

whose ith row is $(x_{g_i^{-1}g_1}, \ldots, x_{g_i^{-1}g_n})$, for $i = 1, \ldots, n$, so that each row (and column) is a permutation of the first. We say that the group identity [8] associated with G holds in S if for each $x_{g_1}, \ldots, x_{g_n}$, the first (and then any) row sum of M_G^* is $(x_{g_1} \vee \cdots \vee x_{g_n})^*$. Finally, we call S an *iteration semiring* [2, 11] if the group identities hold in S for all finite groups of order n.

Classes of examples of (idempotent) iteration semirings are given by the continuous and the *-continuous Kleene algebras defined in the introduction. As mentioned above, the language semirings $P(A^*)$ and the semirings $P(A \times A)$ of binary relations are continuous and hence also *-continuous Kleene algebras, and the semirings $R(A^*)$ of regular languages are *-continuous Kleene algebras.

When S is a *-continuous Kleene algebra and n is a nonnegative integer, then the matrix semiring $S^{n\times n}$ is also a *-continuous Kleene algebra and hence an iteration semiring, cf. [24]. The star operation is defined by

$$M^*_{i,j} = \bigvee_{m\geq 0,\ 1\leq k_1,\ldots,k_m\leq n} M_{i,k_1} M_{k_1,k_2} \cdots M_{k_m,j},$$

for all $M \in S^{n\times n}$ and $1 \leq i, j \leq n$. It is not trivial to prove that the above supremum exists. The fact that M^* is well-defined can be established by induction on n together with the well-known matrix star formula mentioned above.

An idempotent semiring-semimodule pair (S, V) is a *Conway semiring-semimodule pair* if it is equipped with a star operation $^* : S \to S$ and an omega operation $^\omega : S \to V$ such that S is a Conway semiring acting on the semimodule $V = (V, \vee, \bot)$ and the following hold for all $x, y \in S$:

$$\begin{aligned}(x \vee y)^\omega &= (x^*y)^*x^\omega \vee (x^*y)^\omega\\ (xy)^\omega &= x(yx)^\omega.\end{aligned}$$

It is known [2] that when (S, V) is a Conway semiring-semimodule pair, then so is $(S^{n\times n}, V^n)$ for each n, where V^n denotes the $S^{n\times n}$-semimodule of all n-dimensional (column) vectors over V with the action of $S^{n\times n}$ defined similarly to matrix-vector product, and where the omega operation is defined by induction so that when $n > 1$ and $M = \left(\begin{smallmatrix} a & b\\ c & d\end{smallmatrix}\right)$, where a and d are square matrices of dimension $< n$, then

$$M^\omega = \begin{pmatrix}(a \vee bd^*c)^\omega \ \vee\ (a \vee bd^*c)^*bd^\omega\\ (d \vee ca^*b)^\omega \ \vee\ (d \vee ca^*b)^*ca^\omega\end{pmatrix}. \tag{3}$$

We also define *iteration semiring-semimodule pairs* [2, 19] as those Conway semiring-semimodule pairs such that S is an iteration semiring and the omega operation satisfies the following condition: let $M_G = M_G(x_{g_1}, \ldots, x_{g_n})$ like above, with $x_{g_1}, \ldots, x_{g_n} \in S$ for a finite group $G = \{g_1, \ldots, g_n\}$ of order n, then the first (and hence any) entry of M_G^ω is equal to $(x_{g_1} \vee \cdots \vee x_{g_n})^\omega$.

Examples of (idempotent) iteration semiring-semimodule pairs include the semiring-semimodule pairs $(P(A^*), P(A^\omega))$ of languages and ω-languages over an alphabet A mentioned earlier. The omega operation is defined by $X^\omega = \prod_{n\geq 0} X$. More generally, it is known that every continuous Kleene ω-algebra gives rise to an iteration semiring-semimodule pair. The omega operation is defined as for languages: $x^\omega = \prod_{n\geq 0} x_n$ with $x_n = x$ for all $n \geq 0$.

Other not necessarily idempotent examples include the *complete* and the *(symmetric) bi-inductive semiring-semimodule pairs* of [18, 19].

Suppose now that (S, V) is a *-continuous Kleene ω-algebra. Then for each $n \geq 1$, $(S^{n\times n}, V^n)$ is a semiring-semimodule pair. The action of $S^{n\times n}$ on V^n is defined similarly to matrix-vector product (viewing the elements of V^n as column vectors). It is easy to see that $(S^{n\times n}, V^n)$ is a generalized *-continuous Kleene algebra for each $n \geq 1$.

Suppose that $n \geq 2$. We would like to define an infinite product operation $(S^{n\times n})^\omega \to V^n$ on matrices in $S^{n\times n}$ by

$$(\prod_{m\geq 0} M_m)_i = \bigvee_{1\leq i_1, i_2, \ldots \leq n} (M_0)_{i,i_1}(M_1)_{i_1,i_2}\cdots$$

for all $1 \leq i \leq n$. However, unlike in the case of complete semiring-semimodule pairs [19], the supremum on the right-hand side may not exist. Nevertheless it is possible to define an omega operation $S^{n\times n} \to V^n$ and to turn $(S^{n\times n}, V^n)$ into an iteration semiring-semimodule pair.

Lemma 14. *Let (S, V) be a (finitary or non-finitary) *-continuous Kleene ω-algebra. Suppose that $M \in S^{n\times n}$, where $n \geq 2$. Then for every $1 \leq i \leq n$,*

$$(\prod_{m\geq 0} M)_i = \bigvee_{1\leq i_1, i_2, \ldots \leq n} M_{i,i_1}M_{i_1,i_2}\cdots$$

exists, so that we define *M^ω by the above equality. Moreover, when $M = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$, where a and d are square matrices of dimension $< n$, then* (3) *holds.*

Proof. Suppose that $n = 2$. Then by Corollary 1, $(a \vee bd^*c)^\omega$ is the supremum of the set of all infinite products $A_{1,i_1}A_{i_1,i_2}\cdots$ containing a or c infinitely often, and $(a \vee bd^*c)^*bd^\omega$ is the supremum of the set of all infinite products $A_{1,i_1}A_{i_1,i_2}\cdots$ containing a and c only finitely often. Thus, $(a \vee bd^*c)^\omega \vee (a \vee bd^*c)^*bd^\omega$ is the supremum of the set of all infinite products $A_{1,i_1}A_{i_1,i_2}\cdots$. Similarly, $(d \vee ca^*b)^\omega \vee (d \vee ca^*b)^*ca^\omega$ is the supremum of the set of all infinite products $A_{2,i_1}A_{i_1,i_2}\cdots$.

The proof of the induction step is similar. Suppose that $n > 2$, and let a be $k\times k$. Then by induction hypothesis, for every i with $1 \leq i \leq k$, the ith component of $(a \vee bd^*c)^\omega$ is the supremum of the set of all infinite products $A_{i,i_1}A_{i_1,i_2}\cdots$ containing an entry of a or c infinitely often, whereas the ith component of $(a \vee bd^*c)^*bd^\omega$ is the supremum of all infinite products $A_{i,i_1}A_{i_1,i_2}\cdots$ containing entries of a and c only finitely often. Thus, the ith component of $(a \vee bd^*c)^\omega \vee (a \vee bd^*c)^*bd^\omega$ is the supremum of the set of all infinite products $A_{i,i_1}A_{i_1,i_2}\cdots$. A similar fact holds for $(d \vee ca^*b)^\omega \vee (d \vee ca^*b)^*ca^\omega$. The proof is complete. ☐

Theorem 7. *Every (finitary or non-finitary) *-continuous Kleene ω-algebra is an iteration semiring-semimodule pair.*

Proof. Suppose that (S, V) is a finitary *-continuous Kleene ω-algebra. Then

$$(x \vee y)^\omega = (x^* y)^\omega \vee (x^* y)^* x^\omega,$$

since by Lemma 7 and Lemma 12, $(x^* y)^\omega$ is the supremum of the set of all infinite products over $\{x, y\}$ containing y infinitely often, and $(x^* y)^* x^\omega$ is the supremum of the set of infinite products over $\{x, y\}$ containing y finitely often. Thus, $(x^* y)^\omega \vee (x^* y)^* x^\omega$ is equal to $(x \vee y)^\omega$, which by Ax3 is the supremum of all infinite products over $\{x, y\}$. As noted above, also

$$(xy)^\omega = x(yx)^\omega$$

for all $x, y \in S$. Thus, (S, V) is a Conway semiring-semimodule pair and hence so is each $(S^{n \times n}, V^n)$.

To complete the proof of the fact that (S, V) is an iteration semiring-semimodule pair, suppose that $x_1, \ldots, x_n \in S$, and let $x = x_1 \vee \cdots \vee x_n$. Let A be an $n \times n$ matrix whose rows are permutations of the $x_1, \ldots, x_n$. We need to prove that each component of A^ω is x^ω. We use Lemma 14 and Ax3 to show that both are equal to the supremum of the set of all infinite products over the set $X = \{x_1, \ldots, x_n\}$.

By Lemma 14, for each $i_0 = 1, \ldots, n$, the i_0th row of A^ω is $\bigvee_{i_1, i_2, \ldots} a_{i_0, i_1} a_{i_1, i_2} \cdots$. It is clear that each infinite product $a_{i_0, i_1} a_{i_1, i_2} \cdots$ is an infinite product over X. Suppose now that $x_{j_0} x_{j_1} \cdots$ is an infinite product over X. We define by induction on $k \geq 0$ an index i_{k+1} such that $a_{i_k, i_{k+1}} = x_{j_k}$. Suppose that $k = 0$. Then let i_1 be such that $a_{i_0, i_1} = x_{j_0}$. Since x_{j_0} appears in the i_0th row, there is such an i_1. Suppose that $k > 0$ and that i_k has already been defined. Since x_{j_k} appears in the i_kth row, there is some i_{k+1} with $a_{i_k, i_{k+1}} = x_{j_k}$. We have completed the proof of the fact that the i_0th entry of A^ω is the supremum of the set of all infinite products over the set $X = \{x_1, \ldots, x_n\}$.

Consider now $x^\omega = xx \cdots$. We use induction on n to prove that x^ω is also the supremum of the set of all infinite products over the set $X = \{x_1, \ldots, x_n\}$. When $n = 1$ this is clear. Suppose now that $n > 1$ and that the claim is true for $n - 1$. Let $y = x_1 \vee \cdots \vee x_{n-1}$ so that $x = y \vee x_n$. We have:

$$\begin{aligned} x^\omega &= (y \vee x_n)^\omega \\ &= (x_n^* y)^* x_n^\omega \vee (x_n^* y)^\omega \\ &= (x_n^* y)^* x_n^\omega \vee (x_n^* x_1 \vee \cdots \vee x_n^* x_{n-1})^\omega . \end{aligned}$$

Now

$$(x_n^* y)^* x_n^\omega = \bigvee_{k, m_1, \ldots, m_k \geq 0} x_n^{m_1} y \cdots x_n^{m_k} y x_n^\omega$$

by Lemma 8, which is the supremum of all infinite products over X containing $x_1, \ldots, x_{n-1}$ only a finite number of times.

Also, using the induction hypothesis and Ax4,

$$\begin{aligned}(x_n^* x_1 \vee \cdots \vee x_n^* x_{n-1})^\omega &= \bigvee_{1 \le i_1, i_2, \ldots \le n-1} x_n^* x_{i_1} x_n^* x_{i_2} \cdots \\ &= \bigvee_{1 \le i_1, i_2, \ldots \le n-1} \bigvee_{k_0, k_1, \ldots} x_n^{k_0} x_{i_1} x_n^{k_1} x_{i_2} \cdots\end{aligned}$$

which is the supremum of all infinite products over X containing one of $x_1, \ldots, x_{n-1}$ an infinite number of times. Thus, x^ω is the supremum of all infinite products over X as claimed. □

9 Kleene ω-Algebras

Recall that when S is a *-continuous Kleene algebra, then S is a Kleene algebra [24]. Thus, for all $x, y \in S$, $x^* y$ is the least pre-fixed point (and thus the least fixed point) of the function $S \to S$ defined by $z \mapsto xz \vee y$ for all $z \in S$. Moreover, yx^* is the least pre-fixed point and the least fixed point of the function $S \to S$ defined by $z \mapsto zx \vee y$, for all $z \in S$. Similarly, when (S, V) is a generalized *-continuous Kleene algebra, then for all $x \in S$ and $v \in V$, $x^* v$ is the least pre-fixed point and the least fixed point of the function $V \to V$ defined by $z \mapsto xz \vee v$.

As a natural analogy to Kleene algebras in semiring-semimodule pairs, we propose a notion of *Kleene ω-algebra*.

Definition 7. *A* Kleene ω-algebra *is a semiring-semimodule pair (S, V) in which S is a Kleene algebra and equipped with an omega operation $^\omega : S \to V$ such that the following hold for all $x, y \in S$ and $v \in V$:*

- *$x^* v$ is the least pre-fixed point of the function $V \to V$ defined by $z \mapsto xz \vee v$,*
- *$x^\omega \vee x^* v$ is the greatest post-fixed point of the function $V \to V$ defined by $z \mapsto xz \vee v$.*

It is clear that any Kleene ω-algebra is a bi-inductive semiring-semimodule pair in the sense of [19]. By the above remarks we have:

Lemma 15. *Suppose that (S, V) is a (finitary or non-finitary) *-continuous Kleene ω-algebra. When for all $x \in S$ and $v \in V$, $x^\omega \vee x^* v$ is the greatest post-fixed point of the function $V \to V$ defined by $z \mapsto xz \vee v$, then (S, V) is a Kleene ω-algebra.*

We remark that the precondition of the lemma is indeed necessary, and it is not the case that any *-continuous Kleene ω-algebra is a Kleene ω-algebra. As an example, note that the above property implies that 1^ω is the greatest fixed point of the mapping $z \mapsto z$; but we have seen in Theorem 6 that there are finitary *-continuous Kleene ω-algebras with $1^\omega = \bot$.

10 Conclusion

Motivated by an application to energy problems, we have introduced continuous and *-continuous Kleene ω-algebras and exposed some of their basic properties. Continuous Kleene ω-algebras are idempotent complete semiring-semimodule pairs, and conceptually, *-continuous Kleene ω-algebras are a generalization of continuous Kleene ω-algebras in much the same way as *-continuous Kleene algebras are of continuous Kleene algebras: In *-continuous Kleene algebras, suprema of finite sets and of sets of powers are required to exist and to be preserved by the product; in *-continuous Kleene ω-algebras these suprema are also required to be preserved by the infinite product.

We have seen that the sets of finite and infinite languages over an alphabet are the free continuous Kleene ω-algebras, and that the free finitary *-continuous Kleene ω-algebras are given by the sets of regular languages and of finite unions of finitary infinite products of regular languages. A characterization of the free (non-finitary) *-continuous Kleene ω-algebras (and whether they even exist) is left open.

We have seen that every *-continuous Kleene ω-algebra is an iteration semiring-semimodule pair, hence also matrix-vector semiring-semimodule pairs over *-continuous Kleene ω-algebras are iteration semiring-semimodule pairs. In the second paper of the series [15], we will apply the algebraic setting developed here in order to solve energy problems.

References

[1] Jean Berstel and Christophe Reutenauer. *Noncommutative Rational Series With Applications.* Cambridge Univ. Press, 2010.

[2] Stephen L. Bloom and Zoltán Ésik. *Iteration Theories: The Equational Logic of Iterative Processes.* EATCS monographs on theoretical computer science. Springer-Verlag, 1993.

[3] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, and Nicolas Markey. Timed automata with observers under energy constraints. In Karl Henrik Johansson and Wang Yi, editors, *HSCC*, pages 61–70. ACM, 2010.

[4] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, Nicolas Markey, and Jiří Srba. Infinite runs in weighted timed automata with energy constraints. In Franck Cassez and Claude Jard, editors, *FORMATS*, volume 5215 of *Lect. Notes Comput. Sci.*, pages 33–47. Springer-Verlag, 2008.

[5] Patricia Bouyer, Kim G. Larsen, and Nicolas Markey. Lower-bound-constrained runs in weighted timed automata. *Perform. Eval.*, 73:91–109, 2014.

[6] Romain Brenguier, Franck Cassez, and Jean-François Raskin. Energy and mean-payoff timed games. In Martin Fränzle and John Lygeros, editors, *HSCC*, pages 283–292. ACM, 2014.

[7] Krishnendu Chatterjee and Laurent Doyen. Energy parity games. *Theor. Comput. Sci.*, 458:49–60, 2012.

[8] John H. Conway. *Regular Algebra and Finite Machines.* Chapman and Hall, 1971.

[9] Aldric Degorre, Laurent Doyen, Raffaella Gentilini, Jean-François Raskin, and Szymon Toruńczyk. Energy and mean-payoff games with imperfect information. In Anuj Dawar and Helmut Veith, editors, *CSL*, volume 6247 of *Lect. Notes Comput. Sci.*, pages 260–274. Springer-Verlag, 2010.

[10] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted Automata.* EATCS Monographs in Theoretical Computer Science. Springer-Verlag, 2009.

[11] Zoltán Ésik. Iteration semirings. In Masami Ito and Masafumi Toyama, editors, *DLT*, volume 5257 of *Lect. Notes Comput. Sci.*, pages 1–20. Springer-Verlag, 2008.

[12] Zoltán Ésik, Uli Fahrenberg, and Axel Legay. *-continuous Kleene ω-algebras. In Igor Potapov, editor, *DLT*, volume 9168 of *Lect. Notes Comput. Sci.*, pages 240–251. Springer-Verlag, 2015.

[13] Zoltán Ésik, Uli Fahrenberg, and Axel Legay. *-continuous Kleene ω-algebras for energy problems. In Ralph Matthes and Matteo Mio, editors, *FICS*, volume 191 of *Electr. Proc. Theor. Comput. Sci.*, pages 48–59, 2015.

[14] Zoltán Ésik, Uli Fahrenberg, Axel Legay, and Karin Quaas. Kleene algebras and semimodules for energy problems. In Dang Van Hung and Mizuhito Ogawa, editors, *ATVA*, volume 8172 of *Lect. Notes Comput. Sci.*, pages 102–117. Springer-Verlag, 2013.

[15] Zoltán Ésik, Uli Fahrenberg, Axel Legay, and Karin Quaas. An algebraic approach to energy problems II: The algebra of energy functions. *Acta Cyb.*, 2017. In this issue.

[16] Zoltán Ésik and Werner Kuich. Rationally additive semirings. *J. Univ. Comput. Sci.*, 8(2):173–183, 2002.

[17] Zoltán Ésik and Werner Kuich. Inductive star-semirings. *Theor. Comput. Sci.*, 324(1):3–33, 2004.

[18] Zoltán Ésik and Werner Kuich. A semiring-semimodule generalization of ω-regular languages, Parts 1 and 2. *J. Aut. Lang. Comb.*, 10:203–264, 2005.

[19] Zoltán Ésik and Werner Kuich. On iteration semiring-semimodule pairs. *Semigroup Forum*, 75:129–159, 2007.

[20] Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jiří Srba. Energy games in multiweighted automata. In Antonio Cerone and Pekka Pihlajasaari, editors, *ICTAC*, volume 6916 of *Lect. Notes Comput. Sci.*, pages 95–115. Springer-Verlag, 2011.

[21] Uli Fahrenberg, Axel Legay, and Karin Quaas. Büchi conditions for generalized energy automata. In Manfred Droste and Heiko Vogler, editors, *WATA*, page 47, 2012.

[22] Jonathan S. Golan. *Semirings and their Applications.* Springer-Verlag, 1999.

[23] Line Juhl, Kim G. Larsen, and Jean-François Raskin. Optimal bounds for multiweighted and parametrised energy games. In Zhiming Liu, Jim Woodcock, and Huibiao Zhu, editors, *Theories of Programming and Formal Methods*, volume 8051 of *Lect. Notes Comput. Sci.*, pages 244–255. Springer-Verlag, 2013.

[24] Dexter Kozen. On Kleene algebras and closed semirings. In Branislav Rovan, editor, *MFCS*, volume 452 of *Lect. Notes Comput. Sci.*, pages 26–47. Springer-Verlag, 1990.

[25] Dexter Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Inf. Comput.*, 110(2):366–390, 1994.

[26] Dominique Perrin and Jean-Éric Pin. *Infinite Words: Automata, Semigroups, Logic and Games.* Academic Press, 2004.

[27] Karin Quaas. On the interval-bound problem for weighted timed automata. In Adrian Horia Dediu, Shunsuke Inenaga, and Carlos Martín-Vide, editors, *LATA*, volume 6638 of *Lect. Notes Comput. Sci.*, pages 452–464. Springer-Verlag, 2011.

[28] Yaron Velner, Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, Alexander Moshe Rabinovich, and Jean-François Raskin. The complexity of multi-mean-payoff and multi-energy games. *Inf. Comput.*, 241:177–196, 2015.

[29] Thomas Wilke. An Eilenberg theorem for infinity-languages. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez-Artalejo, editors, *ICALP*, volume 510 of *Lect. Notes Comput. Sci.*, pages 588–599. Springer-Verlag, 1991.

Acta Cybernetica **23** (2017) 229–268.

An Algebraic Approach to Energy Problems II The Algebra of Energy Functions*

Zoltán Ésik[a], Uli Fahrenberg[b], Axel Legay[c], and Karin Quaas[d]

Abstract

Energy and resource management problems are important in areas such as embedded systems or autonomous systems. They are concerned with the question whether a given system admits infinite schedules during which certain tasks can be repeatedly accomplished and the system never runs out of energy (or other resources). In order to develop a general theory of energy problems, we introduce energy automata: finite automata whose transitions are labeled with energy functions which specify how energy values change from one system state to another.

We show that energy functions form a *-continuous Kleene ω-algebra, as an application of a general result that finitely additive, locally *-closed and $\top$-continuous functions on complete lattices form *-continuous Kleene ω-algebras. This permits to solve energy problems in energy automata in a generic, algebraic way. In order to put our work in context, we also review extensions of energy problems to higher dimensions and to games.

Keywords: Energy problem, Kleene algebra, *-continuity, *-continuous Kleene ω-algebra

1 Introduction

Energy and *resource management* problems are important in areas such as embedded systems or autonomous systems. They are concerned with the question whether a given system admits infinite schedules during which (1) certain tasks can be repeatedly accomplished and (2) the system never runs out of energy (or

*This research was supported by grant no. K 108448 from the National Foundation of Hungary for Scientific Research (OTKA), by ANR MALTHY, grant no. ANR-13-INSE-0003 from the French National Research Foundation, and by Deutsche Forschungsgemeinschaft (DFG), projects QU 316/1-1 and QU 316/1-2.

[a]University of Szeged, Hungary (deceased)

[b]École polytechnique, Palaiseau, France. Most of this work was carried out while this author was still employed at Inria Rennes.

[c]Inria Rennes, France

[d]Universität Leipzig, Germany

DOI: 10.14232/actacyb.23.1.2017.14

other specified resources). Starting with [11], formal modeling and analysis of such problems has recently attracted some attention [10, 12, 15, 19, 22, 33, 39, 46, 48].

As an example, Fig. 1 shows a simple model of an electric car, modeled as a weighted timed automaton [4, 5]. In the *working* state W, energy is consumed at a rate of 10 energy units per time unit; in the two *recharging* states R_1 and R_2, the battery is charged at a rate of 20, respectively 10, energy units per time unit. As the clock c is reset ($c \leftarrow 0$) when entering state W and has guard $c \geq 1$ on outgoing transitions, we ensure that the car always has to be in state W for at least one time unit. Similarly, the system can only transition back from states R_1, R_2 to W if it has spent at most one time unit in these states.

Passing from state W to R_2 (and back) requires 2 energy units, whereas passing from W to R_1 requires 6 energy units and passing back from R_1 to W requires 4 energy units. Passing from R_2 to R_1 requires 5 energy units, and passing from R_1 to R_2 requires 1 energy unit.

Altogether, this is intended to model the fact that there are two recharge stations available, one close to work but less powerful, and a more powerful one further away and located uphill, so that moving upwards costs more energy than moving downwards. Now assume that the initial state W is entered with a given *initial energy* x_0, then the energy problem of this model is as follows: Does there exist an infinite trace which (1) visits W infinitely often and (2) never has an energy level below 0?

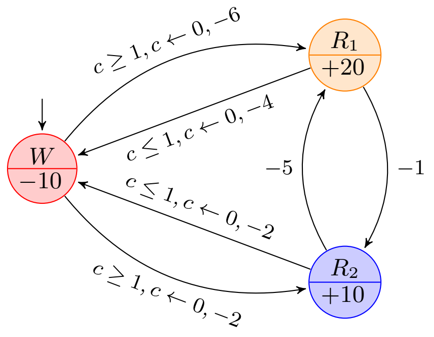

W
-10
R_1
$+20$
R_2
$+10$
$c \geq 1, c \leftarrow 0, -6$
$c \leq 1, c \leftarrow 0, -4$
$c \leq 1, c \leftarrow 0, -2$
$c \geq 1, c \leftarrow 0, -2$
-5
-1

Figure 1: Simple model of an electric car as a weighted timed automaton.

In order to develop a general theory which can be applied to the above and other types of energy problems, we have in [27, 36] introduced *energy automata*. These are finite automata whose transitions are labeled with *energy functions* which specify how energy values change from one system state to another. Using the theory of semiring-weighted automata [24], we have shown in [27] that energy problems in such automata can be solved in a simple static way which only involves manipulations of energy functions.

In order to put the work of [27] on a more solid theoretical footing and with an eye to future generalizations, we have recently introduced a new algebraic structure of **-continuous Kleene* ω*-algebras* [25, 26].

In this paper, we are concerned with conditions under which functions on complete lattices form *-continuous Kleene ω-algebras. We show that sets of functions which are *finitely additive*, *locally* **-closed* and $\top$*-continuous*, all natural conditions which we will introduce later, form *-continuous Kleene ω-algebras. We then show that energy functions are an example of such functions.

Using general results concerning coverability and Büchi acceptance in automata with transition weights in *-continuous Kleene ω-algebras, we are then able to solve energy problems in energy automata in a generic, algebraic way.

In order to put our work in context, we also review extensions of energy problems to higher dimensions and to games. We show that even though our algebraic setting does not apply here, coverability for multi-dimensional energy automata is decidable. Energy games, on the other hand, are shown to be undecidable from dimension two.

Structure of the Paper This is the second in a series of two papers which are concerned with energy problems and their algebraic foundation. In the first paper of the series [28], we have introduced continuous and *-continuous Kleene ω-algebras and exposed some of their algebraic properties. We have shown that every *-continuous Kleene ω-algebra is an iteration semiring-semimodule pair.

In this paper, we continue our work by showing how to compute Büchi acceptance in Section 3. Note that our two papers can be read independently, as we have taken care to recall the relevant results obtained in [28].

We then turn our attention to *-continuous Kleene ω-algebras of functions. In Section 4 we introduce the properties of finite additivity, *-closedness and $\top$-continuity and show that any set S of finitely additive, *-closed and $\top$-continuous functions on a complete lattice L form *-continuous Kleene algebras.

In Section 5 we extend this result and show that if (S, V) is such that S is a *-continuous Kleene algebra of functions $L \to L$, V consists of finitely additive and $\top$-continuous functions $L \to \mathbf{2}$, where $\mathbf{2}$ denotes the Boolean lattice, then (S, V) forms a *-continuous Kleene ω-algebra. We then apply this result to energy automata in Section 6.

In Section 7 we take a more detailed look on two important subclasses of (computable) energy functions and obtain some complexity results. Section 8 reviews a reduction from energy problems on weighted timed automata to our energy automata, in order to further motivate our notions of energy function and energy automaton.

The final Section 9 is concerned with extensions of energy problems to higher dimensions and to games. Using an extension of the Rackoff technique for affine Petri nets, we show that coverability for multi-dimensional energy automata is decidable in exponential time. On the other hand, for a slightly relaxed version of energy function, coverability becomes undecidable from dimension four. Likewise, reachability games on two-dimensional energy automata and on one-dimensional relaxed energy automata are undecidable.

Related Work A simple class of energy automata is the one of *integer-weighted automata*, where all energy functions are updates of the form $x \mapsto x + k$ for some (positive or negative) integer k. Energy problems on these automata, and their extensions to multiple weights (also called *vector addition systems with states* (VASS)) and to games, have been considered for example in [11, 14, 17–20, 33]. The exact complexity of the reachability problem for VASS is one of the most challenging open problems in theoretical computer science; plenty of very recent results aim to get more insight into this problem [7, 38, 42, 43]. Our energy automata may be considered as a generalization of one-dimensional VASS to arbitrary updates; in the final section of this paper we will also be concerned with multi-dimensional energy automata and games.

Energy problems on *timed automata* [2] have been considered in [10–12,46]. Here timed automata are enriched with integer weights in locations and on transitions (the *weighted timed automata* of [4, 5], *cf.* Fig. 1), with the semantics that the weight of a delay in a location is computed by multiplying the length of the delay by the location weight. In [11] it is shown that energy problems for one-clock weighted timed automata without updates on transitions (hence only with weights in locations) can be reduced to energy problems on integer-weighted automata with additive updates.

For one-clock weighted timed automata *with* transition updates, energy problems are shown decidable in [10], using a reduction to energy automata as we use them here. Intuitively, each path in the timed automaton in which the clock is not reset is converted to an edge in an energy automaton, labeled with a *piecewise affine* energy function. We review the reduction from [10] in Section 8 of the present paper. In a recent paper [16], this class of real-time energy problems is treated directly, in the setting of *-continuous Kleene ω-algebras, without a reduction to the untimed setting.

Also another class of energy problems on weighted timed automata is considered in [10], in which weights during delays are increasing *exponentially* rather than linearly. These are shown decidable using a reduction to energy automata with *piecewise polynomial* energy functions; again our present framework applies.

Acknowledgment We are deeply indebted to our colleague and friend Zoltán Ésik with whom we started this research and who has led us along the way. Unfortunately Zoltán could not see this work completed, so any errors are the responsibility of the last three authors.

The second author also acknowledges interesting discussions with Patricia Bouyer, Kim G. Larsen and Nicolas Markey which led to [10] and eventually to Section 8 of this paper.

2 Energy Automata

We recall the energy automata introduced in [28] and the decision problems we are interested in. Let $[0, \infty]_\perp = \{\perp\} \cup [0, \infty]$ denote the complete lattice of non-negative real numbers together with extra elements $\perp$ and ∞, with the standard order on $\mathbb{R}_{\geq 0}$ extended by $\perp < x < \infty$ for all $x \in \mathbb{R}_{\geq 0}$. Also, $\perp + x = \perp - x = \perp$ for all $x \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ and $\infty + x = \infty - x = \infty$ for all $x \in \mathbb{R}_{\geq 0}$.

Definition 1. *An (extended)* energy function *is a mapping* $f : [0, \infty]_\perp \to [0, \infty]_\perp$, *for which* $\perp f = \perp$ *and*

$$yf \geq xf + y - x \tag{$*$}$$

for all $x \leq y$. Moreover, $\infty f = \infty$, unless $xf = \bot$ for all $x \in [0,\infty]_\bot$. The class of all extended energy functions is denoted $\mathcal{E}$.

We write function composition and application in diagrammatical order, from left to right. Hence we write $f;g$, or simply fg, for the composition $g \circ f$ and $x;f$ or xf for function application $f(x)$. This is because we will be concerned with *algebras* of functions, in which function composition is multiplication, and where it is customary to write multiplication in diagrammatical order.

We define a partial order on $\mathcal{E}$, by $f \leq g$ iff $xf \leq xg$ for all $x \in [0,\infty]_\bot$. We will need three special energy functions, $\perp\!\!\!\perp$, id and $\top\!\!\!\top$; these are given by $x\perp\!\!\!\perp = \bot$, $x;\mathsf{id} = x$ for $x \in [0,\infty]_\bot$, and $\bot\top\!\!\!\top = \bot$, $x\top\!\!\!\top = \infty$ for $x \in [0,\infty]$.

Lemma 1 ([28])**.** *With the ordering $\leq$, $\mathcal{E}$ is a complete lattice with bottom element $\perp\!\!\!\perp$ and top element $\top\!\!\!\top$. The supremum on $\mathcal{E}$ is pointwise, i.e., $x(\sup_{i\in I} f_i) = \sup_{i\in I} xf_i$ for any set I, all $f_i \in \mathcal{E}$ and $x \in [0,\infty]_\bot$. Also, $h(\sup_{i\in I} f_i) = \sup_{i\in I}(hf_i)$ for all $h \in \mathcal{E}$.*

We denote binary suprema using the symbol $\vee$; hence $f \vee g$, for $f, g \in \mathcal{E}$, is the function $x(f \vee g) = \max(xf, xg)$. For a subset $\mathcal{E}' \subseteq \mathcal{E}$, we write $\langle \mathcal{E}' \rangle$ for the set of all finite suprema $a_1 \vee \dots \vee a_m$ with $a_i \in \mathcal{E}'$ for each $i = 1, \dots, m$.

Definition 2. *Let $\mathcal{E}' \subseteq \mathcal{E}$ and $n \geq 1$. An $\mathcal{E}'$-*automaton *of dimension n is a structure (α, M, k), were $\alpha \in \{\perp\!\!\!\perp, \mathsf{id}\}^n$ is the initial vector, $M \in \langle \mathcal{E}' \rangle^{n\times n}$ is the transition matrix, and k is an integer $0 \leq k \leq n$.*

Combinatorially, this may be represented as a transition system whose set of states is $\{1, \dots, n\}$. For any pair of states i, j, the transitions from i to j are determined by the entry $M_{i,j}$ of the transition matrix: if $M_{i,j} = f_1 \vee \dots \vee f_m$, then there are m transitions from i to j, respectively labeled $f_1, \dots, f_m$. The states i with $\alpha_i = \mathsf{id}$ are *initial*, and the states $\{1, \dots, k\}$ are *accepting*.

Recall that an *idempotent semiring* [6,37] $S = (S, \vee, \cdot, \bot, 1)$ consists of a commutative idempotent monoid $(S, \vee, \bot)$ and a monoid $(S, \cdot, 1)$ such that the distributive laws

$$x(y \vee z) = xy \vee xz$$
$$(y \vee z)x = yx \vee zx$$

and the zero laws

$$\bot \cdot x = \bot = x \cdot \bot$$

hold for all $x, y, z \in S$. It follows that the product operation distributes over all finite sums.

Each idempotent semiring S is partially ordered by its *natural order* relation $x \leq y$ iff $x \vee y = y$, and then sum and product preserve the partial order and $\bot$ is the least element. Moreover, for all $x, y \in S$, $x \vee y$ is the least upper bound of the set $\{x, y\}$.

Lemma 2 ([28])**.** *$(\mathcal{E}, \vee, \circ, \perp\!\!\!\perp, \mathsf{id})$ is an idempotent semiring with natural order $\leq$.*

An energy automaton is hence a weighted automaton over the semiring $\mathcal{E}$, in the sense of [24]. We recall the decision problems which we are interested in. As the input to a decision problem must be in some way finitely representable, we will state them for subsets $\mathcal{E}' \subseteq \mathcal{E}$ of *computable* energy functions. Note that we give no technical meaning to the term "computable" here; we simply need to take care that the input can be finitely represented.

Problem 1 (State reachability)**.** Given an $\mathcal{E}'$-automaton (α, M, k) of dimension $n \geq 1$ and a computable initial energy $x_0 \in \mathbb{R}_{\geq 0}$: does there exist a finite sequence $(k_0, \dots, k_m)$ of indices $1 \leq k_i \leq n$ such that $\alpha_{k_0} = \mathsf{id}$, $k_m \leq k$, and $x_0 M_{k_0,k_1} \cdots M_{k_{m-1},k_m} \neq \bot$?

Using the representation of $A = (\alpha, M, k)$ as a transition system, we see that the above problem amounts to asking whether there exists a finite path in A, with transition labels $M_{k_0,k_1}, \dots, M_{k_{m-1},k_m}$, such that the path starts in an initial state, ends in an accepting state, and $x_0 M_{k_0,k_1} \cdots M_{k_{m-1},k_m} \neq \bot$.

Problem 2 (Coverability)**.** Given an $\mathcal{E}'$-automaton (α, M, k) of dimension $n \geq 1$, a computable initial energy $x_0 \in \mathbb{R}_{\geq 0}$ and a computable function $z : \{1, \dots, k\} \to \mathbb{R}_{\geq 0}$: does there exist a sequence $(k_0, \dots, k_m)$ of indices $1 \leq k_i \leq n$ such that $\alpha_{k_0} = \mathsf{id}$, $k_m \leq k$, and $x_0 M_{k_0,k_1} \cdots M_{k_{m-1},k_m} \geq k_m z$?

In the transition system representation of $A = (\alpha, M, k)$, this amounts to asking whether there exists a finite path in A as above, starting in an initial state and ending in an accepting state k_m, and such that $x_0 M_{k_0,k_1} \cdots M_{k_{m-1},k_m} \geq k_m z$. Using the function z with $iz = 0$ for all $i = 1, \dots, k$, coverability reduces to state reachability.

Problem 3 (Büchi acceptance)**.** Given an $\mathcal{E}'$-automaton (α, M, k) of dimension $n \geq 1$ and a computable initial energy $x_0 \in \mathbb{R}_{\geq 0}$: does there exist an infinite sequence $(k_0, k_1, \dots)$ of indices $1 \leq k_i \leq n$ such that $\alpha_{k_0} = \mathsf{id}$, $k_i \leq k$ for infinitely many indices i, and $x_0 M_{k_0,k_1} \cdots M_{k_{m-1},k_m} \neq \bot$ for all $m \geq 1$?

Again using the representation of $A = (\alpha, M, k)$ as a transition system, we see that this last problem amounts to asking whether there exists an infinite path in A, with transition labels $M_{k_0,k_1}, M_{k_1,k_2}, \dots$, such that the path starts in an initial state, visits an accepting state infinitely often, and all finite prefixes $x_0 M_{k_0,k_1} \cdots M_{k_{m-1},k_m} \neq \bot$.

We let $\mathsf{Reach}_{\mathcal{E}'}$ denote the function which maps an $\mathcal{E}'$-automaton together with an initial energy to the Boolean values **ff** or **tt** depending on whether the answer to the concrete state reachability problem is negative or positive. $\mathsf{Cover}_{\mathcal{E}'}$ and $\mathsf{Büchi}_{\mathcal{E}'}$ denote the similar mappings for the coverability and Büchi acceptance problems.

3 Büchi Automata in *-Continuous Kleene ω-Algebras

We recall the notion of **-continuous Kleene ω-algebra* introduced in [28]. First, a **-continuous Kleene algebra* is an idempotent semiring $(S, \vee, \cdot, \bot, 1)$ in which the infinite suprema $\bigvee\{x^n \mid n \geq 0\}$ exist for all $x \in S$ and product preserves such suprema:

$$y(\bigvee_{n\geq 0} x^n) = \bigvee_{n\geq 0} yx^n \quad \text{and} \quad (\bigvee_{n\geq 0} x^n)y = \bigvee_{n\geq 0} x^n y$$

for all $x, y \in S$. One then defines $x^* = \bigvee\{x^n \mid n \geq 0\}$ for every $x \in S$.

A *-continuous Kleene algebra is *continuous* if *all* suprema $\bigvee X$, $X \subseteq S$, exist and are preserved by products. *-continuous Kleene algebras are hence a generalization of continuous Kleene algebras. There are interesting Kleene algebras which are *-continuous but not continuous, for example the Kleene algebra of all regular languages over some alphabet, see [28].

Recall that an *idempotent semiring-semimodule pair* [8, 31] (S, V) consists of an idempotent semiring $S = (S, \vee, \cdot, \bot, 1)$ and a commutative idempotent monoid $V = (V, \vee, \bot)$ which is equipped with a left S-action $S \times V \to V$, $(x, v) \mapsto xv$, satisfying

$$\begin{aligned} (x \vee x')v &= xv \vee x'v & \qquad x(v \vee v') &= xv \vee xv' \\ (xx')v &= x(x'v) & \bot v &= \bot \\ x\bot &= \bot & 1v &= v \end{aligned}$$

for all $x, x' \in S$ and $v \in V$. In that case, we also call V a *(left) S-semimodule.*

A *generalized *-continuous Kleene algebra* [28] is a semiring-semimodule pair (S, V) where $S = (S, \vee, \cdot, ^*, \bot, 1)$ is a *-continuous Kleene algebra such that

$$xy^*v = \bigvee_{n\geq 0} xy^n v$$

for all $x, y \in S$ and $v \in V$.

A **-continuous Kleene ω-algebra* [28] consists of a generalized *-continuous Kleene algebra (S, V) together with an infinite product operation $S^\omega \to V$ which maps every infinite sequence $x_0, x_1, \dots$ in S to an element $\prod_{n\geq 0} x_n$ of V. The infinite product is subject to the following conditions:

Ax1: For all $x_0, x_1, \dots \in S$, $\prod_{n\geq 0} x_n = x_0 \prod_{n\geq 0} x_{n+1}$.

Ax2: Let $x_0, x_1, \dots \in S$ and $0 = n_0 \leq n_1 \cdots$ be a sequence which increases without a bound. Let $y_k = x_{n_k} \cdots x_{n_{k+1}-1}$ for all $k \geq 0$. Then $\prod_{n\geq 0} x_n = \prod_{k\geq 0} y_k$.

Ax3: For all $x_0, x_1, \dots$ and y, z in S, $\prod_{n\geq 0}(x_n(y \vee z)) = \bigvee_{x'_n \in \{y,z\}} \prod_{n\geq 0} x_n x'_n$.

Ax4: For all $x, y_0, y_1, \dots \in S$, $\prod_{n\geq 0} x^* y_n = \bigvee_{k_n \geq 0} \prod_{n\geq 0} x^{k_n} y_n$.

A *continuous Kleene ω-algebra* [31] is a semiring-semimodule pair (S, V) in which S is a continuous Kleene algebra, V is a complete lattice, and the S-action on V preserves all suprema in either argument, together with an infinite product as above which satisfies conditions Ax1 and Ax2 above and preserves all suprema. *-continuous Kleene ω-algebras are hence a generalization of continuous Kleene ω-algebras.

For any idempotent semiring S and $n \geq 1$, we can form the matrix semiring $S^{n\times n}$ whose elements are $n \times n$-matrices of elements of S and whose sum and product are given as the usual matrix sum and product. It is known [41] that when S is a *-continuous Kleene algebra, then $S^{n\times n}$ is also a *-continuous Kleene algebra, with the *-operation defined by

$$M^*_{i,j} = \bigvee_{m\geq 0} \bigvee_{1\leq k_1,\dots,k_m\leq n} M_{i,k_1} M_{k_1,k_2} \cdots M_{k_m,j}$$

for all $M \in S^{n\times n}$ and $1 \leq i, j \leq n$. The above infinite supremum exists, as it is taken over a regular set, see [28]. Also, if $n \geq 2$ and $M = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$, where a and d are square matrices of dimension less than n, then

$$M^* = \begin{pmatrix} (a \vee bd^*c)^* & (a \vee bd^*c)^* bd^* \\ (d \vee ca^*b)^* ca^* & (d \vee ca^*b)^* \end{pmatrix} . \tag{1}$$

For any semiring-semimodule pair (S, V) and $n \geq 1$, we can form the matrix semiring-semimodule pair $(S^{n\times n}, V^n)$ whose elements are $n\times n$-matrices of elements of S and n-dimensional (column) vectors of elements of V, with the action of $S^{n\times n}$ on V^n given by the usual matrix-vector product.

When (S, V) is a *-continuous Kleene ω-algebra, then $(S^{n\times n}, V^n)$ is a generalized *-continuous Kleene algebra. By [28, Lemma 14], there is an ω-operation on $S^{n\times n}$ defined by

$$M^\omega_i = \bigvee_{1\leq k_1,k_2,\dots\leq n} M_{i,k_1} M_{k_1,k_2} \cdots$$

for all $M \in S^{n\times n}$ and $1 \leq i \leq n$. Also, if $n \geq 2$ and $M = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$, where a and d are square matrices of dimension less than n, then

$$M^\omega = \begin{pmatrix} (a \vee bd^*c)^\omega \vee (a \vee bd^*c)^* bd^\omega \\ (d \vee ca^*b)^\omega \vee (d \vee ca^*b)^* ca^\omega \end{pmatrix} . \tag{2}$$

Note [28] that it is not generally the case that $(S^{n\times n}, V^n)$ is again a *-continuous Kleene ω-algebra, as the infinite product may not exist.

Let (S, V) be a *-continuous Kleene ω-algebra and $A \subseteq S$ a subset. We write $\langle A\rangle$ for the set of all finite suprema $a_1 \vee \cdots \vee a_m$ with $a_i \in A$ for each $i = 1, \dots, m$.

A *weighted automaton* [32] over A of dimension $n \geq 1$ is a tuple (α, M, k), where $\alpha \in \{\perp, 1\}^n$ is the initial vector, $M \in \langle A\rangle^{n\times n}$ is the transition matrix, and k is an integer $0 \leq k \leq n$. Combinatorially, this may be represented as a transition system whose set of states is $\{1, \dots, n\}$. For any pair of states i, j, the transitions from i to j are determined by the entry $M_{i,j}$ of the transition matrix: if $M_{i,j} = a_1 \vee \cdots \vee a_m$, then there are m transitions from i to j, respectively labeled $a_1, \dots, a_n$. The states i with $\alpha_i = 1$ are *initial*, and the states $\{1, \dots, k\}$ are *accepting*.

The *finite behavior* of a weighted automaton $A = (\alpha, M, k)$ is defined to be

$$|A| = \alpha M^* \kappa \,,$$

where $\kappa \in \{\bot, 1\}^n$ is the vector given by $\kappa_i = 1$ for $i \leq k$ and $\kappa_i = \bot$ for $i > k$. (Note that α has to be used as a *row* vector for this multiplication to make sense.) It is clear by (1) that $|A|$ is the supremum of the products of the transition labels along all paths in A from any initial to any accepting state.

The *Büchi behavior* of a weighted automaton $A = (\alpha, M, k)$ is defined to be

$$\|A\| = \alpha \begin{pmatrix} (a \vee bd^*c)^\omega \\ d^*c(a \vee bd^*c)^\omega \end{pmatrix},$$

where $a \in \langle A \rangle^{k \times k}$, $b \in \langle A \rangle^{k \times (n-k)}$, $c \in \langle A \rangle^{(n-k) \times n}$ and $d \in \langle A \rangle^{(n-k) \times (n-k)}$ are such that $M = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$. By (2), $\|A\|$ is the supremum of the products of the transition labels along all infinite paths in A from any initial state which infinitely often visit an accepting state.

For completeness we also mention a Kleene theorem for the Büchi automata introduced above, which is a direct consequence of the Kleene theorem for Conway semiring-semimodule pairs, cf. [29, 32].

Theorem 1. *An element of V is the Büchi behavior weighted automaton over A iff it is rational over A, i.e., when it can be generated from the elements of A by the semiring and semimodule operations, the action, and the star and omega operations.*

It is a routine matter to show that an element of V is rational over A iff it can be written as $\bigvee_{i=1}^n x_i y_i^\omega$, where each x_i and y_i can be generated from A by $\vee$, $\cdot$, and *.

4 Generalized *-Continuous Kleene Algebras of Functions

In the following two sections our aim is to establish properties which ensure that semiring-semimodule pairs of *functions* form *-continuous Kleene ω-algebras. We will use these properties in Section 6 to show that energy functions form a *-continuous Kleene ω-algebra.

Let L and L' be complete lattices with bottom and top elements $\bot$ and $\top$. Then a function $f : L \to L'$ is said to be *finitely additive* if $\bot f = \bot$ and $(x \vee y)f = xf \vee yf$ for all $x, y \in L$. (Recall that we write function application and composition in the diagrammatic order, from left to right.) When $f : L \to L'$ is finitely additive, then $(\bigvee X)f = \bigvee Xf$ for all finite sets $X \subseteq L$.

Consider the collection $\mathsf{FinAdd}_{L,L'}$ of all finitely additive functions $f : L \to L'$, ordered pointwise. Since the (pointwise) supremum of any set of finitely additive functions is finitely additive, $\mathsf{FinAdd}_{L,L'}$ is also a complete lattice, in which the supremum of any set of functions can be constructed pointwise. The least and greatest elements are the functions $\underline{\bot}$ and $\overline{\top}$ given by $x\underline{\bot} = \bot$ for $x \in L$, $\bot\overline{\top} = \bot$, and $x\overline{\top} = \top$ for $x \in \backslash\{\bot\}$.

Definition 3. *A function $f \in \mathsf{FinAdd}_{L,L'}$ is said to be* $\top$-continuous *if $f = \perp$ or for all $X \subseteq L$ with $\bigvee X = \top$, also $\bigvee Xf = \top$.*

Note that if $f \neq \perp$ is $\top$-continuous, then $\top f = \top$. The functions id and $\perp$ are $\top$-continuous. Also, the (pointwise) supremum of any set of $\top$-continuous functions is again $\top$-continuous.

We will first be concerned with functions in $\mathsf{FinAdd}_{L,L}$, which we just denote FinAdd_L. Since the composition of finitely additive functions is finitely additive and the identity function id over L is finitely additive, and since composition of finitely additive functions distributes over finite suprema, FinAdd_L, equipped with the operation $\vee$ (binary supremum), ; (composition), and the constant function $\perp$ and the identity function id as 1, is an idempotent semiring. It follows that when f is finitely additive, then so is $f^* = \bigvee_{n\geq 0} f^n$. Moreover, $f \leq f^*$ and $f^* \leq g^*$ whenever $f \leq g$.

Lemma 3. *Let S be any subsemiring of FinAdd_L closed under the *-operation. Then S is a *-continuous Kleene algebra iff for all $g, h \in S$, $g^*h = \bigvee_{n\geq 0} g^n h$.*

Proof. Suppose that the precondition of the lemma holds. We need to show that $f(\bigvee_{n\geq 0} g^n)h = \bigvee_{n\geq 0} fg^nh$ for all $f, g, h \in S$. But $f(\bigvee_{n\geq 0} g^n)h = f(\bigvee_{n\geq 0} g^n h)$ by assumption, and we conclude that $f(\bigvee_{n\geq 0} g^n h) = \bigvee_{n\geq 0} fg^n h$ since the supremum is pointwise. ☐

Compositions of $\top$-continuous functions in FinAdd_L are again $\top$-continuous, so that the collection of all $\top$-continuous functions in FinAdd_L is itself an idempotent semiring.

Definition 4. *A function $f \in \mathsf{FinAdd}_L$ is said to be* locally *-closed *if for each $x \in L$, either $xf^* = \top$ or there exists $N \geq 0$ such that $xf^* = x \vee \cdots \vee xf^N$.*

The functions id and $\perp$ are locally *-closed. As the next example demonstrates, compositions of locally *-closed (and $\top$-continuous) functions are not necessarily locally *-closed.

Example 1. Let L be the following complete lattice (the linear sum of three infinite chains):

$$\perp < x_0 < x_1 < \cdots < y_0 < y_1 < \cdots < z_0 < z_1 < \cdots < \top$$

Since L is a chain, a function $L \to L$ is finitely additive iff it is monotone and preserves $\perp$.

Let $f, g : L \to L$ be the following functions. First, $\perp f = \perp g = \perp$ and $\top f = \top g = \top$. Moreover, $x_i f = y_i$, $y_i f = z_i g = \top$ and $x_i g = \perp$, $y_i g = x_{i+1}$, and $z_i g = \top$ for all i. Then f, g are monotone, $uf^* = u \vee uf \vee uf^2$ and $ug^* = u \vee ug$ for all $u \in L$. Also, f and g are $\top$-continuous, since if $\bigvee X = \top$ then either $\top \in X$ or $X \cap \{z_0, z_1, \ldots\}$ is infinite, but then $\bigvee Xf = \bigvee Xg = \top$. However, fg is not locally *-closed, since $x_0(fg)^* = x_0 \vee x_0(fg) \vee x_0(fg)^2 \cdots = x_0 \vee x_1 \vee \cdots = y_0$.

Lemma 4. *Let $f \in \mathsf{FinAdd}_L$ be locally *-closed. Then also f^* is locally *-closed. If f is additionally $\top$-continuous, then so is f^*.*

Proof. We prove that $xf^{**} = x \vee xf^* = xf^*$ for all $x \in L$. Indeed, this is clear when $xf^* = \top$, since $f^* \leq f^{**}$. Otherwise $xf^* = \bigvee_{k \leq n} xf^k$ for some $n \geq 0$.

By finite additivity, it follows that $xf^*f^* = \bigvee_{k \leq n} xf^k f^*$. But for each k, $xf^k f^* = xf^k \vee xf^{k+1} \vee \cdots \leq xf^*$, thus $xf^* = xf^*f^*$ and $xf^* = xf^{**}$. It follows that f^* is locally *-closed.

Suppose now that f is additionally $\top$-continuous. We need to show that f^* is also $\top$-continuous. To this end, let $X \subseteq L$ with $\bigvee X = \top$. Since $x \leq xf^*$ for all $x \in X$, it holds that $\bigvee Xf^* \geq \bigvee X = \top$. Thus $\bigvee Xf^* = \top$. □

Proposition 1. *Let S be any subsemiring of FinAdd_L closed under the *-operation. If each $f \in S$ is locally *-closed and $\top$-continuous, then S is a *-continuous Kleene algebra.*

Proof. Let $g, h \in S$. By Lemma 3, it suffices to show that $g^*h = \bigvee_{n \geq 0} g^n h$. Since this is clear when $h = \bot$, assume that $h \neq \bot$. As $g^n h \leq g^* h$ for all $n \geq 0$, it holds that $\bigvee_{n \geq 0} g^n h \leq g^* h$. To prove the opposite inequality, suppose that $x \in L$. If $xg^* = \top$, then $\bigvee_{n \geq 0} xg^n = \top$, so $\bigvee_{n \geq 0} xg^n h = \top$ by $\top$-continuity. Thus, $xg^*h = \top = \bigvee_{n \geq 0} xg^n h$.

Suppose that $xg^* \neq \top$. Then there is $m \geq 0$ with

$$xg^*h = (x \vee \cdots \vee xg^m)h = xh \vee \cdots \vee xg^m h \leq \bigvee_{n \geq 0} xg^n h = x(\bigvee_{n \geq 0} g^n h)\,.$$

The proof is complete. □

Now define a left action of FinAdd_L on $\mathsf{FinAdd}_{L,L'}$ by $fv = f;v$, for all $f \in \mathsf{FinAdd}_L$ and $v \in \mathsf{FinAdd}_{L,L'}$. It is a routine matter to check that $\mathsf{FinAdd}_{L,L'}$, equipped with the above action, the binary supremum operation $\vee$ and the constant $\bot$ is an (idempotent) left FinAdd_L-semimodule, that is, $(\mathsf{FinAdd}_L, \mathsf{FinAdd}_{L,L'})$ is a semiring-semimodule pair.

Lemma 5. *Let $S \subseteq \mathsf{FinAdd}_L$ be a *-continuous Kleene algebra and $V \subseteq \mathsf{FinAdd}_{L,L'}$ an S-semimodule. Then (S, V) is a generalized *-continuous Kleene algebra iff for all $f \in S$ and $v \in V$, $f^*v = \bigvee_{n \geq 0} f^n v$.*

Proof. Similar to the proof of Lemma 3. □

Proposition 2. *Let $S \subseteq \mathsf{FinAdd}_L$ be a *-continuous Kleene algebra and $V \subseteq \mathsf{FinAdd}_{L,L'}$ an S-semimodule. If each $f \in S$ is locally *-closed and $\top$-continuous and each $v \in V$ is $\top$-continuous, then (S, V) is a generalized *-continuous Kleene algebra.*

Proof. Similar to the proof of Proposition 1. □

5 *-Continuous Kleene ω-Algebras of Functions

In this section, let L be an arbitrary complete lattice and $L' = \mathbf{2}$, the two-element lattice $\{\bot, \top\}$. We define an infinite product $\mathsf{FinAdd}_L^\omega \to \mathsf{FinAdd}_{L,\mathbf{2}}$. Let $f_0, f_1, \ldots \in \mathsf{FinAdd}_L$ be an infinite sequence and define $v = \prod_{n\geq 0} f_n : L \to \mathbf{2}$ by

$$xv = \begin{cases} \bot & \text{if there is } n \geq 0 \text{ such that } xf_0 \cdots f_n = \bot, \\ \top & \text{otherwise} \end{cases}$$

for all $x \in L$. We will write $\prod_{n\geq k} f_n$, for $k \geq 0$, as a shorthand for $\prod_{n\geq 0} f_{n+k}$.

It is easy to see that $\prod_{n\geq 0} f_n$ is finitely additive. Indeed, $\bot \prod_{n\geq 0} f_n = \bot$ clearly holds, and for all $x \leq y \in L$, $x \prod_{n\geq 0} f_n \leq y \prod_{n\geq 0} f_n$. Thus, to prove that $(x \vee y) \prod_{n\geq 0} f_n = x \prod_{n\geq 0} f_n \vee y \prod_{n\geq 0} f_n$ for all $x, y \in L$, it suffices to show that if $x \prod_{n\geq 0} f_n = y \prod_{n\geq 0} f_n = \bot$, then $(x \vee y) \prod_{n\geq 0} f_n = \bot$. But if $x \prod_{n\geq 0} f_n = y \prod_{n\geq 0} f_n = \bot$, then there exist $m, k \geq 0$ such that $xf_0 \cdots f_m = yf_0 \cdots f_k = \bot$. Let $n = \max\{m, k\}$. We have $(x \vee y) f_0 \cdots f_n = xf_0 \cdots f_n \vee yf_0 \cdots f_n = \bot$, and thus $(x \vee y) \prod_{n\geq 0} f_n = \bot$.

It is clear that this infinite product satisfies conditions $\mathsf{Ax1}$ and $\mathsf{Ax2}$ in the definition of *-continuous Kleene ω-algebra. Below we show that also $\mathsf{Ax3}$ and $\mathsf{Ax4}$ hold.

Lemma 6. *For all $f_0, f_1, \ldots, g_0, g_1, \ldots \in \mathsf{FinAdd}_L$,*

$$\prod_{n\geq 0} (f_n \vee g_n) = \bigvee_{h_n \in \{f_n, g_n\}} \prod_{n\geq 0} h_n \,.$$

Note that this implies $\mathsf{Ax3}$.

Proof. Since infinite product is monotone, the term on the right-hand side of the equation is less than or equal to the term on the left-hand side. To prove that equality holds, let $x \in L$ and suppose that $x \prod_{n\geq 0} (f_n \vee g_n) = \top$. It suffices to show that there is a choice of the functions $h_n \in \{f_n, g_n\}$ such that $x \prod_{n\geq 0} h_n = \top$.

Consider the infinite ordered binary tree where each node at level $n \geq 0$ is the source of an edge labeled f_n and an edge labeled g_n, ordered as indicated. We can assign to each node u the composition h_u of the functions that occur as the labels of the edges along the unique path from the root to that node.

Let us mark a node u if $xh_u \neq \bot$. As $x \prod_{n\geq 0} (f_n \vee g_n) = \top$, each level contains a marked node. Moreover, whenever a node is marked and has a predecessor, its predecessor is also marked. By König's lemma [40] there is an infinite path going through marked nodes. This infinite path gives rise to the sequence $h_0, h_1, \ldots$ with $x \prod_{n\geq 0} h_n = \top$. $\square$

Lemma 7. *Let $f \in \mathsf{FinAdd}_L$ and $v \in \mathsf{FinAdd}_{L,\mathbf{2}}$ such that f is locally *-closed and v is $\top$-continuous. If $xf^*v = \top$, then there exists $k \geq 0$ such that $xf^k v = \top$.*

Proof. If $xf^* = \bigvee_{n=0}^N xf^n$ for some $N \geq 0$, then $xf^*v = \bigvee_{n=0}^N xf^n v = \top$ implies the claim of the lemma. If $xf^* = \top$, then $\top$-continuity of v implies that $\bigvee_{n\geq 0} xf^n v = \top$, which again implies the claim. $\square$

Lemma 8. *Let $f, g_0, g_1, \ldots \in \mathsf{FinAdd}_L$ be locally *-closed and $\top$-continuous such that for each $m \geq 0$, $g_m \prod_{n \geq m+1} f^* g_n \in \mathsf{FinAdd}_{L,\mathbf{2}}$ is $\top$-continuous. Then*

$$\prod_{n \geq 0} f^* g_n = \bigvee_{k_0, k_1, \ldots \geq 0} \prod_{n \geq 0} f^{k_n} g_n \,.$$

Proof. As infinite product is monotone, the term on the right-hand side of the equation is less than or equal to the term on the left-hand side. To prove that equality holds, let $x \in L$ and suppose that $x \prod_{n \geq 0} f^* g_n = \top$. We want to show that there exist integers $k_0, k_1, \ldots \geq 0$ such that $x \prod_{n \geq 0} f^{k_n} g_n = \top$.

Let $x_0 = x$. By Lemma 7, $x \prod_{n \geq 0} f^* g_n = x_0 f^* g_0 \prod_{n \geq 1} f^* g_n = \top$ implies that there is $k_0 \geq 0$ for which $x_0 f^{k_0} g_0 \prod_{n \geq 1} f^* g_n = \top$. We finish the proof by induction.

Assume that we have $k_0, \ldots, k_m \geq 0$ such that $x f^{k_0} g_0 \cdots f^{k_m} g_m \prod_{n \geq m+1} f^* g_n = \top$ and let $x_{m+1} = x f^{k_0} g_0 \cdots f^{k_m} g_m$. Then $x_{m+1} f^* g_{m+1} \prod_{n \geq m+2} f^* g_n = \top$ implies, using Lemma 7, that there exists an exponent $k_{m+1} \geq 0$ for which $x_{m+1} f^{k_{m+1}} g_{m+1} \prod_{n \geq m+2} f^* g_n = \top$. ☐

Proposition 3. *Let $S \subseteq \mathsf{FinAdd}_L$ and $V \subseteq \mathsf{FinAdd}_{L,\mathbf{2}}$ such that (S, V) is a generalized *-continuous Kleene algebra of locally *-closed and $\top$-continuous functions $L \to L$ and $\top$-continuous functions $L \to \mathbf{2}$. If $\prod_{n \geq 0} f_n \in V$ for all sequences $f_0, f_1, \ldots$ of functions in S, then (S, V) is a *-continuous Kleene ω-algebra.*

Proof. This is clear from Lemmas 6 and 8. ☐

We finish the section by a lemma which exhibits a condition on the lattice L which ensures that infinite products of locally *-closed and $\top$-continuous functions are again $\top$-continuous.

Lemma 9. *Assume that L has the property that whenever $\bigvee X = \top$ for some $X \subseteq L$, then for all $x < \top$ in L there is $y \in X$ with $x \leq y$. If $f_0, f_1, \ldots \in \mathsf{FinAdd}_L$ is a sequence of locally *-closed and $\top$-continuous functions, then $\prod_{n \geq 0} f_n \in \mathsf{FinAdd}_{L,\mathbf{2}}$ is $\top$-continuous.*

Proof. Let $v = \prod_{n \geq 0} f_n$. We already know that v is finitely additive. We need to show that v is $\top$-continuous. But if $v \neq \bot$, then there is some $x < \top$ with $xv = \top$, *i.e.*, such that $x f_0 \cdots f_n > \bot$ for all n. By assumption, there is some $y \in X$ with $x \leq y$. It follows that $y f_0 \cdots f_n \geq x f_0 \cdots f_n > \bot$ for all n and thus $\bigvee X v = \top$. ☐

6 State Reachability, Coverability and Büchi Acceptance in Energy Automata

We now show how the setting developed in the last sections can be applied to solve the energy problems of Section 2. Recall that $L = [0, \infty]_\bot$ denotes the complete lattice of nonnegative real numbers together with ∞ and an extra bottom element $\bot$, and that $\mathcal{E}$ denotes the idempotent semiring of energy functions $L \to L$. Note that L satisfies the precondition of Lemma 9.

Lemma 10. *Energy functions are finitely additive and $\top$-continuous, hence $\mathcal{E} \subseteq \mathsf{FinAdd}_L$.*

Proof. Finite additivity follows from monotonicity. For $\top$-continuity, let $X \subseteq L$ such that $\bigvee X = \infty$ and $f \in \mathcal{E}$, $f \neq \perp$. By $\bigvee X = \infty$, we know that for every $n \in \mathbb{N}$, there exists $x_n \in X$ with $x_n \geq n$. Choose such a sequence (x_n) and let $y_n = x_n f$ for all n.

If $y_n = \perp$ for all $n \geq 0$, then also $nf = \perp$ for all $n \geq 0$ (as $x_n \geq n$), hence $f = \perp$. We must thus have an index N for which $y_N > \perp$. But then $y_{N+k} \geq y_N + k \geq k$ for all $k \geq 0$, hence $\bigvee Xf = \infty$. □

Lemma 11. *For $f \in \mathcal{E}$, f^* is given by $xf^* = x$ if $xf \leq x$ and $xf^* = \infty$ if $xf > x$. Hence f is locally *-closed and $f^* \in \mathcal{E}$.*

Proof. We have $\perp f^* = \perp$ and $\infty f^* = \infty$. Let $x \neq \perp, \infty$. If $xf \leq x$, then $xf^n \leq x$ for all $n \geq 0$, so that $x \leq \bigvee_{n\geq 0} xf^n \leq x$, whence $xf^* = x$. If $xf > x$, then let $a = xf - x > 0$. We have $xf \geq x + a$, hence by $(*)$, $xf^n \geq x + na$ for all $n \geq 0$, so that $xf^* = \bigvee_{n\geq 0} xf^n = \infty$. □

Not all locally *-closed functions $f : L \to L$ are energy functions: the function f defined by $xf = 1$ for $x < 1$ and $xf = x$ for $x \geq 1$ is locally *-closed, but $f \notin \mathcal{E}$.

Corollary 1. *$\mathcal{E}$ is a *-continuous Kleene algebra.*

Proof. This is clear by Proposition 1. □

Remark 1. It is *not* true that $\mathcal{E}$ is a *continuous* Kleene algebra: Let $f_n, g \in \mathcal{E}$ be defined by $xf_n = x + 1 - \frac{1}{n+1}$ for $x \geq 0$, $n \geq 0$ and $xg = x$ for $x \geq 1$, $xg = \perp$ for $x < 1$. Then $0(\bigvee_{n\geq 0} f_n)g = (\bigvee_{n\geq 0} 0f_n)g = 1g = 1$, whereas $0 \bigvee_{n\geq 0}(f_n g) = \bigvee_{n\geq 0}(0f_n g) = \bigvee_{n\geq 0}((1 - \frac{1}{n+1})g) = \perp$.

Lemma 12. *For any $g \in \mathcal{E}$, there exists $f \in \mathcal{E}$ such that $g = f^*$ iff there is $k \in [0,\infty]_\perp$ such that $xg = x$ for all $x < k$, $xg = \infty$ for all $x > k$, and $kg = k$ or $kg = \infty$.*

Proof. We first note that if $g \in \mathcal{E}$ is such that there is k for which $xg = x$ for $x < k$ and $xg = \infty$ for $x > k$, then $xg^* = xg$ for $x \neq k$, and if $kg = k$ or $kg = \infty$, then also $kg^* = kg$.

Now let $g \in \mathcal{E}$. If there is $f \in \mathcal{E}$ with $g = f^*$, then we set $k = \sup\{x \mid xf \leq x\}$. Then $xf > x$ and hence $xg = \infty$ for all $x > k$, and whenever $x < k$, then there is y with $x \leq y \leq k$ and $yf \leq y$, hence by $(*)$, $xf \leq x$, so that $xg = x$. If $kf \leq k$, then $kg = k$, otherwise $kg = \infty$ as claimed. □

Let $\mathcal{V}$ denote the $\mathcal{E}$-semimodule of all $\top$-continuous functions $L \to \mathbf{2}$. For $f_0, f_1, \ldots \in \mathcal{E}$, define the infinite product $f = \prod_{n\geq 0} f_n : L \to \mathbf{2}$ by $xf = \perp$ if there is an index n for which $xf_0 \cdots f_n = \perp$ and $xf = \top$ otherwise, like in Section 5. By Lemma 9, $\prod_{n\geq 0} f_n$ is $\top$-continuous, *i.e.*, $\prod_{n\geq 0} f_n \in \mathcal{V}$.

By Proposition 2, $(\mathcal{E}, \mathcal{V})$ is a generalized *-continuous Kleene algebra.

Corollary 2. *$(\mathcal{E}, \mathcal{V})$ is a *-continuous Kleene ω-algebra.*

Proof. This is clear by Proposition 3. □

Remark 2. As $\mathcal{E}$ is not a continuous Kleene algebra, it also holds that $(\mathcal{E}, \mathcal{V})$ is not a continuous Kleene ω-algebra; in fact it is clear that there is no $\mathcal{E}$-semimodule $\mathcal{V}'$ for which $(\mathcal{E}, \mathcal{V}')$ would be a continuous Kleene ω-algebra. The initial motivation for the work in [25, 26, 28] and the present paper was to generalize the theory of continuous Kleene ω-algebras so that it would be applicable to energy functions.

Lemma 13. *For $f \in \mathcal{E}$, f^ω is given by $\bot f^\omega = \bot$, and for $x \neq \bot$, $x f^\omega = \bot$ if $xf < x$ and $x f^\omega = \top$ if $xf \geq x$.*

Proof. The claim that $\bot f^\omega = \bot$ is clear, and so is the lemma for $f = \bot$. For $f \neq \bot$ and $x = \infty$, $xf^n = \infty$ for all $n \geq 0$, hence $\infty f^\omega = \top$. Now let $x \neq \bot, \infty$. If $xf \geq x$, then $xf^n \geq x$ for all $n \geq 0$, hence $xf^\omega = \top$. If $xf < x$, then let $a = x - xf > 0$. We have $xf \leq x - a$, hence by $(*)$, $xf^n \leq x - na$ for all $n \geq 0$, so that there is $N \geq 0$ for which $xf^N = \bot$, whence $xf^\omega = \bot$. □

We can now solve the state reachability, coverability, and Büchi problems for energy automata. We say that $\mathcal{E}' \subseteq \mathcal{E}$ is *fixed-point decidable* if it is decidable, for any $f \in \mathcal{E}'$ and $x \in L$, whether $xf < x$, $xf = x$ or $xf > x$.

Theorem 2. *Let $A = (\alpha, M, k)$ be an energy automaton of dimension $n \geq 1$, $x_0 \in \mathbb{R}_{\geq 0}$, and $z : \{1, \dots, k\} \to \mathbb{R}_{\geq 0}$. Then*

- $\mathsf{Reach}_{\mathcal{E}'}(A, x_0) = \mathbf{tt}$ *iff* $x_0|A| \neq \bot$*;*
- $\mathsf{Cover}_{\mathcal{E}'}(A, x_0, z) = \mathbf{tt}$ *iff there exists* $i \leq k$ *such that* $(x_0 \alpha M^*)_i \geq iz$*;*
- $\mathsf{Büchi}_{\mathcal{E}'}(A, x_0) = \mathbf{tt}$ *iff* $x_0\|A\| = \top$*.*

Proof. For state reachability and Büchi acceptance the claims are clear. For coverability, we note that

$$(x_0 \alpha M^*)_i = \bigvee_{m \geq 0} \bigvee_{1 \leq k_1, \dots, k_m \leq n} x_0 \alpha_{k_1} M_{k_1, k_2} \cdots M_{k_m, i},$$

and the claim follows. □

Corollary 3. *For fixed-point decidable subalgebras $\mathcal{E}' \subseteq \mathcal{E}$, Problems 1, 2, and 3 are decidable. For an energy automaton of dimension n, the decision procedures use $O(n^3)$, $O(n^3)$, respectively $O(n^4)$, algebra operations.*

Proof. If $\mathcal{E}'$ is fixed-point decidable, then Lemmas 11 and 13 imply that the * and $^\omega$ operations are computable in $\mathcal{E}'$, and the matrix operations in Theorem 2 can be reduced to compositions, binary suprema, and these two operations. The complexity results follow from the fact that computation of M^* uses $O(n^3)$ operations and computation of M^ω uses $O(n^4)$ operations, *cf.* [30]. □

7 Some Complexity Results

We proceed to identify two important subclasses of computable energy functions, which cover most of the related work mentioned in the introduction, and to give complexity results on their reachability and Büchi acceptance problems.

The *integer update functions* in $\mathcal{E}$ are the functions f_k, for $k \in \mathbb{Z}$, given by

$$x f_k = \begin{cases} x + k & \text{if } x \geq \max(0, -k), \\ \perp & \text{otherwise,} \end{cases}$$

together with $f_{-\infty} := \perp\!\!\!\perp$ and $f_\infty := \top\!\!\!\top$. These are the update functions usually considered in integer-weighted automata and VASS [11, 14, 17–20, 33]. We have $f_k f_\ell = f_{k+\ell}$, $f_k \vee f_\ell = f_{\max(k,\ell)}$, and

$$f_k^* = \begin{cases} f_0 & \text{for } k \leq 0, \\ f_\infty & \text{for } k > 0, \end{cases} \qquad f_k^\omega = \begin{cases} f_{-\infty} & \text{for } k < 0, \\ f_\infty & \text{for } k \geq 0, \end{cases}$$

whence the class $\mathcal{E}_{\mathsf{int}}$ of integer update functions forms a subalgebra of $\mathcal{E}$. A function $f_k \in \mathcal{E}_{\mathsf{int}}$ can be represented by the (extended) integer k, and algebra operations can then be performed in constant time. Also, $\mathcal{E}_{\mathsf{int}}$ is trivially fixed-point decidable, so that Corollary 3 implies the following result.

Theorem 3. *For $\mathcal{E}_{\mathsf{int}}$-automata, Problems 1, 2 and 3 are decidable in PTIME.*

Remark 3. This means that state reachability, coverability and Büchi acceptance for one-dimensional VASS are decidable in PTIME, which seems not to have been noted before. (But see the recent [38], where reachability for one-dimensional *branching VASS* is shown decidable in PTIME. In [7] it is claimed that coverability for one-dimensional VASS is NP-complete.)

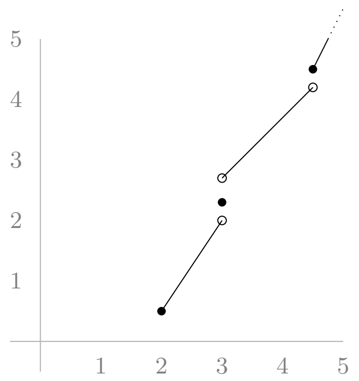
Next we turn our attention to piecewise affine functions.

Definition 5. *A function $f \in \mathcal{E}$ is said to be* (rational) piecewise affine *if there exist* $0 \leq x_0 < x_1 < \cdots < x_k \in \mathbb{Q} \cup \{\infty\}$ *such that*

- $xf = \perp$ *for* $x < x_0$ *and* $xf = \infty$ *for* $x > x_k$*,*
- $x_j f \in \mathbb{Q} \cup \{\perp, \infty\}$ *for all* j*, and*
- *all restrictions* $f_{\restriction]x_j, x_{j+1}[}$ *are affine functions* $x \mapsto a_j x + b_j$ *with* $a_j, b_j \in \mathbb{Q}$*,* $a_j \geq 1$*.*

Let $\mathcal{E}_{\mathsf{pw}} \subseteq \mathcal{E}$ denote the class of piecewise affine energy functions. The notion of *integer piecewise affine* functions, $\mathcal{E}_{\mathsf{pwi}}$, is defined similarly, with all occurrences of $\mathbb{Q}$ above replaced by $\mathbb{Z}$. Clearly $\mathcal{E}_{\mathsf{int}} \subseteq \mathcal{E}_{\mathsf{pwi}} \subseteq \mathcal{E}_{\mathsf{pw}}$.

Note that the definition does not make any assertion about continuity at the x_j, but $(*)$ implies that $\lim_{x \nearrow x_j} xf \leq x_j f \leq \lim_{x \searrow x_j} xf$. A piecewise affine function as above can be represented by its break points $x_0, \dots, x_k$, the values $x_0 f, \dots, x_k f$,



$$xf = \begin{cases} \bot & \text{for } x < 2 \\ .5 & \text{for } x = 2 \\ 1.5\,x - 2.5 & \text{for } 2 < x < 3 \\ 2.3 & \text{for } x = 3 \\ x - .3 & \text{for } 3 < x < 4.5 \\ 4.5 & \text{for } x = 4.5 \\ 2\,x - 4.5 & \text{for } x > 4.5 \end{cases}$$

Figure 2: A piecewise affine energy function

and the numbers $a_0, b_0, \dots, a_k, b_k$. These functions arise in the reduction used in [10] to show decidability of energy problems for one-clock timed automata with transition updates, see Section 8. Fig. 2 shows an example of a piecewise affine energy function.

The class of piecewise affine energy functions forms a subsemiring of $\mathcal{E}$: if $f, g \in \mathcal{E}_{\mathsf{pw}}$ with break points $x_0, \dots, x_k$ and $y_0, \dots, y_\ell$, respectively, then $f \vee g$ is piecewise affine with break points obtained from the break points of f and g together with intersection points of lines (which are rational), and fg is piecewise affine with break points a subset of $\{x_0, \dots, x_k,\ y_0 f^{-1}, \dots, y_\ell f^{-1}\}$ (which are all rational). Hence maxima and compositions of piecewise affine energy functions are computable, but may increase the size of their representation.

Now let, for any $p \in \mathbb{Q}$ with $p \geq 0$, $g_p^-, g_p^+ \in \mathcal{E}_{\mathsf{pw}}$ be the functions defined by

$$x g_p^- = \begin{cases} x & \text{for } x < p\,, \\ \infty & \text{for } x \geq p\,, \end{cases} \qquad x g_p^+ = \begin{cases} x & \text{for } x \leq p\,, \\ \infty & \text{for } x > p\,. \end{cases}$$

Proposition 4. *$\mathcal{E}_{\mathsf{pw}}$ is a *-continuous Kleene algebra.*

Proof. In lieu of Proposition 1, we need to show that $\mathcal{E}_{\mathsf{pw}}$ is closed under the *-operation. Let $f \in \mathcal{E}_{\mathsf{pw}}$, then by Lemma 12, there is a $p \in \mathbb{Q}$ such that $f^* = g_p^-$ or $f^* = g_p^+$. □

Remark that, unlike $\mathcal{E}_{\mathsf{pw}}$, the class $\mathcal{E}_{\mathsf{pwi}}$ of *integer* piecewise affine functions does *not* form a subsemiring of $\mathcal{E}$, as composites of $\mathcal{E}_{\mathsf{pwi}}$-functions are not necessarily integer piecewise affine. As an example, for the functions $f, g \in \mathcal{E}_{\mathsf{pwi}}$ given by

$$xf = 2x\,, \qquad xg = \begin{cases} x + 1 & \text{for } x < 3\,, \\ x + 2 & \text{for } x \geq 3\,, \end{cases}$$

we have

$$xfg = \begin{cases} 2x + 1 & \text{for } x < 1.5\,, \\ 2x + 2 & \text{for } x \geq 1.5\,, \end{cases}$$

which is not integer piecewise affine. The semiring generated by $\mathcal{E}_{\mathsf{pwi}}$ is the subsemiring of $\mathcal{E}_{\mathsf{pw}}$ of functions with *rational* break points $x_0, \dots, x_k$, but *integer* values $a_0, b_0, \dots, a_k, b_k$.

Similarly, the class of rational *affine* functions $x \mapsto ax + b$, $a, b \in \mathbb{Q}$, $a \geq 1$ (without break points) is not closed under maximum, and it can be seen that $\mathcal{E}_{\mathsf{pw}}$ is the semiring generated by rational affine functions.

Lemma 14. *$\mathcal{E}_{\mathsf{pw}}$ is fixed-point decidable.*

Proof. Let $f \in \mathcal{E}_{\mathsf{pw}}$, with representation $(x_0, \dots, x_k, x_0 f, \dots, x_k f, a_0, \dots, a_k, b_0, \dots, b_k)$. Let $x \in \mathbb{R}_{\geq 0}$ be computable; we need to decide whether $xf < x$, $xf = x$ or $xf > x$. If $x < x_0$, then $xf = \bot$. If $x = x_j$ for some j, we can simply compare x_j with $x_j f$.

Assume now that $x \in]x_j, x_{j+1}[$ for some j. If $a_j x_j + b_j < x_j$ and $a_j x_{j+1} + b_j \leq x_{j+1}$, then $xf < x$ by $(*)$. If $a_j x_j + b_j = x_j$ and $a_j x_{j+1} + b_j = x_{j+1}$, then also $xf = x$, and if $a_j x_j + b_j \geq x_j$ and $a_j x_{j+1} + b_j > x_{j+1}$, then $xf > x$. The cases $a_j x_j + b_j > x_j$, $a_j x_{j+1} + b_j \leq x_{j+1}$ and $a_j x_j + b_j \geq x_j$, $a_j x_{j+1} + b_j < x_{j+1}$ cannot occur because of $(*)$.

The last case to consider is $a_j x_j + b_j < x_j$ and $a_j x_{j+1} + b_j > x_{j+1}$. Then we must have $a_j > 1$, and then $xf < x$ if $x < \frac{b_j}{1-a_j}$, $xf = x$ if $x = \frac{b_j}{1-a_j}$, and $xf > x$ if $x > \frac{b_j}{1-a_j}$. ☐

Theorem 4. *For $\mathcal{E}_{\mathsf{pw}}$-automata, Problems 1, 2 and 3 are decidable in EXPTIME.*

Proof. Decidability follows from Corollary 3 and Lemma 14. For the complexity claim, we note that all algebra operations in $\mathcal{E}_{\mathsf{pw}}$ can be performed in time linear in the size of the representations of the involved functions. However, the maximum and composition operations may triple the size of the representations, hence our procedure may take time $O(3^{n^3} p)$ for state reachability and coverability, and $O(3^{n^4} p)$ for Büchi acceptance, for an $\mathcal{E}_{\mathsf{pw}}$-automaton of dimension n and energy functions of representation length at most p. ☐

We believe that the above complexity bound of EXPTIME can be considerably sharpened, but we leave this for future work.

8 Reduction from Weighted Timed Automata

To further motivate the introduction of our notion of energy automata, we review here how the treatment of lower-bound energy problems for one-clock weighted timed automata in [10, 11] naturally leads to our energy functions and energy automata. In this section, and this section only, function application and composition will be written in the standard right-to-left order.

A *weighted timed automaton* $A = (L, l_0, C, I, E, r)$ consists of a finite set of *locations* L with initial location l_0, a finite set of *clocks* C, location *invariants*

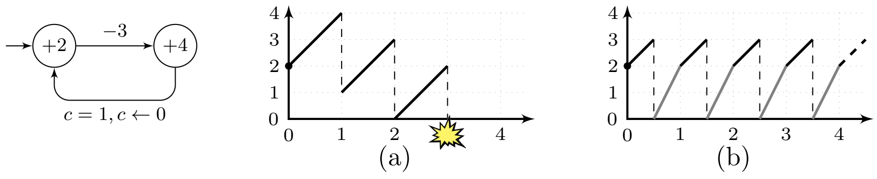


Figure 3: One-clock weighted timed automaton with discrete updates. Any region-stable scheduler (*i.e.,* with switches at *integer* times) is doomed (a), but there exists a feasible schedule with switches at half-integer times (b).

$I : L \to \Phi(C)$, weighted *edges* $E \subseteq L \times \Phi(C) \times 2^C \times \mathbb{Z} \times L$ and location *weight rates* $r : L \to \mathbb{Z}$. Here the set $\Phi(C)$ of clock constraints ϕ is defined by the grammar

$$\phi ::= c \bowtie k \mid \phi_1 \wedge \phi_2 \qquad (c \in C, k \in \mathbb{Z}, \bowtie \in \{\leq, <, \geq, >, =\}).$$

A *clock valuation* is a mapping $C \to \mathbb{R}_{\geq 0}$. For a clock valuation $v : C \to \mathbb{R}_{\geq 0}$ and a clock constraint $\phi \in \Phi(C)$, we write $v \models \phi$ to indicate that v satisfies ϕ. We denote by $v_0 : C \to \mathbb{R}_{\geq 0}$ the clock valuation given by $v_0(c) = 0$ for all $c \in C$. For a clock valuation $v : C \to \mathbb{R}_{\geq 0}$, $d \in \mathbb{R}_{\geq 0}$, and $R \subseteq C$, we denote by $v + d$ and $v[R \leftarrow 0]$ the clock valuations given by $(v + d)(c) = v(c) + d$ for all $c \in C$ and $v[R \leftarrow 0](c) = 0$ for $c \in R$, $v[R \leftarrow 0](c) = v(c)$ for $c \notin R$.

The *semantics* of a weighted timed automaton $A = (L, l_0, C, I, E, r)$ is given by an infinite weighted automaton $[\![A]\!] = (S_A, s_0, T_A)$ with states $S_A = \{(l, v) \mid v \models I(l)\} \subseteq L \times \mathbb{R}^C_{\geq 0}$, initial state $s_0 = (l_0, v_0)$, and transitions $T_A \subseteq S_A \times \mathbb{R} \times S_A$ of two types:

- *delay transitions* $(l, v) \xrightarrow{r(l)d} (l, v + d)$ for all $d \in \mathbb{R}_{\geq 0}$ such that $v + d' \models I(l)$ for all $d' \in [0, d]$;
- *switch transitions* $(l, v) \xrightarrow{p} (l', v')$, where $e = (l, \phi, R, p, l') \in E$ is a transition of A, $v \models \phi$ and $v' = v[R \leftarrow 0]$.

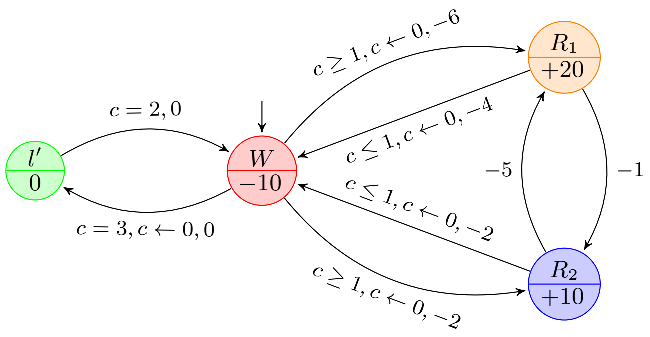
We refer to [34] for a thorough survey on timed automata and weighted timed automata.

The *lower-bound energy problem* for a weighted timed automaton A as above is, given an initial energy $x_0 \in \mathbb{R}_{\geq 0}$, to decide whether there exist an infinite path

$$(l_0, v_0) \xrightarrow{p_0} (l_1, v_1) \xrightarrow{p_1} (l_2, v_2) \xrightarrow{p_2} \cdots$$

of delay and switch transitions in $[\![A]\!]$ for which $x_0 + \sum_{i=0}^n p_i \geq 0$ for all $n \in \mathbb{N}$. We hence want to decide whether there is a run in A where the *accumulated energy* $x_0 + \sum_{i=0}^n p_i$ never drops below zero. We shall say that such a run is *feasible.* Figure 1 in the introduction shows an example of such an energy problem.

For *one-clock* weighted timed automata *without discrete updates*, *i.e.,* with $C = \{c\}$ a singleton and $p = 0$ for all $(l, \phi, R, p, l') \in E$, it was shown in [11] that this problem can be decided via a simple reduction to a refinement of the region



R_1
$+20$
W
-10
l'
0
R_2
$+10$
$c \geq 1, c \leftarrow 0, -6$
$c = 2, 0$
$c \leq 1, c \leftarrow 0, -4$
-5
-1
$c = 3, c \leftarrow 0, 0$
$c \leq 1, c \leftarrow 0, -2$
$c \geq 1, c \leftarrow 0, -2$

Figure 4: Conversion of the weighted timed automaton of Fig. 1 to a 3-bounded weighted timed automaton. To simplify the example, we assume that the invariant of R_1 and R_2 in the automaton of Fig. 1 (and thus here) is $c \leq 1$. The invariant of location W is $c \leq 3$, and the invariant of the new location l' is $c \leq 2$.

graph [2] of A. Figure 3, taken from [10], however, shows that a similar region-stable reduction is not available for one-clock weighted timed automata *with* discrete updates. In the rest of this section we review the substantially more complicated reduction from [10]. For simplicity of presentation we assume the input timed automaton to be *closed*, *i.e.,* only using closed clock constraints $c \leq k$, $c \geq k$, $c = k$ and their conjunctions.

Let $A = (L, l_0, \{c\}, I, E, r)$ be a closed one-clock weighted timed automaton. First, we make sure that A is *bounded*, *i.e.,* that the value of c never exceeds a constant M during any run of A. The construction, based on [5], works essentially by resetting the clock whenever it reaches value M. Formally, let m be the greatest integer constant which appears in any invariant $I(l)$, $l \in L$ and any constraint ϕ in $(l, \phi, R, p, l') \in E$ and set $M = m + 2$. Now for each location $l \in L$, add an invariant $c \leq M$ to $I(l)$ and a new location l' to L, with $I(l') = (c \leq M - 1)$ and $r(l') = 0$, and edges $(l, (c = M), \{c\}, 0, l')$, $(l', (c = M - 1), \emptyset, 0, l)$. Figure 4 shows an example of the construction for the weighted timed automaton in Fig. 1 for $M = 3$. It can be shown [5,10] that the so-constructed bounded automaton admits the same infinite lower-bounded runs for the same initial energies as the old one.

Second, we make sure that the bound $M = 1$. This is done, like in [13], by splitting A into stages, one for every integer $k \in \{0, \dots, M - 1\}$; the intuition is that a state $((l, k), \nu)$ at stage k corresponds to a state (l, ν') with old clock value $\nu' = k + \nu$. Figure 5 shows an example of the construction. Third, we also eliminate edges $(l, (c' \leq 1), \{c'\}, p, l')$ by introducing new locations l'' with $r(l'') = 0$, $I(l'') = c \leq 1$, and edges $(l, (c' \leq 1), \emptyset, 0, l'')$, $(l'', (c' = 1), \{c'\}, p, l')$, see Figure 6. It can again be shown [10] that this construction does not affect energy properties.

Next, noticing that A now only has three types of edges: reset-free edges with constraint $c' \leq 1$ and resetting edges with constraint $c' = 0$ or $c' = 1$, we split the

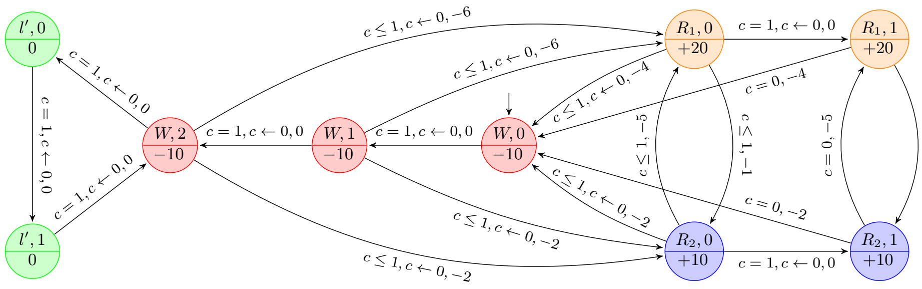


Figure 5: Conversion of the 3-bounded weighted timed automaton of Fig. 4 to a 1-bounded weighted timed automaton.

locations of A so that each location l either has only incoming reset-free edges or only incoming resetting edges. Possibly adding a new initial location, we also make sure that l_0 has no incoming edges. Fig. 7 shows the complete conversion of the weighted timed automaton of Fig. 1.

Let S be the set of locations without incoming reset-free edges, then $l_0 \in S$. For each pair $l, l' \in S$, let $P(l, l')$ be the (finite) set of paths in A from l to l' which

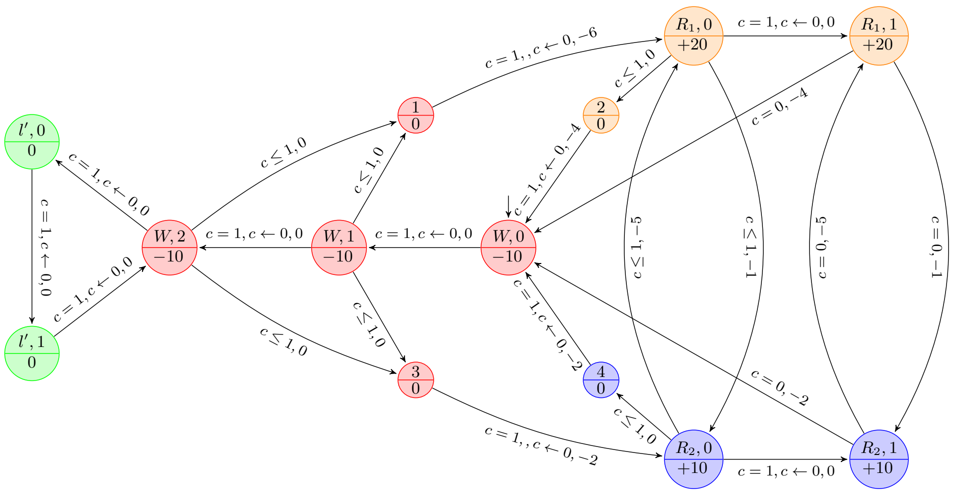


Figure 6: The 1-bounded weighted timed automaton obtained by eliminating resetting edges with clock constraint $c \leq 1$ from the 1-bounded weighted timed automaton in Fig. 5. We eliminate, for instance, the edge $(\langle W, 2\rangle, (c \leq 1), \{c\}, -6, \langle R_1, 0\rangle)$ by introducing an auxiliary location 1 with invariant $c \leq 1$ and rate 0, and edges $(\langle W, 2\rangle, (c \leq 1), \emptyset, 0, 1)$ and $(1, c = 1, \{c\}, -6, \langle R_1, 0, \rangle)$. Note that in the resulting automaton all edges with constraint $c \leq 1$ are reset-free, and all resetting edges have constraint $c = 1$ or $c = 0$.

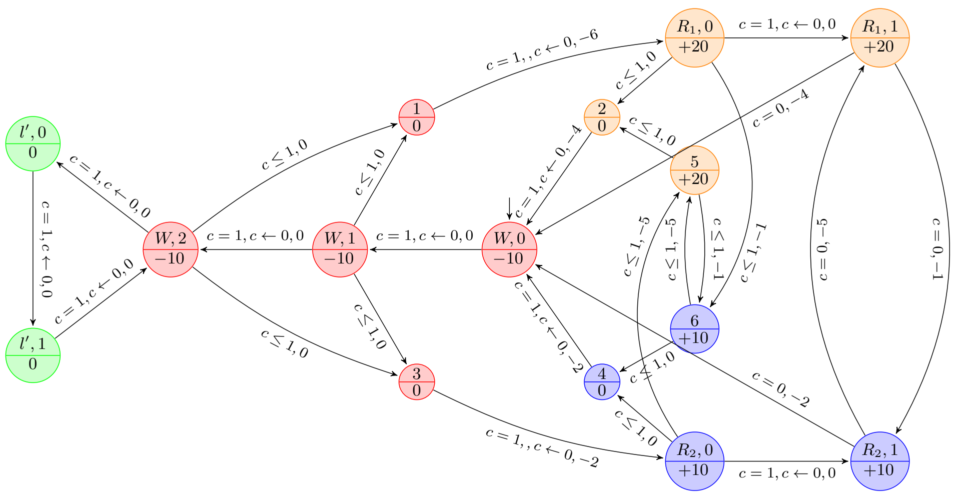


Figure 7: Conversion of the weighted timed automaton in Fig. 6 to one with locations partitioned into locations with only reset-free incoming edges with constraint $c \leq 1$ and locations with only resetting edges and constraint $c = 1$ or $c = 0$ (we omit resets at edges with constraint $c = 0$). Note that in Fig. 6 locations $(R_1, 0)$ and $(R_2, 0)$ have both resetting and reset-free incoming edges. We introduce two auxiliary locations 5 and 6 and redirect edges accordingly.

only go through locations in $L \setminus S$, and which contain at most two copies of any simple cycle. (It is shown in [10] that paths which contain more than two copies of a simple cycle can be reduced to paths in $P(l, l')$ by collecting all delays in the first two copies.) For each such path $\pi \in P(l, l')$, let $f_\pi : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be the function mapping input energy in l to maximal achievable output energy in l' and define $T = \{(l, f_\pi, l') \mid \pi \in P(l, l')\}$. Let $B = (S, l_0, T, S)$; we will show below that B is an energy automaton.

It is clear that any infinite run in A which traverses resetting edges infinitely often will translate to an infinite path in B. We need, however, to take special care of infinite runs in A which are *eventually reset-free*. To do so, it is shown in [10] that we can compute, in EXPTIME, a mapping $z : L \to \mathbb{R}_{\geq 0}$ such that there is a *feasible* infinite reset-free run from a state (l, v) in $[\![A]\!]$ iff $v \geq z(l)$. For the proof of this, one observes that any such run can be converted, by eliminating all delays in locations with nonpositive rates and collecting all delays in locations with positive rates in their first occurrences, into one where, after a finite prefix, no more time elapses. Also, the length of these prefixes is at most $|L|(|L| + 1)$, so $z(l)$ can be computed in finite time.

It can now be shown [10, Lemma 17] that, for any $x_0 \in \mathbb{R}_{\geq 0}$, there is a feasible infinite run from (l_0, x_0) in A iff (1) there is an accepting infinite run in B with initial energy x_0, or (2) a state $l \in S$ is reachable in B, with initial energy x_0, such that the energy in l is at least $z(l)$. Hence the lower-bound energy problem for A

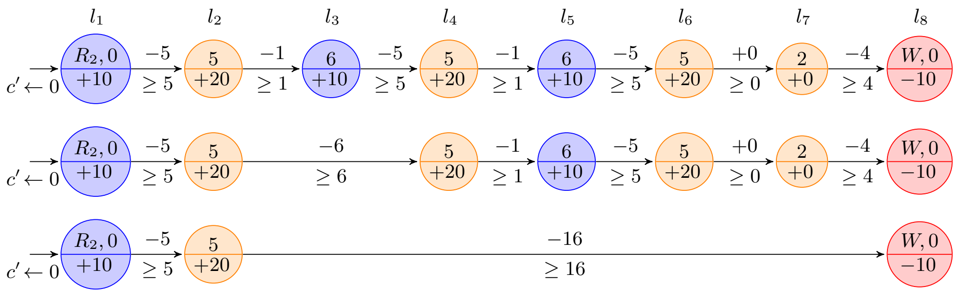


Figure 8: Reset-free path with annotations (top) and after repeated application of the first normalization operation.

from x_0 reduces to $\textsf{Büchi}(B)(x_0)$ and $\textsf{Cover}(B)(x_0, z)$.

We miss to show that $B = (S, l_0, T, S)$ is an energy automaton. Let $l, l' \in S$ and $\pi = (l = l_1, e_1, l_2, \dots, e_n, l_{n+1} = l') \in P(l, l')$; note that e_n is the only resetting edge. If the constraint on e_n is $c' = 0$, then no time elapses during π, and, letting p_i denote the weight of $e_i = (l_i, \phi, \emptyset, p_i, l_{i+1})$ (where $\phi = (c' \leq 1)$), we have

$$f_\pi(x) = \begin{cases} \text{undefined} & \text{if } x + \sum_{i=1}^{k} p_i < 0 \text{ for some } k \in \{1, \dots, n\}\,, \\ x + \sum_{i=1}^{n} p_i & \text{otherwise}\,. \end{cases}$$

If, on the other hand, the last constraint in π is $\phi_n = (c' = 1)$, then we face the task of distributing one time unit of delay optimally through the locations along π. In order to do so, we first annotate the edges along π with *lower-bound constraints*. Hence each e_i is now of the form $e_i = (l_i, \phi, \emptyset, p_i, b_i, l_{i+1})$, with $b_i = -p_i$ initially and the semantics that the edge e_i is enabled for input energy $x \geq b_i$.

We modify π by removing locations in which an optimal path (*i.e.,* with maximal energy output) will not delay. To ease the presentation, we assume that the maximal rate along π is positive, *i.e.,* $\max\{r(l_i) \mid i = 1, \dots, n\} > 0$. The constructions are similar in the other case; see [10] for details. Figure 8 shows as an example a path from $(R_2, 0)$ to $(W, 0)$ in the weighted timed automaton of Fig. 7.

First we note that if $r(l_i) \geq r(l_{i+1})$ for some $i \in \{1, \dots, n-1\}$, then any delay spent in l_{i+1} could just as well (or better) have been spent in l_i. (This is the case for $i = 2$ (location 6 with rate +10) in the example.) Hence we can remove l_{i+1} and update π with an edge $(l_i, \phi, \emptyset, p_i + p_{i+1}, \max(b_i, b_{i+1} - p_i), l_{i+2})$. The new lower-bound constraint $\max(b_i, b_{i+1} - p_i)$ is chosen so that the new edge can be taken precisely when the sequence of the old edges could be taken without intermediate delay.

This modified path π has the property that $r(l_i) < r(l_j)$ for all $1 \leq i < j \leq n$. Next, we see that if there is $i \in \{1, \dots, n-1\}$ for which $b_i + p_i \geq b_{i+1}$, then there is no need to spend any delay in l_{i+1}, as we can go directly to l_{i+2} which has a higher rate. (In our example, this case does not occur.) Hence we modify π once again, removing l_{i+1} and adding a new edge $(l_i, \phi, \emptyset, p_i + p_{i+1}, b_i, l_{i+2})$. The result

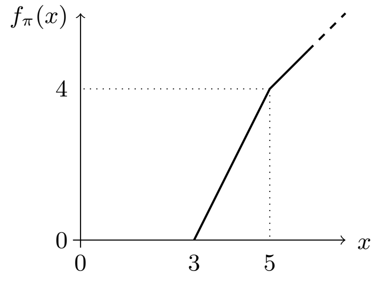


Figure 9: The energy function associated with the path of Fig. 8.

of these two kinds of modifications is a reset-free path in so-called *normal form*, see again Fig. 8.

As the last step, we show by example how to compute the energy function of a path in normal form; we refer to [10] for the general algorithm. Let π be the path at the bottom of Fig. 8; we want to compute the partial function $f_\pi : \mathbb{R}_{\geq 0} \rightharpoonup \mathbb{R}_{\geq 0}$ which maps the input energy x, entering the first location of π, to the maximum available output energy $f_\pi(x)$ when leaving π.

First we notice that if $x < 3$, then we need to spend a delay of $d_1 \geq \frac{5-x}{10} > \frac{1}{5}$ in l_1 to meet its output constraint of $x \geq 5$. The energy value when entering l_2 is then $x + 10d_1 - 5$, which is equal to 0, so that after l_2, the value is $20d_2 \leq 20(1 - d_1) = 20 - 20d_1 < 16$. Hence we cannot match the output constraint $x \geq 16$ on l_2, so that $f_\pi(x)$ is undefined for $x < 3$. On the other hand, when $x = 3$ at the start of π, then we can delay $\frac{1}{5}$ time units in l_1, then delay $\frac{4}{5}$ time units in l_2, and finally achieve $f_\pi(3) = 0$.

It can be shown [10] that the general strategy for maximizing the output energy, given a path in normal form such as the one in the bottom of Fig. 8, is to delay in every location precisely the time necessary for meeting its output constraint, and then to spend any remaining time in the last location. Hence when x is between 3 and 5, we can let $d_1 = \frac{5-x}{10}$ and then $d_2 = 1 - d_1$. This gives a value of $f_\pi(4) = 4 + \frac{1}{10} \cdot 10 - 5 + \frac{9}{10} \cdot 20 - 16 = 2$. For $x \geq 5$, we need not delay in l_1 at all, so in this case, $f_\pi(x) = x - 1$. See Fig. 9 for a graph of the function thus obtained.

Given this general strategy for maximizing output energy, it can be shown that the energy function f_π associated with a path in normal form, or indeed with a general reset-free path π, is a *continuous piecewise affine* function which satisfies $(*)$, *cf.* Definition 5. Now when the input timed automaton is not closed, then the definition interval of f_π ($[1, \infty[$ in the example) may be left-open, and taking maxima of such functions may introduce discontinuities, so that in the end, $B = (S, l_0, T, S)$ is an automaton which has transition weights from the general class $\mathcal{E}_{\mathsf{pw}}$ of piecewise-affine energy functions.

The results of Section 7 thus apply and allow us to compute $\mathsf{Büchi}(B)(x_0)$ and $\mathsf{Cover}(B)(x_0, z)$ in exponential time. As the reduction from A to B shown here may incur an exponential blow-up, our overall procedure for solving lower-bound

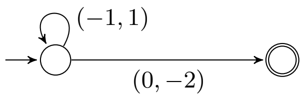


Figure 10: A simple two-dimensional VASS

energy problems in weighted timed automata has double exponential complexity.

9 Multi-dimensional Energy Automata and Games

We turn our attention to several variants of energy automata. We will first be concerned with multi-dimensional energy automata and show that their coverability problem is EXPSPACE-complete. Then we will show that this does not apply to *flat* energy functions, which are not required to satisfy $(*)$; for such functions, coverability is undecidable from dimension four. Finally, we show that reachability games on two-dimensional energy automata and on one-dimensional flat energy automata are undecidable.

An *n-dimensional energy automaton*, or $\mathcal{E}^n$*-automaton* for short, (S, T), for $n \geq 1$, consists of finite sets S of states and $T \subseteq S \times \mathcal{E}^n \times S$ of transitions. By restricting transition labelings, we can define subclasses of $\mathcal{E}^n_{\mathsf{pw}}$-automata, $\mathcal{E}^n_{\mathsf{pwi}}$-automata, and $\mathcal{E}^n_{\mathsf{int}}$-automata.

A *global state* in such an automaton is a pair $(s, \boldsymbol{x}) \in S \times \mathbb{R}^n_{\geq 0}$, and *transitions* are of the form $(s, \boldsymbol{x}) \xrightarrow{\boldsymbol{f}} (s', \boldsymbol{x}')$ such that $(s, \boldsymbol{f}, s') \in T$ and $\boldsymbol{x}'(i) = \boldsymbol{x}(i)\boldsymbol{f}(i)$ for each $i \in \{1, \dots, n\}$. Here, $\boldsymbol{u}(i)$ denotes the ith element of the vector $\boldsymbol{u}$.

A *run* of an $\mathcal{E}^n$-automaton (S, T) from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{x'})$ is a finite sequence $(s_0, \boldsymbol{x_0}), \dots, (s_k, \boldsymbol{x_k})$ of global states such that $(s_0, \boldsymbol{x_0}) = (s, \boldsymbol{x})$, $(s_k, \boldsymbol{x_k}) = (s', \boldsymbol{x'})$, and for all $i \in \{1, \dots, n\}$ there exists $\boldsymbol{f}_i \in \mathcal{E}^n$ such that $(s_{i-1}, \boldsymbol{x}_{i-1}) \xrightarrow{\boldsymbol{f}_i} (s_i, \boldsymbol{x}_i)$. We say that such a run has length k.

We define an ordering $\leq$ on global states by $(s, \boldsymbol{x}) \leq (s', \boldsymbol{x'})$ if, and only if, $s = s'$ and $\boldsymbol{x}(i) \leq \boldsymbol{x'}(i)$ for each $i = 1, \dots, n$. We restate the *coverability problem* and the *state reachability problem*: Given an $\mathcal{E}^n$-automaton, an initial global state $(s, \boldsymbol{x})$, where $\boldsymbol{x} \in \mathbb{R}^n_{\geq 0}$ is a computable initial energy, and some global state $(s', \boldsymbol{x'})$, the coverability problem is to decide whether there exists a run from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{x''})$ such that $\boldsymbol{x'} \leq \boldsymbol{x''}$. The state reachability problem is a special case of the coverability problem for $\boldsymbol{x'} = \boldsymbol{0}$.

For reachability in $\mathcal{E}^n$-automata with $n \geq 2$, our algebraic results do not apply. To see this, we refer to the state reachability problem in Fig. 10: with initial energy $(1, 1)$, the loop needs to be taken precisely once, but with initial energy $(2, 0)$, one needs to loop twice. To make this argument precise, let f denote the function corresponding to the $(-1, 1)$ loop and g the function on the $(0, -2)$ edge. Then we should have $(1, 1)f^*g = (1, 1)fg$ and $(2, 0)f^*g = (2, 0)f^2g$. Hence our framework of computing with energy *functions* will not apply.

9.1 $\mathcal{E}^n_{\mathsf{pw}}$-automata

Recall that n-dimensional VASS form a subclass of our $\mathcal{E}^n_{\mathsf{pw}}$-automata. The coverability problem for VASS is EXPSPACE-complete [44, 47]. In this subsection we aim to show the same complexity for the coverability and the state reachability problem for $\mathcal{E}^n_{\mathsf{pw}}$-automata. For EXPSPACE-membership, we extend Rackoff's proof for VASS [47].

The proof is inspired by a proof for EXPSPACE-completeness for the class of *strongly increasing affine nets* [9]. Affine nets are extensions of classical Petri nets. Recall that in Petri nets the current placement of tokens in the places (called *marking*) changes according to the transition rules, which simply add or subtract tokens from a place. In affine nets, the transition rules are affine functions of the form $\boldsymbol{A}M + \boldsymbol{B}$, where $\boldsymbol{A} \in \mathbb{N}^n \times \mathbb{N}^n$, $\boldsymbol{B} \in \mathbb{N}^n$, and M is a marking of the n-dimensional net. Such a function is *strongly increasing* if $\boldsymbol{A}$ is greater than or equal to the identity matrix, *cf.* condition $(*)$ for energy functions. Note that strongly increasing affine nets operate on vectors over $\mathbb{N}$, while $\mathcal{E}^n_{\mathsf{pw}}$-automata operate on vectors over $\mathbb{R}_{\geq 0}$. However, we remark that even $\mathcal{E}^n_{\mathsf{pwi}}$-automata and strongly increasing affine nets are incomparable in expressiveness: affine nets do not allow for *piecewise* affine functions, and in $\mathcal{E}^n_{\mathsf{pwi}}$-automata the value of an energy variables cannot influence the value of another energy variable as it is the case in affine nets. That is, in $\mathcal{E}^n_{\mathsf{pwi}}$-automata all $\boldsymbol{A}$ matrices are *diagonal*.

Before we prove EXPSPACE-completeness of the coverability problem for $\mathcal{E}^n_{\mathsf{pw}}$-automata, we introduce some helpful notions and prove some lemmas.

Recall that every 1-dimensional integer piecewise affine energy function $f \in \mathcal{E}_{\mathsf{pw}}$ can be represented by its breakpoints $x_0, \dots, x_m \in \mathbb{Q}$, the values $x_0 f, \dots, x_m f \in \mathbb{Q}$, and the numbers $a_0, b_0, a_1, b_1, \dots, a_m, b_m \in \mathbb{Z}$, where $a_j \geq 1$ for all $0 \leq j \leq m$. We use $\mathsf{xmin}_f = x_0$ to denote the *minimal break point* of f. For simplicity we assume that xf is defined iff $x \geq \mathsf{xmin}_f$, but our arguments also apply to the case where the definition interval of f is open. For n-dimensional integer piecewise affine energy functions $\boldsymbol{f} \in \mathcal{E}^n_{\mathsf{pw}}$, we define $\mathsf{xmin}_{\boldsymbol{f}} \in \mathbb{Q}^n$ to be the n-dimensional vector defined by $\mathsf{xmin}_{\boldsymbol{f}}(i) = \mathsf{xmin}_{\boldsymbol{f}(i)}$ for all $i \in \{1, \dots, n\}$.

Fix some $\mathcal{E}^n_{\mathsf{pw}}$-automaton (S, T). We use bmax to denote the maximum of the absolute values of all negative constants smaller than or equal to -1 and occurring in the representation of any energy function in (S, T), or 1 if there is no such constant. We further use $\mathsf{xminmax}$ to denote the maximum of 1 and the minimal break points of all energy functions occurring in (S, T). (Hence $\mathsf{xminmax} \geq 1$.)

The next lemma states the easy fact that the decrease in the value of an energy variable during a run is bounded.

Lemma 15. *For every run from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{x'})$ of length k, $\boldsymbol{x'}(i) \geq \boldsymbol{x}(i) - k\,\mathsf{bmax}$.*

On the other hand, the value of an energy variable can grow quickly even in very short runs. However, for deciding the coverability problem, already for VASS, it is not necessary to store the exact concrete value of an energy variable once it has exceeded a certain high value. Instead, we will represent high values symbolically by ω. The energy variables in the global states of our algorithm will hence take values in $\mathbb{R}_\omega := \mathbb{R}_{\geq 0} \cup \{\omega\}$. Define for every $y \in \mathbb{R}_\omega$, $\omega + y = \omega - y = \omega$, $y \cdot \omega = \omega$, and $y \leq \omega$. Using this, we can extend the definition of $f \in \mathcal{E}_{\mathsf{pw}}$ to a function $f : \mathbb{R}_\omega \to \mathbb{R}_\omega$ in a natural way. We further extend these definitions to the n-dimensional case. In the following, we will use $\boldsymbol{x}$ to denote vectors in $\mathbb{R}^n$, and $\boldsymbol{y}$ to denote vectors in $(\mathbb{R}_\omega)^n$.

Let $\boldsymbol{y} \in (\mathbb{R}_\omega)^n$ and assume $\boldsymbol{y}(i) \in \mathbb{R}_{\geq 0}$. We explain when to replace the concrete value $\boldsymbol{y}(i)$ by ω. The crucial point here is that there is not a single threshold value $t \in \mathbb{R}_{\geq 0}$ such that $\boldsymbol{y}(i)$ is replaced by ω whenever $\boldsymbol{y}(i) \geq t$. Instead, $\boldsymbol{y}(i)$ is replaced by ω whenever $\boldsymbol{y}(i) \geq t(r)$, where r is the number of indices j for which $\boldsymbol{y}(j) \in \mathbb{R}_{\geq 0}$, and t is a mapping $t : \{0, \ldots, n\} \to \mathbb{R}_{\geq 0}$. To make this formal, we define

$$\begin{aligned} \mathsf{omega}(\boldsymbol{y}) &= \{i \in \{1, \ldots, n\} \mid \boldsymbol{y}(i) = \omega\}, \\ \mathsf{real}(\boldsymbol{y}) &= \{i \in \{1, \ldots, n\} \mid \boldsymbol{y}(i) \in \mathbb{R}_{\geq 0}\}. \end{aligned}$$

For a finite set λ we use $|\lambda|$ to denote its cardinality. Let $t : \{0, 1, \ldots, n\} \to \mathbb{R}_{\geq 0}$ be a mapping. We define the vector $\boldsymbol{y}_t \in (\mathbb{R}_\omega)^n$ by

$$\boldsymbol{y}_t(i) = \begin{cases} \boldsymbol{y}(i) & \text{if } \boldsymbol{y}(i) < t(|\mathsf{real}(\boldsymbol{y})|), \\ \omega & \text{if } \boldsymbol{y}(i) \geq t(|\mathsf{real}(\boldsymbol{y})|). \end{cases}$$

Thus, in $\boldsymbol{y}_t$ all entries which are greater than or equal to the value $t(|\mathsf{real}(\boldsymbol{y})|)$ are replaced by ω; other entries do not change.

Next, we define the *abstract t-semantics* of (S, T). For this, let $t : \{0, 1, \ldots, n\} \to \mathbb{R}_{\geq 0}$ be a mapping. A *global t-state* of (S, T) is a pair $(s, \boldsymbol{y}) \in S \times (\mathbb{R}_\omega)^n$. We define a t-transition relation over the set of global t-states by $(s, \boldsymbol{y}) \xrightarrow{\boldsymbol{f}}_t (s', \boldsymbol{y'})$ iff $(s, \boldsymbol{f}, s') \in T$, $\boldsymbol{y}\boldsymbol{f}$ is defined, and $\boldsymbol{y'} = (\boldsymbol{y}\boldsymbol{f})_t$. A t-run of (S, T) from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ is a finite sequence $(s_0, \boldsymbol{y_0}), (s_1, \boldsymbol{y_1}), \ldots, (s_k, \boldsymbol{y_k})$ of global t-states such that $(s_0, \boldsymbol{y_0}) = (s, \boldsymbol{y})$, $(s_k, \boldsymbol{y_k}) = (s', \boldsymbol{y'})$, and $(s_{i-1}, \boldsymbol{y}_{i-1}) \xrightarrow{\boldsymbol{f}_i}_t (s_i, \boldsymbol{y}_i)$ for some $\boldsymbol{f}_i \in \mathcal{E}_{\mathsf{pw}}^n$, for all $i \in \{1, \ldots, k\}$. We say that such a t-run has length k.

The following observation can be easily proved.

Lemma 16. *If $\boldsymbol{y}(i) = \omega$ and $(s, \boldsymbol{y}) \xrightarrow{\boldsymbol{f}}_t (s', \boldsymbol{y'})$, then $\boldsymbol{y'}(i) = \omega$, hence* $\mathsf{omega}(\boldsymbol{y}) \subseteq \mathsf{omega}(\boldsymbol{y'})$.

The following lemma will be needed to prove the completeness of our algorithm. The proof is simple and left to the reader.

Lemma 17. *If there is a run from $(s_1, \boldsymbol{x_1})$ to $(s_2, \boldsymbol{x_2})$ of length k, then there is for every $\boldsymbol{x'_1} \geq \boldsymbol{x_1}$ a t-run from $(s_1, \boldsymbol{x'_1})$ to $(s_2, \boldsymbol{y_2})$ of length k and such that $\boldsymbol{y_2} \geq \boldsymbol{x_2}$.*

For proving the soundness of our algorithm, we need to be able to reverse the process of abstracting concrete data values. Given $\boldsymbol{y} \in (\mathbb{R}_\omega)^n$ with $\boldsymbol{y}(i) = \omega$ for some $i \in \{1, \ldots, n\}$, we define the vector $\boldsymbol{y}_{t^{\mathsf{rev}}} \in \mathbb{R}^n_{\geq 0}$ by

$$\boldsymbol{y}_{t^{\mathsf{rev}}}(i) = \begin{cases} \boldsymbol{y}(i) & \text{if } \boldsymbol{y}(i) \in \mathbb{R}_{\geq 0}, \\ t(|\mathsf{real}(\boldsymbol{y})| + 1) & \text{if } \boldsymbol{y}(i) = \omega. \end{cases} \tag{3}$$

The following lemma will be crucial for proving the soundness of our algorithm.

Lemma 18. *For every $P \in \mathbb{R}_{\geq 0}$, if there is a t-run from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ of length k, $|\mathsf{real}(\boldsymbol{y})| = |\mathsf{real}(\boldsymbol{y'})|$, and $t(|\mathsf{real}(\boldsymbol{y})| + 1) \geq k\,\mathsf{bmax} + \mathsf{xminmax} + P$, then there exists a run from $(s, \boldsymbol{y}_{t^{\mathsf{rev}}})$ to some global state $(s', \boldsymbol{x'})$ of length at most k, where $\boldsymbol{x'}(i) = \boldsymbol{y'}(i)$ for all $i \in \mathsf{real}(\boldsymbol{y'})$ and $\boldsymbol{x'}(i) \geq \mathsf{xminmax} + P$ for all $i \in \mathsf{omega}(\boldsymbol{y'})$.*

Proof. Note that $|\mathsf{real}(\boldsymbol{y})| = |\mathsf{real}(\boldsymbol{y'})|$ implies that in the t-run from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ no ω-abstraction took place. We prove the lemma by induction on k.

For the induction base, let $k = 0$. Hence $(s', \boldsymbol{y'}) = (s, \boldsymbol{y})$. Then we have $\boldsymbol{y}_{t^{\mathsf{rev}}}(i) = t(|\mathsf{real}(\boldsymbol{y})| + 1) \geq \mathsf{xminmax} + P$ for all $i \in \mathsf{omega}(\boldsymbol{y})$, where the first equality holds by (3), and the inequality holds by assumption.

For the induction step, assume $k > 0$. Hence there exists $(s'', \boldsymbol{y''})$ such that $(s, \boldsymbol{y}) \xrightarrow{\boldsymbol{f}}_t (s'', \boldsymbol{y''})$, and there exists a t-run from $(s'', \boldsymbol{y''})$ to $(s', \boldsymbol{y'})$ of length $k - 1$.

Note that by assumption $|\mathsf{omega}(\boldsymbol{y})| = |\mathsf{omega}(\boldsymbol{y'})|$, and by Lemma 16 the global t-states occurring in the t-run have ω-entries in the same dimensions.

We first argue that we can execute the transition labeled with $\boldsymbol{f}$ on the global state $(s, \boldsymbol{y}_{t^{\mathsf{rev}}})$, *i.e.*, $\boldsymbol{y}_{t^{\mathsf{rev}}}(i)\boldsymbol{f}(i)$ is defined for all $i \in \{1, \dots, n\}$. For $i \in \mathsf{real}(\boldsymbol{y})$ this is clear. So let $i \in \mathsf{omega}(\boldsymbol{y})$. By assumption,

$$\boldsymbol{y}_{t^{\mathsf{rev}}}(i) = t(|\mathsf{real}(\boldsymbol{y})| + 1) \geq k\,\mathsf{bmax} + \mathsf{xminmax} + P. \tag{4}$$

It follows by $\mathsf{xminmax} \geq \mathsf{xmin}_{\boldsymbol{f}(i)}$ that $\boldsymbol{y}_{t^{\mathsf{rev}}}(i)\boldsymbol{f}(i)$ is defined.

Let $(s'', \boldsymbol{x''})$ be the global state that results from applying $\boldsymbol{f}$ to $(s, \boldsymbol{y}_{t^{\mathsf{rev}}})$, *i.e.*,

$$(s, \boldsymbol{y}_{t^{\mathsf{rev}}}) \xrightarrow{\boldsymbol{f}} (s'', \boldsymbol{x''}).$$

By (4) and Lemma 15, we have for all $i \in \mathsf{omega}(\boldsymbol{y''})$, and $\boldsymbol{x''}(i) = \boldsymbol{y''}(i)$ for all $i \in \mathsf{real}(\boldsymbol{y''})$.

Define the mapping $t_1 : \{0, 1, \dots, n\} \to \mathbb{R}_{\geq 0}$ by

$$t_1(j) = \begin{cases} t(j) & \text{if } j \neq |\mathsf{real}(\boldsymbol{y''})| + 1, \\ t(|\mathsf{real}(\boldsymbol{y''})| + 1) - \mathsf{bmax} & \text{otherwise.} \end{cases}$$

Note that

$$\begin{aligned} t_1(|\mathsf{real}(\boldsymbol{y''})| + 1) &= t(|\mathsf{real}(\boldsymbol{y''})| + 1) - \mathsf{bmax} \\ &= t(|\mathsf{real}(\boldsymbol{y})| + 1) - \mathsf{bmax} \\ &\geq (k - 1)\mathsf{bmax} + \mathsf{xminmax} + P, \end{aligned}$$

where the first equation holds by definition of t_1, the second equation holds by $|\mathsf{real}(\boldsymbol{y''})| = |\mathsf{real}(\boldsymbol{y})|$, and the inequality holds by assumption. We can thus apply the induction hypothesis on the t_1-run from $(s'', \boldsymbol{y''})$ to $(s', \boldsymbol{y'})$ of length $k - 1$. Hence there exists a run from $(s'', \boldsymbol{y''}_{t_1^{\mathsf{rev}}})$ to (s', x') of length at most $k - 1$ and such that $\boldsymbol{x'}(i) = \boldsymbol{y'}(i)$ for all $i \in \mathsf{real}(\boldsymbol{y'})$, and $\boldsymbol{x'}(i) \geq \mathsf{xminmax}+P$ for all $i \in \mathsf{omega}(\boldsymbol{y'})$.

Observe that $\boldsymbol{x''} \geq \boldsymbol{y''}_{t_1^{\mathsf{rev}}}$: $\boldsymbol{x''}(i) = \boldsymbol{y''}_{t_1^{\mathsf{rev}}}(i)$ for all $i \in \mathsf{real}(\boldsymbol{y''})$, and $\boldsymbol{x''}(i) \geq t(\mathsf{real}(\boldsymbol{y})) - \mathsf{bmax}$ and $\boldsymbol{y''}_{t_1^{\mathsf{rev}}}(i) = t(\mathsf{real}(\boldsymbol{y})) - \mathsf{bmax}$ for all $i \in \mathsf{omega}(\boldsymbol{y''})$. We can thus conclude that there is a run from $(s'', \boldsymbol{x''})$ to some global state $(s', \boldsymbol{x'_1})$ of length at most $k - 1$ and with $\boldsymbol{x'_1} \geq \boldsymbol{x'}$, which yields the statement of the lemma. □

We are finally ready to state the main result of this subsection.

Theorem 5. *State reachability and coverability are EXPSPACE-complete for $\mathcal{E}^n_{\mathsf{pw}}$-automata for $n \geq 3$.*

Proof. The lower bound follows from EXPSPACE-hardness for VASS [44]. For the upper bound, let (S, T) be an $\mathcal{E}^n_{\mathsf{pw}}$-automaton, let $(s, \boldsymbol{x})$ and $(s', \boldsymbol{x'})$, respectively, be the initial global state and the global state to be covered, respectively. We use $\mathsf{cmax} = \max\{\mathsf{bmax}, \max_i \boldsymbol{x'}(i)\}$ to denote the maximum of bmax and the maximum entry in $\boldsymbol{x'}$; note that $\mathsf{cmax} \geq 1$. Define $\mathsf{thd} : \{0, 1, \dots, n\} \to \mathbb{Q}_{\geq 0}$ by

$$\mathsf{thd}(0) = 0, \qquad \mathsf{thd}(i) = \mathsf{bmax} \cdot \mathsf{len}(i-1) + \mathsf{xminmax} + \mathsf{cmax}$$

for every $i \in \{1, \dots, n\}$, where $\mathsf{len} : \{0, \dots, n\} \to \mathbb{Q}_{\geq 0}$ is defined inductively by

$$\mathsf{len}(0) = |S|, \qquad \mathsf{len}(i) = (\mathsf{thd}(i))^i \, |S| + \mathsf{len}(i-1)$$

for every $i \in \{1, \dots, n\}$.

The correctness of our algorithm is based on the following two claims.

Soundness Claim. If there exists a thd-run from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ of length k and such that $\boldsymbol{y'} \geq \boldsymbol{x'}$, then there exists a run from $(s, \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}})$ to $(s', \boldsymbol{x''})$ of length at most k and such that $\boldsymbol{x''} \geq \boldsymbol{x'}$.

Completeness Claim. If there exists a run from $(s_0, \boldsymbol{x_0})$ to $(s', \boldsymbol{x''})$ for some $\boldsymbol{x''} \geq \boldsymbol{x'}$, then there is also a thd-run from $(s_0, \boldsymbol{x_0})$ to $(s', \boldsymbol{y'})$ of length $k \leq \mathsf{len}(n)$ and such that $\boldsymbol{y'} \geq \boldsymbol{x'}$.

By these two claims, there is a run from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{x''})$ for some $\boldsymbol{x''} \geq \boldsymbol{x'}$ if, and only if, there is a thd-run from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{y'})$ such that $\boldsymbol{y'} \geq \boldsymbol{x'}$ and of length bounded by $\mathsf{len}(n)$. The existence of such a thd-run can thus be verified by a non-deterministic Turing machine that keeps in memory one global thd-state as well as one counter counting up to $\mathsf{len}(n)$. Every entry in $(\mathbb{R}_\omega)^n$ occurring in the analysis is either ω or less than $\mathsf{thd}(n) = \mathsf{bmax} \cdot \mathsf{len}(n) + \mathsf{xminmax} + \mathsf{cmax}$. By the Length Claim below, the memory space needed by the algorithm is $\mathcal{O}((n+1)!(\log(\mathsf{bmax}) + \log(\mathsf{cmax}) + \log(\mathsf{xminmax})) + \log(|S|)) = \mathcal{O}(2^{cn \log n}(\log(\mathsf{cmax}) + \log(\mathsf{xminmax}) + \log(|S|)))$, which is in NEXPSPACE.

Length Claim. For all $i \in \mathbb{N}$, $\mathsf{len}(i) \leq (6\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{(i+1)!} \, |S|$.

In the remainder of this subsection, we prove the three claims stated above.

Proof of the Soundness Claim. The proof is by induction on $|\mathsf{real}(\boldsymbol{y})|$.

For the induction base, assume $|\mathsf{real}(\boldsymbol{y})| = 0$, *i.e.,* we have $\boldsymbol{y}(i) = \omega$ for all $i \in \{1, \dots, n\}$. By definition, for every $i \in \{1, \dots, n\}$ we have

$$\begin{aligned} \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}}(i) = \mathsf{thd}(|\mathsf{real}(\boldsymbol{y})| + 1) &= \mathsf{thd}(1) \\ &= \mathsf{bmax} \cdot \mathsf{len}(0) + \mathsf{xminmax} + \mathsf{cmax} \\ &= \mathsf{bmax}\,|S| + \mathsf{xminmax} + \mathsf{cmax}. \end{aligned} \tag{5}$$

For defining the sequence of transitions that leads $(s, \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}})$ to some $(s', \boldsymbol{x''})$ in at most k transition steps and such that $\boldsymbol{x''} \geq \boldsymbol{x'}$, we distinguish two cases. First, assume that the length k of the thd-run from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ satisfies $k > |S|$. There must exist a (syntactical) cycle-free path from s to s' in (S, T) of length bounded by $|S|$. By (5) and Lemma 15, there is a run corresponding to this path from $(s, \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}})$ to $(s', \boldsymbol{x''})$, where $\boldsymbol{x''}$ satisfies $\boldsymbol{x''}(i) \geq \mathsf{xminmax} + \mathsf{cmax}$ for all $i \in \{1, \dots, n\}$. By definition of cmax, we can conclude $\boldsymbol{x''} \geq \boldsymbol{x'}$.

Second, assume that the length k of the thd-run from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ satisfies $k \leq |S|$. By (5) and Lemma 15, we can argue that starting from $(s, \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}})$ we can use the same sequence of transitions as in the thd-run, yielding a run of length k, and reaching a global state $(s', \boldsymbol{x''})$ for some $\boldsymbol{x''} \geq \boldsymbol{x'}$. This finishes the induction base.

For the induction step assume $|\mathsf{real}(\boldsymbol{y})| = i + 1$. We consider two cases.

(i) First assume $|\mathsf{real}(\boldsymbol{y})| = |\mathsf{real}(\boldsymbol{y'})|$. By definition of the thd-semantics, every global thd-state $(s_1, \boldsymbol{y_1})$ that occurs in the thd-run ρ from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ satisfies

- $|\mathsf{real}(\boldsymbol{y_1})| = i + 1$,
- $\boldsymbol{y_1}(j) < \mathsf{thd}(|\mathsf{real}(\boldsymbol{y_1})|) = \mathsf{thd}(i + 1)$ for all $j \in \mathsf{real}(\boldsymbol{y_1})$.

Let ρ' be the run from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$ that is obtained from ρ by removing all cycles between identical global thd-states. The length k' of ρ' is bounded by

$$k' \leq (\mathsf{thd}(i + 1))^{i+1} \, |S| \, .$$

Next we prove that there is a corresponding run from $(s, \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}})$. For this, note that

$$\begin{aligned}
\mathsf{thd}(|\mathsf{real}(\boldsymbol{y})| + 1) &= \mathsf{thd}((i + 1) + 1) \\
&= \mathsf{bmax} \cdot \mathsf{len}(i + 1) + \mathsf{xminmax} + \mathsf{cmax} \\
&= \mathsf{bmax}\,((\mathsf{thd}(i + 1))^{i+1}\,|S| + \mathsf{len}(i)) + \mathsf{xminmax} + \mathsf{cmax} \\
&\geq \mathsf{bmax}\,((\mathsf{thd}(i + 1))^{i+1}\,|S|) + \mathsf{xminmax} + \mathsf{cmax} \\
&\geq \mathsf{bmax} \cdot k' + \mathsf{xminmax} + \mathsf{cmax}.
\end{aligned}$$

By Lemma 18 (for $P = \mathsf{cmax}$), there exists a run from $(s, \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}})$ to $(s', \boldsymbol{x''})$ of length at most k' and such that $\boldsymbol{x''}(j) = \boldsymbol{y'}(j)$ for all $j \in \mathsf{real}(\boldsymbol{y'})$, and $\boldsymbol{x''}(j) \geq \mathsf{xminmax} + \mathsf{cmax}$ for all $j \in \mathsf{omega}(\boldsymbol{y'})$. Then $\boldsymbol{x''} \geq \boldsymbol{x'}$ follows by $\mathsf{cmax} \geq \boldsymbol{x'}(j)$ for all $j \in \{1, \dots, n\}$.

(ii) Now assume $|\mathsf{real}(\boldsymbol{y})| > |\mathsf{real}(\boldsymbol{y'})|$. In the thd-run ρ from $(s, \boldsymbol{y})$ to $(s', \boldsymbol{y'})$, let $(s_1, \boldsymbol{y_1})$ be the last global thd-state with $|\mathsf{real}(\boldsymbol{y_1})| = i + 1$. Partition ρ into three parts: A thd-run ρ_1 of length k_1 from $(s, \boldsymbol{y})$ to $(s_1, \boldsymbol{y_1})$, a thd-transition $(s_1, \boldsymbol{y_1}) \xrightarrow{\boldsymbol{f}}_{\mathsf{thd}} (s_2, \boldsymbol{y_2})$ (where $\mathsf{real}(\boldsymbol{y_2}) \leq i$), and a thd-run ρ_2 of length k_2 from $(s_2, \boldsymbol{y_2})$ to $(s', \boldsymbol{y'})$. Hence $k = k_1 + k_2 + 1$. Note that ρ_1 may be empty, in case $(s_1, \boldsymbol{y_1}) = (s, \boldsymbol{y})$; in that case, $k_1 = 0$ and $k = k_2 + 1$.

As in case (i), we can show that there is a run ρ'_1 from $(s, \boldsymbol{y})$ to $(s_1, \boldsymbol{y_1})$ of length $k'_1 \leq (\mathsf{thd}(i + 1))^{i+1}\,|S|$. Using similar arguments as above, we obtain

$$\mathsf{thd}(|\mathsf{real}(\boldsymbol{y})| + 1) \geq \mathsf{bmax} \cdot k'_1 + \mathsf{bmax} \cdot \mathsf{len}(i) + \mathsf{xminmax} + \mathsf{cmax}.$$

By Lemma 18 (with $P = \mathsf{bmax} \cdot \mathsf{len}(i) + \mathsf{cmax}$) there exists a run from $(s, \boldsymbol{y}_{\mathsf{thd}^{\mathsf{rev}}})$ to $(s_1, \boldsymbol{x_1})$ of length at most k_1' and such that $\boldsymbol{x_1}(j) = \boldsymbol{y_1}(j)$ for all $j \in \mathsf{real}(\boldsymbol{y_1})$, and $\boldsymbol{x_1}(j) \geq \mathsf{bmax} \cdot \mathsf{len}(i) + \mathsf{xminmax} + \mathsf{cmax}$ for all $j \in \mathsf{omega}(\boldsymbol{y_1})$. Note that $\boldsymbol{x_1} \geq \boldsymbol{y_1}_{\mathsf{thd}^{\mathsf{rev}}}$. Hence we also have $(s_1, \boldsymbol{x_1}) \xrightarrow{\boldsymbol{f}} (s_2, \boldsymbol{x_2})$ for some $\boldsymbol{x_2} \geq \boldsymbol{y_2}_{\mathsf{thd}^{\mathsf{rev}}}$. By induction hypothesis, there exists a run from $(s_2, \boldsymbol{y_2}_{\mathsf{thd}^{\mathsf{rev}}})$ to $(s', \boldsymbol{x''})$ for some $\boldsymbol{x''} \geq \boldsymbol{x'}$ and of length at most k_2. But by $\boldsymbol{x_2} \geq \boldsymbol{y_2}_{\mathsf{thd}^{\mathsf{rev}}}$ there also exists such a run from $(s, \boldsymbol{x_2})$. This completes the induction step and thus the proof of the soundness claim.

Proof of the Completeness Claim. Assume that there exists a run from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{x''})$ for some $\boldsymbol{x''} \geq \boldsymbol{x'}$. By Lemma 17 there exists a thd-run ρ from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{y})$ for some $\boldsymbol{y} \geq \boldsymbol{x''}$. Let ρ' be the thd-run from $(s, \boldsymbol{x})$ to $(s', \boldsymbol{y})$ that is obtained from ρ by removing all cycles between identical global thd-states. We prove that $|\rho'| \leq \mathsf{len}(n)$. For all $i \in \{|\mathsf{real}(\boldsymbol{y})|, \ldots, n\}$, let $(s_i, \boldsymbol{y_i})$ be the first global thd-state occurring in ρ' such that $|\mathsf{real}(\boldsymbol{y_i})| \leq i$, and let ρ_i' be the suffix of ρ' that starts in $(s_i, \boldsymbol{y_i})$. We prove by induction on i that $|\rho_i'| \leq \mathsf{len}(i)$.

For the base case, let $i = |\mathsf{real}(\boldsymbol{y})|$. By definition of the thd-semantics, every global thd-state $(s', \boldsymbol{y'})$ occurring in ρ_i' satisfies $\boldsymbol{y'}(j) < \mathsf{thd}(i)$ for every $j \in \mathsf{real}(\boldsymbol{y'})$. Since there are exactly i entries that take values in $\{0, \ldots, \mathsf{thd}(i) - 1\}$, the length $|\rho_i'|$ of ρ_i' is bounded by $\mathsf{thd}(i)^i \cdot |S|$. This and the definition of len yields $|\rho_i'| \leq \mathsf{len}(i)$.

For the induction step, let ρ_{i+1} be the prefix of ρ_{i+1}' that starts in $(s_{i+1}, \boldsymbol{y_{i+1}})$ and ends in $(s_i, \boldsymbol{y_i})$. Every global thd-state $(s', \boldsymbol{y'})$ that occurs in ρ_{i+1} (except for $(s_i, \boldsymbol{y_i})$) satisfies $\boldsymbol{y'}(j) < \mathsf{thd}(i+1)$ for all $j \in \mathsf{real}(\boldsymbol{y'}) = \mathsf{real}(\boldsymbol{y_{i+1}})$. Hence $|\rho_{i+1}| \leq (\mathsf{thd}(i+1))^{i+1} |S|$. By induction hypothesis, $|\rho_i'| \leq \mathsf{len}(i)$. Altogether, we obtain

$$|\rho_{i+1}'| = |\rho_{i+1}| + |\rho_i'| \leq (\mathsf{thd}(i+1))^{i+1} |S| + \mathsf{len}(i) \leq \mathsf{len}(i+1).$$

This completes the induction step and the proof of the completeness claim.

Proof of the Length Claim. The claim is proved by induction on i. The base case, $i = 0$, trivially holds. So let us assume that the claim holds for i. Then

$$\begin{aligned}
\mathsf{len}(i+1) &= (\mathsf{bmax} \cdot \mathsf{len}(i) + \mathsf{xminmax} + \mathsf{cmax})^{i+1} |S| + \mathsf{len}(i) \\
&\leq (2\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax} \cdot \max(\mathsf{len}(i), 1))^{i+1} |S| + \mathsf{len}(i) \\
&\leq 3(2\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax} \cdot \max(\mathsf{len}(i), 1))^{i+1} |S| \\
&\leq 3^{i+1}(2\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{i+1} (\max(\mathsf{len}(i), 1))^{i+1} |S| \\
&\leq (6\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{i+1} ((6\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{(i+1)!})^{i+1} |S| \\
&\leq (6\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{(i+1)!} (6\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{(i+1)(i+1)!} |S| \\
&= (6\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{(i+2)(i+1)!} |S| \\
&= (6\,\mathsf{bmax} \cdot \mathsf{cmax} \cdot \mathsf{xminmax})^{(i+2)!} |S|.
\end{aligned}$$

With the proves of these three lemmas, the proof of Theorem 5 is complete. □

9.2 Flat $\mathcal{E}^n_{\mathsf{pw}}$-automata

Next we show that if the requirement $(*)$ on energy functions, that $yf \geq xf + y - x$ for each $x \leq y$, is lifted, then reachability becomes undecidable from dimension 4. We call such functions *flat* energy functions; remark that we still require them to be strictly increasing, but the derivative, where it exists, may be less than 1. The class of all flat energy functions is denoted $\bar{\mathcal{E}}$ and its restrictions by $\bar{\mathcal{E}}_{\mathsf{pw}}$ and $\bar{\mathcal{E}}_{\mathsf{pwi}}$.

Theorem 6. *State reachability, coverability, and Büchi acceptance are undecidable for $\bar{\mathcal{E}}^4_{\mathsf{pw}}$-automata.*

Proof. The proof is a reduction from the halting problem (respectively, the recurrent state problem) for two-counter machines. A *two-counter machine* $\mathcal{M}$ is a finite sequence $(\mathcal{I}_j)_{j=1}^n$ of instructions operating on two counters denoted by C_1 and C_2, where $\mathcal{I}_j$ is one of the following instructions (with $i \in \{1, 2\}$ and $j, k, m \in \{1, ..., n\}$):

increment	$\mathcal{I}_j : C_i := C_i + 1;$ `go to` $\mathcal{I}_k$
zero test/dec	$\mathcal{I}_j :$ `if` $C_i = 0$ `then goto` $\mathcal{I}_k$ `else` $C_i := C_i - 1;$ `goto` $\mathcal{I}_m$
halt	$\mathcal{I}_j :$ `halt`

A *configuration* of such a two-counter machine $\mathcal{M}$ is a triple $\gamma = (J, c, d) \in \{\mathcal{I}_1, \dots, \mathcal{I}_n\} \times \mathbb{N} \times \mathbb{N}$, where J indicates the current instruction, and c and d are the current values of the counters C_1 and C_2, respectively. A *computation* of $\mathcal{M}$ is a finite or infinite sequence $(\gamma_i)_{i \geq 0}$ of configurations, such that $\gamma_0 = (\mathcal{I}_1, 0, 0)$ and γ_{i+1} is the result of executing the instruction $\mathcal{I}_i$ on γ_i for each $i \geq 0$. Without loss of generality, we assume that $\mathcal{I}_n$ is the only instruction of the form `halt`. The *halting problem* for two-counter machines asks, given a two-counter machine $\mathcal{M}$, whether the (unique) computation of $\mathcal{M}$ reaches a configuration with instruction $\mathcal{I}_n$, *i.e.,* the halting instruction. This problem is Σ^0_1-complete [45]. The *recurrent state problem* for two-counter machines asks, given a two-counter machine $\mathcal{M}$, whether the (unique) computation of $\mathcal{M}$ visits instruction $\mathcal{I}_1$ infinitely often. This problem is Σ^1_1-complete [3].

Given a two-counter machine $\mathcal{M}$, we construct an $\bar{\mathcal{E}}^4_{\mathsf{pw}}$-automaton $\mathcal{A}_{\mathcal{M}}$ with state set $S \subseteq \{\mathcal{I}_1, \dots, \mathcal{I}_n\}$ such that $\mathcal{M}$ halts (visits instruction $\mathcal{I}_1$ infinitely often, respectively) if, and only if, $\mathcal{A}_{\mathcal{M}}$ reaches state $\mathcal{I}_n$ (visits $\mathcal{I}_1$ infinitely often, respectively).

We use two variables x and y to store the values of the counters C_1 and C_2 by requiring that $x = 1/(2^{c_1} 3^{c_2})$ and $y = 2^{c_1} 3^{c_2}$. Two other variables z and z' are used for storing temporary information needed for encoding zero tests and decrementation operations of the 2-counter machine. The initial value of all energy variables is 1.

For encoding the instructions of the 2-counter machine, we first define the following flat energy functions: dec_2 multiplies the current values of x and z' with 3, and divides the current values of y and z by 3; inc_2 turns these operations back, *i.e.,* it divides the current values of x and z' by 3 and multiplies the values of y and z with 3. Functions dec_1 and inc_1 are defined analogously by replacing 3 by 2. Finally, function dec'_1 behaves like dec_1 but does not change the values of z

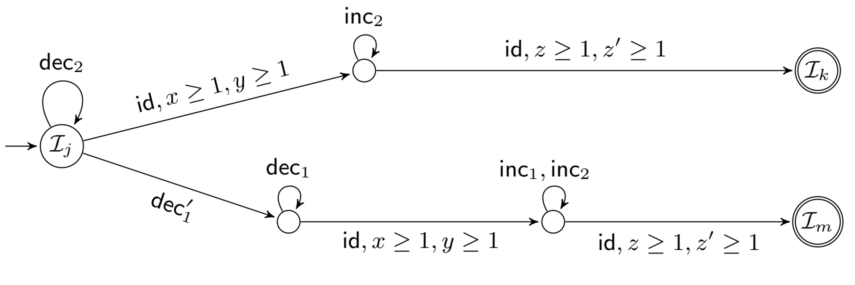


Figure 11: Module for encoding zero test/decrement instructions of a 2-counter machine

and z'; likewise for inc_1'. An increment instruction for counter C_1 is then encoded by a simple transition from state $\mathcal{I}_j$ to state $\mathcal{I}_k$ labeled by inc_1'; analogously for an increment of counter C_2. The encoding zero test/decrement instructions for counter C_1 can be done as shown in Fig. 11. Here, the additional inequalities for the definition intervals of the energy functions are crucial for the correctness of the construction. □

9.3 Reachability games on $\mathcal{E}^n_{\mathsf{pw}}$-automata

Next we extend our energy automata formalism to (turn based) *reachability games*. Let (S,T) be an n-dimensional energy automaton such that $S = S_A \cup S_B$ forms a partition of S and $T \subseteq (S_A \times \mathcal{E}^n_{\mathsf{pw}} \times S_B) \cup (S_B \times \mathcal{E}^n_{\mathsf{pw}} \times S_A)$. Then (S, S_A, S_B, T) induces an n-dimensional *energy game* G. The intuition of the reachability game is that the two players A and B take turns to move along the game graph (S,T), updating energy values at each turn. The goal of player A is to reach a state in F, the goal of player B is to prevent this from happening.

The reachability game is a game on a well-structured transition system as in [1]. In general, the reachability game on well-structured transition systems is undecidable; in particular, the game on 2-dimensional vector addition systems with states is undecidable [1]. It is hence clear that it is undecidable whether player A wins the reachability game in 2-dimensional $\mathcal{E}_{\mathsf{int}}$-automata.

Theorem 7. *Whether player A wins the reachability game in $\mathcal{E}^2_{\mathsf{int}}$-automata is undecidable.*

As a corollary, we can show that for *flat* energy functions, already one-dimensional reachability games are undecidable.

Theorem 8. *It is undecidable for $\bar{\mathcal{E}}_{\mathsf{pw}}$-automata whether player A wins the reachability game.*

Proof. We show a reduction from reachability games on 2-dimensional $\mathcal{E}_{\mathsf{int}}$-automata to reachability games on 1-dimensional $\bar{\mathcal{E}}_{\mathsf{pw}}$-automata. Let (S,T) be a 2-dimensional

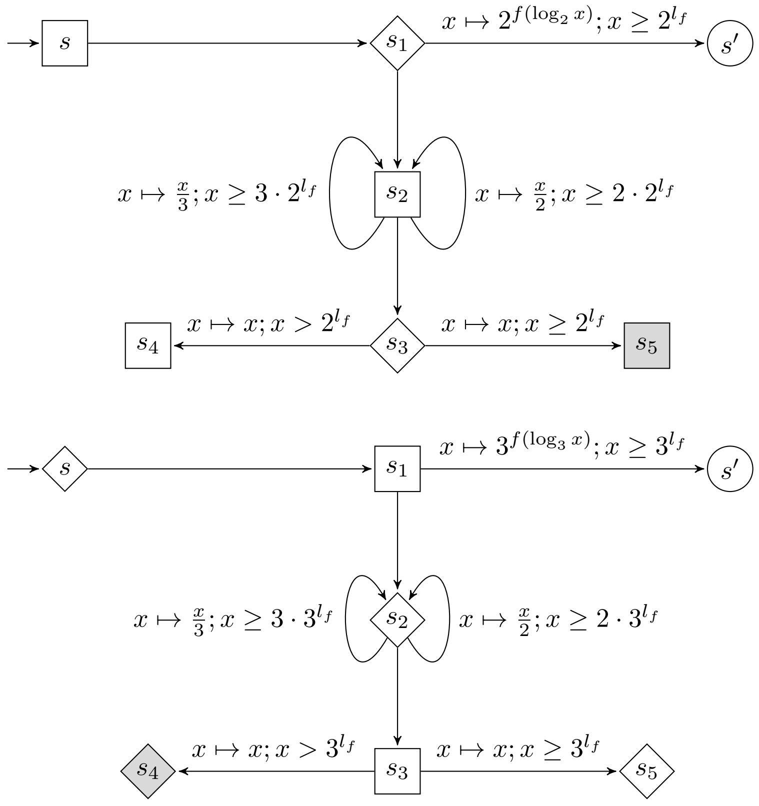

s
s_1
$x \mapsto 2^{f(\log_2 x)}; x \geq 2^{l_f}$
s'
$x \mapsto \frac{x}{3}; x \geq 3 \cdot 2^{l_f}$
s_2
$x \mapsto \frac{x}{2}; x \geq 2 \cdot 2^{l_f}$
s_4
$x \mapsto x; x > 2^{l_f}$
s_3
$x \mapsto x; x \geq 2^{l_f}$
s_5
s
s_1
$x \mapsto 3^{f(\log_3 x)}; x \geq 3^{l_f}$
s'
$x \mapsto \frac{x}{3}; x \geq 3 \cdot 3^{l_f}$
s_2
$x \mapsto \frac{x}{2}; x \geq 2 \cdot 3^{l_f}$
s_4
$x \mapsto x; x > 3^{l_f}$
s_3
$x \mapsto x; x \geq 3^{l_f}$
s_5

Figure 12: Conversion of two types of edges in (S, T). Top: an edge $(s, (f, \mathsf{id}), s')$ from a player-A state s; bottom: an edge $(s, (\mathsf{id}, f), s')$ from a player-B state s. Player-A states are depicted using squares, player-B states are diamonds. Accepting states have a gray background color. The ownership of state s' is unchanged.

$\mathcal{E}_{\mathsf{int}}$-automaton. By inserting extra states (and transitions) if necessary, we can assume that for any $(s, (f, g), s') \in T$, either $f = \mathsf{id}$ with $l_f = 0$, or $g = \mathsf{id}$ with $l_g = 0$. We build an energy automaton (S', T').

Let $(s, (f, \mathsf{id}), s') \in T$ be a player-A transition (*i.e.*, $s \in S_A$) in (S, T) (with lower bound l_f as usual), then we model this in (S', T') using s, s' and the following new states and transitions; see Figure 12 for a pictorial description.

- player-A states: s_2, s_4, s_5 (accepting); player-B states: s_1, s_3
- transitions:
  - $(s, [x \mapsto x; x \geq 0], s_1)$; $(s_1, [x \mapsto 2^{f(\log_2 x)}; x \geq 2^{l_f}], s')$
  - $(s_1, [x \mapsto x; x \geq 0], s_2)$

– $(s_2, [x \mapsto \frac{x}{3}; x \geq 3 \cdot 2^{l_f}], s_2)$; $(s_2, [x \mapsto \frac{x}{2}; x \geq 2 \cdot 2^{l_f}], s_2)$
– $(s_2, [x \mapsto x; x \geq 0], s_3)$
– $(s_3, [x \mapsto x; x > 2^{l_f}], s_4)$; $(s_3, [x \mapsto x; x \geq 2^{l_f}], s_5)$

Note that s_4 is a deadlock state, hence player A loses the reachability game if s_4 is reached. Similarly, she wins if s_5 is reached.

The intuition is that the new energy variable x encodes the two old ones as $x = 2^{x_1} 3^{x_2}$. If player A wants to bring (S', T') from s to s', and commits to this by taking the transition $s \to s_1$, she may be interrupted by player B taking the $s_1 \to s_2$ transition. Here player A has to prove that x_1 was really $\geq l_f$, by using the loops at s_2 to bring x to the precise value 2^{l_f}. If she manages this, then player B has only the $s_3 \to s_5$ transition available in s_3, hence player A wins. Otherwise, player B wins.

The conversions of other types of transitions are similar. One can easily see that player A can reach a state in F in the original energy automaton (S, T) if, and only if, she can reach a state in F, or one of the new accepting states, in the new automaton (S', T').

We miss to argue that all energy functions in (S', T') are piecewise affine. Looking at the defined modules, we see that this is the case except perhaps for the functions defined as $g_2(x) = 2^{f(\log_2 x)}$ and $g_3(x) = 3^{f(\log_3 x)}$. However, f is an integer update function, so that $f(x) = x + k$ for some $k \in \mathbb{Z}$; hence $g_2(x) = 2^k x$ and $g_3(x) = 3^k x$, which are indeed piecewise affine. □

10 Conclusion

We have in this paper introduced a functional framework for modeling and analyzing energy problems, and we have seen that our framework encompasses most existing formal approaches to energy problems. In the first paper of this series [28], we have developed a theory of *-continuous Kleene ω-algebras in order to analyze energy problems algebraically.

We have seen here that the algebraic setting of *-continuous Kleene ω-algebras applies to energy functions and that it allows to solve reachability and Büchi acceptance problems in energy automata in a generic way. For the important class of piecewise affine energy functions, we have shown that reachability and Büchi acceptance are decidable in EXPTIME.

In the last part of this paper, we have seen that one quickly comes into trouble with undecidability if the class of energy functions is extended or if two-player games are considered. This may be remedied by considering *approximate* solutions instead, using notions of distances for energy automata akin to the ones in [35] to provide quantitative measures for similar energy behavior; this is future work.

In the two papers of this series, we have seen that *-continuous Kleene ω-algebras provide a natural generalization of continuous Kleene ω-algebras, much in the same way in which *-continuous Kleene algebras are a natural generalization of continuous Kleene algebras. We have left open a few algebraic problems, in particular a characterization of the free (non-finitary) *-continuous Kleene ω-algebras. We have seen that *-continuous Kleene ω-algebras find a natural application in energy functions and energy problems, but we are confident that they will find numerous other applications. In honor of the late Zoltán Ésik, we propose to rename *-continuous Kleene ω-algebras to "*Ésik algebras*".

References

[1] Parosh Aziz Abdulla, Ahmed Bouajjani, and Julien d'Orso. Monotonic and downward closed games. *J. Log. Comput.*, 18(1):153–169, 2008.

[2] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.

[3] Rajeev Alur and Thomas A. Henzinger. A really temporal logic. *J. ACM*, 41(1):181–204, 1994.

[4] Rajeev Alur, Salvatore La Torre, and George J. Pappas. Optimal paths in weighted timed automata. In Di Benedetto and Sangiovanni-Vincentelli [23], pages 49–62.

[5] Gerd Behrmann, Ansgar Fehnker, Thomas Hune, Kim G. Larsen, Paul Pettersson, Judi Romijn, and Frits W. Vaandrager. Minimum-cost reachability for priced timed automata. In Di Benedetto and Sangiovanni-Vincentelli [23], pages 147–161.

[6] Jean Berstel and Christophe Reutenauer. *Noncommutative Rational Series With Applications.* Cambridge Univ. Press, 2010.

[7] Michael Blondin, Alain Finkel, Stefan Göller, Christoph Haase, and Pierre McKenzie. Reachability in two-dimensional vector addition systems with states is PSPACE-complete. In *LICS*, pages 32–43. IEEE, 2015.

[8] Stephen L. Bloom and Zoltán Ésik. *Iteration Theories: The Equational Logic of Iterative Processes.* EATCS monographs on theoretical computer science. Springer-Verlag, 1993.

[9] Rémi Bonnet, Alain Finkel, and M. Praveen. Extending the Rackoff technique to affine nets. In Deepak D'Souza, Telikepalli Kavitha, and Jaikumar Radhakrishnan, editors, *FSTTCS*, volume 18 of *Leibniz Int. Proc. Inf.*, pages 301–312. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[10] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, and Nicolas Markey. Timed automata with observers under energy constraints. In Karl Henrik Johansson and Wang Yi, editors, *HSCC*, pages 61–70. ACM, 2010.

[11] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, Nicolas Markey, and Jiří Srba. Infinite runs in weighted timed automata with energy constraints. In Franck Cassez and Claude Jard, editors, *FORMATS*, volume 5215 of *Lect. Notes Comput. Sci.*, pages 33–47. Springer-Verlag, 2008.

[12] Patricia Bouyer, Kim G. Larsen, and Nicolas Markey. Lower-bound-constrained runs in weighted timed automata. *Perform. Eval.*, 73:91–109, 2014.

[13] Patricia Bouyer, Kim G. Larsen, Nicolas Markey, and Jacob Illum Rasmussen. Almost optimal strategies in one clock priced timed games. In S. Arun-Kumar and Naveen Garg, editors, *FSTTCS*, volume 4337 of *Lect. Notes Comput. Sci.*, pages 345–356. Springer-Verlag, 2006.

[14] Tomáš Brázdil, Petr Jančar, and Antonín Kučera. Reachability games on extended vector addition systems with states. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *ICALP*, volume 6199 of *Lect. Notes Comput. Sci.*, pages 478–489. Springer-Verlag, 2010.

[15] Romain Brenguier, Franck Cassez, and Jean-François Raskin. Energy and mean-payoff timed games. In Martin Fränzle and John Lygeros, editors, *HSCC*, pages 283–292. ACM, 2014.

[16] David Cachera, Uli Fahrenberg, and Axel Legay. An omega-algebra for real-time energy problems. In Prahladh Harsha and G. Ramalingam, editors, *FSTTCS*, volume 45 of *Leibniz Int. Proc. Inf.*, pages 394–407. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[17] Jakub Chaloupka. Z-reachability problem for games on 2-dimensional vector addition systems with states is in P. In Antonín Kučera and Igor Potapov, editors, *RP*, volume 6227 of *Lect. Notes Comput. Sci.*, pages 104–119. Springer-Verlag, 2010.

[18] Tat-hung Chan. The boundedness problem for three-dimensional vector addition systems with states. *Inf. Proc. Letters*, 26(6):287–289, 1988.

[19] Krishnendu Chatterjee and Laurent Doyen. Energy parity games. *Theor. Comput. Sci.*, 458:49–60, 2012.

[20] Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean-François Raskin. Generalized mean-payoff and energy games. In Kamal Lodaya and Meena Mahajan, editors, *FSTTCS*, volume 8 of *Leibniz Int. Proc. Inf.*, pages 505–516, 2010.

[21] Adrian Horia Dediu, Shunsuke Inenaga, and Carlos Martín-Vide, editors. *Language and Automata Theory and Applications - 5th International Conference, LATA 2011, Tarragona, Spain, May 26-31, 2011. Proceedings*, volume 6638 of *Lect. Notes Comput. Sci.* Springer-Verlag, 2011.

[22] Aldric Degorre, Laurent Doyen, Raffaella Gentilini, Jean-François Raskin, and Szymon Toruńczyk. Energy and mean-payoff games with imperfect information. In Anuj Dawar and Helmut Veith, editors, *CSL*, volume 6247 of *Lect. Notes Comput. Sci.*, pages 260–274. Springer-Verlag, 2010.

[23] Maria Domenica Di Benedetto and Alberto L. Sangiovanni-Vincentelli, editors. *Hybrid Systems: Computation and Control, 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001, Proceedings*, volume 2034 of *Lect. Notes Comput. Sci.* Springer-Verlag, 2001.

[24] Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of Weighted Automata.* EATCS Monographs in Theoretical Computer Science. Springer-Verlag, 2009.

[25] Zoltán Ésik, Uli Fahrenberg, and Axel Legay. *-continuous Kleene ω-algebras. In Igor Potapov, editor, *DLT*, volume 9168 of *Lect. Notes Comput. Sci.*, pages 240–251. Springer-Verlag, 2015.

[26] Zoltán Ésik, Uli Fahrenberg, and Axel Legay. *-continuous Kleene ω-algebras for energy problems. In Ralph Matthes and Matteo Mio, editors, *FICS*, volume 191 of *Electr. Proc. Theor. Comput. Sci.*, pages 48–59, 2015.

[27] Zoltán Ésik, Uli Fahrenberg, Axel Legay, and Karin Quaas. Kleene algebras and semimodules for energy problems. In Dang Van Hung and Mizuhito Ogawa, editors, *ATVA*, volume 8172 of *Lect. Notes Comput. Sci.*, pages 102–117. Springer-Verlag, 2013.

[28] Zoltán Ésik, Uli Fahrenberg, Axel Legay, and Karin Quaas. An algebraic approach to energy problems I: *-continuous Kleene ω-algebras. *Acta Cyb.*, 2017. In this issue.

[29] Zoltán Ésik and Werner Kuich. A semiring-semimodule generalization of ω-regular languages, Parts 1 and 2. *J. Aut. Lang. Comb.*, 10:203–264, 2005.

[30] Zoltán Ésik and Werner Kuich. *Modern Automata Theory.* 2007. `http://dmg.tuwien.ac.at/kuich/mat.pdf`.

[31] Zoltán Ésik and Werner Kuich. On iteration semiring-semimodule pairs. *Semigroup Forum*, 75:129–159, 2007.

[32] Zoltán Ésik and Werner Kuich. Finite automata. In *Handbook of Weighted Automata* [24], pages 69–104.

[33] Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jiří Srba. Energy games in multiweighted automata. In Antonio Cerone and Pekka Pihlajasaari, editors, *ICTAC*, volume 6916 of *Lect. Notes Comput. Sci.*, pages 95–115. Springer-Verlag, 2011.

[34] Uli Fahrenberg, Kim G. Larsen, and Axel Legay. Model-based verification, optimization, synthesis and performance evaluation of real-time systems. In Zhiming Liu, Jim Woodcock, and Huibiao Zhu, editors, *ICTAC Training School on Software Engineering*, volume 8050 of *Lect. Notes Comput. Sci.*, pages 67–108. Springer-Verlag, 2013.

[35] Uli Fahrenberg and Axel Legay. The quantitative linear-time–branching-time spectrum. *Theor. Comput. Sci.*, 538:54–69, 2014.

[36] Uli Fahrenberg, Axel Legay, and Karin Quaas. Büchi conditions for generalized energy automata. In Manfred Droste and Heiko Vogler, editors, *WATA*, page 47, 2012.

[37] Jonathan S. Golan. *Semirings and their Applications.* Springer-Verlag, 1999.

[38] Stefan Göller, Christoph Haase, Ranko Lazić, and Patrick Totzke. A polynomial-time algorithm for reachability in branching VASS in dimension one. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *ICALP*, volume 55 of *Leibniz Int. Proc. Inf.*, pages 105:1–105:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[39] Line Juhl, Kim G. Larsen, and Jean-François Raskin. Optimal bounds for multiweighted and parametrised energy games. In Zhiming Liu, Jim Woodcock, and Huibiao Zhu, editors, *Theories of Programming and Formal Methods*, volume 8051 of *Lect. Notes Comput. Sci.*, pages 244–255. Springer-Verlag, 2013.

[40] Dénes König. Über eine Schlussweise aus dem Endlichen ins Unendliche. *Acta Sci. Math. (Szeged)*, 3(2-3):121–130, 1927.

[41] Dexter Kozen. On Kleene algebras and closed semirings. In Branislav Rovan, editor, *MFCS*, volume 452 of *Lect. Notes Comput. Sci.*, pages 26–47. Springer-Verlag, 1990.

[42] Jérôme Leroux. The general vector addition system reachability problem by Presburger inductive invariants. *Logical Meth. Comput. Sci.*, 6(3), 2010.

[43] Jérôme Leroux. Vector addition system reachability problem: A short self-contained proof. In Dediu et al. [21], pages 41–64.

[44] Richard J. Lipton. The reachability problem requires exponential space. Technical report, Department of Computer Science, Yale University, 1976.

[45] Marvin L. Minsky. Recursive unsolvability of Post's problem of "Tag" and other topics in theory of Turing machines. *Annals Math.*, 74(3):437–455, 1961.

[46] Karin Quaas. On the interval-bound problem for weighted timed automata. In Dediu et al. [21], pages 452–464.

[47] Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.*, 6:223–231, 1978.

[48] Yaron Velner, Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, Alexander Moshe Rabinovich, and Jean-François Raskin. The complexity of multi-mean-payoff and multi-energy games. *Inf. Comput.*, 241:177–196, 2015.

Acta Cybernetica **23** (2017) 269–281.

Synchronous Forest Substitution Grammars[*]

Andreas Maletti[a]

— dedicated to the memory of Zoltán Ésik (1951–2016) —

Abstract

The expressive power of synchronous forest (tree-sequence) substitution grammars (SFSGs) is studied in relation to multi bottom-up tree transducers (MBOTs). It is proved that SFSGs have exactly the same expressive power as compositions of an inverse MBOT with an MBOT. This result is used to derive complexity results for SFSGs and the fact that compositions of an MBOT with an inverse MBOT can compute tree translations that cannot be computed by any SFSG, although the class of tree translations computable by MBOTs is closed under composition.

Keywords: tree transducer, synchronous grammar, regular tree language, machine translation

1 Introduction

Synchronous forest substitution grammars (SFSGs) [19] or the rational binary tree relations [17] computed by them received renewed interest recently due to their applications in Chinese-to-English machine translation [21, 22]. The fact that [19] and [17] arrived independently and with completely different backgrounds at the same model shows that SFSGs are a natural, practically relevant, and theoretically interesting model for tree translations. Roughly speaking, SFSGs are a synchronous grammar formalism [2] that utilizes only first-order substitution as in a regular tree grammar [7, 8], but allows several components that develop simultaneously for both the input and the output side. This feature allows them to model linguistic discontinuity on both the source and target language. The rational binary tree relations or equivalently the tree translations computed by SFSGs can also be characterized by rational expressions [17] and automata [16].

Multi bottom-up tree transducers (MBOTs) [1, 4] are restricted SFSGs, in which only the output side is allowed to have several components. They were rediscovered

[*]This article is an extended and revised version of [Maletti: *Synchronous forest substitution grammars.* In Proc. Algebraic Informatics, LNCS 8080, pages 235–246, 2013]

[a]Universität Leipzig, Institute of Computer Science, PO box 100 920, 04009 Leipzig, Germany, E-mail: `maletti@informatik.uni-leipzig.de`

DOI: 10.14232/actacyb.23.1.2017.15

in [5, 6], but were studied extensively by [3, 11, 1] already in the 70s and 80s. Their properties [13] are desirable in statistical syntax-based machine translation [10]. This led to a closer inspection [4, 15, 9] of their properties in recent years. Overall, their expressive power is rather well-understood by now.

In this contribution, we investigate the expressive power of SFSGs in terms of MBOTs. We show that the expressive power of SFSGs coincides exactly with that of compositions of an inverse MBOT followed by an MBOT. This characterization is natural in terms of bimorphisms and shows that the input and the output tree are independently obtained by a full MBOT from an intermediate tree language, which is always regular [7, 8]. This paves the way to complementary results. In particular, we derive the first complexity results for SFSGs and we demonstrate that the composition in the other order (first an MBOT followed by an inverse MBOT) contains tree translations that cannot be computed by any SFSG. This shows a limitation of MBOTs, which are closed under composition [4]. Overall, we can thus also characterize the expressive power of SFSGs by an arbitrary chain of inverse MBOTs followed by an arbitrary chain of MBOTs.

2 Preliminaries

We use $\mathbb{N}$ for the set of nonnegative integers, and $\mathbb{N}_+ = \mathbb{N} \setminus \{0\}$ for the set of positive integers. For all $k \in \mathbb{N}$, the set $\{i \in \mathbb{N}_+ \mid i \leq k\}$ is abbreviated to $[k]$. In particular, $[0] = \emptyset$. For all relations $R \subseteq A \times B$ and subsets $A' \subseteq A$, we let $R(A') = \{b \in B \mid \exists a \in A' \colon (a, b) \in R\}$. Moreover,

$$R^{-1} = \{(b, a) \mid (a, b) \in R\} \qquad \mathrm{dom}(R) = R^{-1}(B) \qquad \mathrm{ran}(R) = \mathrm{dom}(R^{-1}) \enspace ,$$

which are called the *inverse* of R, the *domain* of R, and the *range* of R, respectively. Given relations $R \subseteq A \times B$ and $S \subseteq B \times C$, the *composition* $R \,;\, S \subseteq A \times C$ of R followed by S is $R \,;\, S = \{(a, c) \in A \times C \mid \exists b \in B \colon (a, b) \in R,\, (b, c) \in S\}$. These notions and notations are lifted to sets and classes of relations as usual. For every $k \in \mathbb{N}$, we also write $A^k = A \times \cdots \times A$ containing the factor A exactly k times.

Given a set Σ, the set $\Sigma^* = \bigcup_{k \in \mathbb{N}} \Sigma^k$ is the set of all words over Σ, which includes the empty word $\varepsilon \in \Sigma^0$. The concatenation of two words $u, w \in \Sigma^*$ is denoted by $u.w$ or just uw. The length $|w|$ of a word $w \in \Sigma^*$ is the unique $k \in \mathbb{N}$ such that $w \in \Sigma^k$. We simply write w_i for the i-th letter of w, so $w_i = \sigma_i$ for all $i \in [k]$ provided that $w = \sigma_1 \cdots \sigma_k$ with letters $\sigma_i \in \Sigma$ for all $i \in [k]$. Given a set A, the set $T_\Sigma(A)$ of all Σ-*trees indexed by* A is the smallest set T such that $A \subseteq T$ and $\sigma\vec{u} \in T$ for all $\sigma \in \Sigma$ and $\vec{u} \in T^*$. Such a sequence $\vec{u}$ of trees is also called *forest*. Consequently, a tree t is either an element of A, or it consists of a root node labeled σ followed by a forest $\vec{u}$ of $|\vec{u}|$ children. To improve the readability, we often write a forest '$t_1 \cdots t_k$' as '$t_1, \ldots, t_k$', where $t_1, \ldots, t_k \in T_\Sigma(A)$. In addition, we identify the tree t with the forest (t). The *positions* $\mathrm{pos}(t) \subseteq \mathbb{N}_+^*$ of a tree $t \in T_\Sigma(A)$ are inductively defined by

$$\mathrm{pos}(a) = \{\varepsilon\} \qquad \text{and} \qquad \mathrm{pos}(\sigma\vec{u}) = \{\varepsilon\} \cup \bigcup_{i=1}^{|\vec{u}|} \{i.p \mid p \in \mathrm{pos}(\vec{u}_i)\}$$

for every $a \in A$, $\sigma \in \Sigma$, and $\vec{u} \in T_\Sigma(A)^*$. For each forest $\vec{u} \in T_\Sigma(A)^*$, we let $\mathrm{pos}(\vec{u}) = \bigcup_{i=1}^{|\vec{u}|} \{\#^{i-1}.p \mid p \in \mathrm{pos}(\vec{u}_i)\}$. Positions are totally ordered via the (standard) lexicographic ordering $\preceq$ on $\mathbb{N}_+^*$, which can be extended to $(\mathbb{N}_+ \cup \{\#\})^*$ with the convention that the additional letter # is larger than all numbers; i.e., $n \prec \#$ for every $n \in \mathbb{N}_+$. Let $t, t' \in T_\Sigma(A)$ and $p \in \mathrm{pos}(t)$. The label of t at position p is $t(p)$, the subtree rooted at position p is $t|_p$, and the tree obtained by replacing the subtree at position p by t' is denoted by $t[t']_p$. Formally, they are defined by $a(\varepsilon) = a|_\varepsilon = a$ and $a[t']_\varepsilon = t'$ for every $a \in A$ and

$$t(p) = \begin{cases} \sigma & \text{if } p = \varepsilon \\ \vec{u}_i(p') & \text{if } p = i.p' \text{ with } i \in \mathbb{N}_+ \end{cases} \qquad t|_p = \begin{cases} t & \text{if } p = \varepsilon \\ \vec{u}_i|_{p'} & \text{if } p = i.p' \text{ with } i \in \mathbb{N}_+ \end{cases}$$

$$t[t']_p = \begin{cases} t' & \text{if } p = \varepsilon \\ \sigma(\vec{u}_1, \dots, \vec{u}_{i-1}, \vec{u}_i[t']_{p'}, \vec{u}_{i+1}, \dots, \vec{u}_{|\vec{u}|}) & \text{if } p = i.p' \text{ with } i \in \mathbb{N}_+ \end{cases}$$

for all $t = \sigma\vec{u}$ with $\sigma \in \Sigma$ and $\vec{u} \in T_\Sigma(A)^*$. We immediately also extend this notion to forests $\vec{u} \in T_\Sigma(A)^*$ for all $\#^i p \in \mathrm{pos}(\vec{u})$ with $i \in \mathbb{N}$ and $p \in \mathrm{pos}(\vec{u}_{i+1})$ by $\vec{u}(\#^i p) = \vec{u}_{i+1}(p)$ and $\vec{u}|_{\#^i p} = \vec{u}_{i+1}|_p$ and

$$\vec{u}[t']_{\#^i p} = (\vec{u}_1, \dots, \vec{u}_{i-1}, \vec{u}_i[t']_p, \vec{u}_{i+1}, \dots, \vec{u}_{|\vec{u}|}) \ .$$

In the following, let $\vec{u} \in T_\Sigma(A)^*$ be a forest. By our identification of trees $T_\Sigma(A)$ with forests $T_\Sigma(A)^1$ of length 1, this choice includes trees. A position $p \in \mathrm{pos}(\vec{u})$ is a *leaf* in $\vec{u}$ if $p.1 \notin \mathrm{pos}(\vec{u})$. For every selection $S \subseteq A \cup \Sigma$ of labels, we let $\mathrm{pos}_S(\vec{u}) = \{p \in \mathrm{pos}(\vec{u}) \mid \vec{u}(p) \in S\}$ and $\mathrm{pos}_s(\vec{u}) = \mathrm{pos}_{\{s\}}(\vec{u})$ for every $s \in A \cup \Sigma$. The forest $\vec{u} \in T_\Sigma(A)$ is *linear in* $S \subseteq A$ if $|\mathrm{pos}_s(\vec{u})| \leq 1$ for every $s \in S$. The *variables* of $\vec{u}$ are $\mathrm{var}(\vec{u}) = \{a \in A \mid \mathrm{pos}_a(\vec{u}) \neq \emptyset\}$. Given a selection $S \subseteq A$ and a mapping $\theta \colon S \to T_\Sigma(A)^*$ such that $|\theta(s)| = |\mathrm{pos}_s(\vec{u})|$ for all $s \in S$, also called *suitable substitution for S in $\vec{u}$*, the forest $\vec{u}\theta$ is obtained from $\vec{u}$ by replacing for every $s \in S$ the leaves $\mathrm{pos}_s(\vec{u})$ in lexicographic order by the trees $\theta(s)$. Formally, for every $s \in S$, let $\mathrm{pos}_s(\vec{u}) = \{p_{s1}, \dots, p_{sk_s}\}$ with $p_{s1} \prec \cdots \prec p_{sk_s}$. Then

$$\vec{u}\theta = \vec{u}[\theta(s_1)_1]_{p_{s_1 1}} \cdots [\theta(s_1)_{|\theta(s_1)|}]_{p_{s_1 k_{s_1}}} \cdots [\theta(s_\ell)_1]_{p_{s_\ell 1}} \cdots [\theta(s_\ell)_{|\theta(s_\ell)|}]_{p_{s_\ell k_{s_\ell}}} \ ,$$

where $S = \{s_1, \dots, s_\ell\}$.

Given two sets Σ and Δ with $\Box \notin \Delta$, a mapping $d \colon \Sigma \to (\Delta \cup \{\Box\})$ is a *delabeling*. Thus, a delabeling is similar to a relabeling [7, 8], but it can also map symbols to a special symbol $\Box$, which will yield that those symbols are deleted, when they occur with exactly one child and project on the delabeling of the child. The delabeling induces a mapping $\tau_d \colon T_\Sigma(A) \to T_{\Sigma\cup\Delta}(A)$ such that $\tau_d(a) = a$ for all $a \in A$ and

$$\tau_d(\sigma\vec{u}) = \begin{cases} \tau_d(\vec{u}_1) & \text{if } d(\sigma) = \square \text{ and } |\vec{u}| = 1 \\ \sigma(\tau_d(\vec{u}_1), \dots, \tau_d(\vec{u}_{|\vec{u}|})) & \text{if } d(\sigma) = \square \text{ and } |\vec{u}| \neq 1 \\ d(\sigma)(\tau_d(\vec{u}_1), \dots, \tau_d(\vec{u}_{|\vec{u}|})) & \text{otherwise} \end{cases}$$

for all $\sigma \in \Sigma$ and $\vec{u} \in T_\Sigma(A)^*$.

Finally, let us recall the regular tree languages [7, 8]. A *(finite-state) tree automaton* (TA) is a tuple $G = (Q, \Sigma, I, R)$ such that Q is a finite set of *states*, Σ is an alphabet of symbols, $I \subseteq Q$ is a set of *initial states*, and $R \subseteq Q \times \Sigma \times Q^*$ is a finite set of *rules*. A rule $(q, \sigma, \vec{r}) \in R$ is typically written $q \to \sigma\vec{r}$. Given sentential forms $\xi, \zeta \in T_\Sigma(Q)$ we write $\xi \Rightarrow_G \zeta$ if there exists a rule $q \to \sigma\vec{r} \in R$ and an occurrence $p \in \mathrm{pos}_q(\xi)$ of q in ξ such that $\zeta = \xi[\sigma\vec{r}]_p$. The tree automaton G generates the tree language $L(G) = \{t \in T_\Sigma \mid \exists q \in I \colon q \Rightarrow_G^* t\}$, where $\Rightarrow_G^*$ is the reflexive and transitive closure of $\Rightarrow_G$. A tree language $L \subseteq T_\Sigma$ is *regular* if there exists a TA G such that $L = L(G)$. The class of regular tree languages is denoted by 'Reg'. Moreover, 'FTA' denotes the class of partial identities computed by the regular tree languages; i.e., $\mathrm{FTA} = \{\mathrm{id}_L \mid L \in \mathrm{Reg}\}$, where $\mathrm{id}_L = \{(t, t) \mid t \in L\}$.

3 Synchronous forest substitution grammars

In this section, we introduce our main model, the *(finite-state) synchronous forest-substitution grammar* (SFSG), which is the natural finite-state generalization of the (local non-contiguous) synchronous tree-sequence substitution grammars of [19]. Although we often speak about grammars in the following, we will continue to use 'states' instead of 'nonterminals'. SFSGs naturally coincide in expressive power with the binary rational relations studied by [17, 16], which we will show later. We immediately present it in a form inspired by tree bimorphisms [1] and tree grammars with multi-variables [17].

Definition 1. *A* (finite-state) synchronous forest-substitution grammar *(SFSG) is a tuple $G = (Q, \Sigma, \Delta, I, R)$, where*
- *Q is a finite set of states,*
- *Σ and Δ are alphabets of input and output symbols,*
- *$I \subseteq Q$ is a set of initial states, and*
- *$R \subseteq T_\Sigma(Q)^* \times Q \times T_\Sigma(Q)^*$ is a finite set of* rules.

It is a multi bottom-up tree transducer *(MBOT) if $R \subseteq T_\Sigma(Q) \times Q \times T_\Sigma(Q)^*$ and a* multiple regular tree grammar *(MRTG) if $R \subseteq T_\Sigma(Q) \times Q \times \{\varepsilon\}$.*

In simple terms, an SFSG consists of a finite set of rules that specify a state, for which the rule applies together with a sequence of input tree fragments and a sequence of output tree fragments. In an application of such a rule all fragments replace occurrences of the guarding state at the same time in the input and output tree. This also yields that all occurrences of the same state in those fragments are implicitly linked and prepared to be replaced in parallel in a future rule application.

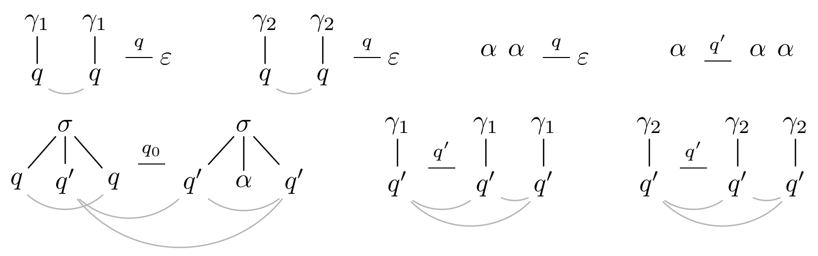


Figure 1: Example rules of the SFSG of Example 1.

An MBOT is a restricted SFSG, in which only a single input tree fragment is allowed in each rule. Compared to its traditional definition [4] the linearity of the single input tree fragment in the states Q is not required here, but as we will see nonlinear rules will not be useful in our version of MBOTs. To make the rules more readable, we also write $\ell_1 \cdots \ell_k \overset{q}{—} r_1 \cdots r_{k'}$ or $\vec{\ell} \overset{q}{—} \vec{r}$ for a rule $(\ell_1, \dots, \ell_k, q, r_1, \dots, r_{k'}) \in R$.

Example 1. Let $G = (Q, \Sigma, \Sigma, \{q_0\}, R)$ be the SFSG such that

- $Q = \{q_0, q, q'\}$ and $\Sigma = \{\alpha, \gamma_1, \gamma_2, \sigma\}$, and
- for every $\gamma \in \{\gamma_1, \gamma_2\}$ the following rules are in R:

$$\begin{aligned} \rho_0 &= \sigma(q, q', q) \overset{q_0}{—} \sigma(q', \alpha, q') & \rho_\gamma &= \gamma(q)\,\gamma(q) \overset{q}{—} \varepsilon & \rho_\alpha &= \alpha\,\alpha \overset{q}{—} \varepsilon \\ \rho'_\gamma &= \gamma(q') \overset{q'}{—} \gamma(q')\,\gamma(q') & \rho'_\alpha &= \alpha \overset{q'}{—} \alpha\,\alpha\ . \end{aligned}$$

The rules are illustrated in Figure 1, where we indicate the implicit links by splines. Clearly, this SFSG G is neither an MBOT nor an MRTG, although the rules for q are valid MRTG rules and the rules for q' are valid MBOT rules.

It remains to define the semantics of SFSGs. We use a bottom-up variant of the classical fixed-points semantics of an SFSG G. It closely corresponds to a semantics based on the evaluation of derivation trees (and that of bimorphisms), which we also define as well. We inductively define the pairs of input and output tree sequences generated by each state, which we call pre-translations. Each pre-translation for a state $q \in Q$ is obtained from a rule $\rho = \vec{\ell} \overset{q}{—} \vec{r}$ of R by replacing all occurrences of a state $q' \in \operatorname{var}(\vec{\ell}.\vec{r})$ by the corresponding components of a pre-translation for q'.

Definition 2. *Let $G = (Q, \Sigma, \Delta, I, R)$ be an SFSG. A* pre-translation *for $q \in Q$ is a pair $\langle \vec{u}, \vec{v} \rangle$ consisting of an input tree sequence $\vec{u} \in T_\Sigma^*$ and an output tree sequence $\vec{v} \in T_\Delta^*$. For every state $q \in Q$, the* pre-translations $G_q \subseteq T_\Sigma^* \times T_\Delta^*$ generated by q *are defined to be the smallest set T_q such that $\langle \vec{\ell}\theta, \vec{r}\theta' \rangle \in T_q$ for all rules $\rho = \vec{\ell} \overset{q}{—} \vec{r} \in R$ and suitable substitutions $\theta\colon \operatorname{var}(\vec{\ell}) \to T_\Sigma^*$ and $\theta'\colon \operatorname{var}(\vec{r}) \to T_\Delta^*$ for* $\operatorname{var}(\vec{\ell})$ *in $\vec{\ell}$ and for* $\operatorname{var}(\vec{r})$ *in $\vec{r}$, respectively, with pre-translations $\langle \theta(q'), \theta'(q') \rangle$ of $T_{q'}$ for every $q' \in \operatorname{var}(\vec{\ell}.\vec{r})$. The derivation tree corresponding to the newly constructed pre-translation $\langle \vec{\ell}\theta, \vec{r}\theta' \rangle$ is $\rho(t_{q_1}, \dots, t_{q_k})$, where* $\operatorname{var}(\vec{\ell}.\vec{r}) = \{q_1, \dots, q_k\}$

with $q_1 <_N \cdots <_Q q_k$ for some fixed total order $\leq_Q$ on Q and $t_{q'}$ is the derivation tree corresponding to the pre-translation $\langle \theta(q'), \theta'(q') \rangle$ for every $q' \in \mathrm{var}(\vec{\ell}.\vec{r})$. The derivation tree language $D_q(G) \subseteq T_R$ contains all derivation trees for the pre-translations $\langle \vec{u}, \vec{v} \rangle \in G_q$.

Example 2. Let us recall the SFSG G of Example 1. The rules $\rho_\alpha = \alpha\,\alpha \overset{q}{\text{---}} \varepsilon$ and $\rho'_\alpha = \alpha \overset{q'}{\text{---}} \alpha\,\alpha$ immediately yield the corresponding pre-translations $\langle \alpha\,\alpha, \varepsilon \rangle \in G_q$ and $\langle \alpha, \alpha\,\alpha \rangle \in G_{q'}$ with derivation trees ρ_α and ρ'_α, respectively. The former pre-translation can be combined with the rule ρ_γ for $\gamma \in \{\gamma_1, \gamma_2\}$ to obtain the pre-translation $\langle \gamma(\alpha)\,\gamma(\alpha), \varepsilon \rangle \in G_q$ with derivation tree $\rho_\gamma(\rho_\alpha)$, and more generally, the pre-translations

$$\langle \gamma_{i_1}(\cdots(\gamma_{i_k}(\alpha))\cdots)\,\gamma_{i_1}(\cdots(\gamma_{i_k}(\alpha))\cdots), \varepsilon \rangle \in G_q$$

for all $k \in \mathbb{N}$ and $i_1, \dots, i_k \in \{1, 2\}$. The derivation tree corresponding to the displayed pre-translation is $\rho_{\gamma_{i_1}}(\cdots(\rho_{\gamma_{i_k}}(\rho_\alpha))\cdots)$. Similarly, if we use the rules ρ'_γ with $\gamma \in \{\gamma_1, \gamma_2\}$ on the already mentioned pre-translation $\langle \alpha, \alpha\,\alpha \rangle \in G_{q'}$ and the such obtained pre-translations, then we derive the pre-translation

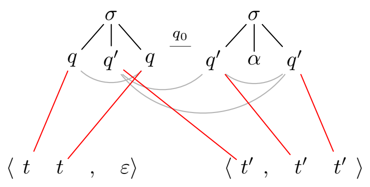
$$\langle \gamma_{i_1}(\cdots(\gamma_{i_k}(\alpha))\cdots), \gamma_{i_1}(\cdots(\gamma_{i_k}(\alpha))\cdots)\,\gamma_{i_1}(\cdots(\gamma_{i_k}(\alpha))\cdots) \rangle \in G_{q'}$$

using the derivation tree $\rho'_{\gamma_{i_1}}(\cdots(\rho'_{\gamma_{i_k}}(\rho'_\alpha))\cdots)$ for all $k \in \mathbb{N}$ and $i_1, \dots, i_k \in \{1, 2\}$. Plugging those pre-translations into the rule ρ_0, we obtain pre-translations of the form $\langle \sigma(t, t', t), \sigma(t', \alpha, t') \rangle \in G_{q_0}$. We illustrate the last step of the combination process in Figure 2.

The tree translation computed by an SFSG is now simply the set of all those pre-translations computed by the initial states that have sequences of length 1 for the input and output side. The restriction to sequences of length 1 is necessary to obtain a relation on trees. Finally, we also formally define the tree language generated by an SFSG although this notion is most suitable for MRTGs.

Definition 3. *Let $G = (Q, \Sigma, \Delta, I, R)$ be an SFSG. It computes the tree translation $\tau_G \subseteq T_\Sigma \times T_\Delta$ defined by $\tau_G = (\bigcup_{q \in I} G_q) \cap (T_\Sigma \times T_\Delta)$. The tree language $L(G) \subseteq T_\Sigma$ generated by G is $L(G) = (\bigcup_{q \in I} G_q) \cap (T_\Sigma \times \{\varepsilon\})$. Two SFSGs are (translation) equivalent if their computed tree translations coincide and language equivalent if their generated tree languages coincide. The classes* `SFSG` *and* `MBOT` *contain all tree translations computable by SFSGs and MBOTs, respectively, and the class* `MRTG` *denotes the class of all tree languages generated by MRTGs.*

In the rest of this section, we present a normal form for MBOTs and an alternative characterization of SFSGs in terms of classical bimorphisms [1] using a tree language of `MRTG` as center language. The former result demonstrates that our MBOTs are as expressive as the notion discussed in [4]. We conclude with some simple properties of `SFSG`, but we start with the normal form for MBOTs.



σ
q
q'
q
q_0
σ
q'
α
q'
$\langle\, t \quad t \;,\; \varepsilon \rangle$
$\langle\, t' \;,\; t' \quad t' \,\rangle$

Figure 2: Illustration of the combination of a rule with pre-translations.

Lemma 1. *For every MBOT $G = (Q, \Sigma, \Delta, I, R)$ there is a translation equivalent MBOT $G' = (Q, \Sigma, \Delta, I, R')$ such that t is linear in Q and $\operatorname{var}(\vec{r}) \subseteq \operatorname{var}(t)$ for every $t \overset{q}{—} \vec{r} \in R'$.*

Proof. We set $R' = \{t \overset{q}{—} \vec{r} \in R \mid t \text{ linear in } Q, \operatorname{var}(\vec{r}) \subseteq \operatorname{var}(t)\}$, which makes sure that the MBOT G' obeys the required restrictions. The translation equivalence of G and G' remains a proof obligation. We first observe that $|\vec{u}| = 1$ for every state $q \in Q$ and pre-translation $\langle \vec{u}, \vec{v} \rangle \in G_q$ due to the rule shape of G. Now, let $\rho = t \overset{q}{—} \vec{r} \in R$ be a rule that admits a state $q' \in \operatorname{var}(\vec{r}) \setminus \operatorname{var}(t)$. To build a pre-translation utilizing ρ (whose derivation tree has root label ρ), we need a pre-translation $\langle \varepsilon, \vec{v} \rangle \in G_{q'}$ because $q' \in \operatorname{var}(t.\vec{r})$, but $q' \notin \operatorname{var}(t)$. Such pre-translations do not exist, hence the rule ρ is useless (i.e., there are no derivation trees that contain ρ), which proves that deleting it does not affect the semantics. Similarly, let $\rho = t \overset{q}{—} \vec{r} \in R$ be a rule such that t is not linear in Q; i.e., there exists a state $q' \in Q$ such that $|\operatorname{pos}_{q'}(t)| \geq 2$. To utilize such a rule, we need a pre-translation $\langle \vec{u}, \vec{v} \rangle \in G_{q'}$ with $|\vec{u}| = |\operatorname{pos}_{q'}(t)| \geq 2$, which again do not exist. Consequently, both types of rules can be deleted without effect, which proves that G and G' are translation equivalent. □

Consequently, our class `MBOT` coincides with the notion of [4], so we can freely use the known properties of `MBOT`. Already in [12, 4] MBOTs were transformed into a normal form before composition. In this normal form, at most one (input or output) symbol is allowed in each rule. For our purposes, a slightly less restricted variant, in which at most one input symbol may occur in each rule is sufficient since we compose the input parts of two MBOTs. Let us recall the relevant normalization result [4].

Lemma 2 (see [4, Lemma 14])**.** *For every MBOT $G = (Q, \Sigma, \Delta, I, R)$ there exists a translation equivalent MBOT $G' = (Q', \Sigma, \Delta, I', R')$ in normal form, which means that $|\operatorname{pos}_\Sigma(t)| \leq 1$ for every rule $t \overset{q}{—} \vec{r} \in R'$.*

Proof. By Lemma 1 we can construct a translation equivalent MBOT G'' in the sense of [4]. With the help of [4, Lemma 14], we can then construct a translation equivalent MBOT G' in normal form. □

For MBOTs in normal form, we can now define the determinism property, which we use to avoid the k-morphisms of [1]. We note that deterministic MBOTs are slightly more expressive than k-morphisms.

Definition 4. *An MBOT* $(Q, \Sigma, \Delta, I, R)$ *in normal form is* deterministic *if* $|I| = 1$, $t \notin Q$ *for every* $t \overset{q}{—} \vec{r} \in R$, *and for every* $q \in Q$ *and* $\sigma \in \Sigma$ *there exists at most one rule* $t \overset{q}{—} \vec{r} \in R$ *with* $t(\varepsilon) = \sigma$. *It is a* deterministic linear top-down tree transducer with regular look-ahead *(deterministic LTOP^R) if additionally* $|\vec{r}| \leq 1$ *for all* $t \overset{q}{—} \vec{r} \in R$.

We conclude with the presentation of some simple properties of SFSG including one characterization of it in terms of bimorphisms. We will develop another bimorphism characterization in the next section.

Lemma 3. *We observe that (i)* $\mathtt{SFSG} = \mathtt{SFSG}^{-1}$, *(ii) both the domain* $\mathrm{dom}(\tau)$ *and the range* $\mathrm{ran}(\tau)$ *of a tree translation* $\tau \in \mathtt{SFSG}$ *are not necessarily regular, and (iii)* $\mathtt{MBOT} \subsetneq \mathtt{SFSG}$.

Proof. The first property is immediate because the syntactic definition of SFSGs is completely symmetric. The tree translation τ_G computed by the SFSG G of Example 1 is such that both its domain and its range are not regular, which proves the second property. Finally, the inclusion in the third item is obvious, and its strictness follows because $\mathrm{dom}(\tau)$ is regular for every $\tau \in \mathtt{MBOT}$ by Lemma 1 and [4, Theorem 25], so $\tau_G \notin \mathtt{MBOT}$. □

Theorem 1. *For every SFSG* G *there exists an MRTG* G_0 *and two deterministic LTOP^{R}s* G_1 *and* G_2 *such that* $\tau_G = \{(\tau_{G_1}(t), \tau_{G_2}(t)) \mid t \in L(G_0)\}$.

Proof. Let $G = (Q, \Sigma, \Delta, I, R)$ be the SFSG. We start with the construction of the MRTG $G_0 = (Q \cup \{\star\}, \Sigma \cup \Delta \cup \{\gamma\}, \emptyset, \{\star\}, R_0)$ such that $\star \notin Q$, $\gamma \notin \Sigma \cup \Delta$, and

$$R_0 = \{\gamma(q_0, q_0) \overset{\star}{—} \varepsilon \mid q_0 \in I\} \cup \{\vec{\ell}.\vec{r} \overset{q}{—} \varepsilon \mid \vec{\ell} \overset{q}{—} \vec{r} \in R\} \enspace .$$

Let $\Gamma = \Sigma \cup \Delta \cup \{\gamma\}$. The two deterministic LTOP$^{\mathrm{R}}$s G_1 and G_2 simply project on the first and second subtree, respectively. We omit their straightforward, albeit technical specification and the obvious correctness proof. □

Using Theorem 1 the relation of SFSG to the binary rational relations of [17] should be apparent. The main difference that remains is that we cannot specify the order, in which components are substituted. However, this does not restrict the expressive power. For the converse inclusion between SFSG and certain bimorphism, we restrict ourselves to linear tree homomorphisms [7, 8], which are slightly cumbersome to define in our notation. Note that the LTOP$^{\mathrm{R}}$s constructed in the previous proof are actually linear tree homomorphisms. We let LHOM denote the class of all linear tree homomorphisms, and assume that each tree homomorphism h is extended to act on state leaves as the identity; i.e., $h(q) = q$ for all $q \in Q$.

Theorem 2. *For all MRTGs* $G_0 = (Q, \Gamma, \emptyset, I, R_0)$ *and all tree homomorphisms* $h_1 \colon T_\Gamma \to T_\Sigma$ *and* $h_2 \colon T_\Gamma \to T_\Delta$ *there exists an SFSG* $G = (Q, \Sigma, \Delta, I, R)$ *such that* $\tau_G = \{(\tau_{G_1}(t), \tau_{G_2}(t)) \mid t \in L(G_0)\}$.

Proof. We let $R = \{h_1(\ell_1) \cdots h_1(\ell_k) \overset{q}{—} h_2(\ell_1) \cdots h_2(\ell_k) \mid \ell_1 \cdots \ell_k \overset{q}{—} \varepsilon \in R_0\}$. We again omit the straightforward correctness proof. □

Consequently, `SFSG` can be characterized by bimorphisms [1] with linear tree homomorphisms and a center language from `MRTG`.

4 Composition and decomposition

For our second characterization of `SFSG`, we first characterize it in terms of `MBOT`. Since we already showed that `MBOT` $\subsetneq$ `SFSG` in Lemma 3, we need a composition of MBOTs to characterize the expressive power of SFSGs. The relevant decomposition is presented in Theorem 3, and the corresponding composition is presented in Theorem 5.

Theorem 3 (see [17, Proposition 4.5])**.** *For every SFSG G, there exist two deterministic MBOTs G_1 and G_2 such that $\tau_G = \tau_{G_1}^{-1} ; \tau_{G_2}$.*

Proof. Let $G = (Q, \Sigma, \Delta, I, R)$ be the SFSG. As usual, we assume a total order $\leq$ on Q, and whenever we explicitly list states like $\{q_1, \ldots, q_k\}$, we assume that $q_1 < \cdots < q_k$. We construct the two MBOTs $G_1 = (Q, R, \Sigma, I, R_1)$ and $G_2 = (Q, R, \Delta, I, R_2)$ such that

- $R_1 = \{\rho(q_1, \ldots, q_k) \overset{q}{—} \vec{\ell} \mid \rho = \vec{\ell} \overset{q}{—} \vec{r} \in R,\ \mathrm{var}(\vec{\ell}.\vec{r}) = \{q_1, \ldots, q_k\}\}$, and
- $R_2 = \{\rho(q_1, \ldots, q_k) \overset{q}{—} \vec{r} \mid \rho = \vec{\ell} \overset{q}{—} \vec{r} \in R,\ \mathrm{var}(\vec{\ell}.\vec{r}) = \{q_1, \ldots, q_k\}\}$.

Obviously, both G_1 and G_2 are deterministic MBOTs. A straightforward induction can be used to prove that G_1 and G_2 translate derivation trees of $D_q(G)$ with $q \in Q$ into the corresponding input and output tree, respectively. Since each derivation tree $t \in D_q(G)$ uniquely determines the corresponding input and output tree, we immediately obtain that $\tau_G = \tau_{G_1}^{-1} ; \tau_{G_2}$. A more detailed proof can be found in [17]. □

In the proof of Theorem 3 the rule ρ uniquely determines the state q. Nevertheless, the constructed MBOTs have (potentially) several states as we need to check the finite-state behavior of the SFSG. It follows straightforwardly from the proof of Theorem 3 that each SFSG can be characterized by a regular derivation tree language and two deterministic MBOTs mapping the derivation trees to the input and output trees. This view essentially coincides with the bimorphism approach [1], and SFSGs are equally expressive as the bimorphisms of [1], in which both the input and output morphisms are allowed to be k-morphisms. We reuse this characterization later on, so we make it explicit here.

Theorem 4. `SFSG` = `dMBOT`$^{-1}$; `FTA` ; `dMBOT`*, where* `dMBOT` *is the class of all tree translations computed by deterministic MBOTs.*

Now we are ready to state our first composition result. We first prove it using several known results on decompositions and compositions together with a few new results.

Theorem 5. $\texttt{MBOT}^{-1} \mathbin{;} \texttt{MBOT} \subseteq \texttt{SFSG}$.

Proof. Let G_1 and G_2 be the given input MBOTs. Without loss of generality, let G_1 and G_2 be in normal form (see Lemma 2). With the help of the construction of [4, Lemma 6] applied to both G_1 and G_2 we obtain delabelings d_1 and d_2, regular tree languages $L_1, L_2 \in \mathrm{Reg}$, and deterministic MBOTs G'_1 and G'_2 such that

$$\tau_{G_1} = d_1^{-1} \mathbin{;} \mathrm{id}_{L_1} \mathbin{;} \tau_{G'_1} \qquad \text{and} \qquad \tau_{G_1} = d_2^{-1} \mathbin{;} \mathrm{id}_{L_2} \mathbin{;} \tau_{G'_2} \enspace .$$

This situation is depicted in Figure 3. We observe that

$$\tau_{G_1}^{-1} \mathbin{;} \tau_{G_2} = (d_1^{-1} \mathbin{;} \mathrm{id}_{L_1} \mathbin{;} \tau_{G'_1})^{-1} \mathbin{;} (d_2^{-1} \mathbin{;} \mathrm{id}_{L_2} \mathbin{;} \tau_{G'_2}) = \tau_{G'_1}^{-1} \mathbin{;} \mathrm{id}_{L_1} \mathbin{;} d_1 \mathbin{;} d_2^{-1} \mathbin{;} \mathrm{id}_{L_2} \mathbin{;} \tau_{G'_2} \enspace .$$

Next, we show that the composition $d_1 \mathbin{;} d_2^{-1}$ can equivalently be expressed as the composition $e_2^{-1} \mathbin{;} e_1$ for some delabelings e_1 and e_2 following the construction of [3, Sect. II-1-4-2-1]. To this end, let $d_1 \colon \Sigma \to \Delta \cup \{\square\}$, and we set $\Sigma' = \{\underline{\sigma} \mid \sigma \in \Sigma,\, d_1(\sigma) = \square\}$, which is an alphabet containing copies of the elements of Σ that are erased by d_1. Similarly, let $d_2 \colon \Gamma \to \Delta \cup \{\square\}$, and we set $\Gamma' = \{\overline{\gamma} \mid \gamma \in \Gamma,\, d_2(\gamma) = \square\}$ to an alphabet that contains copies of those elements of Γ that are erased by d_2. Moreover, let

$$\Delta'' = \{\langle \sigma, \gamma\rangle \mid \sigma \in \Sigma,\, \gamma \in \Gamma,\, d_1(\sigma) = d_2(\gamma) \neq \square\}$$

and $\Delta' = \Sigma' \cup \Gamma' \cup \Delta''$. Then we construct the two delabelings $e_1 \colon \Delta' \to \Sigma \cup \{\square\}$ and $e_2 \colon \Delta' \to \Gamma \cup \{\square\}$ as follows:

$$\begin{array}{lll} e_2(\underline{\sigma}) = \sigma & e_2(\overline{\gamma}) = \square & e_2(\langle \sigma, \gamma\rangle) = \sigma \\ e_1(\underline{\sigma}) = \square & e_1(\overline{\gamma}) = \gamma & e_1(\langle \sigma, \gamma\rangle) = \gamma \end{array}$$

for all $\underline{\sigma} \in \Sigma'$, $\overline{\gamma} \in \Gamma'$, and $\langle \sigma, \gamma\rangle \in \Delta''$. We leave the formal proof of $d_1;d_2^{-1} = e_2^{-1};e_1$ to the interested reader, but mention that it can be achieved by a simple induction. Thus, we arrive at

$$\tau_{G_1}^{-1} \mathbin{;} \tau_{G_2} = \tau_{G'_1}^{-1} \mathbin{;} \mathrm{id}_{L_1} \mathbin{;} d_1 \mathbin{;} d_2^{-1} \mathbin{;} \mathrm{id}_{L_2} \mathbin{;} \tau_{G'_2} = (\tau_{G'_1}^{-1} \mathbin{;} \mathrm{id}_{L_1} \mathbin{;} e_2^{-1}) \mathbin{;} (e_1 \mathbin{;} \mathrm{id}_{L_2} \mathbin{;} \tau_{G'_2})$$

using the just explained exchange of the delabelings. Since inverse delabelings preserve regular tree languages, we let $L'_1 = e_2^{-1}(L_1)$ and $L'_2 = e_1^{-1}(L_2)$, which are clearly both regular, so also their intersection $L'_1 \cap L'_2$ is regular [7, 8]. Consequently,

$$\tau_{G_1}^{-1} \mathbin{;} \tau_{G_2} = (\tau_{G'_1}^{-1} \mathbin{;} e_2^{-1}) \mathbin{;} \mathrm{id}_{L'_1 \cap L'_2} \mathbin{;} (e_1 \mathbin{;} \tau_{G'_2}) \enspace ,$$

which we can further simplify to $\tau_{G''_1}^{-1} \mathbin{;} \mathrm{id}_{L'_1 \cap L'_2} \mathbin{;} \tau_{G''_2}$ by composing the delabelings e_1 and e_2 with the deterministic MBOTs G'_1 and G'_2 to obtain the deterministic MBOTs G''_1 and G''_2, respectively, using [4, Theorem 23]. With this final step, we obtain a bimorphism representation of $\tau_{G_1}^{-1} \mathbin{;} \tau_{G_2}$ and according to Theorem 4 we have $\tau_{G_1}^{-1} \mathbin{;} \tau_{G_2} \in \texttt{SFSG}$. $\square$

$$\begin{tikzcd}
 & & L_1' \cap L_2' \arrow[dl, "e_2"'] \arrow[dr, "e_1"] & & \\
 & L_1 \arrow[dl, "\tau_{G_1'}"'] \arrow[dr, "d_1"] & & L_2 \arrow[dl, "d_2"'] \arrow[dr, "\tau_{G_2'}"] & \\
 {} & & {} & & {}
\end{tikzcd}$$

Figure 3: Illustration of the approach used in the proof of Theorem 5.

Problem	String level	Tree level
Parsing	$\mathcal{O}\big(\lvert G\rvert \cdot (\lvert w_1\rvert \cdot \lvert w_2\rvert)^{2r+2}\big)$	$\mathcal{O}\big(\lvert G\rvert \cdot \lvert t_1\rvert \cdot \lvert t_2\rvert\big)$
Translation	$\mathcal{O}\big(\lvert G\rvert \cdot \lvert w_1\rvert^{r+2}\big)$	$\mathcal{O}\big(\lvert G\rvert \cdot \lvert t_1\rvert\big)$

Table 1: Complexity results for an SFSG G and input strings (w_1, w_2) as well as trees (t_1, t_2), where $r = \max\, \{\lvert\vec{\ell}.\vec{r}\rvert \mid \vec{\ell} \overset{q}{—} \vec{r} \in R\}$ is the length of the longest sequence of input and output tree fragments in a rule of G.

Corollary 1 (of Theorems 3 and 5)**.** $\texttt{SFSG} = \texttt{MBOT}^{-1}\,;\texttt{MBOT}$.

We conclude with some additional properties of $\texttt{SFSG}$ and their consequences for $\texttt{MBOT}$ using our main result of Corollary 1. In particular, it is known [9] that the output string language of an MBOT is a language generated by an LCFRS (linear context-free rewriting system) [20, 18]. Using Corollary 1, we can conclude that both the input and the output string language of an SFSG are generated by an LCFRS as well. Similarly, together with Theorem 1 we can also conclude that the input and output tree languages are in $\texttt{MRTG}$. Moreover, we can import several complexity results from $\texttt{MBOT}$ [14] to $\texttt{SFSG}$ as indicated in Table 1.

Lemma 4 (see [16, Example 5])**.** $\texttt{SFSG}$ *is not closed under composition.*

Corollary 2. $\texttt{MBOT}\,;\texttt{MBOT}^{-1} \not\subseteq \texttt{SFSG}$.

Proof. Assume on the contrary that $\texttt{MBOT}\,;\texttt{MBOT}^{-1} \subseteq \texttt{SFSG}$. Then

$$\begin{aligned} \texttt{SFSG}\,;\texttt{SFSG} &\subseteq (\texttt{MBOT}^{-1}\,;\texttt{MBOT})\,;(\texttt{MBOT}^{-1}\,;\texttt{MBOT}) \subseteq \texttt{MBOT}^{-1}\,;\texttt{SFSG}\,;\texttt{MBOT} \\ &\subseteq \texttt{MBOT}^{-1}\,;(\texttt{MBOT}^{-1}\,;\texttt{MBOT})\,;\texttt{MBOT} \quad \subseteq \texttt{MBOT}^{-1}\,;\texttt{MBOT} = \texttt{SFSG} \end{aligned}$$

using Corollary 1, our assumption, Corollary 1, the closure under composition for $\texttt{MBOT}$ [4, Theorem 23], and Corollary 1 once more. However, the result contradicts Lemma 4, thus our assumption is false, proving the result. □

References

[1] Arnold, André and Dauchet, Max. Morphismes et bimorphismes d'arbres. *Theoret. Comput. Sci.*, 20(1):33–93, 1982.

[2] Chiang, David. An introduction to synchronous grammars. In *Proc. 44th ACL*. ACL, 2006. Part of a tutorial given with K. Knight.

[3] Dauchet, Max. *Transductions de forêts — Bimorphismes de magmoïdes.* Première thèse, Université de Lille, 1977.

[4] Engelfriet, Joost, Lilin, Eric, and Maletti, Andreas. Composition and decomposition of extended multi bottom-up tree transducers. *Acta Inform.*, 46(8):561–590, 2009.

[5] Fülöp, Zoltán, Kühnemann, Armin, and Vogler, Heiko. A bottom-up characterization of deterministic top-down tree transducers with regular look-ahead. *Inf. Process. Lett.*, 91(2):57–67, 2004.

[6] Fülöp, Zoltán, Kühnemann, Armin, and Vogler, Heiko. Linear deterministic multi bottom-up tree transducers. *Theoret. Comput. Sci.*, 347(1–2):276–287, 2005.

[7] Gécseg, Ferenc and Steinby, Magnus. *Tree Automata.* Akadémiai Kiadó, 1984. 2nd edition available at `https://arxiv.org/abs/1509.06233`.

[8] Gécseg, Ferenc and Steinby, Magnus. Tree languages. In Rozenberg, Grzegorz and Salomaa, Arto, editors, *Handbook of Formal Languages*, volume 3, chapter 1, pages 1–68. Springer, 1997.

[9] Gildea, Daniel. On the string translations produced by multi bottom-up tree transducers. *Comput. Linguist.*, 38(3):673–693, 2012.

[10] Knight, Kevin and Graehl, Jonathan. An overview of probabilistic tree transducers for natural language processing. In *Proc. 6th CICLing*, volume 3406 of LNCS, pages 1–24. Springer, 2005.

[11] Lilin, Eric. Propriétés de clôture d'une extension de transducteurs d'arbres déterministes. In *Proc. 6th CAAP*, volume 112 of LNCS, pages 280–289. Springer, 1981.

[12] Maletti, Andreas. Compositions of extended top-down tree transducers. *Inform. and Comput.*, 206(9–10):1187–1196, 2008.

[13] Maletti, Andreas. Why synchronous tree substitution grammars? In *Proc. 2010 HLT-NAACL*, pages 876–884. ACL, 2010.

[14] Maletti, Andreas. An alternative to synchronous tree substitution grammars. *J. Nat. Lang. Engrg.*, 17(2):221–242, 2011.

[15] Maletti, Andreas. How to train your multi bottom-up tree transducer. In *Proc. 49th ACL*, pages 825–834. ACL, 2011.

[16] Radmacher, Frank G. An automata theoretic approach to rational tree relations. In *Proc. 34th SOFSEM*, volume 4910 of LNCS, pages 424–435. Springer, 2008.

[17] Raoult, Jean-Claude. Rational tree relations. *Bull. Belg. Math. Soc. Simon Stevin*, 4(1):149–176, 1997.

[18] Seki, Hiroyuki, Matsumura, Takashi, Fujii, Mamoru, and Kasami, Tadao. On multiple context-free grammars. *Theoret. Comput. Sci.*, 88(2):191–229, 1991.

[19] Sun, Jun, Zhang, Min, and Tan, Chew Lim. A non-contiguous tree sequence alignment-based model for statistical machine translation. In *Proc. 47th ACL*, pages 914–922. ACL, 2009.

[20] Vijay-Shanker, K., Weir, David J., and Joshi, Aravind K. Characterizing structural descriptions produced by various grammatical formalisms. In *Proc. 25th ACL*, pages 104–111. ACL, 1987.

[21] Zhang, Min, Jiang, Hongfei, Aw, Aiti, Li, Haizhou, Tan, Chew Lim, and Li, Sheng. A tree sequence alignment-based tree-to-tree translation model. In *Proc. 46th ACL*, pages 559–567. ACL, 2008.

[22] Zhang, Min, Jiang, Hongfei, Li, Haizhou, Aw, Aiti, and Li, Sheng. Grammar comparison study for translational equivalence modeling and statistical machine translation. In *Proc. 22nd CoLing*, pages 1097–1104. ACL, 2008.

Acta Cybernetica **23** (2017) 283–317.

Weighted Recognizability over Infinite Alphabets

Maria Pittou[a] and George Rahonis[a]

Dedicated to the memory of Zoltán Ésik

Abstract

We introduce weighted variable automata over infinite alphabets and commutative semirings. We prove that the class of their behaviors is closed under sum, and under scalar, Hadamard, Cauchy, and shuffle products, as well as star operation. Furthermore, we consider rational series over infinite alphabets and we state a Kleene-Schützenberger theorem. We introduce a weighted monadic second order logic and a weighted linear dynamic logic over infinite alphabets and investigate their relation to weighted variable automata. An application of our theory, to series over the Boolean semiring, concludes to new results for the class of languages accepted by variable automata.

Keywords: infinite alphabets, semirings, weighted variable automata, weighted *MSO*, weighted *LDL*

1 Introduction

The last two decades a large body of research has been devoted to the development of models for infinite state systems which have finite control structure and handle data from an unbounded domain. This research led to the concept of finite automata over infinite alphabets. Motivating examples for such models consist, for instance, XML schemas, software with integer parameters, and system specification and verification. Later on, it came up that finite automata over infinite alphabets can contribute also to a series of interesting topics namely, the problem of query graph databases [33], reasoning about systems with resource generation capabilities [10, 11], learning theories [30], and systems with freshness needed in object-oriented languages and security protocols [6].

Several models of automata with data values, i.e., over infinite alphabets have been investigated, namely register [24, 28, 29, 37], data [5], pebble [28, 36, 39], nominal [10], variable [21, 22], and P automata [9]. All these models refer to qualitative aspects of infinite state systems. Furthermore, rational [1, 25] and logic definable languages [4, 36] have been studied over infinite alphabets.

[a]Department of Mathematics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece. E-mail: {mpittou,grahonis}@math.auth.gr

DOI: 10.14232/actacyb.23.1.2017.16

In this paper we intend to study automata models over infinite alphabets in the quantitative setup. Our motivation origins from the fact that several applications require a quantitative analysis of systems, for instance the resource usage control where resource variables are mapped to infinite domains [10, 11]. It is well-known that weighted automata is a reasonable tool for the description of quantitative features of computing systems [14]. According to our best knowledge, a quantitative counterpart for automata over infinite alphabets does not exist. In [8] the authors considered quantitative infinite alphabets to model controlled variables for the controller synthesis problem from incompatible situations. For our investigation, we chose the concept of variable automata from [21, 22]. Variable automata are simple in their definition and implementation in contrast to other proposed models. Despite their simplicity, variable automata and their extensions appeared to be expressive enough for several applications. Indeed, in [2] the authors introduced fresh variable automata to describe web services in which the agents exchange data ranging over infinite domains. Furthermore, in [3], fresh variable automata were equipped with guards consisting of equalities and disequalities. In [10] variable automata were extended to consume data words, in order to express security policies (safety properties) for model checking programs that dynamically generate and operate over resources. Very recently, variable automata have been also used for querying graph databases [43]. In a similar approach, a variable *LTL* was has been investigated in [23]. More precisely, the atomic propositions in that logic were parameterized with variables over some finite or infinite domain in order to express specifications over large, possibly infinite domains. The model checking problem has been also studied for that setting (cf. also [38]).

We consider our weighted variable automata over an infinite alphabet Σ and a commutative semiring K, and provide a systematic study of the class of their behaviors. Our framework builds upon the techniques which were developed in [26, 27] for variable tree automata over infinite ranked alphabets. We prove that, if in addition the semiring K is idempotent, then the class of series accepted by our models is closed under sum, and scalar, Hadamard, Cauchy, and shuffle products, as well as under star operation. As we indicate by a simple example, the proofs for the aforementioned properties require new techniques than the well-known ones for recognizable series [14]. We define rational series over infinite alphabets and state a Kleene-Schützenberger type theorem. Furthermore, we introduce a weighted monadic second order logic and a weighted linear dynamic logic over infinite alphabets. We show the expressive equivalence of the latter logic to our weighed automata, whereas the corresponding equivalence requires fragments on the weighted monadic second order logic. Therefore, several well-known results from classical weighted automata theory hold also for our weighted automata over infinite alphabets. Moreover, by considering the Boolean semiring $\mathbb{B}$, we derive as an application of our theory new results for the class of variable automata of [21, 22]. This shows the robustness of our theory and the theory of variable automata [21, 22].

Apart from this Introduction, the paper contains 7 sections. In Section 2 we present some preliminary background. In Section 3 we introduce our weighted variable automata and in Section 4 we establish the closure properties of the class of series accepted by our models. Then, in Section 5 we consider rational series over infinite alphabets and state our Kleene-Schützenberger theorem. Sections 6 and 7, respectively are devoted to weighted monadic second order logic and weighted linear dynamic logic, and their relation to weighted variable automata. In Section 8 we expose the new results on variable automata derived by our theory. Finally, in the Conclusion, we present some ideas for future research.

A preliminary version of this paper appeared in [32] (cf. also [31]).

2 Preliminaries

Let Σ be an alphabet, i.e., a nonempty (potentially infinite) set. As usually, we denote by Σ^* the set of all finite words over Σ and $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$, where ε is the empty word. A subset $L \subseteq \Sigma^*$ is a language over Σ. A word $w = \sigma_0 \ldots \sigma_{n-1}$, where $\sigma_0, \ldots, \sigma_{n-1} \in \Sigma$ $(n \geq 1)$, is written also as $w = w(0) \ldots w(n-1)$ where $w(i) = \sigma_i$ for every $0 \leq i \leq n-1$. For every finite word $w = w(0) \ldots w(n-1)$ and every $0 \leq i \leq n-1$ we denote by $w_{\geq i}$ the suffix $w(i) \ldots w(n-1)$. If S is a set, then $\mathcal{P}(S)$ will denote the powerset of S, and the notation $S' \subseteq_{fin} S$ means that S' is a finite subset of S.

A *semiring* $(K, +, \cdot, 0, 1)$ is an algebraic structure such that $(K, +, 0)$ is a commutative monoid, $(K, \cdot, 1)$ is a monoid, $0 \neq 1$, $\cdot$ is both left- and right-distributive over $+$, and $0 \cdot k = k \cdot 0 = 0$ for every $k \in K$. If no confusion arises, we shall denote the semiring simply by K and the $\cdot$ operation simply by concatenation. The semiring K is called *commutative* if the monoid $(K, \cdot, 1)$ is commutative. Moreover, K is called *additively idempotent* (or simply *idempotent*), if $k + k = k$ for every $k \in K$. Finally, K is called *locally finite* if every finitely generated subsemiring is finite. Interesting examples of semirings are the following:

- the semiring $(\mathbb{N}, +, \cdot, 0, 1)$ of *natural numbers*,
- the *Boolean semiring* $\mathbb{B} = (\{0, 1\}, +, \cdot, 0, 1)$,
- the *tropical* or *min-plus semiring* $(\mathbb{R}_+ \cup \{\infty\}, \min, +, \infty, 0)$ where $\mathbb{R}_+ = \{r \in \mathbb{R} \mid r \geq 0\}$,
- the *arctical* or *max-plus semiring* $(\mathbb{R}_+ \cup \{-\infty\}, \max, +, -\infty, 0)$,
- the *Viterbi semiring* $([0, 1], \max, \cdot, 0, 1)$,
- every bounded distributive lattice with the operations supremum and infimum, and especially the *fuzzy semiring* $F = ([0, 1], \max, \min, 0, 1)$.

All the above semirings, except the first one, are idempotent.

Let Σ be an alphabet and K a semiring. A *formal series* (or simply *series*) *over* Σ *and* K is a mapping $s : \Sigma^* \to K$. For every $w \in \Sigma^*$ we write (s, w) for the value $s(w)$ and refer to it as the *coefficient of* s *on* w. The *support of* s is the set $supp(s) = \{w \in \Sigma^* \mid (s,w) \neq 0\}$. A series with finite support is called a *polynomial.* The *constant series* $\widetilde{k}$ $(k \in K)$ is defined, for every $w \in \Sigma^*$, by $\left(\widetilde{k}, w\right) = k$. Moreover, for every $w \in \Sigma^*$, we denote by $\overline{w}$ the series determined, for every $u \in \Sigma^*$, by $(\overline{w}, u) = 1$ if $u = w$ and 0, otherwise. The class of all series over Σ and K is denoted as usual by $K\langle\langle\Sigma^*\rangle\rangle$, and the class of polynomials over Σ and K by $K\langle\Sigma^*\rangle$. The characteristic series $1_L \in K\langle\langle\Sigma^*\rangle\rangle$ of a language $L \subseteq \Sigma^*$ is defined by $(1_L, w) = 1$ if $w \in L$ and $(1_L, w) = 0$ otherwise.

Let $s, r \in K\langle\langle\Sigma^*\rangle\rangle$ and $k \in K$. The *sum* $s + r$, the *scalar products* ks and sk as well as the *Hadamard product* $s \odot r$ are defined elementwise by $(s + r, w) = (s, w) + (r, w)$, $(ks, w) = k \cdot (s, w)$, $(sk, w) = (s, w) \cdot k$, and $(s \odot r, w) = (s, w) \cdot (r, w)$, respectively, for every $w \in \Sigma^*$. It is well-known that the structures $\left(K\langle\langle\Sigma^*\rangle\rangle, +, \odot, \widetilde{0}, \widetilde{1}\right)$ and $\left(K\langle\Sigma^*\rangle, +, \odot, \widetilde{0}, \widetilde{1}\right)$ are semirings, which moreover are commutative (resp. idempotent) whenever K is commutative (resp. idempotent).

The *Cauchy product of* r *and* s is the series $r \cdot s \in K\langle\langle\Sigma^*\rangle\rangle$ defined for every $w \in \Sigma^*$ by

$$(r \cdot s, w) = \sum_{\substack{u,v \in \Sigma^* \\ w = uv}} ((r, u) \cdot (s, v)).$$

The nth-iteration $r^n \in K\langle\langle\Sigma^*\rangle\rangle$ $(n \geq 0)$ of a series $r \in K\langle\langle\Sigma^*\rangle\rangle$ is defined inductively by

$$r^0 = \overline{\varepsilon} \quad \text{and} \quad r^{n+1} = r \cdot r^n \quad \text{for } n \geq 0.$$

Then, we have $(r^n, w) = \sum\limits_{\substack{u_1,\ldots,u_n \in \Sigma^* \\ w = u_1 \ldots u_n}} ((r, u_1) \cdot \ldots \cdot (r, u_n))$ for every $w \in \Sigma^*$. A series $r \in K\langle\langle\Sigma^*\rangle\rangle$ is called *proper* whenever $(r, \varepsilon) = 0$. If r is proper, then for every $w \in \Sigma^*$ and $n > |w|$ we have $(r^n, w) = 0$. The *star* $r^* \in K\langle\langle\Sigma^*\rangle\rangle$ *of a proper series* $r \in K\langle\langle\Sigma^*\rangle\rangle$ is defined by $r^* = \sum\limits_{n \geq 0} r^n$. Thus, for every $w \in \Sigma^*$ we have $(r^*, w) = \sum\limits_{0 \leq n \leq |w|} (r^n, w)$.

Finally, the *shuffle product of* r *and* s is the series $r \sqcup\!\sqcup\, s \in K\langle\langle\Sigma^*\rangle\rangle$ defined for every $w \in \Sigma^*$ by

$$(r \sqcup\!\sqcup\, s, w) = \sum_{\substack{u,v \in \Sigma^* \\ w \in u \sqcup\!\sqcup\, v}} ((r, u) \cdot (s, v))$$

where $u \sqcup\!\sqcup\, v$ denotes the shuffle product of u and v.

Next we turn to weighted automata. For this we assume the alphabet Σ to be finite. A *weighted automaton over* Σ *and* K is a quadruple $A = (Q, in, wt, ter)$ where Q is the *finite state set*, $in : Q \to K$ is the *initial distribution*, $wt : Q \times \Sigma \times Q \to K$ is a mapping assigning *weights* to the transitions of the automaton, and $ter : Q \to K$ is the *final* (or *terminal*) *distribution.*

Let $w = w(0) \ldots w(n-1) \in \Sigma^*$. A *path of* A *over* w is a sequence of transitions

$P_w := ((q_i, w(i), q_{i+1}))_{0 \leq i \leq n-1}$. The *weight* of P_w is given by the value

$$weight(P_w) = in(q_0) \cdot \prod_{0 \leq i \leq n-1} wt\left((q_i, w(i), q_{i+1})\right) \cdot ter(q_n).$$

The *behavior of* A is the series $\|A\| : \Sigma^* \to K$ whose coefficients are given by

$$(\|A\|, w) = \sum_{P_w} weight(P_w)$$

for every $w \in \Sigma^*$.

A series $s \in K\langle\langle \Sigma^* \rangle\rangle$ is called recognizable if $s = \|A\|$ for some weighted automaton A over Σ and K. As usual we denote by $Rec(K, \Sigma)$ the class of recognizable series over Σ and K. Two weighted automata $A = (Q, in, wt, ter)$ and $A' = (Q', in', wt', ter')$ over Σ and K are called *equivalent* if $\|A\| = \|A'\|$.

Finally, a weighted automaton $A = (Q, in, wt, ter)$ over Σ and K is called *normalized* if there exist two states $q_{in}, q_{ter} \in Q$, $q_{in} \neq q_{ter}$, such that:

- $in(q) = 1$ if $q = q_{in}$, and $in(q) = 0$ otherwise,
- $ter(q) = 1$ if $q = q_{ter}$, and $ter(q) = 0$ otherwise, and
- $wt((q, \sigma, q_{in})) = wt((q_{ter}, \sigma, q)) = 0$

for every $q \in Q, \sigma \in \Sigma$. We shall denote a normalized weighted automaton $A = (Q, in, wt, ter)$ simply by $A = (Q, q_{in}, wt, q_{ter})$. The next result has been proved by several authors, cf. for instance [18].

Proposition 1. *Let $A = (Q, in, wt, ter)$ be a weighted automaton over Σ and K. We can effectively construct a normalized weighted automaton A' such that $(\|A'\|, w) = (\|A\|, w)$ for every $w \in \Sigma^+$ and $(\|A'\|, \varepsilon) = 0$.*

3 Weighted variable automata

In this section we introduce the notion of our weighted variable automata. We show that the well-known constructions on weighted automata are not sufficient to obtain the closure properties of the class of series recognized by our models. Therefore, we provide some supplementary matter and we state Lemma 1 which will be needed in the sequel in our constructions.

Let Σ, Σ' be (infinite) alphabets. A *relabeling from* Σ *to* Σ' is a mapping $h : \Sigma \to \mathcal{P}(\Sigma')$. Next let $\Gamma \subseteq_{fin} \Sigma$, Z be a finite set whose elements are called *bounded variables* and y an element which is called a *free variable.* We assume that the sets Σ, Z, and $\{y\}$ are pairwise disjoint. A relabeling h from $\Gamma \cup Z \cup \{y\}$ to Σ is called *valid* if

(i) it is the identity on Γ,[1]

[1]Abusing notation we identify $\{\sigma\}$ with σ, for every $\sigma \in \Gamma$.

(ii) $card(h(z)) = 1$ for every $z \in Z$,

(iii) h is injective on Z and $\Gamma \cap h(Z) = \emptyset$, and

(iv) $h(y) = \Sigma \setminus (\Gamma \cup h(Z))$.

The above definition means that the application of h on a word w over $\Gamma \cup Z \cup \{y\}$ assigns to every occurrence of a symbol $z \in Z$ in w the same symbol from Σ, but it is possible to assign different symbols from Σ to different occurrences of y in w. This justifies the names bounded and free for the set of variables Z and the variable y, respectively. It should be clear that a valid relabeling from $\Gamma \cup Z \cup \{y\}$ to Σ is well-defined if it is defined only on Z satisfying conditions (ii) and (iii). We shall denote by $VR(\Gamma \cup Z \cup \{y\}, \Sigma)$ the set of all valid relabelings from $\Gamma \cup Z \cup \{y\}$ to Σ, and simply by $VR(\Gamma \cup Z \cup \{y\})$ if the alphabet Σ is understood.

We set $\Delta = \Gamma \cup Z \cup \{y\}$ and let $w \in \Sigma^*$. The *preimage of w over Δ* is the set $preim_\Delta(w) = \{u \in \Delta^* \mid \text{ there exists } h \in VR(\Delta) \text{ such that } u \in h^{-1}(w)\}$.

Now we are ready to introduce our weighted variable automata over the infinite alphabet Σ and a semiring K.

Definition 1. *A* weighted variable automaton *(*wva *for short)* over Σ and K *is a pair $\mathcal{A} = \langle \Sigma, A \rangle$ where Σ is an infinite alphabet and $A = (Q, in, wt, ter)$ is a weighted automaton over Γ_A and K. The input alphabet Γ_A of A is defined by $\Gamma_A = \Sigma_A \cup Z \cup \{y\}$, where $\Sigma_A \subseteq_{fin} \Sigma$, Z is a finite alphabet of* bounded variables*, and y is a* free variable*.*

The *behavior of $\mathcal{A}$* is the series $\|\mathcal{A}\| : \Sigma^* \to K$ whose coefficients are determined by

$$(\|\mathcal{A}\|, w) = \sum_{u \in preim_{\Gamma_A}(w)} (\|A\|, u)$$

for every $w \in \Sigma^*$. Clearly, the above sum is finite and thus $(\|\mathcal{A}\|, w)$ is well-defined for every $w \in \Sigma^*$.

Two wva $\mathcal{A}$ and $\mathcal{A}'$ over Σ and K are called *equivalent* whenever $\|\mathcal{A}\| = \|\mathcal{A}'\|$.

A series r over Σ and K is called *v-recognizable* if there exists a wva $\mathcal{A}$ such that $r = \|\mathcal{A}\|$. We shall denote by $VRec(K, \Sigma)$ the class of v-recognizable series over Σ and K. It should be clear that every weighted automaton A over a subalphabet $\Sigma' \subseteq_{fin} \Sigma$ and K can be considered as a wva such that its transitions labelled by variables carry the weight 0. Therefore, we get the next result, where the strictness of the inclusion trivially holds by the definition of wva.

Proposition 2. $\bigcup_{\Sigma' \subseteq_{fin} \Sigma} Rec(K, \Sigma') \subsetneq VRec(K, \Sigma)$.

Throughout the paper Σ will denote an infinite alphabet, Z a finite set of bounded variables, y a free variable, and K a commutative semiring. In addition, in the present and the next section, K will be assumed to be idempotent.

In the sequel, we will call a wva $\mathcal{A} = \langle \Sigma, A\rangle$ over Σ and K, simply a wva.

Definition 2. *A wva $\mathcal{A} = \langle \Sigma, A\rangle$ is called* normalized *if A is normalized.*

Proposition 3. *Let $\mathcal{A} = \langle \Sigma, A\rangle$ be a wva. We can effectively construct a normalized wva $\mathcal{A}'$ such that $(\|\mathcal{A}'\|, w) = (\|\mathcal{A}\|, w)$ for every $w \in \Sigma^+$ and $(\|\mathcal{A}'\|, \varepsilon) = 0$.*

Proof. We immediately obtain our result by Proposition 1 and Definition 2. □

In the sequel, we wish to investigate closure properties of the class $VRec\,(K, \Sigma)$. For this, we cannot apply the well-known constructions from classical weighted automata theory. For instance, let $\mathcal{A} = \langle \Sigma, A\rangle$ be a normalized wva, where $A = (\{q_{in}, q, q_{ter}\}, q_{in}, wt_A, q_{ter})$, $\Gamma_A = \{a\} \cup \{z\} \cup \{y\}$ and transitions with non-zero weights given by $wt_A((q_{in}, a, q)) = wt_A((q, z, q_{ter})) = 1$. Consider also the normalized wva $\mathcal{A}' = \langle \Sigma, A'\rangle$ where $A' = (\{q'_{in}, q'_{ter}\}, q'_{in}, wt_{A'}, q'_{ter})$, $\Gamma_{A'} = \{a'\} \cup \{z'\} \cup \{y'\}$ and $wt_{A'}((q'_{in}, a', q'_{ter})) = wt_{A'}((q'_{in}, y', q'_{ter})) = 1$. Moreover, let us assume that $a \neq a'$. Clearly, $(\|\mathcal{A}\|, aa') = 1$ and $(\|\mathcal{A}'\|, a') = 1$. Nevertheless, if we consider the disjoint union of A and A', say the weighted automaton B, then $a, a' \in \Gamma_B$ which implies that we cannot apply a valid relabeling assigning the letter a' to z. This in turn, implies that the word aa' does not belong to the support of the wva derived by the weighted automaton B. Furthermore, another problem of this construction is the choice of the free variable among y and y' which moreover causes new inconsistencies. Similar, even more complex, situations arise for the constructions of wva proving closure under further properties like Hadamard, Cauchy, and shuffle product. Therefore, we state Lemma 1 below which will be of great importance to our constructions for the closure properties of the class $VRec\,(K, \Sigma)$. We shall need some preliminary matter.

Let $\mathcal{A} = \langle \Sigma, A\rangle$ be a wva where $A = (Q, in, wt, ter)$ with $\Gamma_A = \Sigma_A \cup Z \cup \{y\}$, and $\Sigma' \subseteq_{fin} \Sigma$ such that $\Sigma' \setminus \Sigma_A \neq \emptyset$. We define on $VR\,(\Gamma_A)$ the relation $\equiv_{\Sigma'}$ determined for every $h_1, h_2 \in VR\,(\Gamma_A)$ by

$$h_1 \equiv_{\Sigma'} h_2 \quad \text{iff} \quad h_1(\sigma) \cap \Sigma' = h_2(\sigma) \cap \Sigma' \text{ for every } \sigma \in Z \cup \{y\}.$$

It should be clear that $\equiv_{\Sigma'}$ is an equivalence relation. Moreover, since $Z \cup \{y\}$ and Σ' are finite, the index of $\equiv_{\Sigma'}$ is finite. Let V be a set of representatives of $VR\,(\Gamma_A) \,/ \equiv_{\Sigma'}$. For every $h \in V$, we let $Z_h = \{z \in Z \mid h(z) \in \Sigma'\}$ and $\Gamma_h = \Sigma_A \cup \Sigma' \cup (Z \setminus Z_h) \cup \{y\}$, and we consider the weighted automaton $A_h = (Q_h, in_h, wt_h, ter_h)$ over Γ_h and K, where $Q_h = \{q_h \mid q \in Q\}$ is a copy of Q, $in_h(q_h) = in(q)$ and $ter_h(q_h) = ter(q)$ for every $q_h \in Q_h$. The weight assignment mapping wt_h is defined as follows. For every $q_h, q'_h \in Q_h, \sigma \in \Gamma_h$, we let

$$wt_h\left((q_h, \sigma, q'_h)\right) = \begin{cases} wt\left((q, \sigma, q')\right) & \text{if } \sigma \in \Sigma_A \cup (Z \setminus Z_h) \cup \{y\} \\ wt\left((q, z, q')\right) & \text{if } \sigma = h(z) \text{ and } z \in Z_h \\ wt\left((q, y, q')\right) & \text{if } \sigma \in h\,(y) \cap \Sigma' \\ 0 & \text{otherwise} \end{cases} .$$

Without any loss, we assume that the sets Q_h are pairwise disjoint. We let $Q_V = \bigcup_{h\in V} Q_h$, $\Gamma_V = \Sigma_A \cup \Sigma' \cup Z \cup \{y\}$, and consider the wva $\mathcal{A}_{(\Sigma',V)} = \langle \Sigma, A_{(\Sigma',V)} \rangle$ over Σ and K, where $A_{(\Sigma',V)} = (Q_V, in_V, wt_V, ter_V)$ is a weighted automaton with input alphabet Γ_V. Its initial and final distribution are defined, respectively, by $in_V(q) = in_h(q)$, $ter_V(q) = ter_h(q)$ for every $q \in Q_h$, $h \in V$. The weight assignment mapping $wt_V : Q_V \times \Gamma_V \times Q_V \to K$ is given by

$$wt_V((q,\sigma,q')) = \begin{cases} wt_h((q,\sigma,q')) & \text{if } q,q' \in Q_h \text{ for some } h \in V \\ 0 & \text{otherwise} \end{cases}$$

for every $q, q' \in Q_V, \sigma \in \Gamma_V$.

Since the weighted automaton $A_{(\Sigma',V)}$ is the disjoint union of A_h, $h \in V$, we get that $\left\|A_{(\Sigma',V)}\right\| = \sum_{h\in V} \|A_h\|$. Therefore, for every $w \in \Sigma^*$, we have

$$\left(\left\|\mathcal{A}_{(\Sigma',V)}\right\|, w\right) = \sum_{u \in preim_{\Gamma_V}(w)} \left(\left\|A_{(\Sigma',V)}\right\|, u\right) = \sum_{h\in V} \sum_{u \in preim_{\Gamma_h}(w)} \left(\|A_h\|, u\right).$$

Lemma 1. $\|\mathcal{A}\| = \left\|\mathcal{A}_{(\Sigma',V)}\right\|$.

Proof. Let $w = w(0)\dots w(n-1) \in \Sigma^*$. Consider a word $u = u(0)\dots u(n-1) \in preim_{\Gamma_A}(w)$ and a valid relabeling $h \in VR(\Gamma_A)$ with $w \in h(u)$. We define the word $u' = u'(0)\dots u'(n-1) \in \Gamma_V^*$ as follows:

$$u'(i) = \begin{cases} u(i) & \text{if } (u(i) \in \Sigma_A \cup Z \setminus Z_h) \text{ or } (u(i) = y \text{ and } w(i) \notin \Sigma' \setminus \Sigma_A) \\ w(i) & \text{if } (u(i) \in Z_h) \text{ or } (u(i) = y \text{ and } w(i) \in \Sigma' \setminus \Sigma_A) \end{cases}$$

for every $0 \le i \le n-1$.

We consider the set of valid relabelings $V' \subseteq V$ as follows: $g \in V'$ implies that $g(z) = h(z)$ for every $z \in Z_h \cap \{u(i) \mid 0 \le i \le n-1\}$ and $g(y) \cap \Sigma' = h(y) \cap \Sigma'$ whenever $u(i) = y$ and $w(i) \in \Sigma'$ for some $0 \le i \le n-1$. Let $P_u^{(A)}$ be a path of A over u. Then, by construction of $A_{(\Sigma',V)}$, for every $g \in V'$, there exists a path $P_{u'}^{(A_g)}$ of A_g over u' with $weight\left(P_{u'}^{(A_g)}\right) = weight\left(P_u^{(A)}\right)$. Clearly, there are $r = card(V')$ such paths and since K is idempotent, we get $\sum_{g\in V'} weight\left(P_{u'}^{(A_g)}\right) = weight\left(P_u^{(A)}\right)$. On the other hand, for every $g \in V \setminus V'$ and path $P_{u'}^{(A_g)}$ of A_g, we have $weight\left(P_{u'}^{(A_g)}\right) = 0$. Therefore, we obtain

$$\sum_{P_u^{(A)}} weight\left(P_u^{(A)}\right) = \sum_{g\in V} \sum_{P_{u'}^{(A_g)}} weight\left(P_{u'}^{(A_g)}\right).$$

We define the valid relabeling $h' \in VR(\Gamma_V)$ as follows:

- $h'(z) = h(z)$ for every $z \in Z \setminus Z_h$,

and we let, nondeterministically,

- $h'(z) \in \Sigma \setminus (\Sigma_A \cup \Sigma' \cup h(Z \setminus Z_h) \cup \{w(i) \mid 0 \le i \le n-1 \text{ and } w(i) \in h(y)\})$ for every $z \in Z_h$.

Then we have $w \in h'(u')$ which implies that $u' \in preim_{\Gamma_V}(w)$.

Conversely, let $u' = u'(0) \dots u'(n-1) \in preim_{\Gamma_V}(w)$. Hence, there is a valid relabeling $h' \in VR(\Gamma_V)$ such that $w \in h'(u')$. By construction of $A_{(\Sigma',V)}$, there is a valid relabeling h from Γ_A to Σ and a word $u = u(0) \dots u(n-1) \in \Gamma_A^*$ such that

$$u(i) = \begin{cases} u'(i) & \text{if } u'(i) \in \Sigma_A \cup Z \setminus Z_h \\ z & \text{if } u'(i) = h(z) \text{ and } z \in Z_h \\ y & \text{if } u'(i) \in (h(y) \cap \Sigma') \cup \{y\} \end{cases}$$

for every $0 \le i \le n-1$. Keeping the previous notations, for every $g \in V'$, there is a path $P_{u'}^{(A_g)}$ of the weighted automaton A_g over u'. By construction of $A_{(\Sigma',V)}$, all such paths $P_{u'}^{(A_g)}$ $(g \in V')$ have the same weight and there exist $r = card(V')$ such paths. Furthermore, for every $g \in V'$ and $P_{u'}^{(A_g)}$ there is a path $P_u^{(A)}$ of A over u with $weight\left(P_u^{(A)}\right) = weight\left(P_{u'}^{(A_g)}\right)$, and since K is idempotent we get $weight\left(P_u^{(A)}\right) = \sum_{g \in V'} weight\left(P_{u'}^{(A_g)}\right)$. On the other hand, for every $g \in V \setminus V'$ and path $P_{u'}^{(A_g)}$ of A_g, we have that $weight\left(P_{u'}^{(A_g)}\right) = 0$. Therefore $\sum_{g \in V} \sum_{P_{u'}^{(A_g)}} weight\left(P_{u'}^{(A_g)}\right) = \sum_{P_u^{(A)}} weight\left(P_u^{(A)}\right)$. We consider the relabeling h'' from Γ_A to Σ defined in the following way. It is the identity on Σ_A, $h''(z) = h'(z)$ for every $z \in Z \setminus Z_h$, $h''(z) = h(z)$ for every $z \in Z_h$, and $h''(y) = h'(y) \cup ((h(y) \cap \Sigma') \setminus h(Z_h))$ (in fact $(h(y) \cap \Sigma') \cap h(Z_h) = \emptyset$ since h is a valid relabeling on Γ_A). Trivially h'' is a valid relabeling and $w \in h''(u)$ which implies that $u \in preim_{\Gamma_A}(w)$.

We conclude that for every $w \in \Sigma^*$ we have

$$\begin{aligned}
\left(\left\|\mathcal{A}_{(\Sigma',V)}\right\|, w\right) &= \sum_{u' \in preim_{\Gamma_V}(w)} \left(\left\|A_{(\Sigma',V)}\right\|, u'\right) = \sum_{u' \in preim_{\Gamma_V}(w)} \sum_{g \in V} \left(\|A_g\|, u'\right) \\
&= \sum_{u' \in preim_{\Gamma_V}(w)} \sum_{g \in V} \sum_{P_{u'}^{(A_g)}} weight\left(P_{u'}^{(A_g)}\right) \\
&= \sum_{u \in preim_{\Gamma_A}(w)} \sum_{P_u^{(A)}} weight\left(P_u^{(A)}\right) \\
&= \sum_{u \in preim_{\Gamma_A}(w)} \left(\|A\|, u\right) = \left(\|\mathcal{A}\|, w\right)
\end{aligned}$$

and we are done. □

4 Closure properties of the class $VRec\,(K,\Sigma)$

In this section, we investigate closure properties of the class of v-recognizable series over the infinite alphabet Σ and the semiring K. More precisely, we show that the class $VRec\,(K,\Sigma)$ is closed under sum, and under scalar, Hadamard, Cauchy and shuffle products, as well as star operation.

Proposition 4. *The class* $VRec\,(K,\Sigma)$ *is closed under sum.*

Proof. Let $r^{(i)} \in VRec\,(K,\Sigma)$ with $i=1,2$. Then there exist two wva $\mathcal{A}^{(i)} = \left\langle \Sigma, A^{(i)} \right\rangle$ with $A^{(i)} = \left(Q^{(i)}, in^{(i)}, wt^{(i)}, ter^{(i)}\right)$ and $\Gamma^{(i)} = \Sigma^{(i)} \cup Z^{(i)} \cup \left\{y^{(i)}\right\}$, accepting $r^{(i)}$, for $i=1,2$. Without any loss, we assume that $Q^{(1)} \cap Q^{(2)} = \emptyset$ and $\left(Z^{(1)} \cup \left\{y^{(1)}\right\}\right) \cap \left(Z^{(2)} \cup \left\{y^{(2)}\right\}\right) = \emptyset$. We consider the wva $\mathcal{A}^{(1)}_{\left(\Sigma^{(2)},V_1\right)} = \left\langle \Sigma, A^{(1)}_{\left(\Sigma^{(2)},V_1\right)} \right\rangle$ with $A^{(1)}_{\left(\Sigma^{(2)},V_1\right)} = \left(Q^{(1)}_{V_1}, in^{(1)}_{V_1}, wt^{(1)}_{V_1}, ter^{(1)}_{V_1}\right)$ over $\Gamma^{(1)} \cup \Sigma^{(2)}$ and K and the wva $\mathcal{A}^{(2)}_{\left(\Sigma^{(1)},V_2\right)} = \left\langle \Sigma, A^{(2)}_{\left(\Sigma^{(1)},V_2\right)} \right\rangle$ with $A^{(2)}_{\left(\Sigma^{(1)},V_2\right)} = \left(Q^{(2)}_{V_2}, in^{(2)}_{V_2}, wt^{(2)}_{V_2}, ter^{(2)}_{V_2}\right)$ over $\Gamma^{(2)} \cup \Sigma^{(1)}$ and K, determined by the procedure before Lemma 1. Moreover, without any loss, we assume that $Q^{(1)}_{V_1} \cap Q^{(2)}_{V_2} = \emptyset$. By Lemma 1 we have $\left\| \mathcal{A}^{(1)}_{\left(\Sigma^{(2)},V_1\right)} \right\| = r^{(1)}$ and $\left\| \mathcal{A}^{(2)}_{\left(\Sigma^{(1)},V_2\right)} \right\| = r^{(2)}$. Let $Q = Q^{(1)}_{V_1} \cup Q^{(2)}_{V_2}$ and $\Gamma = \Sigma^{(1)} \cup \Sigma^{(2)} \cup Z^{(1)} \cup Z^{(2)} \cup \{y\}$, where y denotes a new free variable different from $y^{(1)}$ and $y^{(2)}$. We consider the wva $\mathcal{A} = \langle \Sigma, A\rangle$ with $A = (Q, in, wt, ter)$ where in and ter are defined, for every $q \in Q$, respectively by

$$in\,(q) = \begin{cases} in^{(1)}_{V_1}(q) & \text{if } q \in Q^{(1)}_{V_1} \\ in^{(2)}_{V_2}(q) & \text{if } q \in Q^{(2)}_{V_2} \end{cases} \quad \text{and} \quad ter\,(q) = \begin{cases} ter^{(1)}_{V_1}(q) & \text{if } q \in Q^{(1)}_{V_1} \\ ter^{(2)}_{V_2}(q) & \text{if } q \in Q^{(2)}_{V_2} \end{cases}.$$

The weight assignment mapping $wt : Q \times \Gamma \times Q \to K$ is defined as follows:

$$wt\,((q,\sigma,q')) = \begin{cases} wt^{(1)}_{V_1}((q,\sigma,q')) & \text{if } q,q' \in Q^{(1)}_{V_1}, \sigma \in \Gamma \setminus \{y\} \\ wt^{(2)}_{V_2}((q,\sigma,q')) & \text{if } q,q' \in Q^{(2)}_{V_2}, \sigma \in \Gamma \setminus \{y\} \\ wt^{(1)}_{V_1}\left(\left(q,y^{(1)},q'\right)\right) & \text{if } q,q' \in Q^{(1)}_{V_1}, \sigma = y \\ wt^{(2)}_{V_2}\left(\left(q,y^{(2)},q'\right)\right) & \text{if } q,q' \in Q^{(2)}_{V_2}, \sigma = y \\ 0 & \text{otherwise} \end{cases}$$

for every $q,q' \in Q$, $\sigma \in \Gamma$.

We show that $\|\mathcal{A}\| = \left\|\mathcal{A}^{(1)}\right\| + \left\|\mathcal{A}^{(2)}\right\|$. For this, let $w \in \Sigma^*$, $u \in preim_{\Gamma}(w)$, and $h \in VR(\Gamma)$ such that $w \in h(u)$. Then, for every path $P^{(A)}_u$ of A over u, by construction of A, we point out the following cases. (i) There exists a path $P_{u^{(1)}}$ of $A^{(1)}_{\left(\Sigma^{(2)},V_1\right)}$ over $u^{(1)}$ with $weight\,(P_{u^{(1)}}) = weight\left(P^{(A)}_u\right)$, where $u^{(1)}$ is obtained from u by replacing every occurrence of y with $y^{(1)}$. (ii) There exists a path $P_{u^{(2)}}$ of $A^{(2)}_{\left(\Sigma^{(1)},V_2\right)}$ over $u^{(2)}$ with $weight\,(P_{u^{(2)}}) = weight\left(P^{(A)}_u\right)$, where $u^{(2)}$

is obtained from u by replacing every occurrence of y with $y^{(2)}$. Suppose firstly that (i) holds. We consider the valid relabeling $h^{(1)} \in VR\left(\Gamma^{(1)} \cup \Sigma^{(2)}\right)$ such that $h^{(1)}$ coincides with h on $\Sigma^{(1)} \cup \Sigma^{(2)} \cup Z^{(1)}$ and $h^{(1)}\left(y^{(1)}\right) = h(y) \cup h\left(Z^{(2)}\right)$. Trivially, $w \in h^{(1)}\left(u^{(1)}\right)$ which implies that $u^{(1)} \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w)$. Similarly, in case (ii) we get that $u^{(2)} \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w)$.

Conversely, let $w \in \Sigma^*$, $u^{(1)} \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w)$, and $h^{(1)} \in VR\left(\Gamma^{(1)} \cup \Sigma^{(2)}\right)$ such that $w \in h^{(1)}\left(u^{(1)}\right)$. Then, for every path $P_{u^{(1)}}$ of $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}$ over $u^{(1)}$, by construction of A, there exists a path $P_u^{(A)}$ of A over u with $weight\left(P_u^{(A)}\right) = weight\left(P_{u^{(1)}}\right)$, where u is obtained from $u^{(1)}$ by replacing every occurrence of $y^{(1)}$ with y. We define the valid relabeling $h \in VR(\Gamma)$ which coincides with $h^{(1)}$ on $\Sigma^{(1)} \cup \Sigma^{(2)} \cup Z^{(1)}$, $h(z) \in \Sigma \setminus (\Sigma^{(1)} \cup \Sigma^{(2)} \cup h^{(1)}\left(Z^{(1)}\right) \cup (h^{(1)}\left(y^{(1)}\right)$
$\cap\{w(i) \mid 0 \leq i \leq n-1\}))$ for $z \in Z^{(2)}$ and $h(y) = h^{(1)}\left(y^{(1)}\right) \setminus \{h(z) \mid z \in Z^{(2)}\}$. Trivially, $w \in h(u)$ which implies that $u \in preim_{\Gamma}(w)$.
Next assume that $u^{(2)} \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w)$ and $h^{(2)} \in VR\left(\Gamma^{(2)} \cup \Sigma^{(1)}\right)$ such that $w \in h^{(2)}\left(u^{(2)}\right)$. Then, for every path $P_{u^{(2)}}$ of $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}$ over $u^{(2)}$, by construction of A, there exists a path $P_{u'}^{A}$ of A over u' with $weight\left(P_{u'}^{(A)}\right) = weight\left(P_{u^{(2)}}\right)$, where u' is obtained from $u^{(2)}$ by replacing every occurrence of $y^{(2)}$ with y. We define the valid relabeling $h' \in VR(\Gamma)$ which coincides with $h^{(2)}$ on $\Sigma^{(1)} \cup \Sigma^{(2)} \cup Z^{(2)}$, $h'(z) \in \Sigma \setminus \left(\Sigma^{(1)} \cup \Sigma^{(2)} \cup h^{(2)}\left(Z^{(2)}\right) \cup \left(h^{(2)}\left(y^{(2)}\right) \cap \{w(i) \mid 0 \leq i \leq n-1\}\right)\right)$ for $z \in Z^{(1)}$ and $h'(y) = h^{(2)}\left(y^{(2)}\right) \setminus \{h'(z) \mid z \in Z^{(1)}\}$. Trivially, $w \in h'(u')$ which implies that $u' \in preim_{\Gamma}(w)$.

We conclude that for every $w \in \Sigma^*$ we have

$$\begin{aligned}
(\|\mathcal{A}\|, w) &= \sum_{u \in preim_{\Gamma}(w)} (\|A\|, u) = \sum_{u \in preim_{\Gamma}(w)} \sum_{P_u^{(A)}} weight\left(P_u^{(A)}\right) \\
&= \sum_{u^{(1)} \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w)} \sum_{P_{u^{(1)}}} weight\left(P_{u^{(1)}}\right) \\
&\qquad + \sum_{u^{(2)} \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w)} \sum_{P_{u^{(2)}}} weight\left(P_{u^{(2)}}\right) \\
&= \sum_{u^{(1)} \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w)} \left(\left\|A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, u^{(1)}\right) \\
&\qquad + \sum_{u^{(2)} \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w)} \left(\left\|A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, u^{(2)}\right) \\
&= \left(\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, w\right) + \left(\left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, w\right) \\
&= \left(\left\|\mathcal{A}^{(1)}\right\|, w\right) + \left(\left\|\mathcal{A}^{(2)}\right\|, w\right) \\
&= \left(\left\|\mathcal{A}^{(1)}\right\| + \left\|\mathcal{A}^{(2)}\right\|, w\right) = \left(r^{(1)} + r^{(2)}, w\right)
\end{aligned}$$

where the sixth equality holds by Lemma 1, and we are done. □

Proposition 5. *The class $VRec\,(K,\Sigma)$ is closed under the scalar products.*

Proof. Let $r \in VRec\,(K,\Sigma)$ and $k \in K$. Then there exists a wva $\mathcal{A} = \langle \Sigma, A\rangle$ with $A = (Q, in, wt, ter)$ accepting r. We consider the wva $\mathcal{A}' = \langle \Sigma, A'\rangle$ with $A' = (Q, in', wt, ter)$ where $in'(q) = k \cdot in\,(q)$ for every $q \in Q$. Then, by standard arguments we get $\|\mathcal{A}'\| = k\,\|\mathcal{A}\|$, and we are done. □

Proposition 6. *The class $VRec\,(K,\Sigma)$ is closed under Hadamard product.*

Proof. Let $r^{(i)} \in VRec\,(K,\Sigma)$ with $i = 1,2$. Then there exist two wva $\mathcal{A}^{(i)} = \left\langle \Sigma, A^{(i)}\right\rangle$ with $A^{(i)} = \left(Q^{(i)}, in^{(i)}, wt^{(i)}, ter^{(i)}\right)$ over $\Gamma^{(i)} = \Sigma^{(i)} \cup Z^{(i)} \cup \left\{y^{(i)}\right\}$, accepting $r^{(i)}$ for $i = 1,2$. Without any loss, we assume that $Q^{(1)} \cap Q^{(2)} = \emptyset$ and $\left(Z^{(1)} \cup \left\{y^{(1)}\right\}\right) \cap \left(Z^{(2)} \cup \left\{y^{(2)}\right\}\right) = \emptyset$. We consider the wva $\mathcal{A}^{(1)}_{\left(\Sigma^{(2)},V_1\right)} = \left\langle \Sigma, A^{(1)}_{\left(\Sigma^{(2)},V_1\right)}\right\rangle$ with $A^{(1)}_{\left(\Sigma^{(2)},V_1\right)} = \left(Q^{(1)}_{V_1}, in^{(1)}_{V_1}, wt^{(1)}_{V_1}, ter^{(1)}_{V_1}\right)$ over $\Gamma^{(1)} \cup \Sigma^{(2)}$ and $\mathcal{A}^{(2)}_{\left(\Sigma^{(1)},V_2\right)} = \left\langle \Sigma, A^{(2)}_{\left(\Sigma^{(1)},V_2\right)}\right\rangle$ with $A^{(2)}_{\left(\Sigma^{(1)},V_2\right)} = \left(Q^{(2)}_{V_2}, in^{(2)}_{V_2}, wt^{(2)}_{V_2}, ter^{(2)}_{V_2}\right)$ over $\Gamma^{(2)} \cup \Sigma^{(1)}$ determined by the procedure described before Lemma 1. Moreover, without any loss, we assume that $Q^{(1)}_{V_1} \cap Q^{(2)}_{V_2} = \emptyset$. By Lemma 1 we get $\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)},V_1\right)}\right\| = r^{(1)}$ and $\left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)},V_2\right)}\right\| = r^{(2)}$.

We consider the set $\left(Z^{(1)} \cup \left\{y^{(1)}\right\}\right) \times \left(Z^{(2)} \cup \left\{y^{(2)}\right\}\right) \setminus \{y\}$ where $y = \left(y^{(1)}, y^{(2)}\right)$, and a maximal subset $G \subseteq \left(Z^{(1)} \cup \left\{y^{(1)}\right\}\right) \times \left(Z^{(2)} \cup \left\{y^{(2)}\right\}\right) \setminus \{y\}$ satisfying the next condition: every element of $Z^{(1)}$ (resp. of $Z^{(2)}$) occurs in at most one pair in G as a left (resp. as a right) coordinate. Assume that $G_1, \ldots, G_m$ is an enumeration of all such sets of pairs of variables. Moreover, we let $Q = Q^{(1)}_{V_1} \times Q^{(2)}_{V_2}$ and $\Gamma_{G_j} = \Sigma^{(1)} \cup \Sigma^{(2)} \cup G_j \cup \{y\}$ for every $1 \leq j \leq m$, and we consider the wva $\mathcal{A}_{G_j} = \left\langle \Sigma, A_{G_j}\right\rangle$ with $A_{G_j} = \left(Q, in_{G_j}, wt_{G_j}, ter_{G_j}\right)$ over Γ_{G_j}. For every $1 \leq j \leq m$, the initial and terminal distribution are given respectively, by $in_{G_j}\left(\left(q^{(1)}, q^{(2)}\right)\right) = in^{(1)}_{V_1}\left(q^{(1)}\right) \cdot in^{(2)}_{V_2}\left(q^{(2)}\right)$ and $ter_{G_j}\left(\left(q^{(1)}, q^{(2)}\right)\right) = ter^{(1)}_{V_1}\left(q^{(1)}\right) \cdot ter^{(2)}_{V_2}\left(q^{(2)}\right)$, and the weight assignment mapping $wt_{G_j} : Q \times \Gamma_{G_j} \times Q \to K$ is defined by

$$wt_{G_j}\left(\left(q^{(1)}, q^{(2)}\right), \sigma, \left(q'^{(1)}, q'^{(2)}\right)\right) =$$
$$\begin{cases} wt^{(1)}_{V_1}\left(\left(q^{(1)}, \sigma, q'^{(1)}\right)\right) \cdot wt^{(2)}_{V_2}\left(\left(q^{(2)}, \sigma, q'^{(2)}\right)\right) & \text{if } \sigma \in \Sigma^{(1)} \cup \Sigma^{(2)} \\ wt^{(1)}_{V_1}\left(\left(q^{(1)}, x^{(1)}, q'^{(1)}\right)\right) \cdot wt^{(2)}_{V_2}\left(\left(q^{(2)}, x^{(2)}, q'^{(2)}\right)\right) & \text{if } \sigma = \left(x^{(1)}, x^{(2)}\right) \in G_j \cup \{y\} \\ 0 & \text{otherwise} \end{cases}$$

for every $\left(q^{(1)}, q^{(2)}\right), \left(q'^{(1)}, q'^{(2)}\right) \in Q, \sigma \in \Gamma_{G_j}$.

By Proposition 4, the series $\sum_{1\leq j\leq m} \left\| \mathcal{A}_{G_j} \right\|$ is recognizable. We will show that $\left\|\mathcal{A}^{(1)}\right\| \odot \left\|\mathcal{A}^{(2)}\right\| = \sum_{1\leq j\leq m} \left\|\mathcal{A}_{G_j}\right\|$.

To this end, let $w = w(0) \dots w(n-1) \in \Sigma^*$, $u^{(1)} \in preim_{\Gamma^{(1)}\cup\Sigma^{(2)}}(w)$, $u^{(2)} \in preim_{\Gamma^{(2)}\cup\Sigma^{(1)}}(w)$, $h^{(1)} \in VR\left(\Gamma^{(1)} \cup \Sigma^{(2)}\right)$, and $h^{(2)} \in VR\left(\Gamma^{(2)} \cup \Sigma^{(1)}\right)$ such that $w \in h^{(1)}\left(u^{(1)}\right) \cap h^{(2)}\left(u^{(2)}\right)$. For every $w(t) \in \Sigma$, $0 \leq t \leq n-1$, we have either $w(t) \in \Sigma^{(1)} \cup \Sigma^{(2)}$ and hence $u^{(1)}(t) = u^{(2)}(t) = w(t)$, or $w(t) \in \Sigma \setminus \Sigma^{(1)} \cup \Sigma^{(2)}$ and one of the following cases holds.

- There exist bounded variables $z^{(1)} \in Z^{(1)}$, $z^{(2)} \in Z^{(2)}$ such that $u^{(1)}(t) = z^{(1)}$, $u^{(2)}(t) = z^{(2)}$ and $h^{(1)}\left(u^{(1)}(t)\right) = h^{(2)}\left(u^{(2)}(t)\right) = w(t)$.
- There exists a bounded variable $z^{(1)} \in Z^{(1)}$ such that $u^{(1)}(t) = z^{(1)}$, $u^{(2)}(t) = y^{(2)}$ and $h^{(1)}\left(u^{(1)}(t)\right) = w(t) \in h^{(2)}\left(u^{(2)}(t)\right)$.
- There exists a bounded variable $z^{(2)} \in Z^{(2)}$ such that $u^{(1)}(t) = y^{(1)}$, $u^{(2)}(t) = z^{(2)}$ and $h^{(2)}\left(u^{(2)}(t)\right) = w(t) \in h^{(1)}\left(u^{(1)}(t)\right)$.
- $u^{(1)}(t) = y^{(1)}$, $u^{(2)}(t) = y^{(2)}$, and $w(t) \in h^{(1)}\left(u^{(1)}(t)\right) \cap h^{(2)}\left(u^{(2)}(t)\right)$.

We consider the word $u = u(0) \dots u(n-1)$ by

$$u(t) = \begin{cases} w(t) & \text{if } w(t) \in \Sigma^{(1)} \cup \Sigma^{(2)} \\ \left(u^{(1)}(t), u^{(2)}(t)\right) & \text{otherwise} \end{cases}$$

for every $0 \leq t \leq n-1$. For every $1 \leq j \leq m$, we define a valid relabeling $h_j \in VR\left(\Gamma_{G_j}\right)$ such that $h_j(\sigma) = h^{(1)}\left(x^{(1)}\right)$ for every $\sigma = \left(x^{(1)}, x^{(2)}\right) \in \left(Z^{(1)} \times \left(Z^{(2)} \cup \left\{y^{(2)}\right\}\right)\right) \cap G_j$, and $h_j(\sigma) = h^{(2)}\left(x^{(2)}\right)$ for every $\sigma = \left(x^{(1)}, x^{(2)}\right) \in \left(\left\{y^{(1)}\right\} \times Z^{(2)}\right) \cap G_j$. Hence $u \in preim_{\Gamma_{G_j}}(w)$ for some $1 \leq j \leq m$.

By the definition of the list $G_1, \dots, G_m$, there is a set $J \subseteq \{1, \dots, m\}$, such that for every path

$$P_{u^{(1)}} : \left(q_0^{(1)}, u^{(1)}(0), q_1^{(1)}\right) \dots \left(q_{n-1}^{(1)}, u^{(1)}(n-1), q_n^{(1)}\right)$$

of $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}$ over $u^{(1)}$, and

$$P_{u^{(2)}} : \left(q_0^{(2)}, u^{(2)}(0), q_1^{(2)}\right) \dots \left(q_{n-1}^{(2)}, u^{(2)}(n-1), q_n^{(2)}\right)$$

of $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}$ over $u^{(2)}$, there exists a path

$$P_u^{(G_j)} : \left(\left(q_0^{(1)}, q_0^{(2)}\right), u(0), \left(q_1^{(1)}, q_1^{(2)}\right)\right) \dots \left(\left(q_{n-1}^{(1)}, q_{n-1}^{(2)}\right), u(n-1), \left(q_n^{(1)}, q_n^{(2)}\right)\right)$$

of A_{G_j} over u, for every $j \in J$. Conversely, for every path $P_u^{(G_j)}$ of A_{G_j} over u ($j \in J$) there are two paths $P_{u^{(1)}}$ of $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}$ over $u^{(1)}$ and $P_{u^{(2)}}$ of $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}$ over $u^{(2)}$ respectively, obtained in the obvious way. Moreover, in case $weight\left(P_u^{(G_j)}\right) \neq 0$, for every $j \in J$, it holds

$$\begin{aligned}
&weight\left(P_u^{(G_j)}\right)\\
&= in_{G_j}\left(\left(q_0^{(1)}, q_0^{(2)}\right)\right) \cdot \prod_{0\le t\le n-1} wt_{G_j}\left(\left(\left(q_t^{(1)}, q_t^{(2)}\right), u(t), \left(q_{t+1}^{(1)}, q_{t+1}^{(2)}\right)\right)\right)\\
&\qquad \cdot ter_{G_j}\left(\left(q_n^{(1)}, q_n^{(2)}\right)\right)\\
&= in_{V_1}^{(1)}\left(q_0^{(1)}\right) \cdot in_{V_2}^{(2)}\left(q_0^{(2)}\right) \cdot \prod_{0\le t\le n-1}\begin{pmatrix} wt_{V_1}^{(1)}\left(\left(q_t^{(1)}, u^{(1)}(t), q_{t+1}^{(1)}\right)\right)\\ \cdot wt_{V_2}^{(2)}\left(\left(q_t^{(2)}, u^{(2)}(t), q_{t+1}^{(2)}\right)\right)\end{pmatrix}\\
&\qquad \cdot ter_{V_1}^{(1)}\left(q_n^{(1)}\right) \cdot ter_{V_2}^{(2)}\left(q_n^{(2)}\right)\\
&= in_{V_1}^{(1)}\left(q_0^{(1)}\right) \cdot \prod_{0\le t\le n-1} wt_{V_1}^{(1)}\left(\left(q_t^{(1)}, u^{(1)}(t), q_{t+1}^{(1)}\right)\right) \cdot ter_{V_1}^{(1)}\left(q_n^{(1)}\right)\\
&\qquad \cdot in_{V_2}^{(2)}\left(q_0^{(2)}\right) \cdot \prod_{0\le t\le n-1} wt_{V_2}^{(2)}\left(\left(q_t^{(2)}, u^{(2)}(t), q_{t+1}^{(2)}\right)\right) \cdot ter_{V_2}^{(2)}\left(q_n^{(2)}\right)\\
&= weight\left(P_{u^{(1)}}\right) \cdot weight\left(P_{u^{(2)}}\right).
\end{aligned}$$

Conversely, if $weight\left(P_{u^{(1)}}\right) \neq 0$, $weight\left(P_{u^{(2)}}\right) \neq 0$, then by the consideration of the list $G_1, \ldots, G_m$, there is at least one $1 \le j \le m$ with $weight\left(P_u^{(G_j)}\right) = weight\left(P_{u^{(1)}}\right) \cdot weight\left(P_{u^{(2)}}\right)$. Therefore, and since K is idempotent, we obtain[2]

$$\sum_{1\le j\le m}\sum_{P_u^{(G_j)}} weight\left(P_u^{(G_j)}\right) = \sum_{P_{u^{(1)}}, P_{u^{(2)}}} weight\left(P_{u^{(1)}}\right) \cdot weight\left(P_{u^{(2)}}\right).$$

We conclude

$$\begin{aligned}
\left(\sum_{1\le j\le m}\left\|\mathcal{A}_{G_j}\right\|, w\right) &= \sum_{1\le j\le m}\left(\left\|\mathcal{A}_{G_j}\right\|, w\right) = \sum_{1\le j\le m}\ \sum_{u\in preim_{\Gamma_{G_j}}(w)}\left(\left\|A_{G_j}\right\|, u\right)\\
&= \sum_{1\le j\le m}\ \sum_{u\in preim_{\Gamma_{G_j}}(w)}\ \sum_{P_u^{(G_j)}} weight\left(P_u^{(G_j)}\right)\\
&= \sum_{\substack{u^{(1)}\in preim_{\Gamma^{(1)}\cup\Sigma^{(2)}}(w)\\ u^{(2)}\in preim_{\Gamma^{(2)}\cup\Sigma^{(1)}}(w)}}\ \sum_{\substack{P_{u^{(1)}}\\ P_{u^{(2)}}}}\left(weight\left(P_{u^{(1)}}\right) \cdot weight\left(P_{u^{(2)}}\right)\right)\\
&= \sum_{u^{(1)}\in preim_{\Gamma^{(1)}\cup\Sigma^{(2)}}(w)}\ \sum_{P_{u^{(1)}}} weight\left(P_{u^{(1)}}\right)
\end{aligned}$$

[2]It should be clear that for $j \in \{1, \ldots, m\} \setminus J$ the paths $P_u^{(Gj)}$ do not exist, hence by definition $weight\left(P_u^{(Gj)}\right) = 0$.

$$\cdot \sum_{u^{(2)} \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w)} \sum_{P_{u^{(2)}}} weight\left(P_{u^{(2)}}\right)$$

$$= \sum_{u^{(1)} \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w)} \left(\left\| A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} \right\|, u^{(1)}\right)$$

$$\cdot \sum_{u^{(2)} \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w)} \left(\left\| A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} \right\|, u^{(2)}\right)$$

$$= \left(\left\| \mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} \right\|, w\right) \cdot \left(\left\| \mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} \right\|, w\right)$$

for every $w \in \Sigma^*$, which, by Lemma 1, implies

$$\left(\left\|\mathcal{A}^{(1)}\right\| \odot \left\|\mathcal{A}^{(2)}\right\|, w\right) = \left(\sum_{1 \leq j \leq m} \left\|\mathcal{A}_{G_j}\right\|, w\right)$$

for every $w \in \Sigma^*$, i.e., $\left\|\mathcal{A}^{(1)}\right\| \odot \left\|\mathcal{A}^{(2)}\right\| = \sum\limits_{1 \leq j \leq m} \left\|\mathcal{A}_{G_j}\right\|$, as required. □

Proposition 7. *The class $VRec\,(K, \Sigma)$ is closed under Cauchy product.*

Proof. Let $r^{(i)} \in VRec\,(K, \Sigma)$ with $i = 1, 2$. We consider the proper series $r'^{(1)}, r'^{(2)}$ over Σ and K defined, for every $w \in \Sigma^*$, by

- $\left(r'^{(1)}, w\right) = \begin{cases} \left(r^{(1)}, w\right) & \text{if } w \in \Sigma^+ \\ 0 & \text{otherwise} \end{cases}$, and
- $\left(r'^{(2)}, w\right) = \begin{cases} \left(r^{(2)}, w\right) & \text{if } w \in \Sigma^+ \\ 0 & \text{otherwise.} \end{cases}$

Then $r^{(1)} \cdot r^{(2)} = r'^{(1)} \cdot r'^{(2)} + \left(r^{(1)}, \varepsilon\right) r^{(2)} + r^{(1)} \left(r^{(2)}, \varepsilon\right) + \left(r^{(1)}, \varepsilon\right) \left(r^{(2)}, \varepsilon\right) \bar{\varepsilon}$ and by Propositions 2, 4, and 5, it suffices to show that $r'^{(1)} \cdot r'^{(2)} \in VRec\,(K, \Sigma)$. By Proposition 3, there are normalized wva $\mathcal{A}^{(i)} = \left\langle \Sigma, A^{(i)} \right\rangle$ with $A^{(i)} = \left(Q^{(i)}, q^{(i)}_{in}, wt^{(i)}, q^{(i)}_{ter}\right)$ over $\Gamma^{(i)} = \Sigma^{(i)} \cup Z^{(i)} \cup \left\{y^{(i)}\right\}$ and K, accepting respectively $r'^{(i)}$, with $i = 1, 2$. Without any loss, we assume that $Q^{(1)} \cap Q^{(2)} = \emptyset$ and $\left(Z^{(1)} \cup \left\{y^{(1)}\right\}\right) \cap \left(Z^{(2)} \cup \left\{y^{(2)}\right\}\right) = \emptyset$. We consider the wva $\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} = \left\langle \Sigma, A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} \right\rangle$ and $\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} = \left\langle \Sigma, A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} \right\rangle$ determined by the procedure before Lemma 1. By Proposition 3 and Lemma 1, $\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}$ and $\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}$ can be assumed to be normalized hence, let $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} = \left(Q^{(1)}_{V_1}, q^{(1)}_{in_{V_1}}, wt^{(1)}_{V_1}, q^{(1)}_{ter_{V_1}}\right)$ over $\Gamma^{(1)} \cup \Sigma^{(2)}$ and $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} = \left(Q^{(2)}_{V_2}, q^{(2)}_{in_{V_2}}, wt^{(2)}_{V_2}, q^{(2)}_{ter_{V_2}}\right)$ over $\Gamma^{(2)} \cup \Sigma^{(1)}$. Moreover, without any loss, we assume that $Q^{(1)}_{V_1} \cap Q^{(2)}_{V_2} = \emptyset$. We let $y = \left(y^{(1)}, y^{(2)}\right)$ and consider the set $H = \left(Z^{(1)} \cup \left\{y^{(1)}\right\}\right) \times \left(Z^{(2)} \cup \left\{y^{(2)}\right\}\right) \setminus \{y\}$, and a maximal subset $G \subseteq H \cup Z^{(1)} \cup Z^{(2)}$ satisfying the following condition: every element of $Z^{(1)}$ (resp. of $Z^{(2)}$) occurs either in at most one pair of H as a left (resp. as a right) coordinate, or as a single element of G. Assume that $G_1, \ldots, G_m$ is an enumeration of all such sets. We let $Q = Q^{(1)}_{V_1} \cup Q^{(2)}_{V_2} \setminus \left\{q^{(1)}_{ter_{V_1}}\right\}$ and consider, for every $1 \leq j \leq m$, the normalized wva $\mathcal{A}_{G_j} = \left\langle \Sigma, A_{G_j} \right\rangle$ where $A_{G_j} = \left(Q, q^{(1)}_{in_{V_1}}, wt_{G_j}, q^{(2)}_{ter_{V_2}}\right)$ and $\Gamma_{G_j} = \Sigma^{(1)} \cup \Sigma^{(2)} \cup G_j \cup \{y\}$. The weight assignment mapping wt_{G_j} is defined, for every $1 \leq j \leq m$, as follows:

$$wt_{G_j}\left((q,\sigma,q')\right) =$$
$$\begin{cases} wt^{(1)}_{V_1}\left((q,\sigma,q')\right) & \text{if } q,q' \in Q^{(1)}_{V_1} \setminus \left\{q^{(1)}_{ter_{V_1}}\right\} \text{ and} \\ & \qquad \sigma \in \Sigma^{(1)} \cup \Sigma^{(2)} \cup \left(Z^{(1)} \cap G_j\right) \\ wt^{(2)}_{V_2}\left((q,\sigma,q')\right) & \text{if } q,q' \in Q^{(2)}_{V_2} \text{ and } \sigma \in \Sigma^{(1)} \cup \Sigma^{(2)} \cup \left(Z^{(2)} \cap G_j\right) \\ wt^{(1)}_{V_1}\left(\left(q,\sigma,q^{(1)}_{ter_{V_1}}\right)\right) & \text{if } q \in Q^{(1)}_{V_1} \setminus \left\{q^{(1)}_{ter_{V_1}}\right\}, q' = q^{(2)}_{in_{V_2}}, \text{ and} \\ & \qquad \sigma \in \Sigma^{(1)} \cup \Sigma^{(2)} \cup \left(Z^{(1)} \cap G_j\right) \\ wt^{(1)}_{V_1}\left(\left(q,x^{(1)},q'\right)\right) & \text{if } q,q' \in Q^{(1)}_{V_1} \setminus \left\{q^{(1)}_{ter_{V_1}}\right\} \text{ and} \\ & \qquad \sigma = \left(x^{(1)},x^{(2)}\right) \in G_j \cup \{y\} \\ wt^{(2)}_{V_2}\left(\left(q,x^{(2)},q'\right)\right) & \text{if } q,q' \in Q^{(2)}_{V_2} \text{ and } \sigma = \left(x^{(1)},x^{(2)}\right) \in G_j \cup \{y\} \\ wt^{(1)}_{V_1}\left(\left(q,x^{(1)},q^{(1)}_{ter_{V_1}}\right)\right) & \text{if } q \in Q^{(1)}_{V_1} \setminus \left\{q^{(1)}_{ter_{V_1}}\right\}, q' = q^{(2)}_{in_{V_2}}, \text{ and} \\ & \qquad \sigma = \left(x^{(1)},x^{(2)}\right) \in G_j \cup \{y\} \\ 0 & \text{otherwise} \end{cases}$$

for every $q, q' \in Q, \sigma \in \Gamma_{G_j}$.

By Lemma 1 and Proposition 4, it suffices to show that $\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)},V_1\right)}\right\| \cdot \left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)},V_2\right)}\right\| = \sum\limits_{1 \leq j \leq m} \left\|\mathcal{A}_{G_j}\right\|$.

Let $w_1, w_2 \in \Sigma^+$, $u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)$, and $u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)$. We set $w_1 w_2 = w = w(0) \ldots w(n-1)$ hence, $w_1 = w(0) \ldots w(k)$ and $w_2 = w(k+1) \ldots w(n-1)$ for some $0 \leq k < n-1$. Furthermore, we set $u_1 u_2 = u = u(0) \ldots u(n-1)$ and, by our assumption, we have $u_1 = u(0) \ldots u(k)$ and $u_2 = u(k+1) \ldots u(n-1)$. Let $h^{(1)} \in VR\left(\Gamma^{(1)} \cup \Sigma^{(2)}\right)$, $h^{(2)} \in VR\left(\Gamma^{(2)} \cup \Sigma^{(1)}\right)$ such that $w_1 \in h^{(1)}(u_1)$ and $w_2 \in h^{(2)}(u_2)$. Moreover, let

$$P_{u_1} : \left(q^{(1)}_{in_{V_1}}, u(0), q^{(1)}_1\right) \ldots \left(q^{(1)}_{k-1}, u(k), q^{(1)}_{ter_{V_1}}\right)$$

be a path of $A^{(1)}_{\left(\Sigma^{(2)},V_1\right)}$ over u_1, and

$$P_{u_2} : \left(q^{(2)}_{in_{V_2}}, u\left(k+1\right), q^{(2)}_{k+1}\right) \ldots \left(q^{(2)}_{n-1}, u\left(n-1\right), q^{(2)}_{ter_{V_2}}\right)$$

be a path of $A^{(2)}_{\left(\Sigma^{(1)},V_2\right)}$ over u_2.

We point out the following cases.

- The sets $\{w(0),\ldots,w(k)\}\cap\left(\Sigma\setminus\left(\Sigma^{(1)}\cup\Sigma^{(2)}\right)\right)$ and $\{w(k+1),\ldots,w(n-1)\}\cap\left(\Sigma\setminus\left(\Sigma^{(1)}\cup\Sigma^{(2)}\right)\right)$ are disjoint. Then, if $weight(P_{u_1})\neq 0\neq weight(P_{u_2})$, by the definition of the list $G_1,\ldots,G_m$, there is a set $J\subseteq\{1,\ldots,m\}$ such that for every $j\in J$

  $$P^{(G_j)}_{u'} : \left(q^{(1)}_{in_{V_1}}, u'\left(0\right), q^{(1)}_{1}\right) \ldots \left(q^{(1)}_{k-1}, u'\left(k\right), q^{(2)}_{in_{V_2}}\right)\left(q^{(2)}_{in_{V_2}}, u'\left(k+1\right), q^{(2)}_{k+1}\right)$$
  $$\ldots\left(q^{(2)}_{n-1}, u'\left(n-1\right), q^{(2)}_{ter_{V_2}}\right)$$

  is a path of A_{G_j} over u', where u' is obtained by u by replacing every occurrence of $y^{(1)}$ in u_1 and $y^{(2)}$ in u_2, respectively by y. Clearly, it holds $weight\left(P^{(G_j)}_{u'}\right) = weight(P_{u_1})weight(P_{u_2})$. Moreover, since K is idempotent, we get $\sum_{j\in J} weight\left(P^{(G_j)}_{u'}\right) = weight(P_{u_1})weight(P_{u_2})$. By the definition of the list $G_1,\ldots,G_m$ we have also $weight\left(P^{(G_j)}_{u'}\right) = 0$ for every $j\in\{1,\ldots,m\}\setminus J$, and thus $\sum_{1\leq j\leq m} weight\left(P^{(G_j)}_{u'}\right) = weight(P_{u_1})weight(P_{u_2})$. Furthermore, the relabeling $h^{(j)}:\Gamma_{G_j}\to\mathcal{P}(\Sigma)$, for every $1\leq j\leq m$, which is defined as the identity on $\Sigma^{(1)}\cup\Sigma^{(2)}$, and

  - $h^{(j)}(z^{(1)}) = h^{(1)}(z^{(1)})$ for every $z^{(1)}\in Z^{(1)}\cap G_j$,
  - $h^{(j)}(z^{(2)}) = h^{(2)}(z^{(2)})$ for every $z^{(2)}\in Z^{(2)}\cap G_j$, and
  - $h^{(j)}((x^{(1)},x^{(2)}))$ are defined nondeterministically in $\Sigma\setminus(\Sigma^{(1)}\cup\Sigma^{(2)}\cup\{w(0),\ldots,w(n-1)\}\cup\{h^{(j)}(z^{(i)})\mid z^{(i)}\in Z^{(i)}\cap G_j, i=1,2\})$ whenever $(x^{(1)},x^{(2)})\in G_j\setminus\{y\}$

  is valid, and clearly $w\in h^{(j)}(u')$ for every $j\in J$.

- The sets $\{w(0),\ldots,w(k)\}\cap\left(\Sigma\setminus\left(\Sigma^{(1)}\cup\Sigma^{(2)}\right)\right)$ and $\{w(k+1),\ldots,w(n-1)\}\cap\left(\Sigma\setminus\left(\Sigma^{(1)}\cup\Sigma^{(2)}\right)\right)$ are not disjoint. For simplicity, let us assume that the two sets have only one common letter σ, and let $0\leq l_1<\ldots<l_r\leq k$ and $k+1\leq l_{r+1}<\ldots<l_s\leq n-1$ be the positions in w such that $w(l_1)=\ldots=w(l_r)=w(l_{r+1})=\ldots=w(l_s)=\sigma$.

  Since $u_1\in preim_{\Gamma^{(1)}\cup\Sigma^{(2)}}(w_1)$ and $u_2\in preim_{\Gamma^{(2)}\cup\Sigma^{(1)}}(w_2)$ we get that $u(l_1)=\ldots=u(l_r)=x^{(1)}$ and $u(l_{r+1})=\ldots=u(l_s)=x^{(2)}$ for some $x^{(1)}\in Z^{(1)}\cup\{y^{(1)}\}$ and $x^{(2)}\in Z^{(2)}\cup\{y^{(2)}\}$. If $weight(P_{u_1})\neq 0\neq weight(P_{u_2})$, by the definition of the list $G_1,\ldots,G_m$, there is a set $J\subseteq\{1,\ldots,m\}$ such that for every $j\in J$, the path $P^{(G_j)}_{u'}$ which is determined by

$$\left(q^{(1)}_{in_{V_1}}, u'(0), q^{(1)}_1\right) \ldots \left(q^{(1)}_{l_1}, (x^{(1)}, x^{(2)}), q^{(1)}_{l_1+1}\right) \ldots \left(q^{(1)}_{l_r}, (x^{(1)}, x^{(2)}), q^{(1)}_{l_r+1}\right)$$
$$\ldots \left(q^{(1)}_{k-1}, u'(k), q^{(2)}_{in_{V_2}}\right) \left(q^{(2)}_{in_{V_2}}, u'(k+1), q^{(2)}_{k+1}\right) \ldots \left(q^{(2)}_{l_{r+1}}, (x^{(1)}, x^{(2)}), q^{(2)}_{l_{r+1}+1}\right)$$
$$\ldots \left(q^{(2)}_{l_s}, (x^{(1)}, x^{(2)}), q^{(2)}_{l_s+1}\right) \ldots \left(q^{(2)}_{n-1}, u'(n-1), q^{(2)}_{ter_{V_2}}\right)$$

is a path of A_{G_j} over u', and $weight\left(P^{(G_j)}_{u'}\right) = weight(P_{u_1})weight(P_{u_2})$. The word u' is obtained by u by replacing the letters $u(l_1), \ldots, u(l_s)$ by $(x^{(1)}, x^{(2)})$ and from the remaining letters we replace every occurrence of $y^{(1)}$ and $y^{(2)}$ with y. With the same argument, as in the previous case, we obtain $\sum_{1\leq j\leq m} weight\left(P^{(G_j)}_{u'}\right) = weight(P_{u_1})weight(P_{u_2})$, and in the obvious way, we define an $h^{(j)} \in VR(\Gamma_{G_j})$ such that $w \in h^{(j)}(u')$ for every $j \in J$.

In case the sets $\{w(0), \ldots, w(k)\} \cap \left(\Sigma \setminus \Sigma^{(1)} \cup \Sigma^{(2)}\right)$ and $\{w(k+1), \ldots, w(n-1)\} \cap \left(\Sigma \setminus \Sigma^{(1)} \cup \Sigma^{(2)}\right)$ have more than one common elements, then we similarly merge the labels of the corresponding transitions and construct the paths $P^{(G_j)}_{u'}$.

Conversely, let $w \in \Sigma^+$, $u'^{(j)} \in preim_{\Gamma_{G_j}}(w)$ for some $1 \leq j \leq m$, and

$$P^{(G_j)}_{u'^{(j)}} : \left(q^{(1)}_{in_{V_1}}, u'^{(j)}(0), q^{(1)}_1\right) \ldots \left(q^{(1)}_{k-1}, u'^{(j)}(k), q^{(2)}_{in_{V_2}}\right) \left(q^{(2)}_{in_{V_2}}, u'^{(j)}(k+1), q^{(2)}_{k+1}\right)$$
$$\ldots \left(q^{(2)}_{n-1}, u'^{(j)}(n-1), q^{(2)}_{ter_{V_2}}\right)$$

be a path of A_{G_j} over $u'^{(j)}$. Then, there are two paths P_{u_1}, P_{u_2} of $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}$ and $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}$ over u_1 and u_2 respectively, where the word $u = u_1u_2$ is obtained by the word $u'^{(j)}$ as follows. For every $0 \leq t \leq k$ we replace every occurrence of $(x^{(1)}, x^{(2)})$ (resp. y) by $x^{(1)}$ (resp. $y^{(1)}$) and for every $k+1 \leq t \leq n-1$ we replace every occurrence of $(x^{(1)}, x^{(2)})$ (resp. y) by $x^{(2)}$ (resp. $y^{(2)}$). Moreover, it is not difficult to show that $u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)$, $u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)$, where $w = w_1w_2$ and $weight(P_{u_1})weight(P_{u_2}) = weight\left(P^{(G_j)}_{u'^{(j)}}\right)$.

We conclude that for every $w \in \Sigma^+$ it holds

$$\left(\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\| \cdot \left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, w\right)$$
$$= \sum \left\{ \left(\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, w_1\right) \left(\left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, w_2\right) \mid w = w_1w_2 \right\}$$
$$= \sum \left\{ \begin{array}{l} \sum\limits_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \left(\left\|A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, u_1\right) \\ \qquad\qquad \sum\limits_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \left(\left\|A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, u_2\right) \mid w = w_1w_2 \end{array} \right\}$$

$$= \sum \left\{ \begin{array}{l} \sum\limits_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \sum\limits_{P_{u_1}} weight(P_{u_1}) \\ \qquad \sum\limits_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \sum\limits_{P_{u_2}} weight(P_{u_2}) \mid w = w_1 w_2 \end{array} \right\}$$

$$= \sum \left\{ \begin{array}{l} \sum\limits_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \sum\limits_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \\ \qquad \sum\limits_{P_{u_1}} \sum\limits_{P_{u_2}} weight(P_{u_1}) weight(P_{u_2}) \mid w = w_1 w_2 \end{array} \right\}$$

$$= \sum_{1 \le j \le m} \sum_{u'^{(j)} \in preim_{\Gamma_{G_j}}(w)} \sum_{P^{(G_j)}_{u'^{(j)}}} weight\left(P^{(G_j)}_{u'^{(j)}}\right)$$

$$= \sum_{1 \le j \le m} \sum_{u'^{(j)} \in preim_{\Gamma_{G_j}}(w)} \left(\left\|A_{G_j}\right\|, u'^{(j)}\right)$$

$$= \left(\sum_{1 \le j \le m} \left\|\mathcal{A}_{G_j}\right\|, w \right)$$

where the fifth equality holds since K is idempotent. Hence, we get

$$\left\| \mathcal{A}^{(1)}_{(\Sigma^{(2)}, V_1)} \right\| \cdot \left\| \mathcal{A}^{(2)}_{(\Sigma^{(1)}, V_2)} \right\| = \sum_{1 \le j \le m} \left\|\mathcal{A}_{G_j}\right\|$$

as required, and our proof is completed. □

Proposition 8. *The class $VRec(K, \Sigma)$ is closed under the star operation applied to proper series.*

Proof. Let $r \in VRec(K, \Sigma)$ be proper and $\mathcal{A} = \langle \Sigma, A \rangle$ a normalized wva, with $A = (Q, q_{in}, wt, q_{ter})$ over $\Gamma_A = \Sigma_A \cup Z \cup \{y\}$, accepting r. We consider the wva $\mathcal{A}' = \langle \Sigma, A' \rangle$ over Σ and K with $Q' = Q \setminus \{q_{ter}\}$ and $A' = (Q', q_{in}, wt', q_{in})$ over Γ_A. The weight assignment mapping $wt' : Q' \times \Gamma_A \times Q' \to K$ is defined for every $q, q' \in Q', \sigma \in \Gamma_A$ as follows:

$$wt'((q, \sigma, q')) = \begin{cases} wt((q, \sigma, q')) & \text{if } q' \neq q_{in} \\ wt((q, \sigma, q_{ter})) & \text{if } q' = q_{in} \end{cases} .$$

By standard arguments on weighted automata, we can show that $\|A'\| = \|A\|^*$. Moreover, for every $w \in \Sigma^*$ we have

$$\begin{aligned} (\|\mathcal{A}'\|, w) &= \sum_{u \in preim_{\Gamma_A}(w)} (\|A'\|, u) \\ &= \sum_{u \in preim_{\Gamma_A}(w)} \left(\|A\|^*, u\right) \\ &= \sum_{u \in preim_{\Gamma_A}(w)} \sum_{n \ge 0, u = u_1 \ldots u_n} ((\|A\|, u_1) \cdot \ldots \cdot (\|A\|, u_n)) \end{aligned}$$

$$\begin{aligned}
&= \sum_{n\geq 0, w=w_1\ldots w_n} \sum_{\substack{u_i\in preim_{\Gamma_A}(w_i)\\ 1\leq i\leq n}} ((\|A\|, u_1)\cdot \ldots \cdot (\|A\|, u_n))\\
&= \sum_{n\geq 0, w=w_1\ldots w_n} \sum_{u_1\in preim_{\Gamma_A}(w_1)} (\|A\|, u_1)\cdot \ldots \cdot \sum_{u_n\in preim_{\Gamma_A}(w_n)} (\|A\|, u_n)\\
&= \sum_{n\geq 0, w=w_1\ldots w_n} (r, w_1)\cdot \ldots \cdot (r, w_n)\\
&= (r^*, w)
\end{aligned}$$

which implies that $\|\mathcal{A}'\| = r^*$ hence, $r^* \in VRec\,(K, \Sigma)$ and our proof is completed.

□

Proposition 9. *The class $VRec\,(K, \Sigma)$ is closed under the shuffle product.*

Proof. Let $r^{(i)} \in VRec\,(K, \Sigma)$ with $i = 1, 2$. We consider the proper series $r'^{(1)}, r'^{(2)}$ over Σ and K defined, for every $w \in \Sigma^*$, by

- $\left(r'^{(1)}, w\right) = \begin{cases} \left(r^{(1)}, w\right) & \text{if } w \in \Sigma^+ \\ 0 & \text{otherwise} \end{cases}$, and
- $\left(r'^{(2)}, w\right) = \begin{cases} \left(r^{(2)}, w\right) & \text{if } w \in \Sigma^+ \\ 0 & \text{otherwise.} \end{cases}$

Then $r^{(1)} \sqcup\!\sqcup\, r^{(2)} = r'^{(1)} \sqcup\!\sqcup\, r'^{(2)} + \left(r^{(1)}, \varepsilon\right) r^{(2)} + r^{(1)} \left(r^{(2)}, \varepsilon\right) + \left(r^{(1)}, \varepsilon\right) \left(r^{(2)}, \varepsilon\right) \bar{\varepsilon}$. By Propositions 2, 4, and 5 it suffices to show that $r'^{(1)} \sqcup\!\sqcup\, r'^{(2)} \in VRec\,(K, \Sigma)$. By Proposition 3, we can construct normalized wva $\mathcal{A}^{(i)} = \langle \Sigma, A^{(i)} \rangle$ with $A^{(i)} = \left(Q^{(i)}, q^{(i)}_{in}, wt^{(i)}, q^{(i)}_{ter}\right)$ over $\Gamma^{(i)} = \Sigma^{(i)} \cup Z^{(i)} \cup \{y^{(i)}\}$ and K, accepting respectively $r'^{(i)}$, with $i = 1, 2$. Without any loss, we assume that $Q^{(1)} \cap Q^{(2)} = \emptyset$ and $\left(Z^{(1)} \cup \{y^{(1)}\}\right) \cap \left(Z^{(2)} \cup \{y^{(2)}\}\right) = \emptyset$. We consider the wva $\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} = \left\langle \Sigma, A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} \right\rangle$ and $\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} = \left\langle \Sigma, A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} \right\rangle$ determined by the procedure before Lemma 1. By Proposition 3 and Lemma 1 these wva can be also assumed to be normalized hence, let $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} = \left(Q^{(1)}_{V_1}, q^{(1)}_{in_{V_1}}, wt^{(1)}_{V_1}, q^{(1)}_{ter_{V_1}}\right)$ over $\Gamma^{(1)} \cup \Sigma^{(2)}$ and $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} = \left(Q^{(2)}_{V_2}, q^{(2)}_{in_{V_2}}, wt^{(2)}_{V_2}, q^{(2)}_{ter_{V_2}}\right)$ over $\Gamma^{(2)} \cup \Sigma^{(1)}$. Moreover, without any loss, we assume that $Q^{(1)}_{V_1} \cap Q^{(2)}_{V_2} = \emptyset$. We let $y = \left(y^{(1)}, y^{(2)}\right)$ and consider the set $H = \left(Z^{(1)} \cup \{y^{(1)}\}\right) \times \left(Z^{(2)} \cup \{y^{(2)}\}\right) \setminus \{y\}$ and a maximal subset $G \subseteq H \cup Z^{(1)} \cup Z^{(2)}$ satisfying the following condition: every element of $Z^{(1)}$ (resp. of $Z^{(2)}$) occurs either in at most one pair of H as a left (resp. as a right) coordinate, or as a single element of G. Assume that $G_1, \ldots, G_m$ is an enumeration of all such sets. We let $Q = Q^{(1)}_{V_1} \times Q^{(2)}_{V_2}$, $\Gamma_{G_j} = \Sigma^{(1)} \cup \Sigma^{(2)} \cup G_j \cup \{y\}$, for every $1 \leq j \leq m$, and consider the normalized wva $\mathcal{A}_{G_j} = \langle \Sigma, A_{G_j} \rangle$ over Σ and K

with $A_{G_j} = \left(Q, \left(q^{(1)}_{in_{V_1}}, q^{(2)}_{in_{V_2}}\right), wt_{G_j}, \left(q^{(1)}_{ter_{V_1}}, q^{(2)}_{ter_{V_2}}\right)\right)$ over Γ_{G_j}, where the weight assignment mapping wt_{G_j} is defined for every $1 \leq j \leq m$ as follows:

$$wt_{G_j}\left(\left(\left(q^{(1)}, q^{(2)}\right), \sigma, \left(q'^{(1)}, q'^{(2)}\right)\right)\right) =$$
$$\begin{cases} wt^{(1)}_{V_1}\left(\left(q^{(1)}, \sigma, q'^{(1)}\right)\right) & \text{if } q^{(2)} = q'^{(2)} \text{ and } \sigma \in \Sigma^{(1)} \cup \Sigma^{(2)} \cup \left(Z^{(1)} \cap G_j\right) \\ wt^{(2)}_{V_2}\left(\left(q^{(2)}, \sigma, q'^{(2)}\right)\right) & \text{if } q^{(1)} = q'^{(1)} \text{ and } \sigma \in \Sigma^{(1)} \cup \Sigma^{(2)} \cup \left(Z^{(2)} \cap G_j\right) \\ wt^{(1)}_{V_1}\left(\left(q^{(1)}, x^{(1)}, q'^{(1)}\right)\right) & \text{if } q^{(2)} = q'^{(2)} \text{ and } \sigma = \left(x^{(1)}, x^{(2)}\right) \in G_j \cup \{y\} \\ wt^{(2)}_{V_2}\left(\left(q^{(2)}, x^{(2)}, q'^{(2)}\right)\right) & \text{if } q^{(1)} = q'^{(1)} \text{ and } \sigma = \left(x^{(1)}, x^{(2)}\right) \in G_j \cup \{y\} \\ 0 & \text{otherwise} \end{cases}$$

for every $\left(q^{(1)}, q^{(2)}\right), \left(q'^{(1)}, q'^{(2)}\right) \in Q, \sigma \in \Gamma_{G_j}$.
Next, we show that $\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\| ⊔ \left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\| = \sum_{1 \leq j \leq m} \|\mathcal{A}_{G_j}\|$.

For this let $w, w_1 = w_1(0) \ldots w_1(n_1 - 1), w_2 = w_2(0) \ldots w_2(n_2 - 1) \in \Sigma^+$ such that $w \in w_1 ⊔ w_2$, and $u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)$, $u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)$. Hence, there exist valid relabelings $h^{(1)} \in VR\left(\Gamma^{(1)} \cup \Sigma^{(2)}\right)$ and $h^{(2)} \in VR\left(\Gamma^{(2)} \cup \Sigma^{(1)}\right)$ such that $w_1 \in h^{(1)}(u_1), w_2 \in h^{(2)}(u_2)$. We consider a path
$P_{u_1} : \left(q^{(1)}_{in_{V_1}}, u_1(0), q^{(1)}_1\right) \ldots \left(q^{(1)}_{n_1-1}, u_1(n_1 - 1), q^{(1)}_{ter_{V_1}}\right)$
of $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}$ over u_1 and a path
$P_{u_2} : \left(q^{(2)}_{in_{V_2}}, u_2(0), q^{(2)}_1\right) \ldots \left(q^{(2)}_{n_2-1}, u_2(n_2 - 1), q^{(2)}_{ter_{V_2}}\right)$
of $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}$ over u_2. We distinguish the following cases.

- The sets $\{w_1(0), \ldots, w_1(n_1 - 1)\} \cap \left(\Sigma \setminus \left(\Sigma^{(1)} \cup \Sigma^{(2)}\right)\right)$ and $\{w_2(0), \ldots, w_2(n_2 - 1)\} \cap \left(\Sigma \setminus \left(\Sigma^{(1)} \cup \Sigma^{(2)}\right)\right)$ are disjoint. Then, if $weight(P_{u_1}) \neq 0 \neq weight(P_{u_2})$, by the definition of the list $G_1, \ldots, G_m$, there is a set $J \subseteq \{1, \ldots, m\}$ such that for every $j \in J$ there is a path $P^{(G_j)}_u$ of A_{G_j} over u, for $u \in (u_1 ⊔ u_2) \cap preim_{\Gamma_{G_j}}(w)$ with $weight\left(P^{(G_j)}_u\right) = weight(P_{u_1})weight(P_{u_2})$. Since K is idempotent it holds $\sum_{j \in J} weight\left(P^{(G_j)}_u\right) = weight(P_{u_1})weight(P_{u_2})$ and thus $\sum_{1 \leq j \leq m} weight\left(P^{(G_j)}_u\right) = weight(P_{u_1})weight(P_{u_2})$.

- The sets $\{w_1(0), \ldots, w_1(n_1 - 1)\} \cap \left(\Sigma \setminus \left(\Sigma^{(1)} \cup \Sigma^{(2)}\right)\right)$ and $\{w_2(0), \ldots, w_2(n_2 - 1)\} \cap \left(\Sigma \setminus \left(\Sigma^{(1)} \cup \Sigma^{(2)}\right)\right)$ are not disjoint. Moreover, for simplicity, we assume that the two sets have only one common letter σ, and let $0 \leq l_1 < \ldots < l_r \leq n_1 - 1$ and $0 \leq g_1 < \ldots < g_s \leq n_2 - 1$ be the positions in w_1, w_2 respectively, such that $w_1(l_1) = \ldots = w_1(l_r) = w_2(g_1) = \ldots = w_2(g_s) = \sigma$.

Since $u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)$ and $u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)$ we get that $u_1(l_1) = \ldots = u_1(l_r) = x^{(1)}$ and $u_2(g_1) = \ldots = u_2(g_s) = x^{(2)}$ for some $x^{(1)} \in Z^{(1)} \cup \{y^{(1)}\}$ and $x^{(2)} \in Z^{(2)} \cup \{y^{(2)}\}$. If $weight(P_{u_1}) \neq 0 \neq weight(P_{u_2})$, by the definition of the list $G_1, \ldots, G_m$, there is a set $J \subseteq \{1, \ldots, m\}$ such that for every $j \in J$ there is a path $P_{u'}^{(G_j)}$of A_{G_j} over u', where u' is obtained by u by replacing $x^{(1)}$ (resp. $x^{(2)}$) in u_1 (resp. u_2) at the positions $l_1, \ldots, l_r$ (resp. $g_1, \ldots, g_s$) by the pair $\left(x^{(1)}, x^{(2)}\right)$, and from the remaining letters we replace every occurrence of $y^{(1)}$ and $y^{(2)}$ with y, for $u \in u_1 \sqcup\!\sqcup\, u_2$. Again, we have $weight\left(P_{u'}^{(G_j)}\right) = weight(P_{u_1}) weight(P_{u_2})$ and hence, $\sum_{1 \leq j \leq m} weight\left(P_{u'}^{(G_j)}\right) = weight(P_{u_1}) weight(P_{u_2})$. On the other hand, it is trivially shown that $u' \in preim_{\Gamma_{G_j}}(w)$.

Conversely, keeping the previous notations, for every $w \in \Sigma^+$, $u' \in preim_{\Gamma_{G_j}}(w)$ for some $1 \leq j \leq m$, there are $u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1), u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)$ with $w \in w_1 \sqcup\!\sqcup\, w_2$, such that for every path $P_{u'}^{(G_j)}$of A_{G_j} over u', there are paths P_{u_1} of $A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}$ over u_1 and P_{u_2} of $A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}$ over u_2, with $weight\left(P_{u'}^{(G_j)}\right) = weight(P_{u_1}) weight(P_{u_2})$. Using the same as above argument, we can show that $\sum_{1 \leq j \leq m} weight\left(P_{u'}^{(G_j)}\right) = weight(P_{u_1}) weight(P_{u_2})$.

Now for every $w_1, w_2 \in \Sigma^+$, it holds

$$\begin{aligned}
&\left(\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, w_1\right)\left(\left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, w_2\right) \\
&= \sum_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \left(\left\|A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, u_1\right) \sum_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \left(\left\|A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, u_2\right) \\
&= \sum_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \sum_{P_{u_1}} weight(P_{u_1}) \sum_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \sum_{P_{u_2}} weight(P_{u_2}) \\
&= \sum_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \sum_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \sum_{P_{u_1}} \sum_{P_{u_2}} weight(P_{u_1}) weight(P_{u_2}).
\end{aligned}$$

Hence, for every $w \in \Sigma^+$, we get

$$\begin{aligned}
&\left(\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\| \sqcup\!\sqcup \left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, w\right) \\
&= \sum_{\substack{w_1, w_2 \in \Sigma^+ \\ w \in w_1 \sqcup\!\sqcup\, w_2}} \left(\left(\left\|\mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, w_1\right)\left(\left\|\mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)}\right\|, w_2\right)\right) \\
&= \sum_{\substack{w_1, w_2 \in \Sigma^+ \\ w \in w_1 \sqcup\!\sqcup\, w_2}} \sum_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \left(\left\|A^{(1)}_{\left(\Sigma^{(2)}, V_1\right)}\right\|, u_1\right)
\end{aligned}$$

$$\sum_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \left(\left\| A^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} \right\|, u_2 \right)$$

$$= \sum_{\substack{w_1, w_2 \in \Sigma^+ \\ w \in w_1 \sqcup\!\sqcup w_2}} \sum_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \sum_{P_{u_1}} weight(P_{u_1})$$

$$\sum_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)} \sum_{P_{u_2}} weight(P_{u_2})$$

$$= \sum_{\substack{w_1, w_2 \in \Sigma^+ \\ w \in w_1 \sqcup\!\sqcup w_2}} \sum_{u_1 \in preim_{\Gamma^{(1)} \cup \Sigma^{(2)}}(w_1)} \sum_{u_2 \in preim_{\Gamma^{(2)} \cup \Sigma^{(1)}}(w_2)}$$

$$\sum_{P_{u_1}} \sum_{P_{u_2}} weight(P_{u_1}) weight(P_{u_2})$$

$$= \sum_{1 \le j \le m} \sum_{u \in preim_{\Gamma_{G_j}}(w)} \sum_{P_u^{(G_j)}} weight \left(P_u^{(G_j)} \right)$$

$$= \left(\sum_{1 \le j \le m} \left\| \mathcal{A}_{G_j} \right\|, w \right)$$

which implies that $\left\| \mathcal{A}^{(1)}_{\left(\Sigma^{(2)}, V_1\right)} \right\| \sqcup\!\sqcup \left\| \mathcal{A}^{(2)}_{\left(\Sigma^{(1)}, V_2\right)} \right\| = \sum_{1 \le j \le m} \left\| \mathcal{A}_{G_j} \right\|$ hence, $r'^{(1)} \sqcup\!\sqcup r'^{(2)} = \sum_{1 \le j \le m} \left\| \mathcal{A}_{G_j} \right\|$. Therefore, by Proposition 4, we conclude that $r'^{(1)} \sqcup\!\sqcup r'^{(2)} \in VRec(K, \Sigma)$, as required. $\square$

Remark 1. The definition of the weight assignment mappings wt_{G_j} for $1 \le j \le m$, in the above proof, is not completely right. More precisely, it may happen that there are $q^{(1)} \in Q^{(1)}_{V_1}, q^{(2)} \in Q^{(2)}_{V_2}$ and $\sigma \in \left(\Sigma^{(1)} \cap \Sigma^{(2)}\right)$ (resp. $\sigma = (x^{(1)}, x^{(2)}) \in G_j \cup \{y\}$) such that $wt^{(1)}_{V_1}\left(\left(q^{(1)}, \sigma, q^{(1)}\right)\right) = k_1 \ne 0$, $wt^{(2)}_{V_2}\left(\left(q^{(2)}, \sigma, q^{(2)}\right)\right) = k_2 \ne 0$ with $k_1 \ne k_2$ (resp. $wt^{(1)}_{V_1}((q^{(1)}, x^{(1)}, q^{(1)})) = k_1 \ne 0$, $wt^{(2)}_{V_2}\left(\left(q^{(2)}, x^{(2)}, q^{(2)}\right)\right) = k_2 \ne 0$ with $k_1 \ne k_2$). Then, the value $wt_{G_j}\left(\left(\left(q^{(1)}, q^{(2)}\right), \sigma, \left(q^{(1)}, q^{(2)}\right)\right)\right)$ is not well-defined[3]. Therefore, if this is the case, for every such pair of states $\left(q^{(1)}, q^{(2)}\right)$ and $\sigma \in \left(\Sigma^{(1)} \cap \Sigma^{(2)}\right) \cup G_j \cup \{y\}$, we introduce a new state r and we set

- $wt_{G_j}\left(\left(\left(q^{(1)}, q^{(2)}\right), \sigma, \left(q^{(1)}, q^{(2)}\right)\right)\right) = k_1$,
- $wt_{G_j}\left(\left(\left(q^{(1)}, q^{(2)}\right), \sigma, r\right)\right) = k_1$,
- $wt_{G_j}\left(\left(r, \sigma, \left(q^{(1)}, q^{(2)}\right)\right)\right) = k_2$,

[3]Other authors proving the closure of the class of recognizable series under the shuffle product have also ignored this case (cf. for instance [34]).

- $wt_{G_j}\left((r,\sigma,r)\right) = k_2$, and
- $wt_{G_j}\left(\left(\left(q'^{(1)},q'^{(2)}\right),\sigma',r\right)\right) = wt_{G_j}\left(\left(\left(q'^{(1)},q'^{(2)}\right),\sigma',\left(q^{(1)},q^{(2)}\right)\right)\right)$,
- $wt_{G_j}\left(\left(r,\sigma',\left(q'^{(1)},q'^{(2)}\right)\right)\right) = wt_{G_j}\left(\left(\left(q^{(1)},q^{(2)}\right),\sigma',\left(q'^{(1)},q'^{(2)}\right)\right)\right)$,

for every $q'^{(1)} \in Q^{(1)}_{V_1}, q'^{(2)} \in Q^{(2)}_{V_2}, \sigma' \in \Gamma_{G_j}$.

The following theorem summarizes the results of this section.

Theorem 1. *Let K be a commutative and idempotent semiring and Σ an infinite alphabet. Then the class of v-recognizable series over Σ and K is closed under sum, and under scalar, Hadamard, Cauchy and shuffle products, and under star operation applied to proper series.*

5 Rational series over infinite alphabets

In this section, we extend the notion of rational series over the infinite alphabet Σ and the semiring K[4]. In fact, we state a Kleene-Schützenberger type result for v-recognizable series over Σ and K. For this, we define the notion of rationality for series over Σ in the same way we did it for v-recognizable series. A similar approach for defining regular expressions over infinite alphabets has been followed in [1, 25]. Firstly, we recall the concept of rational series over finite alphabets. Let Δ be a finite alphabet. The class $Rat(K,\Delta)$ of rational series over Δ and K is the least class of series containing the polynomials over Δ and K and being closed under sum, Cauchy product, and star operation applied to proper series. The subsequent result is the fundamental theorem of Schützenberger stating the coincidence of rational and recognizable series.

Theorem 2. *[35, 18, 34] Let K be a semiring and Δ a finite alphabet. Then a series $s \in K\langle\langle\Delta^*\rangle\rangle$ is recognizable iff it is rational.*

Definition 3. *A series s over Σ and K is called* v-rational *if there is a subalphabet $\Gamma \subseteq_{fin} \Sigma$ and a rational series s' over $\Delta = \Gamma \cup Z \cup \{y\}$ and K such that*

$$(s,w) = \sum_{u \in preim_\Delta(w)} (s',u)$$

for every $w \in \Sigma^*$.

Now we discuss why we adopted the above definition for rational series over infinite alphabets. One could think of alternative definitions, more precisely, by defining rational series over the infinite alphabet Σ in the same way we do it for rational series over finite alphabets. It is not difficult to see that such a consideration should

[4]In this section we can relax the commutativity property of K.

not derive an expressively equivalent notion to wva. Consider for instance the normalized wva $\mathcal{A} = \langle \Sigma, A \rangle$ where $A = (\{q_{in}, q_{ter}\}, q_{in}, wt, q_{ter})$ with $\Sigma_A = \{a\}$ and $Z = \{z\}$. The only non-zero assignment of wt is given by $wt((q_{in}, z, q_{ter})) = k \neq 0$. Then trivially, $\|\mathcal{A}\| = \sum_{a' \in \Sigma \setminus \{a\}} ka'$ and it is not difficult to see that this series is not rational in the sense of rational series over finite alphabets. Even if we should consider our rational series to contain, by definition, series of the above form, then still this is not sufficient. For instance let us consider the normalized wva $\mathcal{B} = \langle \Sigma, B \rangle$ where $B = (\{p_{in}, p, p_{ter}\}, p_{in}, wt, p_{ter})$ with $\Sigma_B = \{b\}$, $Z = \{z, z'\}$ and non-zero weights $wt((p_{in}, z, p)) = k, wt((p, z', p_{ter})) = k'$. Then it is easily obtained that

$$\|\mathcal{B}\| = \sum_{\substack{a, a' \in \Sigma \setminus \{b\} \\ a \neq a'}} kk'aa'.$$

On the other hand, the Cauchy product of the series $\sum\limits_{a \in \Sigma \setminus \{b\}} ka \sum\limits_{a' \in \Sigma \setminus \{b\}} k'a'$ clearly differs from $\|\mathcal{B}\|$.

Next, we state our Kleene-Schützenberger type theorem for series over Σ and K.

Theorem 3. *Let K be a semiring and Σ an infinite alphabet. Then a series $s \in K \langle\langle \Sigma^* \rangle\rangle$ is v-recognizable iff it is v-rational.*

Proof. Let $s \in VRec(K, \Sigma)$. Then, there exists a wva $\mathcal{A} = \langle \Sigma, A \rangle$ where A is a weighted automaton over $\Gamma_A = \Sigma_A \cup Z \cup \{y\}$ such that

$$s = \|\mathcal{A}\| = \sum_{u \in preim_{\Gamma_A}(w)} (\|A\|, u).$$

By Theorem 2 the recognizable series $\|A\|$ over Γ_A and K is also rational. This implies that s is v-rational. By a similar argument, we show that if s is v-rational, then it is also v-recognizable, and this concludes our proof. □

6 Weighted monadic second order logic over infinite alphabets

Droste and Gastin in their seminal paper [12] (cf. also [13]), introduced a weighted monadic second order logic (*MSO* logic for short) and proved in the quantitative setup the fundamental result of Büchi [7], Elgot [17], and Trakhtenbrot [41] relating recognizable and *MSO*-definable languages. More precisely, they determined two fragments of their weighted *MSO* logic, namely the restricted, and the existential restricted one, and proved that the classes of series defined by sentences in these two fragments coincide with the class of recognizable series over a finite alphabet and a commutative semiring. We would like to extend this result for the class of v-recognizable series. For this, we introduce a weighted *MSO* logic over the infinite alphabet Σ and the commutative semiring K.

Firstly we recall, for the reader's convenience, the basic definitions of weighted *MSO* logic (cf. [12, 13]) by adopting the notations of [19].

Let Δ be a finite alphabet. The syntax of *MSO* logic formulas over Δ is given by the grammar

$$\phi ::= true \mid P_a(x) \mid x \leq x' \mid x \in X \mid \neg\phi \mid \phi \vee \phi \mid \exists x \,\boldsymbol{.}\, \phi \mid \exists X \,\boldsymbol{.}\, \phi$$

where $a \in \Delta$ and we let $false = \neg true$. The set $free(\phi)$ of free variables of an *MSO* logic formula ϕ is defined as usual. In order to define the semantics of *MSO* logic formulas we need the notions of the extended alphabet and valid assignment (cf. for instance [40]). Let $\mathcal{V}$ be a finite set of first and second order variables. For every word $u = u(0) \dots u(n-1) \in \Delta^*$ we let $Dom(u) = \{0, \dots, n-1\}$. A $(\mathcal{V}, u)$-*assignment* σ is a mapping associating first order variables from $\mathcal{V}$ to elements of $Dom(u)$, and second order variables from $\mathcal{V}$ to subsets of $Dom(u)$. If x is a first order variable and $i \in Dom(u)$, then $\sigma[x \to i]$ denotes the $(\mathcal{V} \cup \{x\}, u)$-assignment which associates i to x and coincides with σ on $\mathcal{V} \setminus \{x\}$. For a second order variable X and $I \subseteq Dom(u)$, the notation $\sigma[X \to I]$ has a similar meaning.

We shall encode pairs of the form (u, σ), where $u \in \Delta^*$ and σ is a $(\mathcal{V}, u)$-assignment, using the extended alphabet $\Delta_{\mathcal{V}} = \Delta \times \{0,1\}^{\mathcal{V}}$. Indeed, every word in $\Delta_{\mathcal{V}}^*$ can be considered as a pair (u, σ) where u is the projection over Δ and σ is the projection over $\{0,1\}^{\mathcal{V}}$. Then σ is a valid assignment if for every first order variable $x \in \mathcal{V}$ the x-row contains exactly one 1. In this case, σ is the $(\mathcal{V}, u)$-assignment such that for every first order variable $x \in \mathcal{V}$, $\sigma(x)$ is the position of the 1 on the x-row, and for every second order variable $X \in \mathcal{V}$, $\sigma(X)$ is the set of positions labelled with 1 along the X-row. It is not difficult to see that the language

$$N_{\mathcal{V}} = \{(u, \sigma) \in \Delta_{\mathcal{V}}^* \mid \sigma \text{ is a valid } (\mathcal{V}, u)\text{-assignment}\}$$

is recognizable.

For every $(u, \sigma) \in \mathcal{N}_{\mathcal{V}}$ we define the satisfaction relation $(u, \sigma) \models \phi$ by induction on the structure of ϕ, as follows:

- $(u, \sigma) \models true$,
- $(u, \sigma) \models P_a(x)$ iff $u(\sigma(x)) = a$,
- $(u, \sigma) \models x \leq x'$ iff $\sigma(x) \leq \sigma(x')$,
- $(u, \sigma) \models x \in X$ iff $\sigma(x) \in \sigma(X)$,
- $(u, \sigma) \models \neg\phi$ iff $(u, \sigma) \not\models \phi$,
- $(u, \sigma) \models \phi \vee \phi'$ iff $(u, \sigma) \models \phi$ or $(u, \sigma) \models \phi'$,
- $(u, \sigma) \models \exists x \,\boldsymbol{.}\, \phi$ iff there exists an $i \in Dom(u)$ such that $(u, \sigma[x \to i) \models \phi$,
- $(u, \sigma) \models \exists X \,\boldsymbol{.}\, \phi$ iff there exists an $I \subseteq Dom(u)$ such that $(u, \sigma[X \to I) \models \phi$.

If $(u, \sigma) \in \Delta_{\mathcal{V}}^* \setminus \mathcal{N}_{\mathcal{V}}$, then we let $(u, \sigma) \not\models \phi$.

Definition 4. *The syntax of formulas of the* weighted *MSO* logic over Δ and K *is given by the grammar*

$$\begin{aligned}\phi &::= true \mid P_a(x) \mid x \leq x' \mid x \in X \mid \neg\phi \mid \phi \vee \phi \mid \exists x \centerdot \phi \mid \exists X \centerdot \phi\\ \varphi &::= k \mid \phi \mid \varphi \oplus \varphi \mid \varphi \otimes \varphi \mid \bigoplus_x \centerdot \varphi \mid \bigoplus_X \centerdot \varphi \mid \bigotimes_x \centerdot \varphi \mid \bigotimes_X \centerdot \varphi\end{aligned}$$

where $k \in K$, $a \in \Delta$.

We denote by $MSO(K,\Delta)$ the set of all weighted *MSO* logic formulas φ over Δ and K. We represent the semantics of formulas $\varphi \in MSO(K,\Delta)$ as series $\|\varphi\| \in K\langle\langle\Delta^*\rangle\rangle$. For the semantics of *MSO* logic formulas ϕ we use the satisfaction relation as defined above. Therefore, the semantics of *MSO* logic formulas ϕ gets only the values 0 and 1.

Definition 5. *Let* $\varphi \in MSO(K,\Delta)$ *and* $\mathcal{V}$ *be a finite set of variables with* $free(\varphi) \subseteq \mathcal{V}$. *The semantics of* φ *is a series* $\|\varphi\|_{\mathcal{V}} \in K\langle\langle\Delta^*_{\mathcal{V}}\rangle\rangle$. *Consider an element* $(u,\sigma) \in \Delta^*_{\mathcal{V}}$. *If* $(u,\sigma) \notin N_{\mathcal{V}}$, *then we let* $(\|\varphi\|_{\mathcal{V}}, (u,\sigma)) = 0$. *Otherwise, we define* $(\|\varphi\|_{\mathcal{V}}, (u,\sigma)) \in K$, *inductively on the structure of* φ, *as follows:*

- $(\|k\|_{\mathcal{V}}, (u,\sigma)) = k$,
- $(\|\phi\|_{\mathcal{V}}, (u,\sigma)) = \begin{cases} 1 & \text{if } (u,\sigma) \models \phi \\ 0 & \text{otherwise} \end{cases}$,
- $(\|\varphi \oplus \psi\|_{\mathcal{V}}, (u,\sigma)) = (\|\varphi\|_{\mathcal{V}}, (u,\sigma)) + (\|\psi\|_{\mathcal{V}}, (u,\sigma))$,
- $(\|\varphi \otimes \psi\|_{\mathcal{V}}, (u,\sigma)) = (\|\varphi\|_{\mathcal{V}}, (u,\sigma)) \cdot (\|\psi\|_{\mathcal{V}}, (u,\sigma))$,
- $\left(\|\bigoplus_x \centerdot \varphi\|_{\mathcal{V}}, (u,\sigma)\right) = \sum_{0 \leq i \leq n-1} \left(\|\varphi\|_{\mathcal{V}\cup\{x\}}, (u,\sigma[x \to i])\right)$,
- $\left(\|\bigoplus_X \centerdot \varphi\|_{\mathcal{V}}, (u,\sigma)\right) = \sum_{I \subseteq Dom(u)} \left(\|\varphi\|_{\mathcal{V}\cup\{X\}}, (u,\sigma[X \to I])\right)$,
- $\left(\|\bigotimes_x \centerdot \varphi\|_{\mathcal{V}}, (u,\sigma)\right) = \prod_{0 \leq i \leq n-1} \left(\|\varphi\|_{\mathcal{V}\cup\{x\}}, (u,\sigma[x \to i])\right)$,
- $\left(\|\bigotimes_X \centerdot \varphi\|_{\mathcal{V}}, (u,\sigma)\right) = \prod_{I \subseteq Dom(u)} \left(\|\varphi\|_{\mathcal{V}\cup\{X\}}, (u,\sigma[X \to I])\right)$.

We simply denote $\|\varphi\|_{free(\varphi)}$ by $\|\varphi\|$. If φ is a sentence, then $\|\varphi\| \in K\langle\langle\Delta^*\rangle\rangle$. Furthermore, it holds [12]

$$(\|\varphi\|_{\mathcal{V}}, (u,\sigma)) = \left(\|\varphi\|, (u,\sigma|_{free(\varphi)})\right)$$

for every $(u,\sigma) \in N_{\mathcal{V}}$.

Definition 6. *A formula* $\varphi \in MSO(K,\Delta)$ *will be called* restricted *if*

- *it contains no universal quantification of the form $\bigotimes_X . \psi$, and*
- *whenever it contains a universal first order quantification $\bigotimes_x . \psi$, then ψ is a formula of the form $\psi = \oplus_{1 \le i \le n} (k_i \otimes \phi_i)$ where $k_i \in K$ and ϕ_i is an MSO logic formula for every $1 \le i \le n$.*

We denote with $RMSO(K, \Delta)$ the subclass of all restricted formulas in $MSO(K, \Delta)$.

Definition 7. *A formula $\varphi \in RMSO(K, \Delta)$ is called* restricted existential *if it is of the form $\bigoplus_{X_1,\dots,X_n} . \psi$ with $\psi \in RMSO(K, \Delta)$ and ψ contains no set quantification.*

The subclass of all restricted existential formulas in $MSO(K, \Delta)$ is denoted by $REMSO(K, \Delta)$.

Definition 8. *A series s over Δ and K is called MSO (resp. RMSO, REMSO)* definable *if there is sentence φ in $MSO(K, \Delta)$ (resp. $RMSO(K, \Delta)$, $REMSO(K, \Delta)$) such that $s = \|\varphi\|$.*

Theorem 4. *[12, 13] Let K be a commutative semiring and Δ a finite alphabet. Then for every series $s \in K \langle\langle \Delta^* \rangle\rangle$ the following statements are equivalent.*

i) s is recognizable.

ii) s is RMSO-definable.

iii) s is REMSO-definable.

Furthermore, if the semiring K is locally finite, then the above statements are also equivalent to

iv) s is MSO-definable.

Now we are ready to introduce the *MSO* logic characterization for series over the infinite alphabet Σ and the commutative semiring K.

Definition 9. *A series s over Σ and K is called VMSO (resp. VRMSO, VREMSO)* definable *if there is a subalphabet $\Gamma \subseteq_{fin} \Sigma$ and an MSO (resp. RMSO, REMSO) definable series s' over $\Delta = \Gamma \cup Z \cup \{y\}$ and K such that*

$$(s, w) = \sum_{u \in preim_\Delta(w)} (s', u)$$

for every $w \in \Sigma^*$.

Theorem 5. *Let K be a commutative semiring and Σ an infinite alphabet. Then for every series $s \in K \langle\langle \Sigma^* \rangle\rangle$ the following statements are equivalent.*

i) s is v-recognizable.

ii) s is VRMSO-definable.

iii) s is VREMSO-definable.

Furthermore, if the semiring K is locally finite, then the above statements are also equivalent to

iv) s is VMSO-definable.

Proof. We obtain our result by Theorem 4 and Definition 9, using similar arguments as the ones in the proof of Theorem 3. □

7 Weighted linear dynamic logic over infinite alphabets

Vardi in 2011 introduced a linear dynamic logic (*LDL* for short) over infinite words and stated the expressive equivalence of *LDL* formulas to ω-rational expressions (cf. [42]). *LDL* is a combination of propositional dynamic logic and of classical *LTL*. In [20] the authors proved the coincidence of the classes of rational and *LDL*-definable languages interpreted over finite words. Recently, *LDL* has been investigated in the quantitative setup for both finite and infinite words [16]. More precisely, the authors proved the expressive equivalence of weighted *LDL* formulas to weighted automata for finite words over commutative semirings, and for infinite words over totally commutative complete semirings. In this section, we introduce a weighted *LDL* over the infinite alphabet Σ and the commutative semiring K, and we show the expressive equivalence of weighted *LDL* formulas to weighted variable automata.

Let us firstly recall the basic definitions for weighted *LDL* logic over finite alphabets [16]. Let Δ be a finite alphabet. For every letter $a \in \Delta$ we consider an atomic proposition p_a, and we let $P = \{p_a \mid a \in \Delta\}$. Moreover, for every $p \in P$ we identify $\neg\neg p$ with p.

Definition 10. *The syntax of LDL formulas ψ* over Δ *is given by the grammar*

$$\psi ::= true \mid p_a \mid \neg\psi \mid \psi \wedge \psi \mid \langle\theta\rangle\,\psi$$
$$\theta ::= \phi \mid \psi? \mid \theta + \theta \mid \theta;\theta \mid \theta^{+}$$

where $p_a \in P$ and ϕ denotes a propositional formula over the atomic propositions in P.

For every *LDL* formula ψ and $u \in \Delta^*$ we define the satisfaction relation $u \models \psi$, inductively on the structure of ψ, as follows:

- $u \models true$,
- $u \models p_a$ iff $u(0) = a$,
- $u \models \neg\psi$ iff $u \not\models \psi$,
- $u \models \psi_1 \wedge \psi_2$ iff $u \models \psi_1$ and $u \models \psi_2$,

- $u \models \langle \phi \rangle \psi$ iff $u \models \phi$ and $u_{\geq 1} \models \psi$,
- $u \models \langle \psi_1 ? \rangle \psi_2$ iff $u \models \psi_1$ and $u \models \psi_2$,
- $u \models \langle \theta_1 + \theta_2 \rangle \psi$ iff $u \models \langle \theta_1 \rangle \psi$ or $u \models \langle \theta_2 \rangle \psi$,
- $u \models \langle \theta_1 ; \theta_2 \rangle \psi$ iff $u = vv'$, $v \models \langle \theta_1 \rangle true$, and $v' \models \langle \theta_2 \rangle \psi$,
- $u \models \langle \theta^+ \rangle \psi$ iff there exists n with $1 \leq n \leq |u|$ such that $u \models \langle \theta^n \rangle \psi$,

where θ^n, $n \geq 1$ is defined inductively by $\theta^1 = \theta$ and $\theta^n = \theta^{n-1}; \theta$ for $n > 1$.

Definition 11. *The syntax of formulas φ of the* weighted *LDL* over Δ and K *is given by the grammar*

$$\varphi ::= k \mid \psi \mid \varphi \oplus \varphi \mid \varphi \otimes \varphi \mid \langle \rho \rangle \varphi$$
$$\rho ::= \phi \mid \varphi ? \mid \rho \oplus \rho \mid \rho \cdot \rho \mid \rho^{\oplus}$$

where $k \in K$, ϕ denotes a propositional formula over the atomic propositions in P, and ψ denotes an LDL formula as in Definition 10.

We denote by $LDL(K, \Delta)$ the set of all weighted LDL formulas φ over Δ and K. We represent the semantics $\|\varphi\|$ of the formulas $\varphi \in LDL(K, \Delta)$ as series over Δ and K.

Definition 12. *Let $\varphi \in LDL(K, \Delta)$. The* semantics *of φ is a series $\|\varphi\| \in K \langle\langle \Delta^* \rangle\rangle$. For every $u \in \Delta^*$ the value $(\|\varphi\|, u)$ is defined inductively as follows:*

- $(\|k\|, u) = k$,
- $(\|\psi\|, u) = \begin{cases} 1 & \text{if } u \models \psi \\ 0 & \text{otherwise} \end{cases}$,
- $(\|\varphi_1 \oplus \varphi_2\|, u) = (\|\varphi_1\|, u) + (\|\varphi_2\|, u)$,
- $(\|\varphi_1 \otimes \varphi_2\|, u) = (\|\varphi_1\|, u) \cdot (\|\varphi_2\|, u)$,
- $(\|\langle \phi \rangle \varphi\|, u) = (\|\phi\|, u) \cdot (\|\varphi\|, u_{\geq 1})$,
- $(\|\langle \varphi_1 ? \rangle \varphi_2\|, u) = (\|\varphi_1\|, u) \cdot (\|\varphi_2\|, u)$,
- $(\|\langle \rho_1 \oplus \rho_2 \rangle \varphi\|, u) = (\|\langle \rho_1 \rangle \varphi\|, u) + (\|\langle \rho_2 \rangle \varphi\|, u)$,
- $(\|\langle \rho_1 \cdot \rho_2 \rangle \varphi\|, u) = \sum\limits_{\substack{v, v' \in \Delta^* \\ u = vv'}} \left((\|\langle \rho_1 \rangle true\|, v) \cdot (\|\langle \rho_2 \rangle \varphi\|, v') \right)$,
- $(\|\langle \rho^{\oplus} \rangle \varphi\|, u) = \sum\limits_{n \geq 1} (\|\langle \rho^n \rangle \varphi\|, u)$

where for the definition of $(\|\langle \rho^{\oplus} \rangle \varphi\|, u)$ we assume that $\|\langle \rho \rangle true\|$ is proper, and ρ^n, $n \geq 1$ is defined inductively by $\rho^1 = \rho$ and $\rho^n = \rho^{n-1} \cdot \rho$ for $n > 1$.

A series $s \in K\langle\langle \Delta^* \rangle\rangle$ is called *LDL-definable* if there is a formula $\varphi \in LDL(K, \Delta)$ such that $s = \|\varphi\|$.

Theorem 6. *[16] Let K be a commutative semiring and Δ a finite alphabet. A series $s \in K\langle\langle \Delta^* \rangle\rangle$ is recognizable iff it is LDL-definable.*

Now we are ready to introduce the *LDL* characterization for series over the infinite alphabet Σ and the semiring K.

Definition 13. *A series s over Σ and K is called VLDL-definable if there is a subalphabet $\Gamma \subseteq_{fin} \Sigma$ and an LDL-definable series s' over $\Delta = \Gamma \cup Z \cup \{y\}$ and K such that*

$$(s, w) = \sum_{u \in preim_{\Delta}(w)} (s', u)$$

for every $w \in \Sigma^*$.

By Theorem 6 and Definition 13, using similar arguments as the ones in the proof of Theorem 3, we obtain the subsequent result.

Theorem 7. *Let K be a commutative semiring and Σ an infinite alphabet. Then a series $s \in K\langle\langle \Sigma^* \rangle\rangle$ is v-recognizable iff it is VLDL-definable.*

8 Application to variable finite automata

In this section, we derive new results for the class of languages accepted by variable finite automata (vfa for short) over the infinite alphabet Σ (cf. [21, 22]). We need first to recall the definition of vfa from [21, 22] but we follow the terminology used previously for wva. Let Z be a finite set of bounded variables and y a free variable. Then, a variable finite automaton over Σ is a pair $\mathcal{A} = \langle \Sigma, A \rangle$ where $A = (Q, \Gamma_A, I, E, F)$ is a finite automaton with input alphabet $\Gamma_A = \Sigma_A \cup Z \cup \{y\}$ where $\Sigma_A \subseteq_{fin} \Sigma$. The language of $\mathcal{A}$ is defined by

$$L(\mathcal{A}) = \bigcup_{\substack{u \in L(A) \\ h \in VR(\Gamma_A)}} h(u).$$

Then the vfa $\mathcal{A} = \langle \Sigma, A \rangle$ can be considered, in the obvious way, as a wva $\mathcal{A}'$ over the Boolean semiring $\mathbb{B}$. Moreover, it holds

$$w \in L(\mathcal{A}) \quad \text{iff} \quad (\|\mathcal{A}'\|, w) = 1$$

for every $w \in \Sigma^*$.

A language $L \subseteq \Sigma^*$ is called *recognizable* if there is a vfa $\mathcal{A} = \langle \Sigma, A \rangle$ such that $L = L(\mathcal{A})$.

Theorem 8. *Let Σ be an infinite alphabet. The class of recognizable languages over Σ is closed under union, intersection, concatenation, Kleene star, and shuffle product.*[5]

[5]The closure under union and intersection has been also proved in [21, 22].

Proof. In Theorem 1 we let $K = \mathbb{B}$. Then for every $L \subseteq \Sigma^*$, clearly L is recognizable iff its characteristic series $1_L \in \mathbb{B}\langle\langle\Sigma^*\rangle\rangle$ is v-recognizable. We conclude our result by the idempotency property of $\mathbb{B}$. □

Next we define the notions of rational (resp. *MSO*-definable, *LDL*-definable) languages over the infinite alphabet Σ.

Definition 14. *Let Σ be an infinite alphabet. A language L over Σ is called* rational *(resp. MSO*-definable*, LDL*-definable*) if there is a subalphabet $\Gamma \subseteq_{fin} \Sigma$ and a rational (resp. MSO-definable, LDL-definable) language L' over $\Delta = \Gamma \cup Z \cup \{y\}$ such that*

$$L = \bigcup_{\substack{u \in L' \\ h \in VR(\Delta)}} h(u).$$

The next theorem establishes new characterizations for the class of languages accepted by vfa.

Theorem 9. *Let Σ be an infinite alphabet and $L \subseteq \Sigma^*$. Then the following statements are equivalent.*

i) L is recognizable.

ii) L is rational.

iii) L is MSO-definable.

iv) L is LDL-definable.

Proof. We take into account the definition of recognizable languages over the infinite alphabet Σ and Definition 14. Then we obtain the equivalence $(i) \iff (ii)$ by Kleene's theorem, the equivalence $(i) \iff (iii)$ by Büchi's theorem, and the equivalence $(i) \iff (iv)$ by [20]. □

Conclusion

We introduced weighted variable automata over an infinite alphabet Σ and a commutative semiring K. Our model is based on the variable automaton model of [21, 22] but we followed the terminology used in [26, 27] for variable tree automata over infinite ranked alphabets. Indeed, that terminology presents in a strict mathematical way the operation of (weighted) variable automata models. With the additional assumption that K is idempotent we proved the closure of the class of series accepted by our models under the operations of sum, and scalar, Hadamard, Cauchy, and shuffle products, as well as star operation applied to proper series. We introduced the notion of rational series over the infinite alphabet Σ and an arbitrary semiring K and stated a Kleene-Schützenberger type theorem. We defined a weighted *MSO* logic over the infinite alphabet Σ and the commutative semiring K

and proved a Büchi type theorem, extending the work of Droste and Gastin [12, 13] to series over infinite alphabets. Finally, we considered a weighted *LDL* over the infinite alphabet Σ and the commutative semiring K and proved the expressive equivalence of its formulas to weighted variable automata.

It is an open problem whether we can relax the idempotency property of the semiring K for the closure properties of the class $VRec(K, \Sigma)$. Furthermore, it should be very interesting to study the weighted variable automata theory over more general weight structures, contributing to real world applications, like for instance over valuation monoids [15].

References

[1] P. Barceló, J. Reutter, L. Libkin, Parameterized regular expressions and their languages, *Theoret. Comput. Sci.* 474(2013) 21–45.

[2] W. Belkhir, Y. Chevalier, M. Rusinowitch, Fresh-variable automata: Application to service composition, in: *Proceedings of SYNASC 2013*, pp.153–160.

[3] W. Belkhir, Y. Chevalier, M. Rusinowitch, Guarded variable automata over infinite alphabets, *CoRR* abs/1304.6297, 2013.

[4] M. Benedikt, C. Ley, G. Puppis, Automata vs. logics on data words, in: *Proceedings of CSL 2010, LNCS* 6247(2010) 110–124.

[5] M. Bojańczyk, C. David, A. Muscholl, T. Schwentick, L. Segoufin, Two-variable logic on data words, *ACM Trans. Comput. Log.* 12(4)(2011) 27.

[6] B. Bollig, P. Habermehl, M. Leucker, B. Monmege, A robust class of data languages and an application to learning, *Log. Methods in Comput. Sci.* 10(4)(2014) 19.

[7] J.R. Büchi, Weak second-order arithmetic and finite automata, *Z. Math. Logik Grundlager Math.* 6(1960) 66–92.

[8] P. Černý, S. Gopi, T.A. Henzinger, A. Radhakrishna, N. Totla, Synthesis from incompatible specifications, in: *Proceedings of EMSOFT 2012, ACM* (2012) 53–62.

[9] J. Dassow, G. Vaszil, P finite automata and regular languages over countably infinite alphabets, in: *Proceedings of WMC 7, LNCS* 4361(2006) 367–381.

[10] P. Degano, G.L. Ferrari, G. Mezzetti, Nominal automata for resource usage control, in: *Proceedings of CIAA 2012, LNCS* 7381(2012) 125–137.

[11] A. Deharbe, F. Peschanski, The omniscient garbage collector: A resource analysis framework, in: *Proceedings of ACSD 2014, IEEE Computer Society* 102–111.

[12] M. Droste, P. Gastin, Weighted automata and weighted logics, *Theoret. Comput. Sci.* 380 (2007) 69–86.

[13] M. Droste, P. Gastin, Weighted automata and weighted logics, chapter 5, in [14].

[14] M. Droste, W. Kuich, H. Vogler (eds), *Handbook of Weighted Automata.* EATCS Monographs in Theoretical Computer Science, Springer, Berlin, 2009.

[15] M. Droste, I. Meinecke, Weighted automata and regular expressions over valuation monoids, *Internat. J. Found. Comput. Sci.* 22(2011) 1829–1844.

[16] M. Droste, G. Rahonis, Weighted linear dynamic logic, in: *Proceedings of GandALF 2016, EPTCS* 226(2016) 149–163.

[17] C. Elgot, Decision problems of finite automata design and related arithmetics, *Trans. Amer. Math. Soc.* 98(1961) 21-52.

[18] Z. Ésik, W. Kuich, Finite automata, chapter 3, in [14].

[19] P. Gastin, B. Monmege, A unifying survey on weighted logics and weighted automata, *Soft Computing*, to appear.

[20] G. De Giacomo, M. Vardi, Linear temporal logic and linear dynamic logic on finite traces, in: *Proceedings of IJCAI, IJCAI/AAAI.* Available at http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6997

[21] O. Grumberg, O. Kupferman, S. Sheinvald, Variable automata over infinite alphabets, in: *Proceedings of LATA 2010, LNCS* 6031(2010) 561–572.

[22] O. Grumberg, O. Kupferman, S. Sheinvald, Variable automata over infinite alphabets, http://www.cs.huji.ac.il/˜ornak/publications/lata10.pdf

[23] O. Grumberg, O. Kupferman, S. Sheinvald, Model checking systems and specifications with parameterized atomic propositions, in: *Proceedings of ATVA 2012, LNCS* 7561(2012) 122–136.

[24] M. Kaminski, N. Francez, Finite-memory automata, *Theoret. Comput. Sci.* 134(1994) 329–363.

[25] M. Kaminski, T. Tan, Regular expressions for languages over infinite alphabets, *Fund. Inform.* 69(2006) 301–318.

[26] I.-E. Mens, *Tree Automata over Infinite Ranked Alphabets*, Master thesis, Thessaloniki 2011, http://invenio.lib.auth.gr/record/128884/files/GRI-2012-8361.pdf

[27] I.-E. Mens, G. Rahonis, Variable tree automata over infinite ranked alphabets, in: *Proceedings of CAI 2011, LNCS* 6742(2011) 247–260.

[28] F. Neven, T. Schwentick, V. Vianu, Towards regular languages over infinite alphabets, in: *Proceedings of MFCS 2001, LNCS* 2136(2001), 560–572.

[29] F. Neven, T. Schwentick, V. Vianu, Finite state machines for strings over infinite alphabets, *ACM Trans. Comput. Log.* 5(2004) 403–435.

[30] G. Pellegrino, Q. Lin, C. Hammerschmidt, S. Verwer, Learning deterministic finite automata from infinite alphabets, in: *Proceedings of ICGI 2016, JMLR: Workshop and Conference Proceedings* 57(2016) 120–131.

[31] M. Pittou, *Weighted Variable Automata over Infinite Alphabets*, Master Thesis, Thessaloniki 2014, http://ikee.lib.auth.gr/record/135664/files/GRI-2015-13622.pdf

[32] M. Pittou, G. Rahonis, Weighted variable automata over infinite alphabets, in: *Proceedings of CIAA 2014, LNCS* 8587(2014) 304–317.

[33] M. Praveen, B. Srivathsan, Nesting depth of operators in graph database queries: Expressiveness vs evaluation complexity, in: *Proceedings of ICALP 2016, LIPIcs* 55(2016), 117:1–117:14.

[34] J. Sakarovitch, *Elements of Automata Theory*, Cambridge University Press, 2009.

[35] M. Schützenberger, On the definition of a family of automata, *Information and Control* 4(1961) 245–270.

[36] L. Segoufin, Automata and logics for words and trees over an infinite alphabet, in: *Proceedings of CSL 2006, LNCS* 4207(2006) 41–57.

[37] Y. Shemesh, N. Francez, Finite-state unification automata and relational languages, *Inform. and Comput.* 114(1994) 192–213.

[38] F. Song, Z. Wu, On temporal logics with data variable quantifications: Decidability and complexity, *Inform. and Comput.* 251(2016) 104–139.

[39] T. Tan, Graph reachability and pebble automata over infinite alphabets, *ACM Trans. Comput. Log.* 14(3)(2013) 19:1–19:31.

[40] W. Thomas, Languages, automata and logic, in: *Handbook of Formal Languages* vol. 3 (G. Rozenberg, A. Salomaa, eds.), Springer, 1997, pp. 389-485.

[41] B. Trakhtenbrot, Finite automata and logic of monadic predicates (in Russian), *Doklady Akademii Nauk SSSR* 140(1961) 326–329.

[42] M.Y. Vardi, The rise and fall of LTL, in: *Proceedings of GandALF 2011, EPTCS* 54(2011).

[43] D. Vrgoč, Using variable automata for querying data graphs, *Inf. Process. Lett.* 115(3)(2015) 425–430.

Acta Cybernetica **23** (2017) 319–326.

Overview of an Abstract Fixed Point Theory for Non-Monotonic Functions and its Applications to Logic Programming

Angelos Charalambidis[a] and Panos Rondogiannis[a]

Abstract

The purpose of the present paper is to give an overview of our joint work with Zoltán Ésik, namely the development of an abstract fixed point theory for a class of non-monotonic functions [4] and its use in providing a novel denotational semantics for a very broad extension of classical logic programming [1]. Our purpose is to give a high-level presentation of the main developments of these two works, that avoids as much as possible the underlying technical details, and which can be used as a mild introduction to the area.

Keywords: fixed point theory, higher-order logic programming, semantics of logic programming

1 Introduction

The purpose of this paper is to present an overview of the authors' joint work with Zoltán Ésik. This work [4] concerned the development of an abstract fixed point theory for a class of functions that exhibit a type of "monotonicity in layers" but which are overall non-monotonic. Such functions prove to be quite common in various investigations in logic programming and formal language theory, and may potentially have other applications. We also describe our development [1], based on the aforementioned abstract framework, of a novel denotational semantics for a very broad extension of classical logic programming. In the rest of this section we provide a short description of the beginnings of our collaboration with Zoltán that led to the above results.

In 2005, the second author together with Bill Wadge proposed [5] the infinite-valued semantics for logic programs with negation. This particular work was somewhat ad-hoc, namely the main results relied on techniques custom-tailored for logic programming. In 2013, the second author of the present paper, together with Zoltán

[a]Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, E-mail: {a.charalambidis,prondo}@di.uoa.gr

DOI: 10.14232/actacyb.23.1.2017.17

Ésik started a collaboration supported by a "Greek-Hungarian Scientific Collaboration Program" with title "*Extensions and Applications of Fixed Point Theory for Non-Monotonic Formalisms*". The purpose of the program was to create an abstract fixed point theory based on the infinite-valued approach, namely a theory that would not only be applicable to logic programs but also to other non-monotonic formalisms. This abstract theory was successfully developed and is described in detail in [4]. As an application of these results, this abstract theory was used in [1] in order to obtain the first extensional semantics for higher-order logic programs with negation. Another application of the new theory to the area of non-monotonic formal grammars was proposed in [3]. Moreover, Zoltán himself further investigated the foundations and the properties of the infinite-valued approach [2], highlighting some of its desirable characteristics. Unfortunately, the further joint development of the abstract infinite-valued approach to non-monotonic fixed point theory, was abruptly interrupted by the untimely loss of Zoltán.

In the next section we describe the basic concepts behind the abstract approach to non-monotonic fixed point theory. In Section 3 we describe the application of the theory to the class of higher-order logic programs with negation. The paper concludes by giving pointers for future work.

2 Non-Monotonic Fixed Point Theory

Suppose that $(L, \leq)$ is a complete lattice in which the least upper bound operation is denoted by $\bigvee$ and the least element is denoted by $\perp$. Let $\kappa > 0$ be a fixed ordinal. We assume that for each ordinal $\alpha < \kappa$, there exists a preordering $\sqsubseteq_\alpha$ on L. We denote with $=_\alpha$ the equivalence relation determined by $\sqsubseteq_\alpha$. We define $x \sqsubset_\alpha y$ iff $x \sqsubseteq_\alpha y$ but $x =_\alpha y$ does not hold. Finally, we define $\sqsubset = \bigcup_{\alpha<\kappa} \sqsubset_\alpha$ and let $x \sqsubseteq y$ iff $x \sqsubset y$ or $x = y$. Given an ordinal $\alpha < \kappa$ and $x \in L$, define $(x]_\alpha = \{y \in L : \forall\beta < \alpha \quad x =_\beta y\}$. We require of our relations to satisfy the following axioms:

Axiom 1. *For all ordinals $\alpha < \beta < \kappa$, $\sqsubseteq_\beta$ is included in $=_\alpha$.*

Axiom 2. $\bigcap_{\alpha<\kappa} =_\alpha$ *is the identity relation on L.*

Axiom 3. *For each $x \in L$, for every ordinal $\alpha < \kappa$, and for any $X \subseteq (x]_\alpha$ there is some $y \in (x]_\alpha$ such that:*

- *$X \sqsubseteq_\alpha y$, and*
- *for all $z \in (x]_\alpha$, if $X \sqsubseteq_\alpha z$ then $y \sqsubseteq_\alpha z$ and $y \leq z$.*

Axiom 4. *If $x_j, y_j \in L$ and $x_j \sqsubseteq_\alpha y_j$ for all $j \in J$ then $\bigvee\{x_j : j \in J\} \sqsubseteq_\alpha \bigvee\{y_j : j \in J\}$.*

The element y specified by the Axiom 3 above, can be shown to be unique and we denote it by $\bigsqcup_\alpha X$.

In the following, we will often talk about "*models of the Axioms 1-4*" (or simply "*models*"). More formally:

Definition 1. *A* model of Axioms 1-4 *or simply* model *consists of a complete lattice $(L, \leq)$, an ordinal $\kappa > 0$ and a set of preorders $\sqsubseteq_\alpha$ for every $\alpha < \kappa$, such that Axioms 1-4 are satisfied.*

Under the above axioms, the following theorem is established in [4]:

Theorem 1. *$(L, \sqsubseteq)$ is a complete lattice.*

The following definition will lead us to the main theorem of [4]:

Definition 2. *Suppose that L is a model and let $\alpha < \kappa$. A function $f : L \to L$ is called α-monotonic if for all $x, y \in L$, if $x \sqsubseteq_\alpha y$ then $f(x) \sqsubseteq_\alpha f(y)$.*

The central fixed point theorem of [4] can now be stated:

Theorem 2. *Let L be a model. Suppose that $f : L \to L$ is α-monotonic for each ordinal $\alpha < \kappa$. Then f has a least pre-fixed point with respect to the partial order $\sqsubseteq$, which is also the least fixed point of f.*

The article [4] contains many more results, but one could say that the above theorem is possibly the main technical achievement. Actually, the above theorem is also the main tool that we will need in the developments of the next section.

3 Higher-Order Logic Programs with Negation

In this section we present the application of the non-monotonic fixed point theory to the class of higher-order logic programs with negation. The approach presented naturally extends the ideas behind the infinite-valued approach proposed in [5] into a higher-order setting. The basic idea behind the approach in [5] is that in order to obtain minimum model semantics for higher-order logic programs with negation it is necessary to consider a multi-valued logic. We first present the syntax and then the semantics of our language.

3.1 Syntax

Our higher-order logic programming language is based on a simple type system that supports two base types: o, the boolean domain, and ι, the domain of individuals (data objects). The composite types are partitioned into three classes: functional (assigned to individual constants, individual variables and function symbols), predicate (assigned to predicate constants and variables) and argument (assigned to parameters of predicates).

Definition 3. *A type τ can either be functional, argument, or predicate, denoted as σ, π and ρ respectively and defined as:*

$$\begin{aligned} \sigma &:= \iota \mid \iota \to \sigma \\ \pi &:= o \mid \rho \to \pi \\ \rho &:= \iota \mid \pi \end{aligned}$$

Definition 4. *The set of* expressions *of our higher-order language is defined as follows:*

1. *Every predicate variable (respectively, predicate constant) of type π is an expression of type π; every individual variable (respectively, individual constant) of type ι is an expression of type ι; the propositional constants* false *and* true *are expressions of type o.*
2. *If f is an n-ary function symbol and $\mathsf{E}_1, \ldots, \mathsf{E}_n$ are expressions of type ι, then $(\mathsf{f}\ \mathsf{E}_1 \cdots \mathsf{E}_n)$ is an expression of type ι.*
3. *If E_1 is an expression of type $\rho \to \pi$ and E_2 is an expression of type ρ, then $(\mathsf{E}_1\ \mathsf{E}_2)$ is an expression of type π.*
4. *If V is an argument variable of type ρ and E is an expression of type π, then $(\lambda\mathsf{V}.\mathsf{E})$ is an expression of type $\rho \to \pi$.*
5. *If $\mathsf{E}_1, \mathsf{E}_2$ are expressions of type π, then $(\mathsf{E}_1 \bigwedge_\pi \mathsf{E}_2)$ and $(\mathsf{E}_1 \bigvee_\pi \mathsf{E}_2)$ are expressions of type π.*
6. *If E is an expression of type o, then $({\sim}\mathsf{E})$ is an expression of type o.*
7. *If $\mathsf{E}_1, \mathsf{E}_2$ are expressions of type ι, then $(\mathsf{E}_1 \approx \mathsf{E}_2)$ is an expression of type o.*
8. *If E is an expression of type o and V is a variable of type ρ then $(\exists_\rho \mathsf{V}\, \mathsf{E})$ is an expression of type o.*

The notions of *free* and *bound* variables of an expression are defined as usual. An expression is called *closed* if it does not contain any free variables.

A *program clause* is a clause $\mathsf{p} \leftarrow_\pi \mathsf{E}$ where p is a predicate constant of type π and E is a closed expression of type π. A *program* is a finite set of program clauses.

3.2 Semantics

We start by examining the semantics of types. The most crucial case is that of the boolean domain o. The boolean values range over a partially ordered set $(V, \leq)$ of truth values. The number of truth values of V will be specified with respect to an ordinal $\kappa > 0$. The set $(V, \leq)$ is the following:

$$F_0 < F_1 < \cdots < F_\alpha < \cdots < 0 < \cdots < T_\alpha < \cdots < T_1 < T_0$$

where $\alpha < \kappa$. Intuitively, F_0 and T_0 are the classical *False* and *True* values and 0 is the undefined value. The new values express different levels of truthness and falsity. The *order* of a truth value is defined as follows: $order(T_\alpha) = \alpha$, $order(F_\alpha) = \alpha$ and $order(0) = +\infty$.

We define the following preorderings $\sqsubseteq_\alpha$ on the set V for each $\alpha < \kappa$:

1. $x \sqsubseteq_\alpha x$ if $order(x) < \alpha$;
2. $F_\alpha \sqsubseteq_\alpha x$ and $x \sqsubseteq_\alpha T_\alpha$ if $order(x) \geq \alpha$;
3. $x \sqsubseteq_\alpha y$ if $order(x), order(y) > \alpha$.

We then have the following result from [1]:

Lemma 1. *$(V, \leq)$ is a complete lattice and a model.*

Let us denote by $[A \overset{m}{\to} B]$ the set of functions from A to B that are α-monotonic for all $\alpha < \kappa$. Based on the above discussion, we can now state the semantics of all the types of our language:

Definition 5. *Let D be a nonempty set. Then:*

- $[\![o]\!]_D = V$*, and* $\leq_o$ *is the partial order of* V*;*
- $[\![\iota]\!]_D = D$*, and* $\leq_\iota$ *is the trivial partial order such that* $d \leq_\iota d$*, for all* $d \in D$*;*
- $[\![\iota^n \to \iota]\!]_D = D^n \to D$*. A partial order in this case will not be needed;*
- $[\![\iota \to \pi]\!]_D = D \to [\![\pi]\!]_D$*, and* $\leq_{\iota\to\pi}$ *is the partial order defined as follows: for all* $f, g \in [\![\iota \to \pi]\!]_D$*,* $f \leq_{\iota\to\pi} g$ *iff* $f(d) \leq_\pi g(d)$ *for all* $d \in D$*;*
- $[\![\pi_1 \to \pi_2]\!]_D = [[\![\pi_1]\!]_D \overset{m}{\to} [\![\pi_2]\!]_D]$*, and* $\leq_{\pi_1\to\pi_2}$ *is the partial order defined as follows: for all* $f, g \in [\![\pi_1 \to \pi_2]\!]_D$*,* $f \leq_{\pi_1\to\pi_2} g$ *iff* $f(d) \leq_{\pi_2} g(d)$ *for all* $d \in [\![\pi_1]\!]_D$*.*

Moreover, we have the following relations $\sqsubseteq_\alpha$ on our domains:

- The relation $\sqsubseteq_\alpha$ on $[\![o]\!]_D$ is the relation $\sqsubseteq_\alpha$ on V.
- The relation $\sqsubseteq_\alpha$ on $[\![\rho \to \pi]\!]_D$ is defined as follows: $f \sqsubseteq_\alpha g$ iff $f(d) \sqsubseteq_\alpha g(d)$ for all $d \in [\![\rho]\!]_D$.

The following lemma can then be established following the results of [4]:

Lemma 2. *Let D be a non-empty set and π be a predicate type. Then, $(\llbracket\pi\rrbracket_D, \leq_\pi)$ is a complete lattice and a model.*

For the rest of the section we focus on Herbrand interpretations and we assume for a program P, $D = U_{\mathsf{P}}$ where U_{P} is the Herbrand universe and therefore we simple write $\llbracket\tau\rrbracket$ instead of $\llbracket\tau\rrbracket_{U_{\mathsf{P}}}$. A Herbrand interpretation I for a program P is a function that maps a predicate of type π to an element of $\llbracket\pi\rrbracket$. The set of all the interpretation of P is denoted by $\mathcal{I}_{\mathsf{P}}$. It follows directly from the results of [4] that $\mathcal{I}_{\mathsf{P}}$ is a complete lattice and a model. A Herbrand state s is a function that assigns to each argument variable V of type ρ, of an element $s(\mathsf{V}) \in \llbracket\rho\rrbracket_{U_{\mathsf{P}}}$.

Let I be a Herbrand interpretation and s be a Herbrand state. The semantics of expressions with respect to I and s, is defined as follows:

1. $\llbracket\mathsf{false}\rrbracket_s(I) = F_0$
2. $\llbracket\mathsf{true}\rrbracket_s(I) = T_0$
3. $\llbracket\mathsf{c}\rrbracket_s(I) = I(\mathsf{c})$, for every individual constant c
4. $\llbracket\mathsf{p}\rrbracket_s(I) = I(\mathsf{p})$, for every predicate constant p
5. $\llbracket\mathsf{V}\rrbracket_s(I) = s(\mathsf{V})$, for every argument variable V
6. $\llbracket(\mathsf{f}\ \mathsf{E}_1 \cdots \mathsf{E}_n)\rrbracket_s(I) = I(\mathsf{f})\ \llbracket\mathsf{E}_1\rrbracket_s(I) \cdots \llbracket\mathsf{E}_n\rrbracket_s(I)$, for every n-ary function symbol f
7. $\llbracket(\mathsf{E}_1\mathsf{E}_2)\rrbracket_s(I) = \llbracket\mathsf{E}_1\rrbracket_s(I)(\llbracket\mathsf{E}_2\rrbracket_s(I))$
8. $\llbracket(\lambda\mathsf{V}.\mathsf{E})\rrbracket_s(I) = \lambda d.\llbracket\mathsf{E}\rrbracket_{s[\mathsf{V}/d]}(I)$, where d ranges over $\llbracket type(\mathsf{V})\rrbracket_D$
9. $\llbracket(\mathsf{E}_1 \bigvee_\pi \mathsf{E}_2)\rrbracket_s(I) = \bigvee_\pi\{\llbracket\mathsf{E}_1\rrbracket_s(I), \llbracket\mathsf{E}_2\rrbracket_s(I)\}$, where $\bigvee_\pi$ is the lub function on $\llbracket\pi\rrbracket_D$
10. $\llbracket(\mathsf{E}_1 \bigwedge_\pi \mathsf{E}_2)\rrbracket_s(I) = \bigwedge_\pi\{\llbracket\mathsf{E}_1\rrbracket_s(I), \llbracket\mathsf{E}_2\rrbracket_s(I)\}$, where $\bigwedge_\pi$ is the glb function on $\llbracket\pi\rrbracket_D$
11. $\llbracket(\sim\mathsf{E})\rrbracket_s(I) = \begin{cases} T_{\alpha+1} & \text{if } \llbracket\mathsf{E}\rrbracket_s(I) = F_\alpha \\ F_{\alpha+1} & \text{if } \llbracket\mathsf{E}\rrbracket_s(I) = T_\alpha \\ 0 & \text{if } \llbracket\mathsf{E}\rrbracket_s(I) = 0 \end{cases}$
12. $\llbracket(\mathsf{E}_1 \approx \mathsf{E}_2)\rrbracket_s(I) = \begin{cases} T_0, & \text{if } \llbracket\mathsf{E}_1\rrbracket_s(I) = \llbracket\mathsf{E}_2\rrbracket_s(I) \\ F_0, & \text{otherwise} \end{cases}$
13. $\llbracket(\exists\mathsf{V}\ \mathsf{E})\rrbracket_s(I) = \bigvee_{d\in\llbracket type(\mathsf{V})\rrbracket_D} \llbracket\mathsf{E}\rrbracket_{s[\mathsf{V}/d]}(I)$

Definition 6. *Let P be a program and let M be a Herbrand interpretation of P. Then M will be called a* model *of P iff for all clauses $\mathsf{p} \leftarrow_\pi \mathsf{E}$ of P, it holds $\llbracket\mathsf{E}\rrbracket(M) \leq_\pi M(\mathsf{p})$, where $M(\mathsf{p}) \in \llbracket\pi\rrbracket$.*

We can now define the immediate consequence operator for our language:

Definition 7. *Let* P *be a program. The mapping* $T_{\mathsf{P}} : \mathcal{I}_{\mathsf{P}} \to \mathcal{I}_{\mathsf{P}}$ *is defined for every* $\mathsf{p} : \pi$ *and for every* $I \in \mathcal{I}_{\mathsf{P}}$ *as*

$$T_{\mathsf{P}}(I)(\mathsf{p}) = \bigvee \{ [\![\mathsf{E}]\!](I) : (\mathsf{p} \leftarrow_{\pi} \mathsf{E}) \in \mathsf{P} \}$$

As it turns out, T_{P} enjoys the α-monotonicity property [1]:

Lemma 3. *For all* $\alpha < \kappa$*,* T_{P} *is* α*-monotonic.*

We now have all we need in order to apply the main Theorem of [4], getting the following result [1]:

Theorem 3 (Least Fixed Point Theorem)**.** *Let* P *be a program and let* $\mathcal{M}$ *be the set of all its Herbrand models. Then,* T_{P} *has a least fixed point* M_{P} *which is the least model of* P*.*

4 Conclusions

We have presented an overview of the abstract fixed point theory developed in [4] and its application [1] on a very broad class of logic programs, namely higher-order logic programs with negation. It is our belief that the framework of [4] can find other interesting applications, especially ones where non-monotonicity plays a prevailing role. In particular, we believe that an area that has not yet been sufficiently explored is that of non-monotonic formal grammars. In [3] it was demonstrated that the semantics of Boolean grammars can be easily captured through an extension of the framework of [4]. However, it is conceivable to have other non-monotonic extensions of formal grammars apart from the Boolean ones, such as for example macro-grammars with conjunction and negation in rule bodies. We believe that the results of [1] can be used as a yardstick in order to approach the semantics of such grammar formalisms.

References

[1] Charalambidis, Angelos, Ésik, Zoltán, and Rondogiannis, Panos. Minimum model semantics for extensional higher-order logic programming with negation. *TPLP*, 14(4-5):725–737, 2014.

[2] Ésik, Zoltán. Equational properties of stratified least fixed points (extended abstract). In de Paiva, Valeria, de Queiroz, Ruy J. G. B., Moss, Lawrence S., Leivant, Daniel, and de Oliveira, Anjolina Grisi, editors, *Logic, Language, Information, and Computation - 22nd International Workshop, WoLLIC 2015, Bloomington, IN, USA, July 20-23, 2015, Proceedings*, volume 9160 of *Lecture Notes in Computer Science*, pages 174–188. Springer, 2015.

[3] Ésik, Zoltán and Rondogiannis, Panos. Theorems on pre-fixed points of non-monotonic functions with applications in logic programming and formal grammars. In Kohlenbach, Ulrich, Barceló, Pablo, and de Queiroz, Ruy J. G. B., editors, *Logic, Language, Information, and Computation - 21st International Workshop, WoLLIC 2014, Valparaíso, Chile, September 1-4, 2014. Proceedings*, volume 8652 of *Lecture Notes in Computer Science*, pages 166–180. Springer, 2014.

[4] Ésik, Zoltán and Rondogiannis, Panos. A fixed point theorem for non-monotonic functions. *Theor. Comput. Sci.*, 574:18–38, 2015.

[5] Rondogiannis, Panos and Wadge, William W. Minimum model semantics for logic programs with negation-as-failure. *ACM Trans. Comput. Log.*, 6(2):441–467, 2005.

Acta Cybernetica **23** (2017) 327–347.

On the Completeness of the Traced Monoidal Category Axioms in (Rel,+)[*]

Miklós Bartha[a]

To the memory of my friend and former colleague Zoltán Ésik

Abstract

It is shown that the traced monoidal category of finite sets and relations with coproduct as tensor is complete for the extension of the traced symmetric monoidal axioms by two simple axioms, which capture the additive nature of trace in this category. The result is derived from a theorem saying that already the structure of finite partial injections as a traced monoidal category is complete for the given axioms. In practical terms this means that if two biaccessible flowchart schemes are not isomorphic, then there exists an interpretation of the schemes by partial injections which distinguishes them.

Keywords: monoidal categories, trace, iteration, feedback, identities in categories, equational completeness

1 Introduction

It was in March, 2012 that Zoltán visited me at the Memorial University in St. John's, Newfoundland. I thought we would find out a research topic together, but he arrived with a specific problem in mind. He knew about the result found by Hasegawa, Hofmann, and Plotkin [12] a few years earlier on the completeness of the category of finite dimensional vector spaces for the traced monoidal category axioms, and wanted to see if there is a similar completeness statement true for the matrix iteration theory of finite sets and relations as a traced monoidal category. Unfortunately, during the short time we spent together, we could not find the solution, and after he had left I started to work on some other problems. Having learned about his shocking untimely death, I felt impelled to finally solve the problem that we started to work on together, and have it published in his memory.

The result obtained in this paper points beyond the original goal of finding a suitable extension of the traced monoidal category axioms for which the category

[*]Work initiated by Zoltán Ésik as part of a research project in 2012.

[a]Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada, E-mail: `bartha@mun.ca`

DOI: 10.14232/actacyb.23.1.2017.18

$(\mathbf{Rel}_{fin}, +)$ is complete. We show that already the category $(\mathbf{Pin}_{fin}, +)$ of finite partial injections is complete for the minimal extension of the traced monoidal axioms that can be expected from these structures. Since $(\mathbf{Pin}_{fin}, +)$ is a subcategory of $(\mathbf{Rel}_{fin}, +)$, the completeness result originally sought by Zoltán follows as a corollary.

We shall assume familiarity with the basic concepts of category theory [19], and it helps if the reader is also familiar with Zoltán's work on iteration theories [9].

2 Traced monoidal categories

A symmetric *monoidal category* consists of a category $\mathcal{C}$ equipped with a bifunctor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ called *tensor*, and a unit object I of $\mathcal{C}$. Furthermore, $\mathcal{C}$ has natural isomorphisms:

$$a_{X,Y,Z} : (X \otimes Y) \otimes Z \to X \otimes (Y \otimes Z)$$

called *associators*,

$$l_X : I \otimes X \to X \text{ and } r_X : X \otimes I \to X$$

called left and right *unitors*, and

$$c_{U,V} : U \otimes V \to V \otimes U$$

called *symmetries*, which isomorphisms are subject to the standard coherence axioms given in [19]. The monoidal category $\mathcal{C}$ is *strict* if the bifunctor $\otimes$ is strictly associative, so that

$$A \otimes (B \otimes C) = (A \otimes B) \otimes C \text{ and } A \otimes I = I \otimes A = A,$$

and all of the associators and unitors are the identities.

Let $\mathcal{C}$ be a monoidal category with tensor $\otimes$ and unit object I as specified above. The following definition of traced monoidal categories uses the terminology of [15], except that trace in is introduced as *left* trace, that is, an operation $\mathcal{C}(U + A, U + B) \to \mathcal{C}(A, B)$, rather than $\mathcal{C}(A + U, B + U) \to \mathcal{C}(A, B)$ (i.e., right trace) as it appears in [15]. The reason is to remain consistent with the notation used in Zoltán's and the author's own work. Also, following in Zoltán's footsteps, we write composition of morphisms in a left-to-right way, that is, the composite of $f : A \to B$ and $g : B \to C$ is $f \circ g : A \to C$.

Definition 1. A *trace* for $\mathcal{C}$ is a family of functions

$$Tr^U_{A,B} : \mathcal{C}(U \otimes A, U \otimes B) \to \mathcal{C}(A, B)$$

natural in A and B, satisfying the following four axioms:

vanishing:

$$Tr^{U \otimes V}_{A,B}(a_{U,V,A} \circ f) = Tr^V_{A,B}(Tr^U_{V \otimes A, V \otimes B}(f \circ a_{U,V,B})),$$

where $f : U \otimes (V \otimes A) \to (U \otimes V) \otimes B$;

superposing:

$$Tr^U_{A,B}(f) \otimes g = Tr^U_{A\otimes C,B\otimes D}(a^{-1}_{U,A,C} \circ (f \otimes g) \circ a_{U,B,D}),$$

where $f : U \otimes A \to U \otimes B$, and $g : C \to D$;

sliding:

$$Tr^U_{A,B}((g \otimes 1_A) \circ f) = Tr^V_{A,B}(f \circ (g \otimes 1_B))$$

where $f : V \otimes A \to U \otimes B$ and $g : U \to V$;

yanking:

$$Tr^U_{U,U}(c_{U,U}) = 1_U.$$

It is well-known that sliding of symmetries (whereby g is chosen as a symmetry isomorphism in the sliding axiom) suffices in the presence of the other axioms. Also, the special vanishing axiom:

$$Tr^I_{A,B}(l_A \circ f \circ l^{-1}_B) = f \qquad \text{for } f : A \to B,$$

which was part of the original system in [15], can easily be derived from the other axioms and can therefore be omitted. As the reader can see, the necessity of using the associator and unitor morphisms $a(X, Y, Z)$, l_X in the traced monoidal axioms makes them look very complicated, even though their graphical representation below shows that in fact they are quite intuitive and simple. It is known, see e.g. [20], that every monoidal category is equivalent to a strict one. Quoting an argument from [14], "most results obtained with the hypothesis that a monoidal category is strict can, in principle, be reformulated and proved without that condition". Our result in this paper is no exception, therefore in the sequel we shall make the technically simplifying assumption that our monoidal categories (traced or not) are strict.

The graphical representation of the traced monoidal category axioms (with the strictness assumption incorporated) is given in Figures 1–5.

Traced monoidal categories (with one additional axiom) and their graphical language first appeared in [1] in an algebraic setting, using the name "scheme algebra" for these structures. The operation corresponding to trace was called feedback. The year was 1987, and already at that time Zoltán and Steve Bloom had a significant number of important results on iterative and iteration theories, the study of which was initiated by Calvin C. Elgot in the early 1970s. The motivation of that study was to find out the equational laws characterizing the iteration operation in flowchart-related algorithms. The algebra (category) of flowcharts itself has also been axiomatized in terms of the iteration operation [8]. This axiomatization was a little awkward, however, because the iteration operation

$$f^\dagger : n \to p \qquad \text{for } f : n \to n + p,$$

where n and p are non-negative integers, was intended to capture the semantical aspects of iteration in the first place. For syntactical purposes the feedback

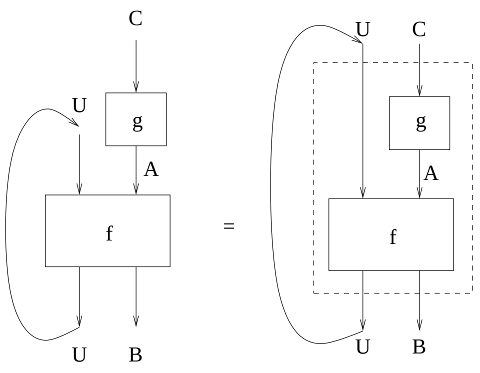


Figure 1: Naturality of trace in A.

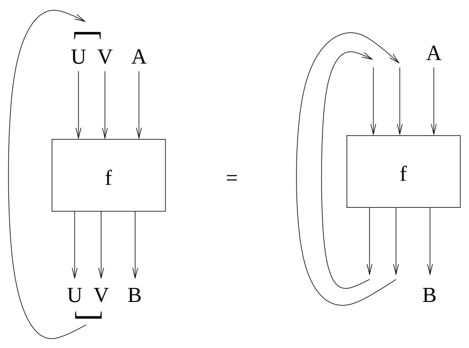


Figure 2: Vanishing

operation

$$\uparrow^n f : p \to q \quad \text{for } f : n + p \to n + q$$

turns out to be a lot more practical and easier to deal with. No loss of generality arises from the switch, provided that the underlying structures are algebraic theories, since there are standard rewriting rules between iteration and feedback in all such theories. See [9, 1, 25]. (The purely syntactical category of flowcharts is of course not such a structure.) Regarding the exact relationship between traced monoidal categories and iteration theories, it was proved in [1] that a single-sorted traced monoidal category is an iteration theory iff it is an algebraic theory and satisfies the commutative axioms discovered by Zoltán in [11].

Continuing the story of traced monoidal categories, essentially the same axiomatization and graphical language as the one presented in [1] was published a few years later in [10] under the name “biflow”. Neither of these pioneer works noticed, however, that the monoidal category of finite dimensional vector spaces, in which tensor is tensor product and feedback is trace, is an obvious scheme algebra/biflow, provided that the axiomatization is lifted from the single-sorted algebraic language

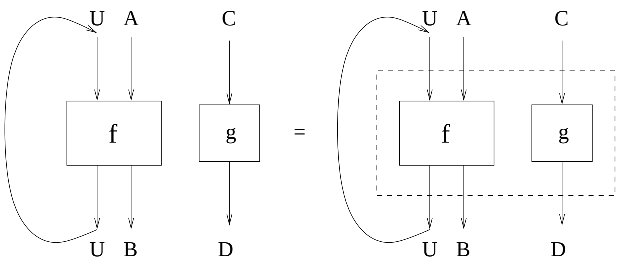


Figure 3: Superposing

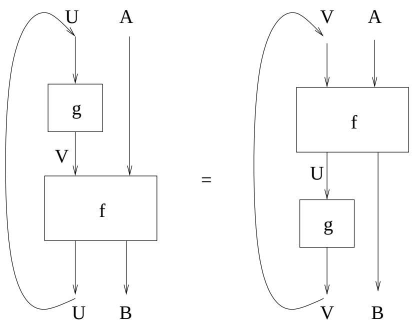


Figure 4: Sliding

to the general "polymorphic" categorical one. Finally, in 1996, Joyal, Street, and Verity made this important point by essentially rediscovering the old scheme axioms in a general new context, which also covered balanced monoidal categories. They also presented the fundamental *Int* construction on the embedding of an arbitrary balanced traced monoidal category into a tortile one [15]. In case braiding is symmetry, as it is in our present study, the *Int* construction transforms an arbitrary traced monoidal category into a compact closed category [17].

By virtue of the above discussion, the feedback operation is deeply rooted in control theory, whereas trace is a concept used primarily in finite dimensional vector spaces as an operation on linear maps (matrices). The usual interpretations of trace and feedback have not much in common, since trace is "multiplicative" style in contrast with feedback, which has a strong "additive" flavour. The informal distinction "additive or multiplicative" uses the very basic category of sets and functions as a basis for comparison. Taking tensor in this category as Cartesian product with $I = \{\emptyset\}$ (multiplicative) or coproduct with $I = \emptyset$ (additive) results in entirely different monoidal categories. On this basis, "multiplicative" in vector spaces means that $\otimes$ is tensor product rather than ordinary product, which happens to coincide with coproduct.

According to the main result of [1], the free single-sorted category with feedback generated by a collection of morphisms (boxes) is the algebra of flowchart schemes,

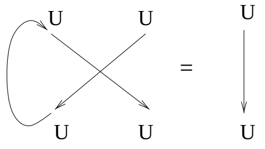


Figure 5: Yanking

which is definitely additive style and reflects the flow of information in flowchart algorithms. In summary, there are countless reasons to call the operation

$$Tr^{U}_{A,B} : \mathcal{C}(U \otimes A, U \otimes B) \to \mathcal{C}(A, B)$$

feedback, and just a few to call it trace. Nevertheless, the name "trace" stuck, and today everyone calls the categorical structures corresponding to scheme algebras traced monoidal categories.

To complicate the issue even further, categories with feedback have also been considered in [16] and a series of works by the present author [2, 3, 4, 5]. In these categories, however, yanking is missing from the axioms imposed. Feedback in such categories is *delayed* like in synchronous systems, e.g. sequential circuits. The meaning of the loop on the left-hand-side of the yanking axiom (Fig. 5) is a *register*, a memory element, which suggests a step-by-step behavior for synchronous systems (circuits/schemes). Note that this kind of categorical interpretation over sets as objects is multiplicative style, since a morphism $A \to B$ is a Mealy automaton $U \times A \to U \times B$ with U being an arbitrary set (of states). As a consequence, there is no explicit control present in the system. In a sequential circuit, for example, every logical gate and flip-flop is engaged in each clock cycle, so that the whole system is massively parallel. What we call "control" in the von Neumann computer architecture is just an abstraction, an extra control line carrying a digital information indicating that the control is present or not. Nevertheless, if the underlying monoidal category of the category of circuits (automata) is additive, then the automata themselves will be additive, too, possessing the equivalent of some kind of control.

As stated in the Introduction of [9], and demonstrated throughout the book, iteration theories are ubiquitous in computer science. If this is true, then traced monoidal categories are twice as ubiquitous and not only in computer science but in the whole of mathematics. Indeed, these categories are more general than iteration theories, therefore they cover more ground. Here are just a few among the most important traced monoidal categories occurring with great frequency.

1. *The additive category* $(\mathbf{Rel}, +)$ *of relations with* $\otimes$ *being coproduct.*

The base category of $(\mathbf{Rel}, +)$ is the category $\mathbf{Rel}$ of sets and relations. That is, objects are sets, and a morphism $A \to B$ is a relation $R \subseteq A \times B$. Tensor of objects is disjoint union $(+)$, $I = \emptyset$, and tensor of morphisms is disjoint union of relations.

For $R : U + A \to U + B$, $Tr^U_{A,B}R$ is given by

$$(a, b) \in Tr^U_{A,B}R \text{ iff } \exists\, u_1, \ldots, u_n \in U,\ n \geq 0, \text{ such that } aRu_1R \ldots Ru_nRb.$$

The category $(\mathbf{Rel}, +)$ restricts naturally to $\mathbf{Rel}_{fin}$, the category of finite sets and relations. This restriction is equivalent to its full subcategory $(\mathbf{Rel}_{\mathbb{N}}, +)$ induced by the objects in $\mathbb{N} = \{0, 1, \ldots n \ldots\}$, where $0 = \emptyset$ and $n + 1 = n \cup \{n\}$ as in Zermelo-Fraenkel set theory. This subcategory is strict and it is closed for trace. It is also single-sorted, since the set of objects $\mathbb{N}$ is generated by the object 1 using tensor. Clearly, $I = 0$ in $(\mathbf{Rel}_{\mathbb{N}}, +)$.

It is very instructive to look at the matrix representation of $(\mathbf{Rel}_{\mathbb{N}}, +)$, for this category is a matrix iteration theory as well. It was shown in [9, Corollary 5.5] that this theory is generated by the initial ω-idempotent Boolean semiring $\mathbf{B}$. In other words, if relations are represented as 0-1 matrices, then composition and tensor (coproduct!) of relations is that of matrices, using the algebraic rules of $\mathbf{B}$ rather than those of GF(2). (That is, 1+1=1 and not 0.) The trace of a finite relation $R : n + p \to n + q$ having a matrix decomposition

$$R = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

with $A : n \to n$, $B : n \to q$, $C : p \to n$, $D : p \to q$ can then be obtained by the well-known Kleene formula:

$$Tr^n_{p,q}R = D + CA^*B, \tag{1}$$

where A^* denotes the infinite sum

$$A^* = \sum_{k=0}^{\infty} A^k$$

according to the Boolean semiring addition and multiplication rules. That is, in the Kleene formula (1), + denotes the Boolean sum of matrices $p \to q$ (rather than + as tensor/coproduct). The present definition of the Kleene star * coincides with the star operation used in matrix iteration theories.

2. *The multiplicative category* $(\mathbf{Rel}, \times)$ *of relations with product as tensor.*

The base category of $(\mathbf{Rel}, \times)$ is also $\mathbf{Rel}$, but tensor is $\times$ (Cartesian product of sets) rather than +. The object I is $\{\emptyset\}$. The tensor of two relations is the product of them in the usual sense. For $R : U \times A \to U \times B$, its trace is defined by

$$(a, b) \in Tr^U_{A,B}R \text{ iff } \exists u \in U \text{ such that } ((u, a), (u, b)) \in R.$$

Again, $(\mathbf{Rel}, \times)$ restricts to the category $\mathbf{Rel}_{fin}$ of finite sets and relations, which category is equivalent to its full subcategory $(\mathbf{Rel}_{\mathbb{N}}, \cdot)$ induced by the objects $\mathbb{N}$, provided that the "set" $n \times m$ is identified with $n \cdot m$ in a given canonical way (e.g. enumeration by rows or columns). In everyday language, take the matrix representation of relations. The unit object I is 1. The category $(\mathbf{Rel}_{\mathbb{N}}, \cdot)$ is no longer single-sorted, but it is still generated by the prime numbers (and 0) as objects, and it is strict.

3. *The multiplicative category* $(\mathbf{FdVect}_K, \otimes)$ *of finite dimensional vector spaces over a given field* K.

The base category is $\mathbf{FdVect}_K$, and $\otimes$ is tensor product of linear maps. The object I is the field K as a 1-dimensional vector space. Analogously to $(\mathbf{Rel}, \times)$, this category is also equivalent to its restriction induced by the concrete n-dimensional spaces F_K^n, in which linear maps are simply $n \times m$ matrices. In this context we can even identify the object F_K^n with the number n, since K is fixed. The reduced category is again strict. For a linear map (matrix, for simplicity) $M : U \to U$, $Tr_{I,I}^U M$ is the sum of diagonal elements in M. The general definition of trace is technically more complicated, and the reader is referred to [15] for the details of this definition. The analogy between $(\mathbf{Rel}_{fin}, \times)$ and $(\mathbf{FdVect}_K, \otimes)$ is that, when relations are represented as 0-1 matrices, their tensor and trace is calculated in the exact same way as in (the strict equivalent of) $(\mathbf{FdVect}_K, \otimes)$, interpreting the ring operations according to the Boolean semiring $\mathbf{B}$ rather than the field K. Since $\mathbf{B}$ is just a commutative semiring (addition is idempotent in $\mathbf{B}$), the morphisms of $(\mathbf{Rel}_{fin}, \times)$ are not linear maps between vector spaces. (They are just linear maps of *free semimodules* over the semiring $\mathbf{B}$.)

4. *The additive category* $(\mathbf{Iso}, \oplus)$ *of quantum control.*

Let $\mathbf{FdHilb}$ denote the category of finite dimensional Hilbert spaces. The base category $\mathbf{Iso}$ is then the subcategory of $\mathbf{FdHilb}$ having isometries only as its morphisms. Tensor is now $\oplus$, that is, coproduct/product in $\mathbf{FdHilb}$. The unit object I is the zero space. Notice the drastic change from the multiplicative tensor $\otimes$ in $(\mathbf{FdVect}_K, \otimes)$ to the additive $\oplus$, which is analogous to $+$ in $(\mathbf{Rel}_{fin}, +)$ as a matrix theory.

Let $\tau : \mathcal{U} \oplus \mathcal{K} \to \mathcal{U} \oplus \mathcal{L}$ be an isometry. Then $Tr_{\mathcal{K},\mathcal{L}}^{\mathcal{U}} \tau : \mathcal{K} \to \mathcal{L}$ is the isometry specified as follows. Consider the matrix of τ

$$\begin{array}{cc} & \begin{array}{cc} \mathcal{U} & \mathcal{L} \end{array} \\ \begin{array}{c} \mathcal{U} \\ \mathcal{K} \end{array} & \left(\begin{array}{cc} \tau_A & \tau_B \\ \tau_C & \tau_D \end{array} \right) \end{array}$$

according to the biproduct decomposition

$$\tau = \langle [\tau_A, \tau_C], [\tau_B, \tau_D] \rangle,$$

where $[\_]$ stands for coproduct and $\langle \_ \rangle$ for product. Trace is defined again by the Kleene formula

$$Tr_{\mathcal{K},\mathcal{L}}^{\mathcal{U}} \tau = \lim_{n \to \infty} (\tau_D + \tau_C \circ \tau_A^{*n} \circ \tau_B). \tag{2}$$

In the present Kleene formula

$$\tau_A^{*n} = \sum_{i=0}^{n} \tau_A^i,$$

where $\tau_A^0 = I_{\mathcal{U}}$ and $\tau_A^{i+1} = \tau_A^i \circ \tau_A$. In other words, τ_A^{*n} is the n-th approximation of τ_A's *Neumann series* well-known in operator theory.

It was proved in [7] that the limit in (2) always exists and the convergence is strong, resulting in an isometry. This result is very surprising, for the monoidal category $(\mathbf{Iso}, \oplus)$ does not even resemble to an algebraic theory and yet, it has a trace completely analogous to iteration in matrix iteration theories. Also, the ring operations addition and multiplication performed in matrix composition and sum are over the *field* $\mathbb{C}$, not over the trivial Boolean *semiring* $\mathbf{B}$. The Kleene formula (2) does not work in the whole category $(\mathbf{FdHilb}, \oplus)$ [21], and there appears to be no way to find a reasonable trace for this monoidal category.

The additive style quantum trace explains how the flow of quantum information can be controlled in a quantum flowchart algorithm [23] or a Turing machine [7]. At the moment its consistency is only proved for finite dimensional Hilbert spaces, which is insufficient to explain the semantics of general Turing machines with an infinite number of tape cells (or the semantics of a von Neumann style analog quantum computer architecture having an infinite memory component). Further analysis is needed to generalize the quantum trace for separable (i.e. countably infinite dimensional) Hilbert spaces. This issue is of utmost importance in the logical design of quantum computers.

At this point the reader might wonder why the very basic monoidal category $(\mathbf{Set}, \times)$ is missing from the above list of examples. What is a reasonable trace for this category, or a suitable extension of it? At the moment this question is still unanswered. Trace cannot simply be adopted from $(\mathbf{Rel}, \times)$, because the trace of a function might turn out to be a relation only. Thus, a deterministic computation would trigger a nondeterministic one, which we certainly do not want to allow. The author proved in [4] that the category with (delayed) circular feedback freely generated from $(\mathbf{Set}_{fin}, \times)$ is the category $Sim(\mathbf{Set}_{fin}, \times)$ of simulation equivalent finite state deterministic Mealy automata. No delay-free model is currently known.

3 Looking at trace in the Hungarian way

In this section we relate the trace operation in $(\mathbf{Rel}_{fin}, +)$ to bipartite graph matchings, and show how the trace of a relation $R : U + A \to U + B$ can be constructed by the help of the well-known Hungarian method. We also take a closer look at Selinger's construction [22] of embedding the category $(\mathbf{Rel}, +)$ into $(\mathbf{Rel}, \times)$, and interpret this embedding as a network flow problem. Let $R : U + A \to U + B$ be a relation over finite sets. The standard graph representation of R is a bipartite graph $G_R = (U + A, U + B)$. See Fig. 6. Extend G_R by edges connecting each vertex $u \in U$ in bipartition $U + A$ with the corresponding u in $U + B$. See Fig. 7. Let G_R^U denote the resulting graph. Consider the collection of newly added edges as a matching M in G_R^U, and construct the relation $R_M : A \to B$ in the following way. For each $(a, b) \in A \times B$, put (a, b) in R_M iff there is an augmenting path in G_R^U from a to b with respect to M. Recall from [18] that an augmenting path is an M-alternating path starting and ending at vertices not covered by M. It is

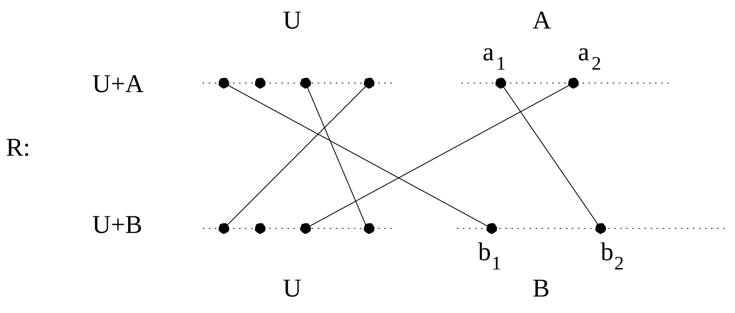


Figure 6: The graph of relation R

immediate by the definitions that

$$(a,b) \in Tr^U_{A,B}R \text{ iff } (a,b) \in R_M.$$

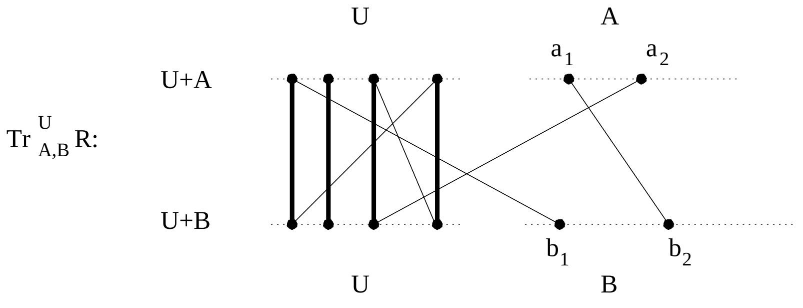


Figure 7: The graph G^U_R with matching M

Thus, in order to calculate $Tr^U_{A,B}R$, it is sufficient to apply the Hungarian method on the graph G^U_R with matching M to find a maximum matching in G^U_R. This is done by looking at the Hungarian forest F obtained at the final stage of the algorithm and see which vertex pairs (a,b) are in the same tree of F. See [18] for the details of the Hungarian method.

The functorial embedding of $(\mathbf{Rel},+)$ into $(\mathbf{Rel},\times)$ by Selinger [22] is yet another example of traced monoidal categories emerging naturally in combinatorial optimization problems. Let $R : A \to B$ be a relation between finite sets A and B, and consider the bipartite graph $G_R = (A,B)$ corresponding to R. Interpret G_R as a network flow problem [18] by assigning non-negative integers to the vertices in A and B. The assignment $f_A : A \to \mathbb{N}$ specifies the *supply* of some merchandise available at each vertex (warehouse) $a \in A$, while the assignment $f_B : B \to \mathbb{N}$ captures the *demand* on the B side for the same merchandise. Assume that

$$t_f = \sum_{a\in A} f_A(a) = \sum_{b\in B} f_B(b)$$

holds, so that the total supply meets the total demand.

The task is to deliver all the goods from side A to side B (in one round) along the roads (edges) between the two sides according to G_R. See Fig. 8. Clearly, a

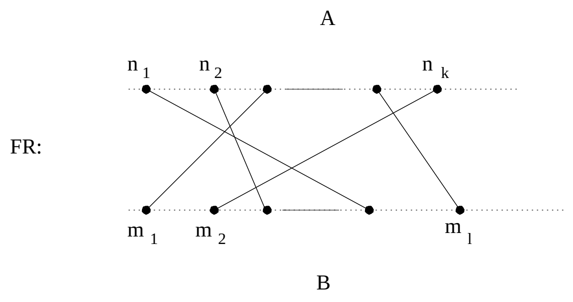


Figure 8: The network flow problem for R

solution to the problem is an assignment $\rho : E(G_R) \to \mathbb{N}$ satisfying the conditions

$$f_A(a) = \sum_{b' \in B} \rho(a, b') \quad \text{and} \quad \sum_{a' \in A} \rho(a', b) = f_B(b)$$

for all $a \in A$ and $b \in B$.

The functor $F : (\mathbf{Rel}, +) \to (\mathbf{Rel}, \times)$ is now defined as follows.

On objects: For every set A, $FA = [A \to \mathbb{N}]_{fin}$, where $[A \to \mathbb{N}]_{fin}$ denotes the set of functions $f : A \to \mathbb{N}$ by which $f(a) = 0$ for all but finitely many $a \in A$.

On morphisms: If $R : A \to B$ is a relation, then for every $f_A \in FA$ and $f_B \in FB$,

$$(f_A, f_B) \in FR \ \text{ iff the network flow problem } (f_A, f_B) \text{ in } G_R \text{ has a solution.}$$

Note that the concrete problem (f_A, f_B) is finite, because both f_A and f_B are in $[A \to \mathbb{N}]_{fin}$.

It was proved in [22] that F is a functorial embedding of traced monoidal categories. Selinger has also proved that there exists no such embedding of $(\mathbf{Rel}_{fin}, +)$ into $(\mathbf{Rel}_{fin}, \times)$. In particular, for the restriction of our concrete embedding F to $(\mathbf{Rel}_{fin}, +)$, one cannot assume that the supply and demand numbers assigned by f_A and f_B to the vertices of G_R remain under a fixed upper bound.

4 Free traced monoidal categories

In general, a *coherence result* for a subcategory μ of monoidal categories is about establishing a left adjoint for a forgetful functor F from the category of μ-monoidal categories into an appropriate syntactical category, and providing a graphical characterization of the free monoidal μ-categories so obtained. For some typical examples, see [19, 20, 17, 24, 1, 2]. In this section we briefly overview the construction of the free traced monoidal category generated by a set of morphisms (variables) presented in [6]. We shall maintain the assumption that the monoidal category to be constructed is strict, even though it is very simple to modify the construction to obtain the non-strict free traced monoidal categories. Our way of choosing the forgetful functor F differs from the method followed e.g. in [17], where the category structure was still preserved. We go one step further in forgetting, and preserve only the alphabet structure of morphisms. To remain consistent with set theory, assume that the hom-sets of our monoidal categories are indeed sets.

For a class O of object variables, a *doubly ranked alphabet* or *monoidal signature*

$$\Sigma = (O, M, r)$$

[14] consists of a set M of *morphism variables* and a mapping r which assigns for each variable $f \in M$ a *domain* $dom(f)$ and a *codomain* $cod(f)$, which are finite sequences (strings) over O. The pair $(dom(f), cod(f))$ is called the *rank* of f. As it is natural, we write $f : dom(f) \to cod(f)$. In case O is the set $\mathbb{N}$, the standard concept of doubly ranked alphabet is recaptured. An *alphabet mapping* between ranked alphabets $\Sigma = (O, M, r)$ and $\Delta = (O', M', r')$ is a mapping ϕ, which assigns to each object variable A in O an object variable ϕA in O' and to each morphism variable $f : u \to v$ in M a morphism variable $\phi f : \phi u \to \phi v$ in M', where ϕ is extended to strings in O^* in the natural way. Thus, an alphabet mapping preserves the rank of morphism variables.

Every (strict) monoidal category $\mathcal{M}$ can trivially be considered as a ranked alphabet $\Sigma = \mathcal{AM}$ in which the object variables are the objects of $\mathcal{M}$ (denoted by O_M as a concrete monoid structure). As an important twist, however, the empty string ϵ in O_M^* must be identified with the object I in O_M. The morphism variables of Σ with rank $u \to v$ are simply all the morphisms $|u| \to |v|$ in $\mathcal{M}$, where $|u|$ and $|v|$ are the evaluations of u and v in the given monoid structure on O_M. We shall use the distinctive subscript $f_{u\to v}$ to refer to the morphism variable $u \to v$ that is actually the morphism $f : |u| \to |v|$. This is necessary in order to define the domain and codomain of variables in a unique way. Since u and v are finite strings, the collection of morphism variables $u \to v$ remains a set.

If $F : \mathcal{M} \to \mathcal{M}'$ is a monoidal functor, then $\mathcal{A}F : \mathcal{AM} \to \mathcal{AM}'$ is the alphabet mapping ϕ by which $\phi A = FA$ and

$$\phi f_{u\to v} = Ff_{Fu\to Fv}$$

for every morphism $f : |u| \to |v|$ in $\mathcal{M}$, where F is extended to strings of objects in the obvious natural way.

With the above definition we have established the functor

$$\mathcal{A} : \mathbf{MonCat} \to \mathbf{Alph}$$

between the category of monoidal categories and that of ranked alphabets. Our aim is to provide a left adjoint $\mathcal{G}$ for the functor $\mathcal{A}$, when restricted to the subcategory **TraMon** of traced monoidal categories. In algebraic terms this amounts to constructing the traced monoidal category freely generated by some doubly ranked alphabet Σ. In order to keep the discussion simple, we shall assume that $O = \mathbb{N}$ that is, our categories are single-sorted. Then the free traced monoidal category generated by a doubly ranked alphabet is essentially the category of flowchart schemes as described in [1]. The only significant difference arises from the absence of the axiom:

$$Tr^U_{I,I} 1_U = 1_I,$$

where $1_U : U \to U$ denotes the identity morphism. In the single-sorted setting this axiom is equivalent to

$$\uparrow 1_1 = 1_0,$$

which was imposed as axiom S6 in [1]. (Read $\uparrow$ simply as $Tr^1_{1,1}$.) If this axiom is not present, then the policy with respect to the so called loop vertex in schemes changes as follows: every subexpression $\uparrow 1_1$ contributes a separate loop vertex $0 \to 0$ to the scheme being constructed. Thus, loop vertices multiply during the construction, as opposed to the policy applied in [8, 1] that there is a unique loop vertex in each scheme. (In other words, loop vertices do not multiply.) This straightforward change covers the whole impact of adding axiom $S6$ to the standard trace monoidal category axioms. The change is clearly visible e.g. in the category $(\mathbf{FdVect}_K, \otimes)$, where the sequence of morphisms

$$Tr^{2^0}_{2^0,2^0} 1_{2^0},\ Tr^{2^1}_{2^1,2^1} 1_{2^1},\ Tr^{2^2}_{2^2,2^2} 1_{2^2}, \ldots, Tr^{2^n}_{2^n,2^n} 1_{2^n}, \ldots$$

produces the sequence of elements

$$1,\ 2 = 1+1,\ 2^2 = 2 \cdot 2, \ldots, 2^n = 2 \cdot 2 \cdot \ldots \cdot 2, \ldots \text{ in } K.$$

The same dilemma, whether to multiply the loop vertices or not in the axiomatization of schemes, occurred already to Bloom and Ésik when writing the paper [8]. The author knows this from Zoltán himself, who told him in 1986 that they decided not to multiply the loop vertices because they only had algebraic theories in mind for possible interpretations. Otherwise the issue was trivial. Since Zoltán used to be the advisor of the author in previous years, the loop vertices did not multiply in the axiomatization [1] either.

Rather than giving the formal definition of Σ-schemes for a doubly ranked alphabet Σ, we just show an example scheme $S : 4 \to 1$ in Fig. 9. The scheme S has 4 input channels and 1 output channel. The alphabet Σ consists of the morphism variables $h : 2 \to 1$ and $g : 1 \to 1$. A variable occurrence in S is a box labelled by that variable. Boxes have numbered input and output ports (numbered from left to right), with as many input (output) ports as the domain (respectively, codomain) of the corresponding variable. The input channels have exactly one output port, while the output channels have a single input port. Each output port in the scheme is connected to exactly one input port, and every input port is the endpoint of a single edge coming from an output port. In addition, there are an arbitrary number of isolated loop vertices (none in our scheme S), which vertices have no input or output ports and are labelled by the special symbol $\perp$ not in Σ.

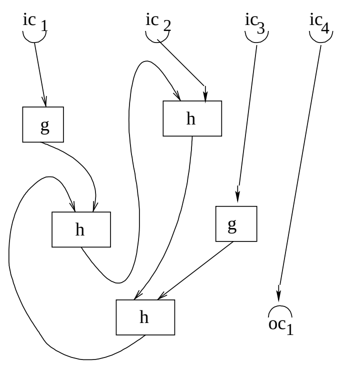


Figure 9: A Σ-scheme $4 \to 1$

Two Σ-schemes are *isomorphic* if they are isomorphic as directed graphs by an isomorphism ϕ which preserves the labeling of the boxes, and the input/output channels. Furthermore, ϕ must respect the numbering of the ports in boxes as well.

The traced monoidal category $Sch(\Sigma)$ of Σ-schemes uses the graph operations disjoint union for tensor, gluing schemes by their input/output channels for composition, and adding an edge from the first output channel to the first input channel for feedback, bypassing the pair of the connected channels themselves. Whenever feedback connects two channels that are already connected to each other in the opposite direction, a new instance of the loop vertex is created and added to the scheme. The interpretation of the identity and symmetry morphisms is straightforward, using straight lines connecting the input channels to the output ones. See [1, 6] for details. With this interpretation, the category $Sch(\Sigma)$ is that of isomorphism classes of Σ-schemes.

If Σ is not single-sorted, then the only modification to the above description is that ports are also labelled by the object variables in such a way that for every box labelled by morphism variable f, the sequence of object variables corresponding to the input (output) ports read from left to right is $dom(f)$ (respectively, $cod(f)$). Obviously, edges must respect the labeling of the ports. The scheme is $u \to v$ if the sequence of input (output) channel labels is u (respectively, v).

In order to interpret a scheme $S : x \to y$ in a traced monoidal category $\mathcal{C}$, one must first assign an object ϕA in $\mathcal{C}$ to each object variable A occurring in S, and concrete morphisms $\phi u \to \phi v$ in $\mathcal{C}$ to each morphism variable $u \to v$ occurring in S as a box label. Then represent S in normal form in the following way. Take the tensor (disjoint union) of the boxes in S, together with some straight lines, apply a suitable permutation from either side, and create the edges of S using feedback. Finally, copy this procedure at the level of semantics in the category $\mathcal{C}$. The result is a morphism $\phi x \to \phi y$, and it will not depend on the concrete syntactical normal form chosen, as long as we do not introduce a cycle of straight lines with feedback in the normal form. See again Fig. 7 in Section 3. This should be avoided by taking only a minimum number of straight lines in the normal form, unless there are loop vertices in the scheme. Each loop vertex $\perp_A$ counts as such a "looping" feedback, which corresponds semantically to $\uparrow^A 1_A$ with the chosen interpretation of the object variable A. Again, see [1, 6] for details.

We shall be interested in two more axioms in Section 5.

$$Acc \text{ (accessibility)}: \quad f = g \quad \text{for } f, g : I \to A, \text{ and}$$

$$Ter \text{ (termination)}: \quad f = g \quad \text{for } f, g : A \to I.$$

By $Trace_{Biacc}$ we shall mean the set of traced monoidal category axioms extended by Acc and $Term$.

Notice that imposing the axioms Acc and $Term$ prompts the addition of constants $0_A : I \to A$ and $0^A : A \to I$ to the traced monoidal language. These constants must then satisfy the following further monoidal axioms:

$$0_A \circ f = 0_B \quad \text{for } f : A \to B,$$

$$0_A \otimes 0_B = 0_{A \otimes B},$$

$$(0_A \otimes 1_A) \circ c_{A,A} = 1_A \otimes 0_A,$$

and the dual counterparts of these axioms for 0^A. See [1, 13]. Regarding schemes as morphisms in the corresponding free category $Sch_{Biacc}(\Sigma)$, one must lift the condition that for each input port of a box (or output channel) there exists an edge arriving at that port, and dually, there need not exist an edge going out from any particular output port. (The ignored ports do not cease to exist, though, they just become idle.) All schemes must be biaccessible, however. Recall from [1, 8, 9] that a Σ-scheme S is *accessible* if each *box* (i.e. not necessarily the output channels) can be reached from at least one input channel through a directed path in S. Dually, S is *terminating* if from each box there exists a path to at least one output channel. Scheme S is *biaccessible* if it is both accessible and terminating. The axiomatization statement for $Sch_{Biacc}(\Sigma)$ as a free category follows from the yet more general axiomatization presented in [1].

5 The completeness result

In this section we take a closer look at the definition of a given monoidal category being complete for the traced monoidal category axioms extended by the axioms Acc and $Term$. We also show that already the initial single-sorted monoidal category satisfying the axioms $Trace_{Biacc}$, the category $(\mathbf{Pin}_{\mathbb{N}}, +)$ of finite partial injections over the objects $\mathbb{N}$ as a subcategory of $(\mathbf{Rel}_{\mathbb{N}}, +)$ is complete for $Trace_{Biacc}$.

The reader might have the impression that this statement is trivial, since $(\mathbf{Pin}_{\mathbb{N}}, +)$, being initial (with or without the additional constants $0_1 : 1 \to 0$ and $0^1 : 0 \to 1$), is present in the category of biaccessible schemes as a subcategory. While this is certainly true, it does not imply that $(\mathbf{Pin}_{\mathbb{N}}, +)$ is complete for $Trace_{Biacc}$. In general, there could be a lot of identities valid in the initial algebra of an equational class that are not valid in the free algebras of that class generated by several variables. For example, the initial D-algebra in [1], as a monoidal category, coincides with the initial algebraic theory, and it can be embedded in the single sorted monoidal category of cycle-free schemes with junctions allowed. (Add a constant $d : 2 \to 1$ to the single-sorted monoidal language, or $d_A : A \otimes A \to A$ in general, as the "diagonal". See [1, 13].) This category is the free D-algebra generated by its boxes, yet, it is very far from being an algebraic theory.

The motivation for our study is the observation made in [12] that the category $(\mathbf{FdVect}_K, \otimes)$ is complete for the traced monoidal axioms, whenever the field K has an infinite characteristic (i.e. the elements $1, 1+1, \ldots, 1+1+\ldots+1, \ldots$ are all distinct.) The authors of [12] rely on the following understanding of completeness.

> If two networks (schemes) have the same value under all interpretations over K, then they are isomorphic.

Our more formal understanding of interpretation and completeness is the following. Consider the doubly ranked alphabet $\Sigma = (O, M, r)$ (fixed for the rest of the paper) in which O is a countably infinite set and $M(p, q)$ is also countably infinite for each $p, q \in O^*$. In other words, we have as many object and morphism variables in Σ as we want in order to use them for labeling schemes. Then a *traced monoidal identity* is a pair $(\mathbf{p}, \mathbf{q})$ of Σ-schemes $u \to v$ for some $u, v \in O^*$. As usual, we write $\mathbf{p} = \mathbf{q}$ for the pair $(\mathbf{p}, \mathbf{q})$. Accordingly, by a $Trace_{Biacc}$ identity we mean a pair of biaccessible schemes.

Definition 2. A $Trace_{Biacc}$ identity $\mathbf{p} = \mathbf{q}$ is *valid* in a traced monoidal category $\mathcal{C}$ if for every alphabet mapping $\phi : \Sigma \to \mathcal{AC}$,

$$(\mathcal{G}\phi)\mathbf{p} = (\mathcal{G}\phi)\mathbf{q}.$$

Recall that $\mathcal{G} : \mathbf{Alph} \to \mathbf{TraMon}$ is the left adjoint of the forgetful functor $\mathcal{A}$. In the present case of biaccessible identities, take $\mathcal{G}$ to be the left adjoint of the appropriate counterpart of $\mathcal{A}$.

Definition 3. A traced monoidal category $\mathcal{C}$ satisfying the additional axioms Acc and $Term$ is *complete* for the traced monoidal axioms $Trace_{Biacc}$ if for every valid identity $\mathbf{p} = \mathbf{q}$ in $\mathcal{C}$, the biaccessible schemes $\mathbf{p}$ and $\mathbf{q}$ are isomorphic.

There is a slight problem with Definitions 2 and 3, which is highlighted by the following example.

Example Let Bi denote the axiom

$$Bi : \quad f = g \quad \text{for } f, g : I \to I,$$

and add it to the the traced monoidal ones to obtain the system $Trace_{Bi}$. Consider the obvious single-sorted traced monoidal category $(\mathbf{Bi}, +)$ of bijections $n \to n$ as a subcategory of $(\mathbf{Rel}_{fin}, +)$. Is it complete for $Trace_{Bi}$? One could immediately answer no, since e.g. Acc and Ter are valid in $(\mathbf{Bi}, +)$, yet they are not provable from $Trace_{Bi}$. This answer is not fair, however, because the axioms Acc and Ter

are just partially meaningful in $(\mathbf{Bi},+)$, and when they are, they are equivalent to Bi.

The anomaly arises from the fact that categories are not simply multi-sorted algebras. Their hom-sets could be empty. Since we defined the alphabet Σ very generously, we must assign a morphism $|\phi p| \to |\phi q|$ to each morphism variable (box) $f : p \to q$ in Σ in order to create an alphabet mapping $\phi : \Sigma \to \mathcal{A}(\mathbf{Bi},+)$. There will not be any, however, unless all object variables are mapped into $I = 0$ by ϕ. Indeed, if $A \neq I$ was mapped into say 1 (i.e. $\phi A = 1$), then – since the empty string $I = \epsilon$ goes automatically to 0 – the morphism variables $\epsilon \to A$ could not be mapped anywhere, for $\mathbf{Bi}(0,1) = \emptyset$. One could (and must) get around this problem by including a "dummy" morphism in each hom-set of $(\mathbf{Bi},+)$ and define the traced monoidal operations on these morphisms e.g. in the strict way (always resulting in the dummy morphism if either of the arguments is dummy). One may even extend $(\mathbf{Bi},+)$ to the traced monoidal category of partial injections. The extension will not make the problem go away, though, because the axioms Acc and Ter would still remain valid in the extended categories but not in the quotient of $Sch(\Sigma)$ by Bi. This fact is already justifiable, however, since "bases are loaded" in the extended categories.

The above discussion shows that the anomaly of the Example is rooted in the polymorphic nature of the categorical language used to specify equational axioms (identities), and more analysis is required to provide a satisfactory explanation. For the moment, however, we shall rely on the straightforward solution of introducing the dummy morphisms if necessary, so at least we escape the contradiction arising from possibly empty hom-sets. Thus, the category $(\mathbf{Bi},+)$ is presently not complete for $Trace_{Bi}$.

Theorem 1. *The category* $(\mathbf{Pin}_{fin},+)$ *is complete for the axioms* $Trace_{Biacc}$.

Proof. Let $S, R : u \to v$ be two biaccessible Σ-schemes. We must find an interpretation ϕ of the alphabet Σ in $(\mathbf{Pin}_{fin},+)$ that distinguishes these two schemes, provided that they are not isomorphic. Obviously, we can assume that both u and v are different from $I = \epsilon$. We can also assume, without loss of generality, that there exists an input-output path in at least one of S and R passing through at least one box. Indeed, otherwise it is trivial to separate S and R directly in $(\mathbf{Pin}_{fin},+)$ by interpreting each object variable A as $\phi A = 1$ and each morphism variable $f : p \to q$ as $\phi f = 0^{\phi(p)} + 0_{\phi(q)}$, i.e., the totally undefined injection.

First let us assume that each box in S and R is labelled by the same morphism variable $f : p \to q$ in M. Since the schemes are biaccessible, $p \neq \epsilon$ and $q \neq \epsilon$. Let us spell out the trajectory of an input-output path in Σ-schemes. As in standard graph theory, it is an alternating sequence

$$q_0, e_1, (p_1, q_1), e_2, \ldots, (p_n, q_n), e_n, p_{n+1} \tag{3}$$

of vertices and edges, starting from and ending at a vertex (port), such that each edge e_i in the sequence is incident with the vertex (output port q_{i-1}) immediately preceding it and with the vertex (input port p_i) immediately following it. Remember that each port is numbered, and it is also labelled by an object variable in O.

For simplicity, the number assigned to the port of an input/output channel is the serial number of that channel.

For a path α of the form (3), $\psi(\alpha)$ will denote the "trace" of α:

$$(l_0, B_0), ((m_1, A_1), (l_1, B_1)), \ldots, ((m_n, A_n), (l_n, B_n)), (m_{n+1}, A_{n+1}).$$

In this sequence, $1 \leq l_0 \leq \mathit{length}(u)$ and $1 \leq m_{n+1} \leq \mathit{length}(v)$ identify the serial number of the input channel the output port of which is q_0 and that of the output channel identified by port p_{n+1}. The pair (m_i, A_i) $((l_i, B_i))$ consists of the serial number and object variable corresponding to the input port p_i (respectively, output port q_i). By definition, $B_i = A_{i+1}$ for every $0 \leq i \leq n$.

Now we turn to defining the separating interpretation ϕ. For all object variables $A \in O$, let $\phi(A) = k$, where k is a sufficiently large integer, the magnitude of which will be specified later. In this way we can think of a morphism variable $f : p \to q$ as a single-sorted one $f_k : k \cdot m \to k \cdot l$, where $m > 0$ and $l > 0$ are the length of p and q, respectively. The morphism $\phi f : k \cdot m \to k \cdot l$ will be the partial injection by which

$$(m_i - 1) \cdot k + i \mapsto (l_i - 1) \cdot k + i + 1 \tag{4}$$

for every $1 \leq i < k$ and appropriately chosen numbers $1 \leq m_i \leq m$, $1 \leq l_i \leq l$.

It is easy to visualize the partial injection $(\mathcal{G}\phi)S$ for scheme S using the Hungarian method discussed in Section 3. Every edge e in S counts as k parallel edges in the corresponding single-sorted scheme S_k, where each port is "multiplied" by k. Let M be the set of all edges in S_k. The set M becomes a matching if we consider S_k as a bipartite graph on the *ports* as vertices rather than one on the boxes. Add ϕf to S_k as edges *inside* the boxes, and let $S_k(\phi)$ denote the resulting bipartite graph. Intuitively, whenever the control reaches a box at input port m_i in "dimension" $i < k$, it will leave at output port l_i in dimension $i + 1$. Thus, $(\mathcal{G}\phi)S$ can be obtained by looking for "dual" augmenting paths in $S_k(\phi)$ with respect to matching M, that is, alternating paths starting and ending with M-positive edges incident with the set of leaves of $S_k(\phi)$ consisting of the input-output channels (ports). If such a path exists between input channel l in dimension i and output channel m in dimension j, then $(l, i) = (l - 1) \cdot k + i$ is mapped to $(m, j) = (m - 1) \cdot k + j$ according to $(\mathcal{G}\phi)S$. (Mind that an input channel is in fact an output port and an output channel counts as an input port.)

Observe that the procedure of calculating $(\mathcal{G}\phi)S$ by the Hungarian method would be exactly the same if we were to interpret S under relations, rather than partial injections.

What we need in order to finish the proof is an understanding of schemes S and R being isomorphic. It is a standard graph theory argument that S and R are isomorphic iff, for each input-output channel pair (l, m), whenever there exists a path α connecting l with m in the one scheme (say S), then there exists such a path β in R, too, so that $\psi(\alpha) = \psi(\beta)$. (Remember that ψ denotes the trace of a path.) Indeed, the numbered ports, together with the biaccessibility restriction, make the graph isomorphism test for S and R straightforward.

Now let us assume that S and R are not isomorphic. Then there exists an input-output path α in say S such that its trace $\psi(\alpha)$ is missing from the traces of paths in R. Clearly, the length of α remains under a fixed number that depends on the size of S and R only. Choose α to be shortest among such paths, and let N denote the length of α. For $k > N$, set the numbers m_i and l_i in (4) according to the parameters of α up to dimension N, and arbitrarily for $N \leq i \leq k$. It is clear that this choice of ϕf will distinguish S and R, since only $(\mathcal{G}\phi)S$ will take the input channel l_0 in dimension 1 to output channel m_N in dimension $N+1$.

If there are several morphism variables assigned as labels to the boxes of S and R, then the proof can be augmented by adding a distinctive "preamble" injection to the interpretation of each morphism variable in each dimension, and driving the control through this preamble. Details are straightforward and left to the reader. The proof is now complete. □

Corollary 1. *The traced monoidal category* $(\mathbf{Rel}_{fin}, +)$ *is complete for* $Trace_{Biacc}$.

Proof. Indeed, $(\mathbf{Pin}_{fin}, +)$ can be embedded in $(\mathbf{Rel}_{fin}, +)$ as a subcategory. Moreover, bases are loaded in both categories, that is, no hom-set is empty. Therefore any valid identity $\mathbf{p} = \mathbf{q}$ in $(\mathbf{Rel}_{fin}, +)$ is valid in $(\mathbf{Pin}_{fin}, +)$ as well. Consequently, by Theorem 1, $\mathbf{p}$ and $\mathbf{q}$ are isomorphic as biaccessible schemes. □

6 Conclusion

We have shown that the traced monoidal category $(\mathbf{Pin}_{fin}, +)$ of finite partial injections is complete for the extension of the traced monoidal axioms by two identities that reflect the biaccesible property of schemes as morphisms in the category freely generated by a given monoidal signature. The proof was a classical separation argument, showing that if two biaccessible schemes are not isomorphic, then there exists an interpretation in terms of finite partial injections which distinguishes them. Since $(\mathbf{Pin}_{fin}, +)$ is the initial single-sorted traced monoidal category satisfying the given axioms, our result holds for every traced monoidal category into which $(\mathbf{Pin}_{fin}, +)$ can be embedded as a sub-monoidal category, and in which the identities Acc and Ter are valid.

References

[1] Bartha, M. A finite axiomatization of flowchart schemes. *Acta Cybernetica*, 8(2):203–217, 1987.

[2] Bartha, M. An equational axiomatization of systolic systems. *Theoret. Comput. Sci.*, 55:265–289, 1987.

[3] Bartha, M. An algebraic model of synchronous systems. *Information and Computation*, 97:97–131, 1992.

[4] Bartha, M. Simulation equivalence of automata and circuits. In: Csuhaj-Varjú, E. and Ésik, Z. editors, *12th International Conference on Automata and Formal Languages*, Balatonfüred, Hungary, Local Proceedings, pages 86–99, 2008.

[5] Bartha, M. Equivalence relations of Mealy automata. In: Bordihn, H., Freund, R., Holzer, M., Kutrib, M., and Otto, F. editors, *First Workshop on Non-Classical Models of Automata and Applications*, Wroclaw, Poland, Proceedings: books@ocg.at 256, Austrian Computer Society, pages 31–45, 2009.

[6] Bartha, M. The monoidal structure of Turing machines. *Mathematical Structures in Computer Science*, 23(2):204–246, 2013.

[7] Bartha, M. (2011) Quantum Turing automata. In Löwe, B. and Winskel, G. editors, *8th International Workshop on Developments in Theoretical Computer Science (DCM 2012)*, pages 17–31, Electronic Proceedings in Theoretical Computer Science, 2014.

[8] Bloom, S.L. and Ésik, Z. Axiomatizing schemes and their behaviors. *J. Comput. System Sci.* 31:375–393, 1985.

[9] Bloom, S.L. and Ésik, Z. *Iteration Theories: The Equational Logic of Iterative Processes.* Springer-Verlag, Berlin, 1993.

[10] Căzănescu, V.E. and Ştefănescu, Gh. Towards a new algebraic foundation of flowchart scheme theory. *Fundamenta Informaticae*, 13:171–210, 1990.

[11] Ésik, Z. Identities in iterative and rational theories. *Computational Linguistics and Computer Languages*, 14:183–207, 1980.

[12] Hasegawa, M., Hofmann, M. and Plotkin, G. Finite dimensional vector spaces are complete for traced symmetric monoidal categories. In *Pillars of Computer Science: Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*, Springer LNCS 4800, pages 367-385, February 2008.

[13] Hasegawa, M. Bialgebras in Rel. In *26th Conference on the Mathematical Foundations of Programming Semantics (MFPS XXVI), May 2010, Ottawa.* Electronic Notes in Theoretical Computer Science 265, pages 337-359, 2010.

[14] Joyal, A. and Street, R. The geometry of tensor calculus I. *Advances in Mathematics*, 88(1):55–112, 1991.

[15] Joyal, A., Street, R. and Verity, D. Traced monoidal categories. *Math. Proc. Camb. Phil. Soc.*, 119:447–468, 1996.

[16] Katis, P., Sabadini, N. and Walters, R.F.C. Feedback, trace, and fixed-point semantics. *Theoret. Informatics Appl.*, 36:181–194, 2002.

[17] Kelly, G. M. and Laplaza, M. L. Coherence for compact closed categories. *J. Pure Appl. Algebra*, 19:193–213, 1980.

[18] Lovász, L. and Plummer, M.D. *Matching Theory.* North Holland, 1986.

[19] Mac Lane, S. *Categories for the Working Mathematician.* Springer-Verlag, 1971.

[20] Mac Lane, S. and Paré, R. Coherence for bicategories and indexed categories. *J. Pure and Appl. Algebra*, 37:59–80, 1985.

[21] Malherbe, O., Scott, P.J., and Selinger, P. Partially traced categories. *Journal of Pure and Applied Algebra*, 216(12):2563–2585, 2011.

[22] Selinger, P. A note on Bainbridge's power set construction. Manuscript 10 pages, 1998.

[23] Selinger, P. Towards a quantum programming language. *Mathematical Structures in Computer Science* 14:527–586, 2004

[24] Selinger, P. A survey of graphical languages for monoidal categories. In Coecke (editor), *New Structures for Physics, Lecture Notes in Physics* 183, Springer-Verlag, Berlin, 2009.

[25] Ştefănescu, Gh. *Network Algebra.* Series in Discrete Mathematics and Theoretical Computer Science, Springer, Heidelberg, 2000.

Acta Cybernetica **23** (2017) 349–369.

Regular Expressions for Muller Context-Free Languages*

Kitti Gelle[a] and Szabolcs Iván[a]

Abstract

Muller context-free languages (MCFLs) are languages of countable words, that is, labeled countable linear orders, generated by Muller context-free grammars. Equivalently, they are the frontier languages of (nondeterministic Muller-)regular languages of infinite trees.

In this article we survey the known results regarding MCFLs, and show that a language is an MCFL if and only if it can be generated by a so-called $\mu\eta$-regular expression.

Keywords: Muller context-free languages, well-ordered induction, regular expressions

1 Introduction

A word, also called "arrangement" in [9], is the isomorphism type of a labeled linear order. Thus, this notion is a generalization of finite and ω-words, permitting e.g. labelings of the integers of the rationals.

Finite automata on ω-words have by now a vast literature, see [21] for a comprehensive treatment. Finite automata acting on well-ordered words longer than ω have been investigated in [1, 6, 7, 24, 25], to mention a few references. Recently, the theory of automata on well-ordered words has been extended to automata on all countable words, including scattered and dense words. In [2, 5, 4], both operational and logical characterizations of the class of languages of countable words recognized by finite automata were obtained.

Context-free grammars generating ω-words were introduced in [8] and subsequently studied in [3, 19]. Context-free grammars generating arbitrary countable words were defined in [10, 11]. Actually, two types of grammars were defined, context-free grammars with Büchi acceptance condition (BCFG), and context-free grammars with Muller acceptance condition (MCFG). These grammars generate the Büchi and the Muller context-free languages of countable words, abbreviated as BCFLs and MCFLs. Every BCFL is clearly an MCFL, but there exists an

*This work was supported by NKFI grant no. 108448.

[a]University of Szeged, Hungary, E-mail: {kgelle,szabivan}@inf.u-szeged.hu

DOI: 10.14232/actacyb.23.1.2017.19

MCFL of well-ordered words that is not a BCFL, for example the set of all countable well-ordered words over some alphabet. In fact, it was shown in [10] that for every BCFL L of well-ordered words there is an integer n such that the order type of the underlying linear order of every word in L is bounded by ω^n.

In this paper we survey the results on Muller context-free languages, and we give an operational ("regular-expression-like") characterization of this class.

2 Notation

2.1 Linear orderings

A (strict) *linear ordering* is a pair $(I, <)$ where $<$ is a strict total ordering on I, that is, an irreflexive, transitive and trichotomous relation. Set-theoretical properties of the domain set I, such as finiteness, membership and cardinality are lifted to the ordering $(I, <)$ thus we can say e.g. that a linear ordering is countable, finite etc. In order to ease notation, we will omit the ordering $<$ from the pair and identify the ordering $(I, <)$ with its domain I, if the ordering relation is not important or is clear from the context. In this paper we will (unless stated otherwise) only deal with countable orderings. A good reference for linear orderings is [23].

An *embedding* of a linear ordering $(I, <)$ into a linear ordering $(J, \prec)$ is a (necessarily) injective mapping $h : I \to J$ preserving the ordering, i.e. $x < y$ implying $h(x) \prec h(y)$. We write $(I, <) \leq (J, \prec)$ if I can be embedded into J. Clearly, this $\leq$ relation is a preorder (a reflexive and transitive relation) between orderings. A surjective embedding is called an *isomorphism* between the orderings I and J; if there exists an isomorphism between I and J, then they are called *isomorphic.* Isomorphism of linear orderings is an equivalence relation, the classes of which are called *order types.* Well-known isomorphism types are the order type ω of the natural numbers $\mathbb{N}$, the type $-\omega$ of the negative integers, the type ζ of integers and the type η of rationals. Since isomorphism is compatible with embeddability, we can say that an order type α can be embedded into an order type β, written as $\alpha \leq \beta$. Note that this relation is not antisymmetric in general as the orderings $(0, 1)$ and $[0, 1]$ can be embedded into each other but they are not isomorphic as the latter one has a least element while the former one does not.

The ordering $(I, <)$ is a *sub-ordering* of $(J, \prec)$ if $I \subseteq J$ and $<$ is the restriction of $\prec$ onto I. An ordering is a *well-ordering* if it has no sub-ordering of type $-\omega$, is *scattered* if it has no sub-ordering of type η, *quasi-dense* if it is not scattered and *dense* if it has at least two elements and for any $x < y$ there exists some z with $x < z < y$. All dense countable orderings have the type η, possibly enriched with either a least or a greatest element (or both). Order types of well-orderings are called *ordinals.* Amongst the class of ordinals, the embeddability relation itself is a well-ordering, moreover, each ordinal α is either a *successor ordinal* $\alpha = \beta + 1$ for some ordinal β, in which case there is no other ordinal between α and β, or is a *limit ordinal* with $\alpha = \bigvee_{\beta<\alpha} \beta$, i.e. is the least upper bound of the set of ordinals strictly smaller than α with respect to the embeddability relation $<$. Note that although the ordinals themselves form a proper class, whenever α is an ordinal, then the class of ordinals smaller than β is a set and whenever X is a set of ordinals, then their least upper bound $\bigvee X$ exists and is an ordinal. Moreover, the class of countable ordinals is the least class which contains the order types 0 and 1 and is closed under taking ω-sums, that is, taking suprema of ω-chains.

When $(I, \prec)$ is some linearly ordered index set and for each $i \in I$, $(J_i, <_i)$ is a linear order, then their *ordered sum* $(J, <) = \sum_{i \in I}(J_i, <_i)$ has the *disjoint union* $J = \{(i, j) : i \in I, j \in J_i\}$ as domain with $(i, j) < (i', j')$ if either $i \prec i'$ or $i = i'$ and $j <_i j'$. In particular, $(J_1, <_1) + (J_2, <_2)$ denotes the sum $\sum_{i \in \{1,2\}} (J_i, <_i)$. It is clear that a well-ordered sum of well-ordered orderings is well-ordered, and a scattered sum of scattered orderings is scattered. The sum operation is also compatible with the isomorphism, thus it can be extended to order types. For example, $-\omega + \omega = \zeta$ and $\eta + \eta = \eta + 1 + \eta = \eta$ where 1 is the order type of the singleton orderings; in general, n is the order type of the n-element orderings for $n \in \mathbb{N}_0 = \{0, 1, \ldots\}$. If α is the order type of I and β is the order type of each J_i, $i \in I$, then $\beta \times \alpha$ stands for the order type of the sum $\sum_{i \in I} J_i$. Hence, product of ordinals is an ordinal. Ordinals are also equipped with an exponentation operator but we will only use finite powers of the form α^n, with n being an integer, that is, $\alpha^0 = 1$ and $\alpha^{n+1} = \alpha^n \times \alpha$.

Hausdorff classified the scattered order types into an infinite hierarchy. We make use of the following variant [17] of this hierarchy: let VD_0 be the class of all finite order types, and when α is some ordinal, then let VD_α consist of the class of all order types that can be written as $\sum_{i \in \zeta} I_i$ where each I_i is a member of VD_{β_i} for some ordinal $\beta_i < \alpha$. Hausdorff's theorem states that a (countable) order type is scattered if and only if it is contained in VD_α for some (countable) ordinal α. The *Hausdorff-rank* $\mathrm{rank}(o)$ of a scattered order type o is the least ordinal α with $o \in VD_\alpha$.

2.2 Words, tree domains, trees

An *alphabet* is a finite nonempty set of symbols, or *letters*. Alphabets are usually denoted Σ, Γ in this paper, and letters are denoted by $a, b, c, \ldots$. A Σ-labeled linear ordering is a tuple $w = (\mathrm{dom}(w), \ell_w)$ where $\mathrm{dom}(w) = (I, <)$ is some linear ordering and $\ell_w : I \to \Sigma$ is the labeling function of w. We usually identify w with ℓ_w and write $w(i)$ for $\ell_w(i)$, $i \in \mathrm{dom}(w)$. Two words u and v are *isomorphic* if there is an isomorphism $h : \mathrm{dom}(u) \to \mathrm{dom}(v)$ which preserves also the labels, that is, $u(i) = v(h(i))$ for each $i \in \mathrm{dom}(u)$. A *word* over Σ is an isomorphism class of countable Σ-labeled linear orderings. For convenience, when u is a word, we take a representant of u and use the notation $\mathrm{dom}(u)$, ℓ_u referring the domain and labeling of the representant. Order theoretic properties are lifted to words. The set of all countable, ω- and finite words over Σ are respectively denoted $\Sigma^\#$, Σ^ω and Σ^*. In particular, ε denotes the empty word (having the empty set as domain). Note that as for any Σ, there is a unique η-word u such that for any $x < y \in \mathrm{dom}(u)$ and letter $a \in \Sigma$ there is some $z \in \mathrm{dom}(u)$ with $x < z < y$ and $u(z) = a$, this u being called the *perfect shuffle of* Σ, and every countable Σ-word v is a *subword* of thus u (that is, $\mathrm{dom}(v)$ is a sub-ordering of $\mathrm{dom}(u)$ and ℓ_v is the restriction of ℓ_u to $\mathrm{dom}(v)$), thus $\Sigma^{\#}$ is indeed a set. A *language* over Σ is an arbitrary subset of $\Sigma^{\#}$. Languages will usually be denoted by $K, L, \ldots$.

When $(I, <)$ is a linear ordering and for each $i \in I$, w_i is a Σ-word, then their *product* or (con)catenation is the word $w = \prod\limits_{i \in I} w_i$ with domain $\sum\limits_{i \in I} \mathrm{dom}(w_i)$ and the labeling of ℓ_w being the *source tupling* of the labelings ℓ_{w_i}, that is, for (i, j), $i \in I$, $j \in \mathrm{dom}(w_i)$ let $w(i, j)$ be $w_i(j)$. Product is extended to languages in the expected way: when $(I, <)$ is the indexing ordering and to each $i \in I$, $L_i \subseteq \Sigma^{\#}$ is a language, then $\prod\limits_{i \in I} L_i$ consists of those words $\prod\limits_{i \in I} w_i$ where $w_i \in L_i$ for each $i \in I$. Binary products are simply written as $u \cdot v$ or $K \cdot L$, or simply uv and KL. When α is some order type, then L^α is the language $\prod\limits_{i \in \alpha} L$, and L^* stands for the union of the languages L^n, n being a natural number. Also, L^+ stands for $\bigcup\limits_{n > 0} L^n$.

A *tree domain* is a prefix closed nonempty (but possibly infinite) subset T of $\mathbb{N}^*$. That is, whenever $x \cdot i$ is in T for $x \in \mathbb{N}^*$ and $i \in \mathbb{N}$, then so is x, in which case x is the *parent* of $x \cdot i$ and $x \cdot i$ is a *child* of x. Members of T are also called *nodes* of T. If $x \cdot y \in T$ for $x, y \in \mathbb{N}^*$, then x is called an *ancestor* of $x \cdot y$ and $x \cdot y$ is a *descendant* of x, denoted $x \preceq x \cdot y$. If y is nonempty, then we talk about *proper* ancestor / descendant, denoted $x \prec x \cdot y$. Nodes of T having no child are called *leaves* of T, the other nodes are called *inner nodes* of T. Clearly, ε is a member of every tree domain.

Subsets of a tree domain T which are tree domains themselves are called *prefixes* of T. A *path* π of T is a prefix in which every node has at most one child. Clearly, to every path π there exists a unique word u_π in $\mathbb{N}^{\leq \omega} = \mathbb{N}^* \cup \mathbb{N}^\omega$ such that $\pi = \{x \in \mathbb{N}^* : x \preceq u_\pi\}$. We will use π and u_π interchargably.

When T is a tree domain and $x \in T$, then the *sub-tree domain* of T is $T|_x = \{y \in \mathbb{N}^* : xy \in T\}$. It is clear that $T|_x$ is also a tree domain, and is a path if so is T.

A (Σ-)*tree* over some alphabet Σ is a labeled tree domain t, that is, a pair $(\mathrm{dom}(t), \ell_t)$ where $\mathrm{dom}(t)$ is a tree domain and $\ell_t : \mathrm{dom}(t) \to \Sigma$ is a labeling function. Similarly to the case of words, we often identify t with ℓ_t and write $t(x)$ in place of $\ell_t(x)$ for $x \in \mathrm{dom}(t)$. Notions of tree domains (nodes, paths, sub-tree domains etc.) are lifted to trees; the *subtree* of t rooted at some node $x \in \mathrm{dom}(t)$ has the domain $\mathrm{dom}(t)|_x$ and labeling $y \mapsto t(xy)$. When $X \subseteq \mathrm{dom}(t)$ is a set of nodes of t, then $\mathrm{labels}(t, X) = \{t(x) : x \in X\}$ is the set of labels occurring on the nodes belonging to X, and $\mathrm{infLabels}(t, X)$ is the set of labels occurring infinitely many times. In particular, if π is a path of t, then a letter $a \in \Sigma$ belongs to $\mathrm{infLabels}(t, \pi)$ if and only if for each $x \in \pi$ there exists some descendant $y \in \pi$ of x with $t(y) = a$. When t is clear from the context, we just write $\mathrm{labels}(X)$ and $\mathrm{infLabels}(X)$, respectively. For any infinite path π, there exists a $\preceq$-minimal node x of π with $\mathrm{infLabels}(\pi|_x) = \mathrm{labels}(\pi|_x)$, this node x is denoted $\mathrm{head}(\pi)$. Clearly,

$\mathrm{infLabels}(\pi) = \mathrm{infLabels}(\pi|_x) \subseteq \mathrm{labels}(\pi|_x) \subseteq \mathrm{labels}(\pi)$ for every path π and node $x \in \pi$.

2.3 Muller context-free grammars and languages

A *Muller context-free grammar* or MCFG for short, is a tuple $G = (N, \Sigma, P, S, \mathcal{F})$ where N and Σ are the disjoint alphabets of *nonterminals* (or variables) and *terminals*, respectively, $S \in N$ is the *start symbol*, P is the finite set of *productions* (or rules) of the form $A \to \alpha$ with $A \in N$ and $\alpha \in (N \cup \Sigma)^+$, and $\mathcal{F} \subseteq P(N)$ is the Muller *acceptance condition.* Here $P(N) = \{N' \subseteq N\}$ is the *power set* of N.

Observe that we explicitly disallow rules of the form $A \to \varepsilon$ here; this makes the treatment of leaf labels more uniform, and as it turns out, such rules can be mimicked by introducing a fresh nonterminal I, rules $A \to I$ and $I \to I$ and adding $\{I\}$ to the acceptance condition. Nevertheless, in our examples we will make use of rules $A \to \varepsilon$ in order to help readability.

An $(N \cup \Sigma)$-tree t is *locally consistent* (with G) if it satisfies the following condition: each inner node x of t is labeled by some nonterminal A and the set of children of x is $\{x{\cdot}1, \ldots, x{\cdot}n\}$ for some integer $n > 0$ with $A \to t(x{\cdot}1)t(x{\cdot}2)\ldots t(x{\cdot}n)$ being a production in P. A locally consistent tree is *complete* if its leaves are labeled in Σ. The leaves of any tree domain are linearly ordered by the *lexicographic ordering* $<_\ell$, that is, $u <_\ell v$ if and only if $u = u_1 \cdot i \cdot u_2$ and $v = u_1 \cdot j \cdot u_3$ for some words $u_1, u_2, u_3 \in \mathbb{N}^*$ and integers $i < j$. The *frontier word* of a tree t is the word $\mathrm{fr}(t)$ having the set of leaves as domain, ordered lexicographically, and labeling inherited from t. That is, $\mathrm{dom}(\mathrm{fr}(t)) = \{x \in \mathrm{dom}(t) : x \text{ is a leaf of } t\}$ and $\mathrm{fr}(t)(x) = t(x)$ for each leaf x.

A locally consistent tree t is a *derivation tree* of G if for each infinite path π of t the set $\mathrm{infLabels}(\pi)$ belongs to the acceptance condition $\mathcal{F}$. Given a symbol $X \in N \cup \Sigma$, we let $\Delta(G, X)$ denote the set of all complete derivation trees t of G whose *root symbol* $t(\varepsilon)$ is X. We write $A \Rightarrow_G^\infty \alpha$ for a symbol $A \in N \cup \Sigma$ and a word $\alpha \in (N \cup \Sigma)^\#$ if α is the frontier word of some derivation tree of G having root symbol A. The *language generated by* G is $L(G) = \{w \in \Sigma^\# : S \Rightarrow_G^\infty w\}$.

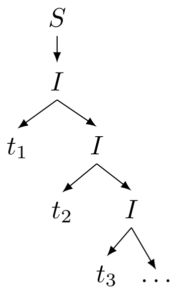
A language $L \subseteq \Sigma^\#$ of Σ-words is a *Muller context-free language*, or MCFL for short, if $L = L(G)$ for some MCFG G. In fact, Muller context-free languages are precisely the frontier languages of (nondeterministic Muller-)regular languages of infinite trees.

Example 1. If $G = (\{S, I\}, \Sigma, P, S, \{\{I\}\})$, with

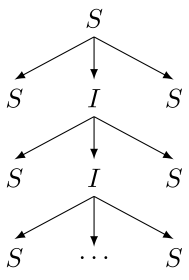
$$P = \{S \to a : a \in \Sigma\} \cup \{S \to \varepsilon, S \to I, I \to SI\},$$

then $L(G)$ consists of all the well-ordered words over Σ.

Indeed, assume $t_1, t_2, \ldots$ are derivation trees. Then so is the tree t depicted in Figure 1a with frontier word $\mathrm{fr}(t_1)\mathrm{fr}(t_2)\ldots$. Thus, $L(G)$ contains the empty word (by $S \to \varepsilon$), the words of length 1 (by $S \to a$ and $S \to b$), and is closed under taking ω-products. Since the least class of order types which contains 0, 1 and which



(a) Tree for Example 1



(b) Tree for Example 2

Figure 1: Derivation trees corresponding to Examples 1 and 2

is closed under ω-sums is the class of all countable ordinals (see e.g. [23]), $L(G)$ contains all the well-ordered words over $\{a,b\}$. For the other direction, assume t is a derivation tree having a frontier word containing an infinite descending chain $\ldots <_\ell u_2 <_\ell u_1$. Then let us define the path $v_0, v_1, \ldots$ in t: $v_0 = \varepsilon$ and v_{i+1} is $v_i \cdot 1$ if this node is an ancestor of infinitely many u_j and $v_i \cdot 2$ otherwise (which happens if v_i corresponds to the production $I \to SI$ and the node $v_i \cdot 1$ (which is labeled S) has no descendant of the form u_j at all). Note that for each u_j there exists a unique v_{i_j} such that v_{i_j} is an ancestor of u_j and v_{i_j+1} is not, since the length of the words v_i grows without a bound. Now these nodes v_{i_j} correspond to the production $I \to SI$ and $v_{i_j+1} = v_{i_j} \cdot 1$, so that the successor of v_{i_j} along the path is labeled by S. Hence $v_0, v_1, \ldots,$ is a path π in t such that $\mathrm{infLabels}(\pi)$ contains S, which is a contradiction since the only accepting set is $\{I\}$.

Example 2. If $G = (\{S, I\}, \Sigma, P, S, \{\{I\}\})$, with

$$P = \{S \to a : a \in \Sigma\} \cup \{S \to \varepsilon, S \to I, S \to SIS\},$$

then $L(G)$ consists of all the scattered words over Σ.

Indeed, the derivation tree depicted on Figure 1b shows that $L(G)$ is closed under $\omega + (-\omega)$-products of words. Since $\varepsilon \in L(G)$, the language is thus closed under ω-products and $-\omega$-products as well, thus closed under ζ-products. Since the one-letter words belong to $L(G)$, we get by Hausdorff's Theorem that $L(G)$ consists of all the scattered Σ-words.

We note that if instead of the Muller condition we define a Büchi-type acceptance condition, then we get a weaker device: the resulting class of "Büchi context-free languages" is strictly contained within the class of MCFLs, see [11, 10].

3 MSO-definable properties are decidable

The logic usually arising when dealing with "regular" structures is that of *monadic second-order* logic, or MSO. In [13] the following general decidability theorem was proved:

Theorem 1. *The following problem is decidable: given an MCFG G generating Σ-words and an MSO formula φ evaluated Σ-words, does it hold that $w \models \varphi$ for every $w \in L(G)$?*

Thus in particular, it is decidable whether G generates scattered, or well-ordered, or dense words only.

We sketch the outline of the proof. First, to each MCFG $G = (N, \Sigma, P, S, \mathcal{F})$ we associate the grammar $G' = (N \cup P, \Sigma, P', S)$ where P' contains the following set of productions:

- For each nonterminal $A \in N$, there is a production $A \to (A \to \alpha_1)(A \to \alpha_2)\ldots(A \to \alpha_n)$ in P' where $A \to \alpha_1, \ldots, A \to \alpha_n$ are the productions in P having A on their left-hand side, in some fixed order.
- For each production $A \to X_1 \ldots X_n$, there is a production $(A \to X_1 \ldots X_n) \to X_1 \ldots X_n$ in P'.

That is, we can rewrite a nonterminal to the sequence of its alternatives, and rewrite a production to its right-hand side. Thus, each nonterminal of G' has exactly one alternative (it is assumed that for each nonterminal A there is at least one production having left-hand side A), thus there is exactly one locally consistent tree of G' having root symbol S. We call this unique tree t_G the *grammar tree* of G.

Example 3. For the MCFG of Example 1, this tree t_G is depicted in Figure 2.

$$\begin{tikzcd}
 & & & S \arrow[dlll] \arrow[dll] \arrow[dl] \arrow[drrrr] & & & & & \\
S \to a \arrow[d] & S \to b \arrow[d] & S \to \varepsilon \arrow[d] & & & & & S \to I \arrow[d] & \\
a & b & \varepsilon & & & & & I \arrow[d] & \\
 & & & & & & & I \to SI \arrow[dll] \arrow[dr] & \\
 & & & & & S \arrow[dl] \arrow[d] \arrow[dr] \arrow[drr] & & & I \arrow[d] \\
 & & & & S \to a \arrow[d] & S \to b \arrow[d] & S \to \varepsilon \arrow[d] & S \to I \arrow[d] & I \to SI \arrow[d] \\
 & & & & a & b & \varepsilon & I \arrow[d] & \ldots \\
 & & & & & & & \ldots &
\end{tikzcd}$$

Figure 2: Grammar tree of the MCFG of Example 1

The cruical fact is that the grammar tree is a *regular* tree, having finitely many (more precisely, at most $|\Sigma| + |N| + |P|$) subtrees. By [22], the MSO theory of a regular tree is decidable, that is, given a regular tree t (which is t_G in our case) and an MSO formula ψ, it is decidable whether $t \vDash \psi$ holds. As t_G can be effectively constructed from G, it remains to construct a formula ψ from G and φ such that $t_G \vDash \psi$ holds if and only if for each $w \in L(G)$ we have $w \vDash \varphi$.

Informally, the formula ψ constructed in [13] has the semantics "whenever T is a derivation tree of G, its frontier word satisfies φ". Now derivation trees are encoded as subsets of $\mathrm{dom}(t_G)$ as follows: a subset $X \subseteq \mathrm{dom}(t_G)$ encodes a derivation tree of G if the following conditions all hold:

- X contains the root node.
- If some $x \in X$ is labeled by some nonterminal $A \in N$, then *exactly one* child of x belongs to X.
- If some $x \in X$ is labeled by some production $A \to \alpha$, then *all the children* of x belong to X.
- On each infinite path $\pi \subseteq X$, the set of symbols from N occurring infinitely many times belongs to $\mathcal{F}$.

These properties can be expressed in MSO and such a set encodes a derivation tree in the obvious way.

Example 4. Figure 3 shows a (part of a) derivation tree t of the grammar of Example 1 and the corresponding subset T of $\mathrm{dom}(t_G)$ (as nodes in boldface).

Then, given a set X of nodes of t_G, we can define the subset $Y \subseteq X$ of the leaves in X and the lexicographic ordering over this Y is also MSO-definable. Hence the formula "whenever X is a subset of $\mathrm{dom}(t_G)$ encoding a derivation tree, and Y is the set of leaves within X, then the word corresponding to Y satisfies φ" is expressible in MSO (moreover, is effectively computable from G and φ), thus proving Theorem 1.

4 A normal form

The general decidability theorem of the previous section does not give us an exact complexity result as model checking MSO is decidable, but nonelementary in general. In this section we give complexity results for several decision problems (and several undecidability results as well) regarding MCFLs, surveying the results of [11]. The decidable properties surveyed here are MSO-definable, thus their decidability is immediate from Theorem 1. For example, $L(G)$ is empty if and only if every member w of $L(G)$ satisfies the false formula $\downarrow$. (On the other hand, universality is not definable this way – and indeed, universality of MFCGs is already undecidable for singleton alphabets.)

(A slightly modified variant of) the normal form of MCFGs introduced in [11] is the following:

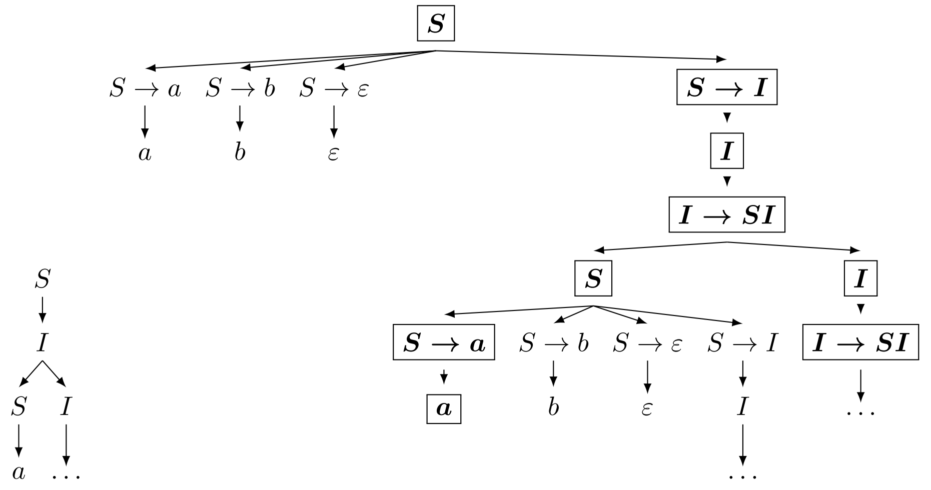


Figure 3: A part of a derivation tree t of Example 1 and the corresponding subset $\boldsymbol{T}$ of t_G

Definition 1. *An MCFG $G = (N, \Sigma, P, S, \mathcal{F})$ is in* normal form *if either P is empty, or P consists of the single production $S \to \varepsilon$, or for each $A \in N$ there exists a nonempty word w with $A \Rightarrow_G^\infty w$ and words u, v with $S \Rightarrow_G^\infty uAv$.*

Moreover, to each $F \in \mathcal{F}$ there exists a path π of some derivation tree t with $\mathrm{infLabels}(\pi) = F$.

In the terminology of [11], each nonterminal has to be +-productive and reachable, and each accepting set has to be viable.

Such a normal form of any MCFG is computable:

Theorem 2 ([11])**.** *For any MFCG G, an equivalent MCFG G' in normal form can be computed in* **PSPACE**. *The resulting grammar G' has a size polynomially bounded by the size of G.*

We sketch the algorithm here. The straightforward modification of the corresponding algorithms for ordinary context-free grammars works: first we check for each symbol X whether X is *productive* (is there a complete derivation tree with root symbol X at all). However, the complexity of this problem is **PSPACE**-complete due to the fact that the emptiness problem of Muller regular tree languages, given by a nondeterministic Muller tree automaton, is **PSPACE**-complete. Since there is a polynomial-time transformation from an MCFG to a corresponding Muller tree automaton and vice versa, we gain **PSPACE**-completeness for deciding productiveness of individual nonterminals, and emptiness of MCFLs as well:

Proposition 1. *[11] Deciding whether $L(G) = \emptyset$ is* **PSPACE***-complete.*

Then, we can throw away all the non-productive nonterminals and get rid of all the productions that have a non-productive nonterminal on either of its sides.

After that, this set is further reduced to the set of *reachable* nonterminals, which can be done by the usual fixed-point construction for CFGs, as for each reachable nonterminal A there is a finite (not necessarily complete) derivation tree with root symbol S and A occurring as a leaf label. Then we can throw away all the non-reachable nonterminals. This can be done in polynomial time.

In the next step, a nonterminal generates A a nonempty word if and only if there is a finite (not necessarily complete) derivation tree rooted A that has some letter $a \in \Sigma$ occurring as a leaf label. This can also be decided in polynomial time by solving a reachability problem. Then, we can simply erase all the symbols from the right-hand sides from which only the empty word can be generated (and if the right-hand side of a rule becomes empty, then erase the rule itself as well), arriving to a grammar in the required normal form, apart from viability of each $F \in \mathcal{F}$. (This way we might lose the word ε from $L(G)$ – if we need the nonempty word, we can allow the production $S \to \varepsilon$ to be present, but in that case S should not appear on the right-hand side of any rule, as in the classical case.)

The normal form can be generated in **PSPACE**. Then, $L(G)$ contains a nonempty word if and only if there are still productions in G.

Proposition 2 ([11]). *It can be decided in* **PSPACE** *whether $L(G)$ contains a nonempty word.*

Now for retaining only the "viable" accepting sets, the following associated graph Γ_G is handy:

Definition 2. *Given an MCFG $G = (N, \Sigma, P, S, \mathcal{F})$, we define the following edge-labeled multigraph Γ_G: the vertices of Γ_G are the nonterminals, and there is an edge from A to B labeled (α, β) for $\alpha, \beta \in (N \cup \Sigma)^*$ if $A \to \alpha B \beta$ is a production of G.*

Now if G already contains only productive and reachable nonterminals, then a set $F \in \mathcal{F}$ is viable if and only if the subgraph of Γ_G induced by F is strongly connected, which is efficiently decidable, finishing the construction of the normal form.

However, language universality (and thus language inclusion and equivalence) is undecidable already for singleton alphabets (contrary to the case of context-free grammars, where undecidability holds only for alphabets of size at least two):

Proposition 3 ([10]). *It is undecidable whether $L(G) = \Sigma^\#$ for an MCFG G, even when Σ is a singleton alphabet.*

In fact, the problem is undecidable already for Büchi context-free grammars. The key for proving this is a reduction from the universality problem of context-free languages of finite words over the binary alphabet $\{a, b\}$: first we encode a by a^ω and b by $a^{-\omega}$. Then, the language L of those words in $a^\#$ *not* belonging to $\{a^\omega, a^{-\omega}\}^*$ is a MCFG and thus, as MCFLs are effectively closed under homomorphisms and finite unions, we get that $L(G) = \{a, b\}^*$ for the CFG G if and only if $L(G') = a^\#$ for some MCFG G' effectively constructed from G.

4.1 Languages of finite words

Given an MCFG $G = (N, \Sigma, P, S, \mathcal{F})$ in normal form, one can decide in (low-degree) polynomial time whether $L(G)$ consists of finite words only. (Note that decidability is clear as the property of being scattered is MSO-definable.)

The key observation here is that $L(G)$ contains an infinite word if and only if there is some $F \in \mathcal{F}$ and an edge $A \xrightarrow{\alpha,\beta} B$ in Γ_G with $\alpha\beta \neq \varepsilon$ and $A, B \in F$ which can be verified efficiently.

Also, it is quite straightforward to see that a language $L \subseteq \Sigma^*$ is context-free if and only if it is an MCFL: for one direction we only have to set $\mathcal{F} = \emptyset$. For the other direction somewhat more care is needed since the frontier word of an infinite tree can be empty. However, it can be decided in **PSPACE** whether $A \Rightarrow_G^\infty \varepsilon$ holds for a nonterminal A: we only have to remove the productions from G having some terminal symbol occurring on the right-hand side (that is, we retain the productions of the form $X \to \alpha$ with $\alpha \in N^*$), and apply an emptyness check for the generated language. Then, for each A generating ε we can include the production $A \to \varepsilon$ and the resulting (classical) context-free grammar will generate $L(G) \cap \Sigma^*$ if G is in normal form.

Thus,

Proposition 4 ([11])**.** *A language $L \subseteq \Sigma^*$ is context-free if and only if it is Muller context-free.*

Also, since emptiness of CFGs is efficiently decidable, we have:

Proposition 5 ([11])**.** *It is decidable in* **PSPACE** *whether an MCFG generates at least one finite word.*

4.2 Languages of well-ordered words

Given an MCFG $G = (N, \Sigma, P, S, \mathcal{F})$ in normal form, one can decide in (low-degree) polynomial time whether $L(G)$ consists of well-ordered words only. Again, decidability itself is already clear, since the property is MSO-definable.

Proposition 6 ([11])**.** *For an MFCG G in normal form, $L(G)$ contains a word which is not well-ordered if there is some set $F \in \mathcal{F}$, nonterminals $A, B \in F$ and an edge $A \xrightarrow{\alpha,\beta} B$ in Γ_G with $\beta \neq \varepsilon$.*

To see this, suppose there is a derivation tree t of G with a frontier word *not* having a well-ordered domain. Then there exists an infinite descending chain $\ldots < x_3 < x_2 < x_1 < x_0$ of leaves of t. Starting from the root, one can then build up an infinite path $\pi = y_0, y_1, \ldots$ such that for each node u of π, an infinite number of these leaves x_i are descendants of u. (This property holds for the root, and at each step we set y_{i+1} to be the first child of y_i which is an ancestor of some x_j.) Then, as each such leaf x_i has some finite depth, there exists an y_j for each x_i such that y_j is an ancestor of x_i but y_{j+1} is not; it is easy to see that in this case x_i is "on the right side" of π.

Hence, for this π it holds that $F = \mathrm{infLabels}(\pi)$ is contained within a nontrivial strongly connected component of Γ_G, moreover, there is an edge $A \xrightarrow{\alpha,\beta} B$ with $A, B \in F$ and $\beta \neq \varepsilon$, the other direction being also straightforward, simply by following a closed path in F visiting each edge at least once, iterating ω times and complete the resulting derivation tree (which already has an infinite descending chain among its leaves due to $\beta \neq \varepsilon$).

Now if $L(G)$ contains well-ordered words only, one can compute an interval of ordinals containing all the order types of $L(G)$:

Proposition 7 ([16])**.** *If the MCFG G generates well-ordered words only, then both the* minimum *and the* supremum *of the order types of the members of $L(G)$ are effectively computable ordinals.*

For the minimum, first we note that to each nonterminal A, if there is a finite word w with $A \Rightarrow_G^\infty w$, then the length n of the shortest such word is computable. Then we can replace each occurrence of these nonterminals by a^n: the minimum order types for the nonterminals in the resulting grammar will coincide with those of G.

Let us fix for each symbol X a complete derivation tree t_X with root symbol X, minimizing the order type of $\mathrm{fr}(t_X)$. It turns out that the members of N can be partially ordered by some properties of these trees t_X and that these trees t_X can be chosen in a way that each subtree of t_X is some t_Y for $Y \in N \cup \Sigma$.

The key construction to see this is the following. When a t is a complete derivation tree of such an MCFG G, then we call t *simple* if it is either finite or has some infinite path π with $\mathrm{infLabels}(\pi) = \mathrm{labels}(\pi)$ and moreover, each *production* corresponding to the nodes of π occur infinitely many times on π. As $A \to uBv \in P$ for some $A, B \in F \in \mathcal{F}$ implies $v = \varepsilon$, this path π has to be the *rightmost* path of t. Then, the order type of $\mathrm{fr}(t)$ is the ω-sum of the order types of the frontiers of the subtrees of t being adjacent to π (that is, rooted at some node x not on π whose parent is on π). As each left-hand side occurs infinitely many times, we get that each nonterminal adjacent to π has a strictly smaller minimum. Thus, if t_X is simple, then we can define t_X after being defined each t_Y where the minimum order type of Y is strictly smaller than that of X, and the order type of its frontier is computable.

Now if t_X is *not* simple, then the order type of its frontier is the sum of the order types corresponding to its direct children. Now all these order types but the last have to be smaller then the order type of $\mathrm{fr}(t_X)$ and the last child has to have one level "closer" for being simple (and this level is finite), establishing the inductive case. Thus, a standard iterative algorithm recomputing the minima from the current estimations eventually terminates and produces the minimum ordinals (in Cantor normal form, say).

For the supremum, the case analysis is slightly more involved. First, one seeks for *reproductive* nonterminals: A is called reproductive if $A \Rightarrow_G^\infty \alpha$ for some α in which A occurs infinite times. It turns out that A is reproductive if and only if there is a production $A \to X_1 \dots X_n B$ and an accepting set $F \in \mathcal{F}$ with $A, B \in F$ and $X_i \Rightarrow_G^\infty uAv$ for some $i \in [n]$ and $u, v \in \Sigma^\infty$, which is decidable.

Then, if A is reproductive, then it is easy to see that arbitrarily large countable ordinals can be generated from A, thus in that case, the supremum in question is ω_1, the smallest uncountable ordinal. Also, if $A \Rightarrow_G^\infty uBv$ for some reproductive nonterminal B, then the same holds for A as well.

Otherwse, to each production of the above form, the nonterminals X_i belong to a strongly connected component strictly below the component of A, again setting a straightforward induction argument: the reader is referred to [16] for the technical details.

As ω-languages are also well-ordered, we note that and a language of ω-words is context-free in the sense of Cohen and Gold [8] if and only if it is an MCFL [11], and moreover, it is decidable whether an MCFG generates only well-ordered words having order type at most ω.

4.3 Languages of scattered words

Analogously for the case of well-ordered words, it is decidable whether an MCFG G generates scattered words only, as this property is also MSO-definable.

Proposition 8 ([11])**.** *It is decidable in polynomial time whether an MCFG G in normal form generates scattered words only.*

The key observation is the following: $L(G)$ contains a quasi-dense word if and only if there exists a *finite* derivation tree t, two leaves x and y of t and a viable accepting set $F \in \mathcal{F}$ such that $\mathrm{labels}(\pi_x) = \mathrm{labels}(\pi_y) = F$ and $t(x) = t(y) = t(\varepsilon)$ where π_x and π_y respectively denote the paths from the root to x and y. As this property is further equivalent to the existence of some production $A \to \alpha B \beta C \gamma$ with $A, B, C \in F$ for a viable $F \in \mathcal{F}$, we have an efficient decision procedure.

For languages of scattered words, the main ingredient of many proofs is that of the *Hausdorff-rank*. Basically, given a derivation tree t, we can tag each node x of t by the rank of $\mathrm{fr}(t|_x)$. Then, consider the subset D of the nodes tagged by the same ordinal as the root. As an infinite sum of scattered orderings of the same rank α has a rank strictly greater than α, this subset D cannot contain an infinite antichain, yielding that D is a finite union of paths. Then, one can partially order the set of derivation trees primarily by the rank of their frontier, secondary by the number of paths covering their respective sets D, and third, by the depth of the first node of D having at least two children in D. The defined ordering becomes then a well-ordering of the derivation trees, allowing us to apply well-founded induction.

The first such application is the following “gap theorem” of MCFLs of scattered words:

Proposition 9 ([11])**.** *The supremum of the Hausdorff-rank of the members of $L(G)$ is computable when G generates scattered words only.*

Moreover, this supremum is either ω_1 or some natural number.

The key observation for this result is again that reproductive nonterminals, and only those, can produce words of arbitrarily large (countable) rank, and for the others, a simple induction works over the strongly connected components of Γ_G.

Interestingly, it is also known [14] that an MCFL consisting only of scatterred words is a BCFL if and only if the second case applies, i.e. if it has a finite upper bound n on the Hausdorff-rank of its members.

In order to define the regular expression-like expressions capturing these MCFLs, we will also consider *pairs* of words over an alphabet Σ, equipped with a *finite* concatenation and an ω-product operation. For pairs (u,v), (u',v') in $\Sigma^\# \times \Sigma^\#$, we define the product $(u,v)\cdot(u',v')$ to be the pair $(uu',v'v)$, and when for each $i \in \omega$, (u_i,v_i) is in $\Sigma^\# \times \Sigma^\#$, then we let $\prod\limits_{i\in\omega}(u_i,v_i)$ be the word $\big(\prod\limits_{i\in\omega} u_i\big)\big(\prod\limits_{i\in-\omega} v_i\big)$. Let $P(\Sigma^\# \times \Sigma^\#)$ denote the set of all subsets of $\Sigma^\# \times \Sigma^\#$. Then $P(\Sigma^\# \times \Sigma^\#)$ is equipped with the operations of set union, concatenation $L\cdot L' = \{(u,v)\cdot(u',v') : (u,v)\in L,\ (u',v')\in L'\}$ and Kleene star $L^* = \{\epsilon\}\cup L\cup L^2\cup\cdots$. We also define an ω-power operation $P(\Sigma^\# \times \Sigma^\#) \to P(\Sigma^\#)$ by L^ω consisting of the words of the form $\prod\limits_{i\in\omega}(u_i,v_i)$ with $(u_i,v_i)\in L$ for each $i\in\omega$.

The motivation behind this notion is the following. If t is a derivation tree with a distinguished leaf x labeled by some nonterminal A, and frontier word $\mathrm{fr}(t) = uAv$, then this frontier word is represented by the pair (u,v). Now if we substitute a tree s with root symbol A in place of the distinguished leaf, having frontier word $\mathrm{fr}(s) = u'Bv'$, yielding the tree t', then we have $\mathrm{fr}(t') = uu'Bv'v$ which is $(u,v)\cdot(u',v')$ according to the product operation we defined on pairs. Similarly, if the root of t is also labeled A, then we can iterate substituting t in place of the distinguished leaf: if we iterate a finite number of times, then the Kleene star contains the "pair representant" of the resulting tree; if we iterate ω times, then the frontier is $u^\omega v^{-\omega} = (u,v)^\omega$.

Then, let the set of $\mu\omega T_s$-expressions over the alphabet Σ be defined by the following grammar (with T being the initial nonterminal):

$$\begin{aligned} T &::= a \mid \varepsilon \mid x \mid T+T \mid T\cdot T \mid \mu x.T \mid P^\omega \\ P &::= T\times T \mid P+P \mid P\cdot P \mid P^* \end{aligned}$$

Here, $a\in\Sigma$ and $x\in\mathcal{X}$ for an infinite countable set of variables. An occurrence of a variable is *free* if it is not in the scope of a μ-operation, and *bound*, if it is not free. A *closed expression* does not have free variable occurrences. The semantics of these expressions are defined as expected using the monotone functions over $P(\Sigma^\#)$ and $P(\Sigma^\# \times \Sigma^\#)$ introduced earlier.

The characterization theorem of [12] states that these notions correspond to each other:

Theorem 3 ([12]). *A language L is an MCFL of scattered words if and only if it can be denoted by some closed $\mu\omega T_s$-expression.*

The direction that such expressions always denote MCFLs is done by a straightforward construction, while the converse direction is again done via the well-ordering of the derivation trees we introduced earlier.

5 Operational characterization of general MCFLs

In this section we give an operational characterization of MCFLs in general, using the operational characterization of Muller regular languages of infinite trees given in [18, 20] which we reproved using slightly different methods in [15].

There, we introduced for each *ranked* alphabet Σ (that is, each symbol $a \in \Sigma$ has some arity $n \geq 0$; the set of n-ary symbols of Σ is denoted Σ_n) the set of *$\mu\eta$-regular tree expressions* (tree expressions for short) over Σ as the least set satisfying all the following conditions:

- If $a \in \Sigma_n$, $n \geq 0$ and $E_1, \ldots, E_n$ are tree expressions over Σ, then $a(E_1, \ldots, E_n)$ is a tree expression over Σ. When $n = 0$, we write a in place of $a()$.
- If E and F are tree expressions over Σ, then so is $(E + F)$.
- If E is a tree expression over $\Sigma \cup \{x\}$ for the nullary symbol x, and F is a tree expression over Δ, then $(E \cdot_x F)$ is a tree expression over $\Sigma \cup \Delta$.
- If E is a tree expression over $\Sigma \cup \{x\}$ for the nullary symbol x, then $(\mu x.E)$ and $(\eta x.E)$ are tree expressions over Σ.

In order to define the semantics of these expressions, we have to define the operations of x-product, μx and ηx on tree languages, for which we use *cuts* and *decompositions* of trees. So let Σ be an alphabet and let $\widehat{\Sigma}$ stand for the disjoint copy $\{\widehat{a} : a \in \Sigma\}$ of Σ. The hatted symbols are of arity zero. Given a tree t, and a subset $X \subseteq \mathrm{dom}(t)$ of its nodes, the *X-cut of t* is the following $\Sigma \cup \widehat{\Sigma}$-tree t/X: a node x belongs to $\mathrm{dom}(t/X)$ if and only if x is a node of t which is *not* a proper descendant of any non-root member of X, i.e. $\mathrm{dom}(t/X) = \mathrm{dom}(t) - \bigcup_{u \in X - \{\varepsilon\}} \{y \in \mathrm{dom}(t) : u \prec y\}$. The labeling of t/X is defined as follows: for a node $x \in \mathrm{dom}(t/X)$ let $(t/X)(x)$ be $t(x)$ if $x \notin X - \{\varepsilon\}$ and $\widehat{t(x)}$ if $x \in X - \{\varepsilon\}$. That is, we "cut" the tree at the nodes of X and add a hat to the symbols occurring at the cut-points.

Example 5. Figure 4 shows a tree t with frontier $a^\omega b^{-\omega}$. Choosing the set X to contain all the inner nodes of t, all the trees of the form $(t|_x)/(X|_x)$ are the same (shown on the right hand side of the Figure). Figure 5 shows a (finite) tree t which gets decomposed into a set of six trees.

Now when K is a language of $\Sigma \cup \widehat{\Sigma}$-trees and L is a language of Δ-trees for some alphabets Σ and Δ, then $K[\widehat{\Sigma}/L]$ is the following language of $\Sigma \cup \Delta$-trees: a tree t belongs to $K[\widehat{\Sigma}/L]$ if there exists some subset $X \subseteq \mathrm{dom}(t)$ such that t/X belongs to K and for each $x \in X$, $t|_x$ belongs to L. (Usually Δ is either Σ or $\Sigma \cup \widehat{\Sigma}$). That is, if we can cut t such that the "retained part" of the tree belongs to K and all the subtrees that are "cut down" belong to L. Observe that if there exists such an X, then the subset X' of $\preceq$-minimal elements of X is also fine (as $t/X = t/X'$ then, and the condition for the subtrees is also valid), thus in that case we can assume that t is cut by some antichain of its nodes.

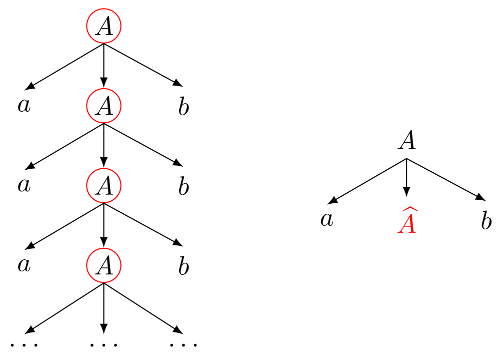


Figure 4: A tree and t one of its possible decompositions.

Given a tree language L over $\Sigma \cup \widehat{\Sigma}$, the function $K \mapsto L[\widehat{\Sigma}/K]$ is a monotone function (with respect to language inclusion), which maps the poset of $\Sigma \cup \widehat{\Sigma}$-tree languages to itself, thus it has a *least fixed point* that can be reached by the Kleene iteration $L_0 = \emptyset$, $L_\alpha = L[\widehat{\Sigma}/L_\beta]$ for successor ordinals $\alpha = \beta + 1$ and $L_\alpha = \bigcup_{\beta<\alpha} L_\beta$ for limit ordinals α, that is, there exists some (least) ordinal α such that L_α is the least fixed point of this function. This least fixed point is denoted L^μ and is a Σ-tree language. It is relatively easy to show that a Σ-tree t belongs to L^μ if and only if there is some subset $X \subseteq t$ of its nodes such that there is no infinite chain $x_1 \prec x_2 \prec \dots$ in X and for each $x \in X$, the tree $(t|_x)/(X|_x)$ belongs to L.

The last operation on trees is the η-product. Given a $\Sigma \cup \widehat{\Sigma}$-tree language L, the language L^η is a language of Σ-trees: a tree t belongs to L^η if and only if there is some $X \subseteq \mathrm{dom}(t)$ of its nodes such that for each $x \in X$, the tree $(t|_x)/(X|_x)$ belongs to L. That is, now an arbitrary set of cut-points in the domain.

Now if L is a language of $\Sigma \cup \{x\}$-trees for a nullary symbol x, then $\mu x.L$ and $\eta x.L$ are the tree languages $\widehat{L}^\mu$ and $\widehat{L}^\eta$ where $\widehat{L}$ is the following language of $\Sigma \cup \widehat{\Sigma}$-trees: a tree t belongs to $\widehat{L}$ if and only if its projection defined by $a \mapsto a$, $\widehat{a} \mapsto x$ belongs to L. That is, the $\Sigma \cup \{x\}$-tree t' we get from t by relabeling each hatted symbol to x. Similarly, if K is a language of $\Sigma \cup \{x\}$-trees and L is some Δ-tree language, then let $K \cdot_x L$ be the language $\widehat{K}[\widehat{\Sigma}/L]$.

Example 6. When K consists of the tree single tree $A(x, \widehat{A})$, then K^η contains a single tree with root symbol A and frontier x^ω. Then, $K^\eta \cdot_x \{\widehat{A}\}$ contains a single tree with root symbol A and frontier $\widehat{A}^\omega$. Finally, the frontier language of $(K^\eta \cdot_x \{\widehat{A}\} \cup \{A(a), A(\widehat{A}, \widehat{A})\})^\mu$ contains all the nonempty well-ordered words over the singleton alphabet $\{a\}$.

After these definitions we are ready to define the semantics of tree expressions in the expected way. A tree expression E *denotes* a tree language $|E|$, a set of Σ-trees defined as follows:

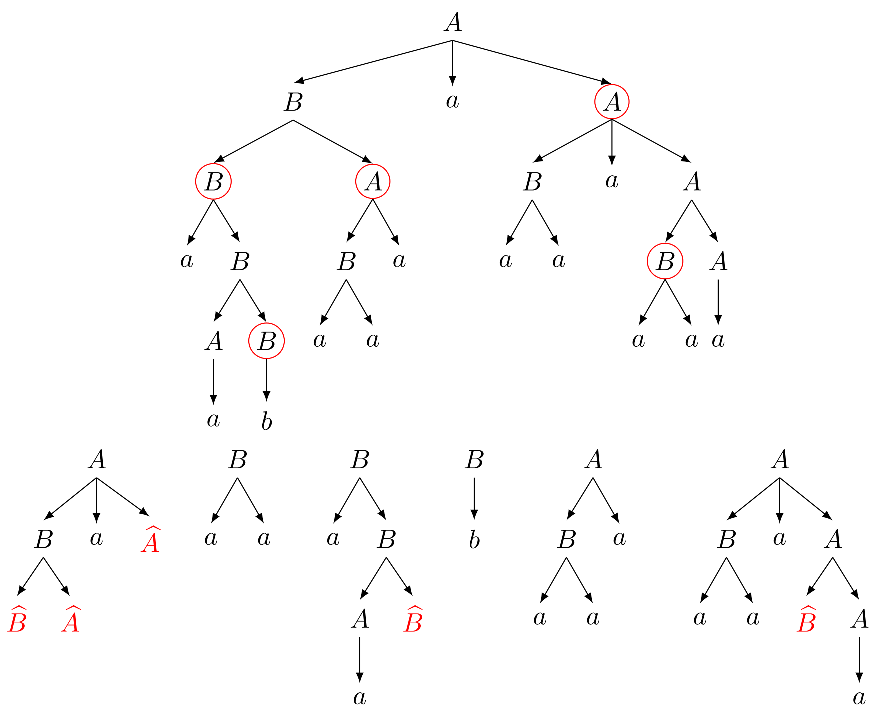


Figure 5: A decomposition of a finite tree.

- $|a(E_1, \dots, E_n)|$ consists of those Σ-trees t whose root symbol is labeled a, the root have exactly the children $1, \dots, n$ and for each $i \in [n]$, $t|_i \in |E_i|$.

- $|(E+F)| = |E| \cup |F|$, $|E \cdot_x F| = |E| \cdot_x |F|$, $|\mu x.E| = \mu x.|E|$ and $|\eta x.E| = \eta x.|E|$.

Now using the terms of [15], a tree language is Muller-regular if and only if it can be denoted by some $\mu\eta$-regular tree expression. Hence it is easy to derive $\mu\eta$-regular *word* expressions for MCFLs as MCFLs are exactly the frontier languages of Muller-regular tree languages. For this, we have to define operations corresponding to the $\cdot_x$, μx and ηx-operations above. Informally, for $\cdot x$ we can define the *substitution operation*: $K[x/L]$ contains those words we get from members of K in which we replace each occurrence of x by some word in L; then, $\mu x.L$ is the least fixed point of the monotone function $X \mapsto L[x/X]$; and, members of $\eta x.L$ are the Σ-words which we get by starting from the word x, then replacing each occurrence of x by some member of L, and repeat this process – the words occurring as "limits" of this (possibly infinite) replacement sequence are members of $\eta x.L$.

To treat the case of $\eta x.L$ formally, we introduce the class of *generalized Σ-trees* as follows. A *generalized tree domain* is a modified tree domain where we do not restrict the set of children of any node to be a finite linearly ordered set, but allow

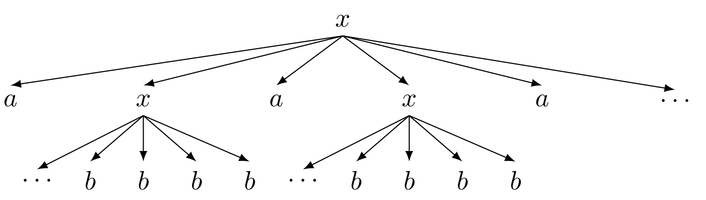


Figure 6: An L-x-substitution tree.

arbitrary countable linear orders. Nevertheless, each node has to have a finite depth.

More formally, given a partially ordered set $P = (P, <)$, a *P-tree domain* is a subset D of P^* satisfying the following conditions:

- D is nonempty and prefix-closed.
- For each node $d \in D$, the set $\{p \in P : d \cdot p \in D\}$ of the *children* of d is a linearly ordered subset of P.

When an alphabet Σ is also given, then a *P-Σ-tree* is a mapping $t : \mathrm{dom}(t) \to \Sigma$ from a P-tree domain to Σ, that is, a Σ-labeled P-tree domain and the *frontier word* of t is the Σ-word $\mathrm{fr}(t)$ whose domain is the set of the *leaves* of t (those nodes having no children) equipped with the *lexicographic* ordering: $p_1 \dots p_n <_\ell p'_1 \dots p'_m$ if and only if for some $i \leq m, n$ we have $p_i < p'_i$ and for each $j < i$, $p_j = p'_j$. Observe that this ordering is total on the leaves since for two different leaves $u = p_1 \dots p_n$ and $v = p'_1 \dots p'_m$ neither of them can be a prefix of the other, hence there exists a unique least index $i \leq m, n$ with $p_i \neq p'_i$; and as the set of the children of the node $p_1 \dots p_{i-1}$ is linearly ordered, it has to be either the case $p_i < p'_i$ or $p'_i < p_i$. Observe also that if each node has a countable children set, the $\mathrm{fr}(t)$ is a countable word.

Given a language $L \subseteq \big(\Sigma \cup \{x\}\big)^\#$, we define the languages $\mu x.L$ and $\eta x.L$ over Σ as follows. Let $P = \biguplus_{u \in L} \mathrm{dom}(u)$ be the disjoint union of the domains of all the words belonging to L. Then, an *L-x-substitition tree* is a P-$\big(\Sigma \cup \{x\}\big)$-tree satisfying the following conditions: the root is not a leaf node, each inner node is labeled by x, each leaf node is labeled in Σ and for each inner node u, the word formed by the labels of the children of u belongs to L.

Example 7. Figure 6 depicts an L-x-substitution tree where L is the language $\{b^{-\omega}, (ax)^\omega\}$.

Then, let $\eta x.L$ contain the frontier words of the L-x-substitution trees, and let $\mu x.L$ contain the frontier words of those L-x-substitution trees having no infinite paths.

Example 8. Figure 7 depicts an L-x-substitution tree for $L = \{axb\}$. This tree shows that $a^\omega b^{-\omega}$ is a member of $\eta x.L$ (but does not, in fact, belong to $\mu x.L$ as

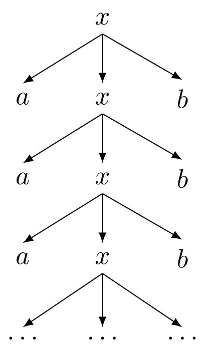


Figure 7: The word $a^\omega b^{-\omega}$ belongs to $\eta x.\{axb\}$.

it has an infinite path. For this language, $\mu x.L = \emptyset$. In contrast, $\mu x.\{axb, c\}$ is $\{a^n cb^n : n \geq 0\}$ and $\eta x.\{axb, c\}$ is $\mu x.\{axb, c\} \cup \{a^\omega b^{-\omega}\}$.)

These operations μ and η on languages over words correspond to the operations μ and η on tree languages in the sense $\mathrm{fr}(\mu x.L) = \mu x.\mathrm{fr}(L)$ and $\mathrm{fr}(\eta x.L) = \eta x.\mathrm{fr}(L)$ for each tree language L. We also make use of the $\cdot_x$ product operation: when $K \subseteq (\Sigma \cup \{x\})^\#$ and $L \subseteq \Delta^\#$ for the alphabets Σ and Δ, then $K \cdot_x L \subseteq (\Sigma \cup \Delta)^\#$ contains those words one can get from a word u in K by replacing each occurrence of x in u by some member of L. Or more technically, the frontier words of those $(K \cup L)$-x-substitution trees of depth at most two in which the word formed by the children of the root symbol belongs to K and each word formed by the children of the depth-one inner nodes belongs to L. In particular, the tree depicted in Figure 6 shows its frontier $(ab^{-\omega})^\omega$ belongs to $(ax)^\omega \cdot_x b^{-\omega}$.

Then also, $\mathrm{fr}(K \cdot_x L) = \mathrm{fr}(K) \cdot_x \mathrm{fr}(L)$ for arbitrary tree languages K and L.

By the characterization of Muller regular tree languages we get the following characterization:

Theorem 4. *A language $L \subseteq \Sigma^\#$ is an MCFL if and only if it can be generated from the singleton languages of one-letter words by a finite number of concatenation, binary union, $\cdot_x$-product, μx and ηx-operations.*

References

[1] Bedon, Nicolas. Finite automata and ordinals. *Theor. Comput. Sci.*, 156(1–2):119–144, 1996.

[2] Bedon, Nicolas, Bès, Alexis, Carton, Olivier, and Rispal, Chloé. *Logic and Rational Languages of Words Indexed by Linear Orderings*, pages 76–85. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[3] Boasson, Luc. Context-free sets of infinite words. In Weihrauch, Klaus, editor, *Theoretical Computer Science*, volume 67 of *Lecture Notes in Computer Science*, pages 1–9. Springer, 1979.

[4] Bruyère, Véronique and Carton, Olivier. Automata on linear orderings. *J. Comput. Syst. Sci.*, 73(1):1–24, 2007.

[5] Bès, Alexis and Carton, Olivier. A Kleene theorem for languages of words indexed by linear orderings. In de Felice, Clelia and Restivo, Antonio, editors, *Developments in Language Theory*, volume 3572 of *Lecture Notes in Computer Science*, pages 158–167. Springer, 2005.

[6] Büchi, J. Richard. *The monadic second order theory of $\omega 1$*, pages 1–127. Springer Berlin Heidelberg, Berlin, Heidelberg, 1973.

[7] Choueka, Yaacov. Finite automata, definable sets, and regular expressions over ω^n-tapes. *Journal of Computer and System Sciences*, 17(1):81 – 97, 1978.

[8] Cohen, Rina S. and Gold, Arie Y. Theory of ω-languages. I. Characterizations of ω-context-free languages. *J. Comput. Syst. Sci.*, 15(2):169–184, 1977.

[9] Courcelle, Bruno. Frontiers of infinite trees. *ITA*, 12(4), 1978.

[10] Ésik, Zoltán and Iván, Szabolcs. Büchi context-free languages. *Theor. Comput. Sci.*, 412(8-10):805–821, 2011.

[11] Ésik, Zoltán and Iván, Szabolcs. On Müller context-free grammars. *Theor. Comput. Sci.*, 416:17–32, 2012.

[12] Ésik, Zoltán and Iván, Szabolcs. Operational characterization of scattered MCFLs. In Béal, Marie-Pierre and Carton, Olivier, editors, *Developments in Language Theory - 17th International Conference, DLT 2013, Marne-la-Vallée, France, June 18-21, 2013. Proceedings*, volume 7907 of *Lecture Notes in Computer Science*, pages 215–226. Springer, 2013.

[13] Ésik, Zoltán and Iván, Szabolcs. MSO-definable properties of Muller context-free languages are decidable. In Câmpeanu, Cezar, Manea, Florin, and Shallit, Jeffrey, editors, *Descriptional Complexity of Formal Systems - 18th IFIP WG 1.2 International Conference, DCFS 2016, Bucharest, Romania, July 5-8, 2016. Proceedings*, volume 9777 of *Lecture Notes in Computer Science*, pages 87–97. Springer, 2016.

[14] Ésik, Zoltán and Okawa, Satoshi. On context-free languages of scattered words. In Yen, Hsu-Chun and Ibarra, Oscar H., editors, *Developments in Language Theory - 16th International Conference, DLT 2012, Taipei, Taiwan, August 14-17, 2012. Proceedings*, volume 7410 of *Lecture Notes in Computer Science*, pages 142–153. Springer, 2012.

[15] Gelle, Kitti and Iván, Szabolcs. Expressions for regular languages of infinite trees. *Manuscript*, 2017.

[16] Iván, Szabolcs and Mészáros, Ágnes. Müller context-free grammars generating well-ordered words. In Dömösi, Pál and Iván, Szabolcs, editors, *Automata and Formal Languages, 13th International Conference, AFL 2011, Debrecen, Hungary, August 17-22, 2011, Proceedings.*, pages 225–240, 2011.

[17] Khoussainov, Bakhadyr, Rubin, Sasha, and Stephan, Frank. Automatic linear orders and trees. *ACM Trans. Comput. Logic*, 6(4):675–700, October 2005.

[18] Mostowski, Andrzej Wlodzimierz. Regular expressions for infinite trees and a standard form of automata. In *Symposium on Computation Theory*, pages 157–168, 1984.

[19] Nivat, Maurice. Sur les ensembles de mots infins engendrés par une grammaire algébrique. *ITA*, 12(3), 1978.

[20] Niwinski, Damian. Fixed points vs. infinite generation. In *Proceedings of the Third Annual IEEE Symposium on Logic in Computer Science (LICS 1988)*, pages 402–409. IEEE Computer Society Press, July 1988.

[21] Perrin, Dominique and Pin, Jean-Éric. Infinite words: automata, semigroups, logic and games, 2004.

[22] Rabin, Michael O. Decidability of second-order theories and automata on infinite trees. *Bull. Amer. Math. Soc.*, 74(5):1025–1029, 09 1968.

[23] Rosenstein, Joseph G. *Linear orderings.* Academic Press New York, 1981.

[24] Wojciechowski, Jerzy. Classes of transfinite sequences accepted by finite automata. *Fundamenta Informaticae*, 7:191–223, 1984.

[25] Wojciechowski, Jerzy. Finite automata on transfinite sequences and regular expressions. *Fundamenta Informaticae*, 8:379–396, 1985.

Acta Cybernetica **23** (2017) 371–378.

Statistical Analysis of DH1 Cryptosystem

Pál Dömösi[a], József Gáll[b], Géza Horváth[b], and Norbert Tihanyi[c]

In memory of Professor Zoltán Ésik

Abstract

In this paper we shall use some standard statistical methods to test the avalanche effect of a previously introduced cryptosystem based on automata compositions, called DH1 cryptosystem. We have generated sample data of encryption and decryption. In our first set of analysis we simply estimated the probabilities of the atoms of the discrete distribution separately in order to compare them with those of the binomial test distribution. In the second statistical analysis, we turned to a goodness-of-fit test. For this we used the χ^2-test. Thirdly, we assumed that the sample comes from a binomial distribution and we calculated the maximum likelihood estimation of the two parameters. Finally we discuss some well-known further tests on randomness and related results. Our main conclusions based on the statistics all confirm that the avalanche effect is fulfilled.

Keywords: automata network, block cypher, statistics, goodness-of-fit, MLE

1 Introduction

Modern block cyphers are symmetric cryptosystems operating on fixed-length groups of bits, called blocks. These blocks contains at least 128 bits. The cryptosystem transforms the plaintext blocks into cyphertext blocks one by one. In [1] the authors introduced a novel block cypher based on abstract automata and Latin cubes, which is called DH1 cryptosystem in [3]. Another type of cryptosystem based on compositions of automata can be found in [2]. The basic idea of DH1 cryptosystem is to use a giant size finite automaton and a pseudorandom generator. The set of states of the automaton consists of all possible plaintext/cyphertext blocks, and the input set of the automaton contains all possible pseudorandom blocks. The size of the pseudorandom blocks are the same as the size of the plaintext/cyphertext blocks: 128 bits. For each plaintext block the pseudorandom generator generates

[a]Institute of Mathematics and Informatics, College of Nyíregyháza, H-4400 Nyíregyháza, Sóstói út 36, Hungary, E-mail: `domosi@nyf.hu`

[b]Faculty of Informatics, University of Debrecen, H-4028 Debrecen, Kassai út 26, Hungary, E-mail: {`gall.jozsef,horvath.geza`}`@inf.unideb.hu`

[c]Faculty of Informatics, Eötvös Loránd University, H-1117 Budapest, Pázmány Péter sétány 1/C, Hungary, E-mail: `tihanyi.pgp@gmail.com`

DOI: 10.14232/actacyb.23.1.2017.20

the next pseudorandom block, and the automaton transforms the plaintext block into a cyphertext block by the effect of the pseudorandom block. The key is the transformation matrix of the automaton.

$$\mathcal{A} = (\{0, 1, ..., n-1\}, \{0, 1, ..., n-1\}, \delta)$$

$$\begin{array}{c|cccc}
\delta & 0 & 1 & ... & n-1 \\
\hline
0 & c_{0,0} & c_{0,1} & \cdots & c_{0,n-1} \\
1 & c_{1,0} & c_{1,2} & \cdots & c_{1,n-1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
n-1 & c_{n-1,1} & c_{n-1,2} & \cdots & c_{n-1,n-1}
\end{array}$$

- $n = 2^{128}$
- the states $(c_{0,0}, ..., c_{n-1,n-1})$ are numbers between 0 and $n-1$
- each row is a permutation of states (other case decryption is impossible)
- each column is a permutation of states (other case statistical attacks are possible)
- input letters are pseudorandom numbers between 0 and $n-1$
- the key is the transition function itself (+ an initial value: the seed of the pseudorandom number generator (PRNG)).

The following example shows the encryption of a secret message which contains the following 3 blocks: 12993,999833,22212211

- plaintext blocks: 12993,999833,22212211
- suppose the (secret) pseudorandom number blocks are: 2012200, 239993,178
- ciphertext blocks: $c_{2012200,12993}, c_{239993,999833}, c_{178,22212211}$ $(\delta(2012200, 12993), \delta(239993, 999833), \delta(178, 22212211))$.

The problem with this idea is the following. The size of the transition matrix of the automaton is huge, namely $2^{128} \times 2^{128} \times 16$ bytes, which is impossible to store in the memory or on a hard disk. The solution is to use an automata network. Automata network consists of smaller automata, and it is able to simulate the work of a huge automaton [4]. In [1] the authors introduced a simple automata network which consist of 16 automaton, each of them calculates only one byte of the cyphertext block. Using this simple automata network makes possible simple calculations, but the authors had to introduce an automata network which main rounds contains 4 steps, and each step contains 2 sub steps to have an appropriate avalanche effect. Avalanche effect is an important property for block cyphers, meaning one bit change in the plaintext block should effect significant change in the cyphertext block, and one bit change in the cyphertext block should effect significant change in the corresponding plaintext block. The experimental results shows that 2 main rounds are enough for appropriate avalanche effect, but in this paper we are going to show detailed statistical analysis based on our test data samples.

2 Data and methodology

To test our system, we calculated the number of the identical bytes in two 16 bytes long independent random strings. We have tested 1.000.000 pairs, and saved the result. We also compared 1.000.000 ciphertext block pairs, where the corresponding plaintext blocks had just 1 bit difference. Finally we compared 1.000.000 plaintext block pairs, where the corresponding ciphertext blocks had just 1 bit difference.

Table 1 : Frequency table of the four samples

identical bytes	EN1R	DE1R	EN2R	DE2R
0	915924	916422	938843	939081
1	43064	42710	59403	59145
2	22670	22397	1717	1746
3	880	921	37	28
4	11050	11064	0	0
5	410	396	0	0
6	179	225	0	0
7	11	4	0	0
8	5574	5594	0	0
9	125	136	0	0
10	72	89	0	0
11	3	1	0	0
12	36	40	0	0
13	0	0	0	0
14	1	1	0	0
15	0	0	0	0
16	0	0	0	0

Basically we have generated this way 4 different samples: on the one hand we had samples obtained after encyption (encoding, denoted by EN) and decryption (decoding, denoted by DE), on the other hand after 1 round and 2 rounds (denoted by 1R and 2R) of encryption or decryption. Hence we shall refer to the 4 samples as EN1R, EN2R, DE1R, DE2R, respectively. (See Table 1, where we show the frequencies of the possible values –i.e. the number of identical bytes– of the distributions for all samples.)

Based on the generated samples we considered three different statistical questions to analyse the distribution of the number of different blocks in the pairs. But the main aim behind all questions, of course, was to check wether the avalanche effect can be confirmed in our case. Clearly, in an ideal situation – i.e. where we have an appropriate avalanche effect – one should get a binomial distribution with parameters $n = 16$ and $p = 1 - 1/256$ for the generated data, since in that case one can get no additional information from the data about the coding method. Therefore in what follows we shall call the binomial distribution with the above parameters simply the 'reference distribution'. With different ways we analyze whether our data show significant difference from the reference distribution or not.

For what follows we shall denote the probabilities of the test (reference) distribution by $p_i^{(0)}$, for $i = 0, 1, \ldots, 16$, whereas the probabilities of the real (true) distribution will be denoted by p_i^*, for $i = 0, 1, \ldots, 16$.

In our first set of analysis we simply estimated the probabilities of the 16 atoms of the discrete distribution separately. We calculated the point estimate of the probabilities, furtherore, considering the interval estimate of the probabilities we used confidence level $\alpha = 0,999$, i.e. 99,9% and calculated the maximum value of the margin of errors, which has the form

$$\Delta = \frac{1}{2} z_{1-\frac{\alpha}{2}} \sqrt{\frac{1}{n}},$$

where $z_{1-\frac{\alpha}{2}}$ is a quantile of the standard normal distribution of order $1 - \frac{\alpha}{2}$ and n is the sample size. It is well known that the margin of error takes its maximum for probability 1/2. Since we have a large sample size we decided to fix the confidence level at a very high value, so that small differences are indicated. This way one can see the difference between the probabilities obtained from the reference binomial distribution and the estimated probabilities from the sample.

In the second statistical analysis, we turned to a goodness-of-fit test. For this we used the χ^2-test of goodness of fit with the well-known test statistics

$$\chi^2 = \sum_{i=1}^{k} \frac{(f_i - f_i^*)^2}{f_i^*},$$

where f_i and f_i^* are the observed and the expected frequencies (the latter one being based on the test distribution) for category (probability) i, $i = 1, \ldots, k$, respectively. The aim is of course to check if the null hypothesis can be confirmed according to what the theoretical distribution of the population what the data is coming from coinsides the reference binomial distribution, which is our test distribution.

In our third analysis, we assumed that the sample comes from a binomial distribution (based on the results obtained to the previous questions) and we calculated simply the maximum likelihood estimation of the two parameters of the distribution and compared them to those of the reference distribution.

3 Statistical results

Before turning to the analysis it is worth to mention that one should check the basic properties of the sample. In other words, since we will use some standard statistical methods which are all based on independent and identically distributed sample (i.i.d. sample) one should verify these properties. Which concerns our case, either after one round or two rounds and either in case of encryption or decryption one can clearly see that the generation method of the data assures us that on the one hand the data element show no independence on the other hand their (theoretical) distribution is the same.

Table 2 : Difference of the point estimates and the theoretical values (i.e. $\hat{p}_i - p_i^{(0)}$, $i = 0, 1, \ldots, 16$)

	EN1R	DE1R	EN2R	DE2R
0	**-2.337318e-02**	**-2.287610e-02**	-4.551571e-04	-2.170959e-04
1	**-1.587231e-02**	**-1.622635e-02**	4.667083e-04	2.086489e-04
2	**2.093660e-02**	**2.066358e-02**	-1.642037e-05	1.257791e-05
3	8.482781e-04	8.892772e-04	5.277280e-06	-3.722757e-06
4	**1.104961e-02**	**1.106360e-02**	-4.043097e-07	-4.043097e-07
5	4.099966e-04	3.959962e-04	-3.805267e-09	-3.805267e-09
6	1.790002e-04	2.250000e-04	-2.735813e-11	-2.735813e-11
7	1.100001e-05	4.000000e-06	-1.532668e-13	-1.532668e-13
8	5.574006e-03	5.594000e-03	-6.761772e-16	-6.761772e-16
9	1.250001e-04	1.360000e-04	-2.357045e-18	-2.357045e-18
10	7.200007e-05	8.900000e-05	-6.470319e-21	-6.470319e-21
11	3.000003e-06	1.000000e-06	-1.384025e-23	-1.384025e-23
12	3.600004e-05	4.000000e-05	-2.261480e-26	-2.261480e-26
13	-2.728784e-29	-2.728784e-29	-2.728784e-29	-2.728784e-29
14	1.000001e-06	1.000000e-06	-2.293096e-32	-2.293096e-32
15	-1.199004e-35	-1.199004e-35	-1.199004e-35	-1.199004e-35
16	-2.938736e-39	-2.938736e-39	-2.938736e-39	-2.938736e-39

Which concerns the point estimations for the four samples, Table 2 contains the results, namely: we show the difference of the point estimates and the theoretical values (obtained from the test distribution). In other words we show $\hat{p}_i - p_i^{(0)}$ for all $i = 0, 1, \ldots, 16$, where $\hat{p}_i$ is clearly the point estimate (namely the relative frequency) of p_i^*. One can compare it with the maximum value of the margin of error Δ descibed in the previous section. With $\alpha = 0,999$ we obtain $\Delta = 0,001545116$. We can see from the table that after 1 round some of the estimates have a relatively large difference from the theoretical value, namely larger than 10^{-2} and hence larger than Δ both in case of encryption or decryption. However, after 2 rounds the results are much better since the largest difference at issue is still clearly under 10^{-3}. Thus we can conclude that after two rounds the generated data do not show difference from the theoretical test distribution, with 99,9% of confidence one could not differentiate between the test probabilities and the obtained empirical probabilities. Thus after four rounds the cryptosystem in this way show to fulfill the appropriate avalanche effect.

The results obtained from the χ^2-test can be seen in Table 3. Note that due to the large sample size we had the following concern. One cannot generally hope a clear confirmation of the null hypothesis, since very small differences of the distributions may lead to the rejection of the null hypothesis in such a case. (That is why sometimes the P-values are used only as an indicator: choosing different test distributions the one giving the largest P-value is accepted even if it does not show perfect fit by the test.) However, the results gave a clear picture, namely they lead to the same conclusions as in the previous analysis: after 2 rounds with both samples we cannot reject the null hypothesis that the data comes from the reference distribution. This again confirm that the cryptosystem seems to fulfill the avalanche effect.

Table 3 : Results obtained from the χ^2-test

	EN1R	DE1R	EN2R	DE2R
test stat.	**4.366657e+19**	**4.367992e+19**	5.357948e-06	1.725129e-06
P-values	**≈0 (<10e-10)**	**≈0 (<10e-10)**	≈ 1	≈ 1
conclusion	$\mathbf{H}_1$	$\mathbf{H}_1$	H_0	H_0

Finally, Table 4 shows the results obtained by the maximum likelihood estimations of the two parameters of the binomial distribution (assuming that the data is from the family of binomial distributions). For the test distribution we have $p = 1 - 1/256 \approx 0,9960938$ and $N = 16$. The results after two rounds support again the acceptence of the reference distribution as the real one, since the errors in the estimates are less than 10^{-4}, which is quite satisfactory.

Table 4 : Results obtained by the maximum likelihood estimations

	EN1R	DE1R	EN2R	DE2R
p	**0.9569296**	**0.9566545**	0.9960694	0.9960817
N	**16.52644**	**16.53131**	15.99994	15.99997

3.1 The Lempel-Ziv, Sárközy and Mauduit randomness tests

One of the criteria used to evaluate the AES candidate algorithms was their demonstrated suitability as random number generators. That is, the evaluation of their output utilizing statistical tests should not provide any means by which to distinguish them computationally from a truly random source. In order to test our cryptosystem we performed some basic randomness testing such as the Lempel-Ziv test and Sárközy and Mauduit methods. Data compression methods are very good starting point for testing pseudo randomness of a finite binary string. Applying the Lempel-Ziv test we were not able to distinguish the output of our cryptosystem from true random sources. In order to fulfill further requirements we performed the Sárközy and Mauduit methods [5, 6] so that to study the behaviour of pseudorandom sequences generated by our cryptosystem. Let $E_N = \{e_1, e_2, \ldots, e_N\} \in \{-1, +1\}^N$ represent a finite binary sequence. Let us define

$$U(E_N, M, a, b) = \sum_{j=1}^{M} e_{a+jb}.$$

The *well-distribution measure* of E_N is defined by

$$W(E_N) = \max_{a,b,t} |U(E_N, t, a, b)| = \max_{a,b,t} \left| \sum_{j=1}^{t} e_{a+jb} \right|$$

where the maximum is taken over all a, b, t such that $a \in \mathbb{Z}, b, t \in \mathbb{N}$ and $1 \leq a+b \leq a + tb \leq N$. Furthermore let us define

$$V(E_N, M, D) = \sum_{n=0}^{M-1} e_{n+d_1}\ e_{n+d_2} \ldots e_{n+d_k}.$$

The *correlation measure of order k* of E_N is defined by

$$C_k(E_N) = \max_{M,D} |V(E_N, M, D)| = \max_{M,D} \left| \sum_{n=0}^{M-1} e_{n+d_1}\ e_{n+d_2} \ldots e_{n+d_k} \right|$$

where the maximum is taken over all M and $D = (d_1, \ldots, d_k)$ such that $0 \leq d_1 \leq \cdots \leq d_k \leq N - M$. The goodness of a PRNG is determined by the order of $W(E_N)$ and $C_k(E_N)$. Our first results on the issue showed that we were not able to distinguish the output of our cryptosystem from true random sources by analyzing the deviation of $W(E_N)$ and $C_k(E_N)$.

There are many different statistical methods for testing the pseudorandomness of a binary string. For instance, The National Institute of Standards and Technology (NIST) published a statistical package consisting of 15 statistical tests that were developed to test the randomness of arbitrarily long binary sequences produced by either hardware or software based cryptographic random or pseudorandom number generators. Our latest (positive) test results confirm that it is meaningful and hopeful to run further tests on the cryptosystem in this direction. Note that according to the first few test results the DH1 cryptosystem successfully passed the criteria of NIST test so we would like to continue our research in this direction.

4 Conclusions

The results from the statistical estimations and tests show that the distributions of the 3 samples are the same with the same parameters, their distribution coincides with the theoretical binomial distribution, which means that the cryptosystem has an appropriate (efficient, statistically significant) avalanche effect. The first few statistical test results suggest that the output of the cryptosystem can not be distinguished from true random sources by statistical tests.

References

[1] P. Dömösi, G. Horváth: *A novel cryptosystem based on abstract automata and Latin cubes*, Studia Scientiarum Mathematicarum Hungarica, 52(2)(2015):221–232.

[2] P. Dömösi, G. Horváth: *A novel cryptosystem based on Gluškov product of automata*, Acta Cybernetica, 22(2015):359–371.

[3] P. Dömösi, J. Gáll, G. Horváth, N. Tihanyi: *Some remarks on the DH1 Cryptosystem based on automata compositions*, in preparation.

[4] P. Dömösi, C. L. Nehaniv: *Algebraic theory of automata networks: An introduction*, ser. SIAM monographs on Discrete Mathematics and Applications, vol. 11, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2005, doi 10.1137/1.9780898718492.

[5] C. Mauduit, A. Sárközy: *On finite pseudorandom binary sequences I : Measure of pseudorandomness, the Legendre symbol*, Acta Arithmetica, 82(1997), 365-377.

[6] C. Mauduit, A. Sárközy: *On finite pseudorandom binary sequences II : The Champernowne, Rubin-Saphiro, and Thue-Morse sequences, a further construction*, J. Number Theory 73(2) (1998), 256-276.

Acta Cybernetica **23** (2017) 379–401.

Minimization of Deterministic Top-down Tree Automata*

Zoltán Fülöp[a] and Sándor Vágvölgyi[a]

To the memory of Zoltán Ésik.

Abstract

We consider offline sensing unranked top-down tree automata in which the state transitions are computed by bimachines. We give a polynomial time algorithm for minimizing such tree automata when they are state-separated.

Keywords: bimachines, top-down unranked tree automata, minimization

1 Introduction

Minimization algorithms are necessary for the practical application of tree automata. Over ranked trees, Björklund and Cleophas [1] presented a taxonomy of algorithms for minimizing deterministic bottom-up tree automata, and Gécseg and Steinby [5] minimized deterministic top-down tree automata.

XML data or XML documents can be adequately represented by finite labeled unranked trees, where unranked means that nodes can have arbitrarily many children. This XML setting motivated the development of a theory of unranked tree automata, both bottom-up and top-down computing were studied [2, 10]. Bottom-up and top-down unranked tree automata have the same recognizing power [3]. Researchers usually abstract XML schema languages as Extended Document Type Definitions (EDTDs for short) instead of tree automata. Minimizing unranked tree automata or EDTDs is of both theoretical and practical importance [7].

In the case of bottom-up computing, Martens and Niehren [7] compared several notions of bottom-up determinism for unranked tree automata, minimized various types of deterministic bottom-up unranked tree automata, and showed that the minimization problem is NP-complete for bottom-up unranked tree automata in which the string languages in the transition functions are represented by deterministic finite state automata. For the size of deterministic bottom-up unranked tree

*The publication is supported by the the NKFI grant no K 108448.

[a]Institute of Informatics, University of Szeged, E-mail: `{fulop,vagvolgy}@inf.u-szeged.hu`

DOI: 10.14232/actacyb.23.1.2017.21

automata, Salomaa and Piao [11] presented upper and lower bounds for the union and intersection operations, and an upper bound for tree concatenation. They [12] presented a lower bound for the size blow-up of determinizing a nondeterministic unranked tree automaton.

For deterministic top-down unranked tree automata a blind version and a sensing version with two variants were introduced. The first variant is the online one, the state of a child depends on the state and the label of its parent and the labels of its left-siblings. Hence the child states are assigned when processing the child string in one pass from left to right. Online deterministic top-down unranked tree automata have been investigated in the context of XML schema languages, they were called as restrained competition EDTDs [9]. The second variant is the offline one, it first reads the complete child string and only then assigns states to all children. The blind, online, and offline sensing deterministic top-down unranked tree automata are increasingly more powerful, and all of them are less powerful than nondeterministic top-down unranked tree automata [8].

Minimization runs in nondeterministic polynomial time for deterministic blind top-down unranked tree automata, but the precise complexity is unknown, and runs in polynomial time for deterministic online sensing top-down unranked tree automata. Martens et al. [8] minimized deterministic offline sensing top-down unranked tree automata, where an unambiguous nondeterministic finite state automaton, associated with the state of the parent, reads the complete child string and assigns to each child the state it enters having read the child's label. They [8] reduced the minimization problem for unambiguous nondeterministic finite state automata, shown to be NP-complete by Jiang and Ravikumar [6], into the minimization for deterministic offline sensing top-down unranked tree automata, hence the minimization is NP-complete for deterministic offline sensing top-down unranked tree automata.

Cristau et al. [4] gave an equivalent formalism for deterministic offline sensing top-down unranked tree automata in terms of bimachines, from now on we refer to this notion simply as a deterministic top-down tree automaton (DTTA for short). A bimachine associated with the state of the parent assigns states to all children during the transition. Its two semi-automaton components read the child string from left-to-right and right-to-left, respectively, and its output function computes the state of a child depending on the label of the node and the states of the two semi-automata. Cristau et al. [4] noted that restrained competition EDTDs can be seen as a restricted version of the DTTAs. Martens and Niehren [7] minimized single-type and restrained completion EDTDs in polynomial time.

We minimize the size of a DTTA by minimizing the number of its states and the number of the states of the bimachines associated with its states. A state of a DTTA is an $\emptyset$-state if it accepts the empty tree language. A DTTA is state-separated if each transition yields a sequence of $\emptyset$-states or a sequence of non $\emptyset$-states. We show that it is decidable if a DTTA is state-separated. As the main result of our paper, we give a polynomial time minimizing algorithm for state-separated DTTAs. We will measure time as the number of elementary steps, assuming that each such step takes a constant time. The core of our algorithm is twofold. Following the ideas of

[5], we compute the connected part of the DTTA, find the equivalent states, and then collapse them into a single state, which is their equivalence class. Then we do similar minimization steps for the semi-automaton components of the bimachines associated with the DTTA states. We compute the connected parts of the semi-automata, find the equivalent semi-automaton states, and then collapse them into a single state, which is their equivalence class. Here two states of either of the semi-automata are equivalent if they yield the same output along all computations on any input word starting from them and any state of the other semi-automaton of the bimachine.

In Section 2, we present a brief review of the notions and notations used in the paper. In Section 3, we recall the concept of a bimachine, then study and minimize bimachines. In Section 4, we recall the concept of a DTTA. Then we present our minimization algorithm for state-separated DTTAs, and show the correctness of our algorithm.

2 Preliminaries

We denote by $\mathbb{N}$ the set of positive integers.

The cardinality of a set A is written as $|A|$. The composition of two mappings $f : A \to B$ and $g : B \to C$ is the mapping $f \circ g : A \to C$ defined by $f \circ g(a) = g(f(a))$ for every $a \in A$.

A *(binary) relation ρ over a set A* is a subset $\rho \subseteq A \times A$. For $(a, b) \in \rho$ we write $a\rho b$. We denote the reflexive and transitive closure of ρ by ρ^*. Let ρ be an *equivalence relation* (i.e., a reflexive, symmetric, and transitive relation) over A. For every $a \in A$, we denote by a/ρ the equivalence class which contains a, i.e., $a/\rho = \{b \in A \mid a\rho b\}$. Moreover, for every $B \subseteq A$ we define $B/\rho = \{a/\rho \mid a \in B\}$. Hence A/ρ is the set of all equivalence classes determined by ρ.

For a set X we denote by X^* the set of all finite words over X. The empty word is denoted by ε. For every $x \in \Sigma^*$, we denote by $|x|$ and x^{-1} the length and the reversal of x, respectively, and define them in the usual way.

A *tree domain* is a non-empty, finite, and prefix-closed subset D of $\mathbb{N}^*$ satisfying the following condition: if $xi \in D$ for $x \in \mathbb{N}^*$ and $i \in \mathbb{N}$, then $xj \in D$ for all j with $1 \leq j < i$.

Let Σ be an *alphabet*, i.e., a finite and non-empty set of symbols. An *unranked tree over* Σ (or just a *tree*) is a mapping $\xi : \mathrm{dom}(\xi) \to \Sigma$, where $\mathrm{dom}(\xi)$ is a tree domain. The elements of $\mathrm{dom}(\xi)$ are called the *nodes* of ξ. For every $x \in \mathrm{dom}(\xi)$ we call the element $\xi(x)$ of Σ the *label* of the node x and the number $\mathrm{rk}_\xi(x) = \max\{i \in \mathbb{N} \mid xi \in \mathrm{dom}(\xi)\}$ the *rank of the node x in ξ*. The *root* of ξ is $\xi(\varepsilon)$. If $xi \in \mathrm{dom}(\xi)$ for some $x \in \mathrm{dom}(\xi)$ and $i \in \mathbb{N}$, then we call xi the successor of x. As usual, a node of ξ without successors is called a *leaf* of ξ. The height $\mathrm{height}(\xi)$ of ξ is defined by $\mathrm{height}(\xi) = \max\{\, |x| \mid x \in \mathrm{dom}(\xi)\,\}$. We denote by T_Σ the set of all trees over Σ.

Furthermore, let $\xi, \xi' \in T_\Sigma$ and $x \in \mathrm{dom}(\xi)$. The *subtree* $\xi|_x$ of ξ at position x is defined by $\mathrm{dom}(\xi|_x) = \{y \in \mathbb{N}^* \mid xy \in \mathrm{dom}(\xi)\}$ and $\xi|_x(y) = \xi(xy)$ for all

$y \in \mathrm{dom}(\xi|_x)$. Moreover, we denote by $\xi[x \leftarrow \xi']$ the tree which is obtained from ξ by "replacing $\xi|_x$ by ξ'", i.e. defined by

$$\mathrm{dom}(\xi[x \leftarrow \xi']) = (\mathrm{dom}(\xi) \setminus \{xy \mid y \in \mathbb{N}^*\}) \cup \{xy \mid y \in \mathrm{dom}(\xi')\}$$

and

$$\xi[x \leftarrow \xi'](z) = \begin{cases} \xi(z) & \text{if } z \in (\mathrm{dom}(\xi) \setminus \{xy \mid y \in \mathbb{N}^*\}) \\ \xi'(y) & \text{if } z = xy \text{ for some } y \in \mathrm{dom}(\xi')\}. \end{cases}$$

If the root of ξ is labeled by a and the root has k successors at which the direct subtrees $\xi_1, \dots, \xi_k$ are rooted, then we write $\xi = a(\xi_1 \dots \xi_k)$.

Throughout the paper Σ and Γ denote arbitrary alphabets.

3 Bimachines

In this section we recall the concept of a bimachine, and establish a pumping lemma for bimachines and give a polynomial time algorithm for minimizing a bimachine.

3.1 General concepts

A *semi-automaton* is a quadruple $\mathcal{S} = (S, \Sigma, s_0, \delta)$, where S is a finite set (*states*), Σ is an alphabet (*input alphabet*), $s_0 \in S$ (*initial state*), and $\delta : S \times \Sigma \to S$ is a mapping (*transition mapping*). Let $s \in S$ be a state and $w = a_1...a_k \in \Sigma^*$ an input word. The *s-run of $\mathcal{S}$ on w* is the sequence $t_0 \dots t_k$ of states such that $t_0 = s$ and $t_i = \delta(t_{i-1}, a_i)$ for all $1 \le i \le k$. We denote the state t_k also by $sw_{\mathcal{S}}$ or by sw if $\mathcal{S}$ is clear from the context. The s_0-run of $\mathcal{S}$ on w is called *the run of $\mathcal{S}$ on w*. A state $s \in S$ *is reachable (in $\mathcal{S}$)* if there is a $w \in \Sigma^*$ such that $s = s_0 w$. The set of all reachable states is $S^c = \{s_0 w \mid w \in \Sigma^*\}$. Moreover, *the connected part of $\mathcal{S}$* is the semi-automaton $\mathcal{S}^c = (S^c, \Sigma, s_0, \delta^c)$, where $\delta^c(s, a) = \delta(s, a)$ for each $s \in S^c$ and $a \in \Sigma$. (Note that $\delta(s, a) \in S^c$.) We call $\mathcal{S}$ *connected* if $S^c = S$. Obviously, $\mathcal{S}^c$ is connected. The following result is well-known.

Proposition 1. There is a polynomial time algorithm which constructs $\mathcal{S}^c$ for a given $\mathcal{S}$.

Proof. The standard algorithm runs in $\mathcal{O}(|S|^2|\Sigma|)$ time. $\square$

A *congruence of $\mathcal{S}$* is an equivalence relation ρ over S such that $s\rho t$ implies $\delta(s, a)\rho\delta(t, a)$ for every $s, t \in S$ and $a \in \Sigma$. The *factor semi-automaton of $\mathcal{S}$ determined by a congruence ρ* is the semi-automaton $\mathcal{S}/\rho = (S/\rho, \Sigma, s_0/\rho, \delta_\rho)$, where $\delta_\rho(s/\rho, a) = \delta(s, a)/\rho$ for all $s \in S$ and $a \in \Sigma$.

Let $\mathcal{T} = (T, \Sigma, t_0, \delta')$ be a semi-automaton. A mapping $\varphi : S \to T$ is a *homomorphism from $\mathcal{S}$ to $\mathcal{T}$* if

- $\varphi(s_0) = t_0$ and,
- $\varphi(\delta(s, a)) = \delta'(\varphi(s), a)$ for every $s \in S$ and $a \in \Sigma$.

$$\begin{tikzcd}
s^{\rightarrow} = t_0^{\rightarrow} & & t_1^{\rightarrow} & & \cdots & & & & t_{k-2}^{\rightarrow} & & t_{k-1}^{\rightarrow} & & t_k^{\rightarrow} \\
& a_1 & & a_2 & & \cdots & & & & a_{k-1} & & a_k & \\
t_k^{\leftarrow} & & t_{k-1}^{\leftarrow} \arrow[d] & & t_{k-2}^{\leftarrow} \arrow[d] & & \cdots & & & & t_1^{\leftarrow} \arrow[d] & & t_0^{\leftarrow} = s^{\leftarrow} \arrow[d] \\
& & b_1 & & b_2 & & \cdots & & & & b_{k-1} & & b_k
\end{tikzcd}$$

Figure 1: Visualization of the definition of the mapping $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}$

For a homomorphism φ from $\mathcal{S}$ to $\mathcal{T}$, we have $\varphi(sw_{\mathcal{S}}) = \varphi(s)w_{\mathcal{T}}$. If φ is a surjective homomorphism, then $\mathcal{T}$ is a *homomorphic image* of $\mathcal{S}$. If, in addition, φ is a bijection, then we say that $\mathcal{S}$ and $\mathcal{T}$ are *isomorphic* and write $\mathcal{S} \cong \mathcal{T}$.

A *bimachine* is a system $\mathcal{B} = (\Sigma, \Gamma, \mathcal{S}^{\rightarrow}, \mathcal{S}^{\leftarrow}, f)$, where Σ and Γ are alphabets (*input* and *output*), $\mathcal{S}^{\rightarrow} = (S^{\rightarrow}, \Sigma, s_0^{\rightarrow}, \delta^{\rightarrow})$ and $\mathcal{S}^{\leftarrow} = (S^{\leftarrow}, \Sigma, s_0^{\leftarrow}, \delta^{\leftarrow})$ are semi-automata, and $f : S^{\rightarrow} \times \Sigma \times S^{\leftarrow} \to \Gamma$ is a mapping (*output function*).

For every $s^{\rightarrow} \in S^{\rightarrow}$ and $s^{\leftarrow} \in S^{\leftarrow}$, we define the mapping

$$\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})} : \Sigma^* \to \Gamma^*$$

as follows. Let $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(\varepsilon) = \varepsilon$. For every $k \geq 1$ and $w = a_1 \dots a_k \in \Sigma^*$, we let $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(w) = b_1 \dots b_k$, where $b_1, \dots, b_k \in \Gamma$ are obtained as follows. Let

- $t_0^{\rightarrow} t_1^{\rightarrow} \dots t_{k-1}^{\rightarrow} t_k^{\rightarrow}$ be the $s^{\rightarrow}$-run of $\mathcal{S}^{\rightarrow}$ on $a_1 \dots a_k$,
- $t_0^{\leftarrow} t_1^{\leftarrow} \dots t_{k-1}^{\leftarrow} t_k^{\leftarrow}$ the $s^{\leftarrow}$-run of $\mathcal{S}^{\leftarrow}$ on the reversed input $a_k \dots a_1$, and
- let $b_i = f(t_{i-1}^{\rightarrow}, a_i, t_{k-i}^{\leftarrow})$ for $1 \leq i \leq k$, see Fig. 1.

We call $\|\mathcal{B}\|_{(s_0^{\rightarrow},s_0^{\leftarrow})}$ *the mapping computed by* $\mathcal{B}$ and denote it by $\|\mathcal{B}\|$.

Throughout the paper, $\mathcal{B}$ and $\mathcal{B}'$ will denote the bimachines

- $\mathcal{B} = (\Sigma, \Gamma, \mathcal{S}^{\rightarrow}, \mathcal{S}^{\leftarrow}, f)$, *with semi-automata* $\mathcal{S}^{\rightarrow} = (S^{\rightarrow}, \Sigma, s_0^{\rightarrow}, \delta^{\rightarrow})$ *and* $\mathcal{S}^{\leftarrow} = (S^{\leftarrow}, \Sigma, s_0^{\leftarrow}, \delta^{\leftarrow})$ *and*
- $\mathcal{B}' = (\Sigma, \Gamma, \mathcal{T}^{\rightarrow}, \mathcal{T}^{\leftarrow}, f')$ *with semi-automata* $\mathcal{T}^{\rightarrow} = (T^{\rightarrow}, \Sigma, t_0^{\rightarrow}, \gamma^{\rightarrow})$ *and* $\mathcal{T}^{\leftarrow} = (T^{\leftarrow}, \Sigma, t_0^{\leftarrow}, \gamma^{\leftarrow})$,

respectively.

The bimachines $\mathcal{B}$ and $\mathcal{B}'$ are *equivalent* if $\|\mathcal{B}\| = \|\mathcal{B}'\|$. Next we prove a pumping lemma for bimachines.

Lemma 1. There is an integer $N > 0$ such that for every $x \in \Sigma^*$ with $|x| > N$ and $\|\mathcal{B}\|(x) = y$, there are $x_1, x_2, x_3 \in \Sigma^*$ and $y_1, y_2, y_3 \in \Gamma^*$ such that

- $x = x_1 x_2 x_3$ and $y = y_1 y_2 y_3$,
- $|x_i| = |y_i|$ for $1 \leq i \leq 3$,
- $0 < |x_2| = |y_2| \leq N$, and
- $\|\mathcal{B}\|(x_1 x_2^n x_3) = y_1 y_2^n y_3$ for every $n \geq 0$.

Proof. Let $N = |S^{\rightarrow}||S^{\leftarrow}||\Sigma|$. Moreover, let $x = a_1 \dots a_k \in \Sigma^*$ be an input string with $a_1, \dots, a_k \in \Sigma$ and $k > N$ and $\|\mathcal{B}\|(x) = y$. Let

- $s_0^{\rightarrow} s_1^{\rightarrow} \dots s_{k-1}^{\rightarrow} s_k^{\rightarrow}$ be the run of $\mathcal{S}^{\rightarrow}$ on $a_1 \dots a_k$,
- $s_0^{\leftarrow} s_1^{\leftarrow} \dots s_{k-1}^{\leftarrow} s_k^{\leftarrow}$ the run of $\mathcal{S}^{\leftarrow}$ on $a_k \dots a_1$, and
- let $b_i = f(s_{i-1}^{\rightarrow}, a_i, s_{k-i}^{\leftarrow})$ for $1 \le i \le k$.

Then $y = b_1 \dots b_k$. Since $k > N$, there are $1 \le i < j \le k$ such that

$$(s_{i-1}^{\rightarrow}, a_i, s_{k-i}^{\leftarrow}) = (s_{j-1}^{\rightarrow}, a_j, s_{k-j}^{\leftarrow}).$$

We may assume w.l.o.g. that the triples in the sequence $(s_i^{\rightarrow}, a_{i+1}, s_{k-i-1}^{\leftarrow}) \dots (s_{j-1}^{\rightarrow}, a_j, s_{k-j}^{\leftarrow})$ are pairwise different. Then we define

$$x_1 = a_1 \dots a_i, \qquad x_2 = a_{i+1} \dots a_j, \text{ and } \qquad x_3 = a_{j+1} \dots a_k,$$

and decompose y into y_1, y_2, and y_3 accordingly. By standard arguments we can show that these decompositions of x and y satisfy the requirements of the lemma. □

Let $s^{\rightarrow} \in S^{\rightarrow}$, $s^{\leftarrow} \in S^{\leftarrow}$, and $x, y \in \Sigma^*$. It should be clear that

$$\|\mathcal{B}\|_{(s^{\rightarrow}, s^{\leftarrow})}(xy) = \|\mathcal{B}\|_{(s^{\rightarrow}, s^{\leftarrow}y^{-1})}(x)\|\mathcal{B}\|_{(s^{\rightarrow}x, s^{\leftarrow})}(y).$$

We will use this fact later in the paper.

Let $s^{\rightarrow} \in S^{\rightarrow}$ and $s^{\leftarrow} \in S^{\leftarrow}$. The pair $(s^{\rightarrow}, s^{\leftarrow})$ is *reachable (in $\mathcal{B}$)* if there is a string $x = a_1 \dots a_k \in \Sigma^*$ with runs

- $s_0^{\rightarrow} s_1^{\rightarrow} \dots s_{k-1}^{\rightarrow} s_k^{\rightarrow}$ of $\mathcal{S}^{\rightarrow}$ on $a_1 \dots a_k$ and
- $s_0^{\leftarrow} s_1^{\leftarrow} \dots s_{k-1}^{\leftarrow} s_k^{\leftarrow}$ of $\mathcal{S}^{\leftarrow}$ on $a_k \dots a_1$

such that $(s^{\rightarrow}, s^{\leftarrow}) = (s_{i-1}^{\rightarrow}, s_{k-i}^{\leftarrow})$ for some $1 \le i \le k$. We note that $(s^{\rightarrow}, s^{\leftarrow})$ is reachable in $\mathcal{B}$ if and only if $s^{\rightarrow}$ is reachable in $\mathcal{S}^{\rightarrow}$ and $s^{\leftarrow}$ is reachable in $\mathcal{S}^{\leftarrow}$.

The connected part of $\mathcal{B}$ is the bimachine $\mathcal{B}^c = (\Sigma, \Gamma, \mathcal{S}^{\rightarrow c}, \mathcal{S}^{\leftarrow c}, f^c)$, where:

- $\mathcal{S}^{\rightarrow c}$ and $\mathcal{S}^{\leftarrow c}$ are the connected parts of $\mathcal{S}^{\rightarrow}$ and $\mathcal{S}^{\leftarrow}$, respectively,
- $f^c(s^{\rightarrow}, a, s^{\leftarrow}) = f(s^{\rightarrow}, a, s^{\leftarrow})$ for every $s^{\rightarrow} \in S^{\rightarrow c}$, $s^{\leftarrow} \in S^{\leftarrow c}$, and $a \in \Sigma$.

It is obvious that $\mathcal{B}^c$ is equivalent to $\mathcal{B}$. We call $\mathcal{B}$ *connected* if $\mathcal{B}^c = \mathcal{B}$. We note that $\mathcal{B}$ is connected if both $\mathcal{S}^{\rightarrow}$ and $\mathcal{S}^{\leftarrow}$ are connected. Hence $\mathcal{B}^c$ is connected. By Proposition 1 we have the following result.

Proposition 2. There is a polynomial time algorithm which constructs $\mathcal{B}^c$ for a given $\mathcal{B}$.

Proof. We compute $\mathcal{S}^{\rightarrow c}$ and $\mathcal{S}^{\leftarrow c}$. Thus, by Proposition 1, the algorithm runs in $\mathcal{O}((|S^{\rightarrow}|^2 + |S^{\leftarrow}|^2)|\Sigma|)$ time. □

A *congruence ρ of $\mathcal{B}$* is a pair $(\rho^{\rightarrow}, \rho^{\leftarrow})$, where

- $\rho^{\rightarrow}$ and $\rho^{\leftarrow}$ are congruences of the semi-automata $\mathcal{S}^{\rightarrow}$ and $\mathcal{S}^{\leftarrow}$, respectively, and
- for all $s^{\rightarrow}, t^{\rightarrow} \in S^{\rightarrow}$, $s^{\leftarrow}, t^{\leftarrow} \in S^{\leftarrow}$, and $a \in \Sigma$, if $s^{\rightarrow} \rho^{\rightarrow} t^{\rightarrow}$ and $s^{\leftarrow} \rho^{\leftarrow} t^{\leftarrow}$, then

$$f(s^{\rightarrow}, a, s^{\leftarrow}) = f(t^{\rightarrow}, a, t^{\leftarrow}).$$

For a congruence $\rho = (\rho^{\rightarrow}, \rho^{\leftarrow})$ of $\mathcal{B}$, we define the *factor bimachine of $\mathcal{B}$ determined by ρ* to be

$$\mathcal{B}/\rho = (\Sigma, \Gamma, \mathcal{S}^{\rightarrow}/\rho^{\rightarrow}, \mathcal{S}^{\leftarrow}/\rho^{\leftarrow}, f_\rho),$$

where $f_\rho(s^{\rightarrow}/\rho^{\rightarrow}, a, s^{\leftarrow}/\rho^{\leftarrow}) = f(s^{\rightarrow}, a, s^{\leftarrow})$ for all $s^{\rightarrow} \in S^{\rightarrow}, s^{\leftarrow} \in S^{\leftarrow}$, and $a \in \Sigma$.

A pair $\varphi = (\varphi^{\rightarrow}, \varphi^{\leftarrow})$ of mappings $\varphi^{\rightarrow} : S^{\rightarrow} \to T^{\rightarrow}$ and $\varphi^{\leftarrow} : S^{\leftarrow} \to T^{\leftarrow}$ is a *homomorphism from $\mathcal{B}$ to $\mathcal{B}'$* if $\varphi^{\rightarrow}$ and $\varphi^{\leftarrow}$ are homomorphisms from $\mathcal{S}^{\rightarrow}$ to $\mathcal{T}^{\rightarrow}$ and $\mathcal{S}^{\leftarrow}$ to $\mathcal{T}^{\leftarrow}$, respectively, and in addition $f(s^{\rightarrow}, a, s^{\leftarrow}) = f'(\varphi^{\rightarrow}(s^{\rightarrow}), a, \varphi^{\leftarrow}(s^{\leftarrow}))$ for every $s^{\rightarrow} \in S^{\rightarrow}$, $s^{\leftarrow} \in S^{\leftarrow}$, and $a \in \Sigma$. If φ is a homomorphism and both $\varphi^{\rightarrow}$ and $\varphi^{\leftarrow}$ are surjective, then $\mathcal{B}'$ is a *homomorphic image* of $\mathcal{B}$. If both $\varphi^{\rightarrow}$ and $\varphi^{\leftarrow}$ are bijections, then we say that $\mathcal{B}$ and $\mathcal{B}'$ are *isomorphic* and write $\mathcal{B} \cong \mathcal{B}'$.

Lemma 2. If there is a homomorphism $\varphi = (\varphi^{\rightarrow}, \varphi^{\leftarrow})$ from $\mathcal{B}$ to $\mathcal{B}'$, then

$$\|\mathcal{B}\|_{(s^{\rightarrow}, s^{\leftarrow})} = \|\mathcal{B}'\|_{(\varphi^{\rightarrow}(s^{\rightarrow}), \varphi^{\leftarrow}(s^{\leftarrow}))}$$

for every $s^{\rightarrow} \in S^{\rightarrow}$ and $s^{\leftarrow} \in S^{\leftarrow}$. In particular, $\|\mathcal{B}\| = \|\mathcal{B}'\|$.

Proof. For every $s^{\rightarrow} \in S^{\rightarrow}$ and $s^{\leftarrow} \in S^{\leftarrow}$, $\|\mathcal{B}\|_{(s^{\rightarrow}, s^{\leftarrow})}(\varepsilon) = \varepsilon$ and $\|\mathcal{B}'\|_{(\varphi^{\rightarrow}(s^{\rightarrow}), \varphi^{\leftarrow}(s^{\leftarrow}))}(\varepsilon) = \varepsilon$.

Let $k \geq 1$ and $w = a_1 \dots a_k \in \Sigma^*$. Then $\|\mathcal{B}\|_{(s^{\rightarrow}, s^{\leftarrow})}(w) = b_1 \dots b_k$, where $b_1, \dots, b_k$ are obtained as follows. Let

- $t_0^{\rightarrow} t_1^{\rightarrow} \dots t_{k-1}^{\rightarrow} t_k^{\rightarrow}$ be the $s^{\rightarrow}$-run of $\mathcal{S}^{\rightarrow}$ on $a_1 \dots a_k$,
- $t_0^{\leftarrow} t_1^{\leftarrow} \dots t_{k-1}^{\leftarrow} t_k^{\leftarrow}$ be the $s^{\leftarrow}$-run of $\mathcal{S}^{\leftarrow}$ on the reversed input $a_k \dots a_1$, and
- $b_i = f(t_{i-1}^{\rightarrow}, a_i, t_{k-i}^{\leftarrow})$ for $1 \leq i \leq k$, see Fig. 1.

Then

- $\varphi^{\rightarrow}(t_0^{\rightarrow})\varphi^{\rightarrow}(t_1^{\rightarrow}) \dots \varphi^{\rightarrow}(t_{k-1}^{\rightarrow})\varphi^{\rightarrow}(t_k^{\rightarrow})$ is the $\varphi^{\rightarrow}(s^{\rightarrow})$-run of $\mathcal{T}^{\rightarrow}$ on $a_1 \dots a_k$, and
- $\varphi^{\leftarrow}(t_0^{\leftarrow})\varphi^{\leftarrow}(t_1^{\leftarrow}) \dots \varphi^{\leftarrow}(t_{k-1}^{\leftarrow})\varphi^{\leftarrow}(t_k^{\leftarrow})$ is the $\varphi^{\leftarrow}(s^{\leftarrow})$-run of $\mathcal{T}^{\leftarrow}$ on the reversed input $a_k \dots a_1$.

As φ is a homomorphism, $b_i = f'(\varphi^{\rightarrow}(t_{i-1}^{\rightarrow}), a_i, \varphi^{\leftarrow}(t_{k-i}^{\leftarrow}))$ for $1 \leq i \leq k$. Hence $\|\mathcal{B}'\|_{(\varphi^{\rightarrow}(s^{\rightarrow}), \varphi^{\leftarrow}(s^{\leftarrow}))}(w) = b_1 \dots b_k$. □

By the corresponding definitions we have the following result.

Lemma 3. If there is a surjective homomorphism φ from $\mathcal{B}$ to $\mathcal{B}'$, then $|T_q^{\rightarrow}| \leq |S_q^{\rightarrow}|$ and $|T_q^{\leftarrow}| \leq |S_q^{\leftarrow}|$.

Lemma 4. If ρ is a congruence of $\mathcal{B}$, then $\mathcal{B}/\rho$ is a homomorphic image of $\mathcal{B}$.

Proof. It is easy to check that the mapping $\varphi^{\rightarrow} : S^{\rightarrow} \to S^{\rightarrow}/\rho^{\rightarrow}$ defined by $\varphi^{\rightarrow}(s^{\rightarrow}) = s^{\rightarrow}/\rho^{\rightarrow}$ is a surjective homomorphism from $\mathcal{S}^{\rightarrow}$ to $\mathcal{S}^{\rightarrow}/\rho^{\rightarrow}$. Also, the mapping $\varphi^{\leftarrow} : S^{\leftarrow} \to S^{\leftarrow}/\rho^{\leftarrow}$ defined analogously is a surjective homomorphism from $\mathcal{S}^{\leftarrow}$ to $\mathcal{S}^{\leftarrow}/\rho^{\leftarrow}$. □

Lemma 5. Let $\rho = (\rho^{\rightarrow}, \rho^{\leftarrow})$ be a congruence of the bimachine $\mathcal{B}$. Then

$$\|\mathcal{B}\|_{(s^{\rightarrow}, s^{\leftarrow})} = \|\mathcal{B}/\rho\|_{(s^{\rightarrow}/\rho^{\rightarrow}, s^{\leftarrow}/\rho^{\leftarrow})}$$

for all $s^{\rightarrow} \in S^{\rightarrow}$ and $s^{\leftarrow} \in S^{\leftarrow}$. In particular, $\|\mathcal{B}\| = \|\mathcal{B}/\rho\|$.

Proof. It follows from Lemmas 2 and 4. □

3.2 Minimization of bimachines

The bimachine $\mathcal{B}$ is called *minimal* if $|S^{\rightarrow}| \leq |T^{\rightarrow}|$ and $|S^{\leftarrow}| \leq |T^{\leftarrow}|$ for any bimachine $\mathcal{B}'$ which is equivalent to $\mathcal{B}$.

We introduce the relation $\rho_{\mathcal{B}}^{\rightarrow} \subseteq S^{\rightarrow} \times S^{\rightarrow}$ as follows: for all $s^{\rightarrow}, t^{\rightarrow} \in S^{\rightarrow}$, we have $s^{\rightarrow} \rho_{\mathcal{B}}^{\rightarrow} t^{\rightarrow}$ if $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})} = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}$ for all $s^{\leftarrow} \in S^{\leftarrow}$. Analogously, we define $\rho_{\mathcal{B}}^{\leftarrow} \subseteq S^{\leftarrow} \times S^{\leftarrow}$ such that for all $s^{\leftarrow}, t^{\leftarrow} \in S^{\leftarrow}$, we have $s^{\leftarrow} \rho_{\mathcal{B}}^{\leftarrow} t^{\leftarrow}$ if $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})} = \|\mathcal{B}\|_{(s^{\rightarrow},t^{\leftarrow})}$ for all $s^{\rightarrow} \in S^{\rightarrow}$. Moreover, let $\rho_{\mathcal{B}} = (\rho_{\mathcal{B}}^{\rightarrow}, \rho_{\mathcal{B}}^{\leftarrow})$.

Lemma 6. The relations $\rho_{\mathcal{B}}^{\rightarrow}$ and $\rho_{\mathcal{B}}^{\leftarrow}$ are congruences of the semi-automata $\mathcal{S}^{\rightarrow}$ and $\mathcal{S}^{\leftarrow}$, respectively. Moreover, $\rho_{\mathcal{B}}$ is a congruence of $\mathcal{B}$.

Proof. We show that $\rho_{\mathcal{B}}^{\rightarrow}$ is a congruence of $\mathcal{S}^{\rightarrow}$. Obviously, $\rho_{\mathcal{B}}^{\rightarrow}$ is an equivalence relation. Now let $s^{\rightarrow}, t^{\rightarrow}, u^{\rightarrow}, v^{\rightarrow} \in \mathcal{S}^{\rightarrow}$, and $a \in \Sigma$ such that $s^{\rightarrow} \rho_{\mathcal{B}}^{\rightarrow} t^{\rightarrow}$, $u^{\rightarrow} = \delta^{\rightarrow}(s^{\rightarrow}, a)$, and $v^{\rightarrow} = \delta^{\rightarrow}(t^{\rightarrow}, a)$. Moreover, let $w \in \Sigma^*$ and $s^{\leftarrow} \in S^{\leftarrow}$. Then we have

$$\begin{aligned} \|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(aw) &= f(s^{\rightarrow}, a, t^{\leftarrow}) \|\mathcal{B}\|_{(u^{\rightarrow},s^{\leftarrow})}(w), \text{ and} \\ \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(aw) &= f(t^{\rightarrow}, a, t^{\leftarrow}) \|\mathcal{B}\|_{(v^{\rightarrow},s^{\leftarrow})}(w), \end{aligned}$$

where $t^{\leftarrow} = s^{\leftarrow} w^{-1}$ in $\mathcal{S}^{\leftarrow}$. Since the left-hand side of both equalities are the same, we obtain $\|\mathcal{B}\|_{(u^{\rightarrow},s^{\leftarrow})}(w) = \|\mathcal{B}\|_{(v^{\rightarrow},s^{\leftarrow})}(w)$, which proves that $u^{\rightarrow} \rho_{\mathcal{B}}^{\rightarrow} v^{\rightarrow}$.

Analogously, we can show that $\rho_{\mathcal{B}}^{\leftarrow}$ is a congruence of the semi-automata $\mathcal{S}^{\leftarrow}$.

Finally, let $s^{\rightarrow}, t^{\rightarrow} \in S^{\rightarrow}$, $s^{\leftarrow}, t^{\leftarrow} \in S^{\leftarrow}$, and $a \in \Sigma$ such that $s^{\rightarrow} \rho_{\mathcal{B}}^{\rightarrow} t^{\rightarrow}$ and $s^{\leftarrow} \rho_{\mathcal{B}}^{\leftarrow} t^{\leftarrow}$. Then

$$\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})} = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})} = \|\mathcal{B}\|_{(t^{\rightarrow},t^{\leftarrow})}.$$

In particular, $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(a) = \|\mathcal{B}\|_{(t^{\rightarrow},t^{\leftarrow})}(a)$, i.e., $f(s^{\rightarrow}, a, s^{\leftarrow}) = f(t^{\rightarrow}, a, t^{\leftarrow})$. Hence, $\rho_{\mathcal{B}}$ is a congruence of $\mathcal{B}$. $\square$

Let us recall that $\mathcal{B}/\rho_{\mathcal{B}} = (\Sigma, \Gamma, \mathcal{S}^{\rightarrow}/\rho_{\mathcal{B}}^{\rightarrow}, \mathcal{S}^{\leftarrow}/\rho_{\mathcal{B}}^{\leftarrow}, f_{\rho_{\mathcal{B}}})$.

A bimachine $\mathcal{B}$ is called *reduced* if both $\rho_{\mathcal{B}}^{\rightarrow}$ and $\rho_{\mathcal{B}}^{\leftarrow}$ are the identity relation. It is easy to check that $\mathcal{B}/\rho_{\mathcal{B}}$ is reduced.

Lemma 7. Assume that $\mathcal{B}$ and $\mathcal{B}'$ are connected and reduced. Then

$$\|\mathcal{B}\| = \|\mathcal{B}'\| \text{ if and only if } \mathcal{B} \cong \mathcal{B}'.$$

Proof. Assume that $\|\mathcal{B}\| = \|\mathcal{B}'\|$. Let us define the relation $\varphi^{\rightarrow} \subseteq S^{\rightarrow} \times T^{\rightarrow}$ as follows:

$$\varphi^{\rightarrow} = \{(s_0^{\rightarrow} x, t_0^{\rightarrow} x) \mid x \in \Sigma^*\}.$$

The domain of $\varphi^{\rightarrow}$ is $S^{\rightarrow}$ because $\mathcal{B}$ is connected. First we show by contradiction that $\varphi^{\rightarrow}$ is a mapping. For this, we assume that there are $x, y \in \Sigma^*$ such that $s_0^{\rightarrow} x = s_0^{\rightarrow} y$ and $t_0^{\rightarrow} x \neq t_0^{\rightarrow} y$. Since $\mathcal{B}'$ is reduced, there are $u, z \in \Sigma^*$ such that $\|\mathcal{B}'\|_{(t_0^{\rightarrow} x, t_0^{\leftarrow} z)}(u) \neq \|\mathcal{B}'\|_{(t_0^{\rightarrow} y, t_0^{\leftarrow} z)}(u)$. Consequently, $\|\mathcal{B}'\|_{(t_0^{\rightarrow} x, t_0^{\leftarrow})}(uz^{-1}) \neq \|\mathcal{B}'\|_{(t_0^{\rightarrow} y, t_0^{\leftarrow})}(uz^{-1})$. On the other hand, by $\|\mathcal{B}\| = \|\mathcal{B}'\|$,

$$\begin{aligned} \|\mathcal{B}\|_{(s_0^{\rightarrow},s_0^{\leftarrow})}(xuz^{-1}) &= \|\mathcal{B}'\|_{(t_0^{\rightarrow},t_0^{\leftarrow})}(xuz^{-1}) \text{ and} \\ \|\mathcal{B}\|_{(s_0^{\rightarrow},s_0^{\leftarrow})}(yuz^{-1}) &= \|\mathcal{B}'\|_{(t_0^{\rightarrow},t_0^{\leftarrow})}(yuz^{-1}). \end{aligned}$$

Thus

$$\|\mathcal{B}\|_{(s_0^{\rightarrow}x, s_0^{\leftarrow})}(uz^{-1}) = \|\mathcal{B}'\|_{(t_0^{\rightarrow}x, t_0^{\leftarrow})}(uz^{-1}) \text{ and}$$
$$\|\mathcal{B}\|_{(s_0^{\rightarrow}y, s_0^{\leftarrow})}(uz^{-1}) = \|\mathcal{B}'\|_{(t_0^{\rightarrow}y, t_0^{\leftarrow})}(uz^{-1}).$$

Hence $\|\mathcal{B}\|_{(s_0^{\rightarrow}x, s_0^{\leftarrow})}(uz^{-1}) \neq \|\mathcal{B}\|_{(s_0^{\rightarrow}y, s_0^{\leftarrow})}(uz^{-1})$, which is a contradiction by our assumption $s_0^{\rightarrow}x = s_0^{\rightarrow}y$.

By interchanging the role of $\mathcal{B}$ and $\mathcal{B}'$, we obtain that $\varphi^{\rightarrow}$ is injective. Moreover, it is obvious that $\varphi^{\rightarrow}$ is surjective.

We can also show that $\varphi^{\rightarrow}$ is a homomorphism. For this, let $x \in \Sigma^*$ and $a \in \Sigma$. Then we have

$$\varphi^{\rightarrow}(\delta^{\rightarrow}(s_0^{\rightarrow}x, a)) = \varphi^{\rightarrow}(s_0^{\rightarrow}xa) = t_0^{\rightarrow}xa = \gamma^{\rightarrow}(t_0^{\rightarrow}x, a) = \gamma^{\rightarrow}(\varphi^{\rightarrow}(s_0^{\rightarrow}x), a).$$

Thus $\mathcal{S}^{\rightarrow}$ and $\mathcal{T}^{\rightarrow}$ are isomorphic.

Analogously, we can define the relation $\varphi^{\leftarrow}$ and show that it is an isomorphism between $\mathcal{S}^{\leftarrow}$ and $\mathcal{T}^{\leftarrow}$. Finally, we show that the pair $\varphi = (\varphi^{\rightarrow}, \varphi^{\leftarrow})$ is an isomorphism between $\mathcal{B}$ and $\mathcal{B}'$. For this, let $x, y \in \Sigma^*$ and $a \in \Sigma$. Then

$$\|\mathcal{B}\|(xay^{-1}) = \|\mathcal{B}'\|(xay^{-1}).$$

By the corresponding definition this means that $f(s_0^{\rightarrow}x, a, s_0^{\leftarrow}y) = f'(t_0^{\rightarrow}x, a, t_0^{\leftarrow}y)$. With this we have proved that $\mathcal{B} \cong \mathcal{B}'$. The proof of the other implication is trivial. □

Lemma 8. Assume that $\mathcal{B}$ and $\mathcal{B}'$ are connected. Then

$$\|\mathcal{B}\| = \|\mathcal{B}'\| \text{ if and only if } \mathcal{B}/\rho_{\mathcal{B}} \cong \mathcal{B}'/\rho_{\mathcal{B}'}.$$

Proof. If $\mathcal{B}/\rho_{\mathcal{B}} \cong \mathcal{B}'/\rho_{\mathcal{B}'}$, then by Lemma 5 we obtain $\|\mathcal{B}\| = \|\mathcal{B}'\|$.

Next assume that $\|\mathcal{B}\| = \|\mathcal{B}'\|$. Again, by Lemma 5 we obtain $\|\mathcal{B}/\rho_{\mathcal{B}}\| = \|\mathcal{B}'/\rho_{\mathcal{B}'}\|$. Moreover, both $\mathcal{B}/\rho_{\mathcal{B}}$ and $\mathcal{B}'/\rho_{\mathcal{B}'}$ are connected and reduced. Hence, by Lemma 7, $\mathcal{B}/\rho_{\mathcal{B}} \cong \mathcal{B}'/\rho_{\mathcal{B}'}$. □

Theorem 1. If the bimachine $\mathcal{B}$ is connected, then $\mathcal{B}/\rho_{\mathcal{B}}$ is minimal.

Proof. Let us assume that $\mathcal{B}$ is connected and that $\|\mathcal{B}\| = \|\mathcal{B}'\|$. By Lemma 4, $\mathcal{B}'/\rho_{\mathcal{B}'}$ is a homomorphic image of $\mathcal{B}'$. By Lemma 8, $\mathcal{B}/\rho_{\mathcal{B}} \cong \mathcal{B}'/\rho_{\mathcal{B}'}$. Hence $\mathcal{B}/\rho_{\mathcal{B}}$ is also a homomorphic image of $\mathcal{B}'$. □

In the rest of this section we give an algorithm which computes $\rho_{\mathcal{B}}$, i.e., $\rho_{\mathcal{B}}^{\rightarrow}$ and $\rho_{\mathcal{B}}^{\leftarrow}$. First we deal with $\rho_{\mathcal{B}}^{\rightarrow}$.

For every $i \geq 1$, we define the relation $\rho_i^{\rightarrow} \subseteq S^{\rightarrow} \times S^{\rightarrow}$, by induction as follows. For all $s^{\rightarrow}, t^{\rightarrow} \in S^{\rightarrow}$,

(i) let $s^{\rightarrow}\rho_1^{\rightarrow}t^{\rightarrow}$ if for all $a \in \Sigma$ and $s^{\leftarrow} \in S^{\leftarrow}$, we have $f(s^{\rightarrow}, a, s^{\leftarrow}) = f(t^{\rightarrow}, a, s^{\leftarrow})$, and

(ii) for each $i \geq 1$, let $s^{\rightarrow}\rho_{i+1}^{\rightarrow}t^{\rightarrow}$ if $s^{\rightarrow}\rho_i^{\rightarrow}t^{\rightarrow}$ and $\delta^{\rightarrow}(s^{\rightarrow},a)\rho_i^{\rightarrow}\delta^{\rightarrow}(t^{\rightarrow},a)$ for each $a \in \Sigma$.

Obviously, we have

$$\rho_1^{\rightarrow} \supseteq \rho_2^{\rightarrow} \supseteq \cdots$$

and thus there is an integer $i \geq 1$ such that $\rho_i^{\rightarrow} = \rho_{i+1}^{\rightarrow}$.

For the rest of this section, let i_0 be the least integer such that $\rho_{i_0}^{\rightarrow} = \rho_{i_0+1}^{\rightarrow}$. We will show that $\rho_{i_0}^{\rightarrow} = \rho_{\mathcal{B}}^{\rightarrow}$.

Claim 1. $\rho_{i_0+1}^{\rightarrow} = \rho_{i_0+2}^{\rightarrow} = \cdots$.

Proof. We prove by contradiction that $\rho_i^{\rightarrow} = \rho_{i+1}^{\rightarrow}$ implies $\rho_{i+1}^{\rightarrow} = \rho_{i+2}^{\rightarrow}$ for every $i \geq 1$. For this we assume that $\rho_i^{\rightarrow} = \rho_{i+1}^{\rightarrow}$ and $\rho_{i+1}^{\rightarrow} \supset \rho_{i+2}^{\rightarrow}$ for some $i \geq 1$. Then there exist two states $s^{\rightarrow}, t^{\rightarrow} \in S^{\rightarrow}$ such that $s^{\rightarrow}\rho_{i+1}^{\rightarrow}t^{\rightarrow}$ but $s^{\rightarrow}\rho_{i+2}^{\rightarrow}t^{\rightarrow}$ does not hold. This means that there exists a symbol $a \in \Sigma$ such that $\delta^{\rightarrow}(s^{\rightarrow},a)\rho_{i+1}^{\rightarrow}\delta^{\rightarrow}(t^{\rightarrow},a)$ does not hold. As $\rho_i^{\rightarrow} = \rho_{i+1}^{\rightarrow}$, we obtain that $\delta^{\rightarrow}(s^{\rightarrow},a)\rho_i^{\rightarrow}\delta^{\rightarrow}(t^{\rightarrow},a)$ does not hold either. Hence $s^{\rightarrow}\rho_{i+1}^{\rightarrow}t^{\rightarrow}$ does not hold either. This contradicts our assumption $\rho_i^{\rightarrow} = \rho_{i+1}^{\rightarrow}$. □

Claim 2. For all $l \geq 1$, $s^{\rightarrow}, t^{\rightarrow} \in S^{\rightarrow}$, if $s^{\rightarrow}\rho_l^{\rightarrow}t^{\rightarrow}$, then for each $s^{\leftarrow} \in S^{\leftarrow}$ and $w \in \Sigma^*$ with $|w| = l$, we have $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(w) = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(w)$.

Proof. We proceed by induction on l. If $l = 1$, then $w = a$ for some $a \in \Sigma$ and hence $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(w) = f(s^{\rightarrow},a,s^{\leftarrow}) = f(t^{\rightarrow},a,s^{\leftarrow}) = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(w)$.

Now assume that the claim holds for $l \geq 1$. Let $w = av \in \Sigma^*$ such that $|v| = l$ (that is, $|w| = l+1$). Then, by the definition of $\rho_{l+1}^{\rightarrow}$, we have $f(s^{\rightarrow},a,s^{\leftarrow}v^{-1}) = f(t^{\rightarrow},a,s^{\leftarrow}v^{-1})$ and $(s^{\rightarrow}a)\rho_l^{\rightarrow}(t^{\rightarrow}a)$. From this, by the induction hypothesis, $\|\mathcal{B}\|_{(s^{\rightarrow}a,s^{\leftarrow})}(v) = \|\mathcal{B}\|_{(t^{\rightarrow}a,s^{\leftarrow})}(v)$ for all $s^{\leftarrow} \in S^{\leftarrow}$. Thus

$$\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(w) = f(s^{\rightarrow},a,s^{\leftarrow}v^{-1})\|\mathcal{B}\|_{(s^{\rightarrow}a,s^{\leftarrow})}(v) =$$
$$f(t^{\rightarrow},a,s^{\leftarrow}v^{-1})\|\mathcal{B}\|_{(t^{\rightarrow}a,s^{\leftarrow})}(v) = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(w)$$

for all $s^{\leftarrow} \in S^{\leftarrow}$. □

Claim 3. $\rho_{i_0}^{\rightarrow} \subseteq \rho_{\mathcal{B}}^{\rightarrow}$.

Proof. Assume that $s^{\rightarrow}\rho_{i_0}^{\rightarrow}t^{\rightarrow}$. Observe that $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(\varepsilon) = \varepsilon = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(\varepsilon)$. Let $w \in \Sigma^*$ with $|w| = l \geq 1$ be arbitrary. By the definition of i_0 and Claim 1, $\rho_{i_0}^{\rightarrow} \subseteq \rho_l^{\rightarrow}$. Consequently, we also have $s^{\rightarrow}\rho_l^{\rightarrow}t^{\rightarrow}$, from which we obtain by Claim 2 that $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(w) = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(w)$. Hence $s^{\rightarrow}\rho_{\mathcal{B}}^{\rightarrow}t^{\rightarrow}$. □

Claim 4. $\rho_{i_0}^{\rightarrow} \supseteq \rho_{\mathcal{B}}^{\rightarrow}$.

Proof. It suffices to show that, for all $s^{\rightarrow}, t^{\rightarrow} \in S^{\rightarrow}$, if for each $s^{\leftarrow} \in S^{\leftarrow}$ we have $\|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})} = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}$, then $s^{\rightarrow}\rho_i^{\rightarrow}t^{\rightarrow}$ for all $i \geq 1$.

We proceed by induction on i. Let $i = 1$ and $a \in \Sigma$. By our assumption, $f(s^{\rightarrow},a,s^{\leftarrow}) = \|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(a) = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(a) = f(t^{\rightarrow},a,s^{\leftarrow})$. Consequently, $s^{\rightarrow}\rho_1^{\rightarrow}t^{\rightarrow}$.

Now assume that the claim holds for $i \geq 1$, i.e., $s^{\rightarrow}\rho_i t^{\rightarrow}$. Let $s^{\leftarrow} \in S^{\leftarrow}$, $a \in \Sigma$ and $v \in \Sigma^*$ be arbitrary. Then $f(s^{\rightarrow}, a, s^{\leftarrow}v^{-1})\|\mathcal{B}\|_{(s^{\rightarrow}a,s^{\leftarrow})}(v) = \|\mathcal{B}\|_{(s^{\rightarrow},s^{\leftarrow})}(av) = \|\mathcal{B}\|_{(t^{\rightarrow},s^{\leftarrow})}(av) = f(t^{\rightarrow}, a, s^{\leftarrow}v^{-1})\|\mathcal{B}\|_{(s^{\rightarrow}a,s^{\leftarrow})}(v)$. Hence $f(s^{\rightarrow}, a, s^{\leftarrow}v^{-1}) = f(t^{\rightarrow}, a, s^{\leftarrow}v^{-1})$ and $\|\mathcal{B}\|_{(s^{\rightarrow}a,s^{\leftarrow})}(v) = \|\mathcal{B}\|_{(t^{\rightarrow}a,s^{\leftarrow})}(v)$. From the latter, we have $\|\mathcal{B}\|_{(s^{\rightarrow}a,s^{\leftarrow})} = \|\mathcal{B}\|_{(t^{\rightarrow}a,s^{\leftarrow})}$, thus by the induction hypothesis, $(s^{\rightarrow}a)\rho_i^{\rightarrow}(t^{\rightarrow}a)$. Then, by the definition of $\rho_{i+1}^{\rightarrow}$, we obtain $s^{\rightarrow}\rho_{i+1}^{\rightarrow}t^{\rightarrow}$ holds as well. □

Lemma 9. We have $\rho_{i_0}^{\rightarrow} = \rho_{\mathcal{B}}^{\rightarrow}$.

Proof. It follows from Claims 3 and 4. □

Analogously, we can define a decreasing sequence

$$\rho_1^{\leftarrow} \supseteq \rho_2^{\leftarrow} \supseteq \cdots$$

of relations over $S^{\leftarrow}$ such that $\rho_{\mathcal{B}}^{\leftarrow} = \rho_{i_0}^{\leftarrow}$ for the least integer i_0 with $\rho_{i_0}^{\leftarrow} = \rho_{i_0+1}^{\leftarrow}$. Hence we can conclude the following.

Proposition 3. There is a polynomial time algorithm which constructs the minimal bimachine which is equivalent to $\mathcal{B}$.

Proof. By Proposition 2 we compute the connected part $\mathcal{B}^c$ of $\mathcal{B}$ in polynomial time. So assume that $\mathcal{B}$ is connected. We compute $\rho_{\mathcal{B}}^{\rightarrow}$ as follows. We compute $\rho_1^{\rightarrow}$ in $\mathcal{O}(|S^{\rightarrow}|^2|\Sigma||S^{\leftarrow}|)$ time and, for every $1 < i \leq i_0$, we compute $\rho_i^{\rightarrow}$ in $\mathcal{O}(|S^{\rightarrow}|^2|\Sigma|^2)$ time. Since $i_0 \leq |S^{\rightarrow}|$, we compute $\rho_{i_0}^{\rightarrow}$ in $\mathcal{O}(|S^{\rightarrow}|^3|\Sigma|^2)$ time. Analogously, we compute $\rho_{\mathcal{B}}^{\leftarrow}$ in polynomial time. □

4 Deterministic top-down tree automata and their minimization

In this section first we recall the concept of a deterministic top-down tree automaton (DTTA for short) from [4]. Then we give a polynomial time algorithm minimizing a state-separated DTTA. The size of a DTTA is the sum of the sizes of the bimachines associated with its states, hence we minimize it by minimizing the number of its states and the number of the states of the bimachines associated with its states.

4.1 Basic concepts

A *deterministic top-down tree automaton* (DTTA for short) is a system

$$\mathcal{A} = \big(Q, \Sigma, f_{\mathrm{in}}, (\mathcal{B}_q \mid q \in Q), F\big),$$

where

- Q is a finite set (*states*),
- Σ is an alphabet (*input alphabet*),
- $f_{\mathrm{in}} : \Sigma \to Q$ is the *initial function*,

- $\mathcal{B}_q = (\Sigma, Q, \mathcal{S}_q^{\rightarrow}, \mathcal{S}_q^{\leftarrow}, f_q)$ is a bimachine for every $q \in Q$ with semi-automata $\mathcal{S}_q^{\rightarrow} = (S_q^{\rightarrow}, \Sigma, s_{q,0}^{\rightarrow}, \delta_q^{\rightarrow})$ and $\mathcal{S}_q^{\leftarrow} = (S_q^{\leftarrow}, \Sigma, s_{q,0}^{\leftarrow}, \delta_q^{\leftarrow})$, and
- $F \subseteq Q$ (*final states*).

Let $\xi \in T_\Sigma$ and $q \in Q$. A *q-run of $\mathcal{A}$ on ξ* is a mapping $r : \mathrm{dom}(\xi) \to Q$ such that $r(\varepsilon) = q$ and for each node $x \in \mathrm{dom}(\xi)$ with $k > 0$ successors $x1, x2, \ldots, xk$, we have

$$r(x1)r(x2)\cdots r(xk) = \|\mathcal{B}_{r(x)}\|\big(\xi(x1)\cdots\xi(xk)\big).$$

Note that for each $\xi \in T_\Sigma$ and $q \in Q$, there is exactly one q-run of $\mathcal{A}$ on ξ. This q-run r is *accepting* if it assigns to each leaf a final state, that is, $r(x) \in F$ for every $x \in \mathrm{dom}(\xi)$ which is a leaf. The *tree language $L(\mathcal{A}, q)$ accepted by $\mathcal{A}$ in q* consists of all trees ξ such that the q-run of $\mathcal{A}$ on ξ is accepting. The $f_{\mathrm{in}}(\xi(\varepsilon))$-run of $\mathcal{A}$ on ξ is called *the run of $\mathcal{A}$ on ξ* and the *tree language $L(\mathcal{A})$ accepted by $\mathcal{A}$* consists of all trees ξ such that the run of $\mathcal{A}$ on ξ is accepting.

Two DTTA $\mathcal{A}$ and $\mathcal{A}'$ are *equivalent* if $L(\mathcal{A}) = L(\mathcal{A}')$.

Remark 1. We note that the root $\xi(\varepsilon)$ of ξ does not play any role in (accepting) q-runs on ξ. Hence if $\xi \in L(\mathcal{A}, q)$, then $\xi' \in L(\mathcal{A}, q)$ for each tree ξ' obtained by replacing the root of ξ with an arbitrary $a \in \Sigma$.

A state $q \in Q$ is called a *$\emptyset$-state* if $L(\mathcal{A}, q) = \emptyset$. We write $Q = Q_+ \cup Q_e$, where Q_e is the set of all $\emptyset$-states and $Q_+ = Q \setminus Q_e$. Note that $F \subseteq Q_+$.

Lemma 10. The set Q_+ is effectively computable.

Proof. We define a sequence $Q_0, Q_1, \ldots$ of sets of states by the following algorithm:
(i) Let $Q_0 = F$ and $i = 0$.
(ii) Let $Q_{i+1} = Q_i \cup \{q \in Q \mid \exists(x \in \Sigma^*) : \|\mathcal{B}_q\|(x) \in Q_i^*\}$.
(iii) If $Q_{i+1} = Q_i$, then $\underline{\text{stop}}$, otherwise $i := i + 1$ and $\underline{\text{goto}}$ (ii).
First we note that for every $i \geq 0$ and $q \in Q$ we can decide whether there is an $x \in \Sigma^*$ with $\|\mathcal{B}_q\|(x) \in Q_i^*$. In fact, it suffices to check if $\|\mathcal{B}_q\|(x) \in Q_i^*$ for input strings x with $|x| \leq N_q$, where N_q is the number provided by Lemma 1 for the bimachine $\mathcal{B}_q$. Hence Q_{i+1} in step (ii) can be computed.

By standard arguments, we can prove the following statements:
- there is an $i \geq 0$ such that $Q_{i+1} = Q_i$,
- if $Q_{i+1} = Q_i$, then $Q_{i+j} = Q_i$ for every $j \geq 1$, and
- if $Q_{i+1} = Q_i$, then $\forall(q \in Q) : \big(q \in Q_i \iff \exists(\xi \in T_\Sigma) : \xi \in L(\mathcal{A}, q)\big)$.

Altogether we obtain that the algorithm terminates with $Q_{i+1} = Q_i$ and in this case $Q_+ = Q_i$. □

Next we introduce the concept of a connected DTTA. For this we define the binary relation $\to_\mathcal{A}$ over Q as follows: for every $q, q' \in Q$, we have $q \to_\mathcal{A} q'$ if there are $k \geq 1$, $a_1 \ldots a_k \in \Sigma^*$ such that $\|\mathcal{B}_q\|(a_1 \ldots a_k) = q_1 \ldots q_k$ and $q' = q_i$ for some $1 \leq i \leq k$. For every $q \in Q$, we define

$$T_q = \{q' \in Q \mid q \to_\mathcal{A}^* q'\}.$$

The DTTA $\mathcal{A}$ is *connected* if, for every $q \in Q$, we have $f_{\mathrm{in}}(a) \to_\mathcal{A}^* q$ for some $a \in \Sigma$.

Proposition 4. There is a polynomial time algorithm which computes T_q for a given state $q \in Q$.

Proof. By Proposition 2 we may assume that $\mathcal{B}_p$ is connected for every $p \in Q$.

(i) Let $T_0 = \{q\}$ and $i = 0$.

(ii) Let

$$T_{i+1} = T_i \cup \{f_p(s^{\rightarrow}, a, s^{\leftarrow}) \mid \\ a \in \Sigma \text{ and } \exists (p \in T_i) : s^{\rightarrow} \in S_p^{\rightarrow}, s^{\leftarrow} \in S_p^{\leftarrow}\}.$$

(iii) If $T_{i+1} = T_i$, then stop, otherwise let $i := i+1$ and goto (ii).

It is an exercise to show that $T_{i+1} = T_i$ for some $i \geq 0$ and for this i we have $T_q = T_i$. The algorithm runs in $\mathcal{O}(|Q|N^{\rightarrow}|\Sigma|N^{\leftarrow})$ time, where $N^{\rightarrow} = \max\{|S_p^{\rightarrow}| \mid p \in Q\}$ and $N^{\leftarrow} = \max\{|S_p^{\leftarrow}| \mid p \in Q\}$. $\square$

For $\mathcal{A}$ we define the DTTA $\mathcal{A}^c = \big(Q^c, \Sigma, f_{\text{in}}^c, (\mathcal{B}_q \mid q \in Q^c), F^c\big)$ called the connected part of $\mathcal{A}$ as follows:

- $Q^c = \bigcup(T_q \mid q = f_{\text{in}}(a) \text{ for some } a \in \Sigma)$,
- $f_{\text{in}}^c(a) = f_{\text{in}}(a)$ for every $a \in \Sigma$, and
- $F^c = F \cap Q^c$.

The following statement is obvious.

Proposition 5. $\mathcal{A}^c$ is connected and is equivalent to $\mathcal{A}$.

By the definition of $\mathcal{A}^c$ and Proposition 4, we have the following result.

Proposition 6. There is a polynomial time algorithm which constructs $\mathcal{A}^c$.

A *congruence of* $\mathcal{A}$ is an equivalence relation $\tau \subseteq Q \times Q$ satisfying the following two conditions:

(i) for all states $p, q \in Q$, and nonempty word $a_1 \dots a_k \in \Sigma^*$, if $p\tau q$, $\|\mathcal{B}_p\|(a_1 \dots a_k) = p_1 \dots p_k$, and $\|\mathcal{B}_q\|(a_1 \dots a_k) = q_1 \dots q_k$, then $p_i \tau q_i$ for all $1 \leq i \leq k$,

(ii) if $p\tau q$, then $p \in F$ if and only if $q \in F$.

Let τ be an equivalence relation on Q. For every $q \in Q$, we introduce the bimachine

$$\mathcal{B}_{q,\tau} = (\Sigma, Q/\tau, \mathcal{S}_q^{\rightarrow}, \mathcal{S}_q^{\leftarrow}, f_{q,\tau}),$$

where $f_{q,\tau}(s^{\rightarrow}, a, s^{\leftarrow}) = f_q(s^{\rightarrow}, a, s^{\leftarrow})/\tau$ for all $s^{\rightarrow} \in S_q^{\rightarrow}, s^{\leftarrow} \in S_q^{\leftarrow}$, and $a \in \Sigma$. Then, for every $a_1 \dots a_k \in \Sigma^+$, we have $\|\mathcal{B}_{q,\tau}\|(a_1 \dots a_k) = p_1/\tau \dots p_k/\tau$, where $\|\mathcal{B}_q\|(a_1 \dots a_k) = p_1 \dots p_k$.

Lemma 11. Let τ be a congruence on $\mathcal{A}$ and $p, q \in Q$ such that $p\tau q$. Then $\|\mathcal{B}_{p,\tau}\| = \|\mathcal{B}_{q,\tau}\|$.

Proof. By definition $\|\mathcal{B}_{p,\tau}\|(\varepsilon) = \varepsilon = \|\mathcal{B}_{q,\tau}\|(\varepsilon)$.

Let $k \geq 1$ and $a_1 \dots a_k \in \Sigma^+$. Then we have $\|\mathcal{B}_{p,\tau}\|(a_1 \dots a_k) = p_1/\tau \dots p_k/\tau$, where $\|\mathcal{B}_p\|(a_1 \dots a_k) = p_1 \dots p_k$ and $\|\mathcal{B}_{q,\tau}\|(a_1 \dots a_k) = q_1/\tau \dots q_k/\tau$, where $\|\mathcal{B}_q\|(a_1 \dots a_k) = q_1 \dots q_k$. By (i) in the definition of a congruence of a DTTA, we have $p_i \tau q_i$ for all $1 \leq i \leq k$. Hence $p_1/\tau \dots p_k/\tau = q_1/\tau \dots q_k/\tau$. Thus $\|\mathcal{B}_{p,\tau}\|(a_1 \dots a_k) = \|\mathcal{B}_{q,\tau}\|(a_1 \dots a_k)$. $\square$

Given a congruence τ of $\mathcal{A}$, we define the *factor DTTA $\mathcal{A}/\tau$ of $\mathcal{A}$ determined by τ* as $\mathcal{A}/\tau = \big(Q/\tau, \Sigma, f_{\text{in},\tau}, (\mathcal{B}_{q/\tau} \mid q/\tau \in Q/\tau), F/\tau\big)$, where

- $f_{\text{in},\tau}(a) = (f_{\text{in}}(a))/\tau$ for every $a \in \Sigma$,
- $\mathcal{B}_{q/\tau} = \mathcal{B}_{q,\tau}$ for every $q \in Q$.

We note that the definition of the bimachine $\mathcal{B}_{q/\tau}$ and hence that of the DTTA $\mathcal{A}/\tau$ is syntactically ambiguous. Indeed, for $p/\tau = q/\tau$, the bimachines $\mathcal{B}_{p,\tau}$ and $\mathcal{B}_{q,\tau}$ may be different syntactically and we can pick any of them. However, our choice has no impact on $\|\mathcal{A}/\tau\|$ because, by Lemma 11, $p\tau q$ implies $\|\mathcal{B}_{p,\tau}\| = \|\mathcal{B}_{q,\tau}\|$. In other words, $\|\mathcal{A}/\tau\|$ is well-defined.

Throughout the paper $\mathcal{A}$ and $\mathcal{A}'$ will denote the DTTA

- $\mathcal{A} = \big(Q, \Sigma, f_{\text{in}}, (\mathcal{B}_q \mid q \in Q), F\big)$ *with bimachines* $\mathcal{B}_q = (\Sigma, Q, \mathcal{S}_q^{\rightarrow}, \mathcal{S}_q^{\leftarrow}, f_q)$ *and semi-automata* $\mathcal{S}_q^{\rightarrow} = (S_q^{\rightarrow}, \Sigma, s_{q,0}^{\rightarrow}, \delta_q^{\rightarrow})$ *and* $\mathcal{S}_q^{\leftarrow} = (S_q^{\leftarrow}, \Sigma, s_{q,0}^{\leftarrow}, \delta_q^{\leftarrow})$ *for every $q \in Q$, and*
- $\mathcal{A}' = \big(Q', \Sigma, f'_{\text{in}}, (\mathcal{B}'_q \mid q \in Q'), F'\big)$ *with bimachines* $\mathcal{B}'_q = (\Sigma, Q', \mathcal{T}_q^{\rightarrow}, \mathcal{T}_q^{\leftarrow}, f'_q)$ *and semi-automata* $\mathcal{T}_q^{\rightarrow} = (T_q^{\rightarrow}, \Sigma, t_{q,0}^{\rightarrow}, \gamma_q^{\rightarrow})$ *and* $\mathcal{T}_q^{\leftarrow} = (T_q^{\leftarrow}, \Sigma, t_{q,0}^{\leftarrow}, \gamma_q^{\leftarrow})$ *for every $q \in Q'$,*

respectively.

Furthermore, let $\varphi : Q \to Q'$ be a mapping and $\varphi^* : Q^* \to Q'^*$ its unique extension to a monoid homomorphism. The mapping φ is a *homomorphism from $\mathcal{A}$ to $\mathcal{A}'$* if

- $f'_{\text{in}} = f_{\text{in}} \circ \varphi$,
- $\|\mathcal{B}'_{\varphi(q)}\| = \|\mathcal{B}_q\| \circ \varphi^*$ for every $q \in Q$, and
- $q \in F \iff \varphi(q) \in F'$ for every $q \in Q$.

If φ is a surjective homomorphism, then $\mathcal{A}'$ is a *homomorphic image* of $\mathcal{A}$. If, in addition, φ is a bijection, then we say that $\mathcal{A}$ and $\mathcal{A}'$ are *isomorphic* and write $\mathcal{A} \cong \mathcal{A}'$.

Lemma 12. If there is a homomorphism φ from $\mathcal{A}$ to $\mathcal{A}'$, then
(i) $L(\mathcal{A}, q) = L(\mathcal{A}', \varphi(q))$ for every $q \in Q$, and
(ii) $L(\mathcal{A}) = L(\mathcal{A}')$.

Proof. Let φ be a *homomorphism from $\mathcal{A}$ to $\mathcal{A}'$*. To show (i), we prove by induction on height(ξ) that for any $q \in Q$ and $\xi \in T_\Sigma$, $\xi \in L(\mathcal{A}, q)$ if and only if $\xi \in L(\mathcal{A}', \varphi(q))$.

Base of induction: height$(\xi) = 0$, i.e., $\xi = a$ for some $a \in \Sigma$. Then $\xi \in L(\mathcal{A}, q)$ if and only if $\xi \in L(\mathcal{A}', \varphi(q))$. Thus the statement holds obviously.

Induction step: height$(\xi) = n > 0$. Then $\xi = a(\xi_1, \ldots, \xi_k)$ for some $a \in \Sigma$, $k \geq 1$, and $\xi_1, \ldots, \xi_k \in T_\Sigma$. Let $a_i = \xi(i)$ for all $1 \leq i \leq k$. Then we have

$$\begin{array}{ll} & \xi \in L(\mathcal{A}, q) \\ \iff & \|\mathcal{B}_q\|(a_1...a_k) = q_1 \ldots q_k \text{ and} \\ & \xi_i \in L(\mathcal{A}, q_i) \text{ for all } 1 \leq i \leq k \\ \iff & \|\mathcal{B}'_{\varphi(q)}\|(a_1...a_k) = \varphi(q_1) \ldots \varphi(q_k) \text{ and} \\ & \xi_i \in L(\mathcal{A}', \varphi(q_i)) \text{ for all } 1 \leq i \leq k \\ \iff & \xi \in L(\mathcal{A}', \varphi(q)). \end{array}$$

We now show (ii). Let $\xi \in L(\mathcal{A})$, i.e., $\xi \in L(\mathcal{A}, f_{\text{in}}(\xi(\varepsilon))$. Then by (i), $\xi \in L(\mathcal{A}', \varphi(f_{\text{in}}(\xi(\varepsilon)))$. As φ is a homomorphism, $f'_{\text{in}}(\xi(\varepsilon)) = \varphi(f_{\text{in}}(\xi(\varepsilon)))$. Thus $\xi \in L(\mathcal{A}', f'_{\text{in}}(\xi(\varepsilon))$, which implies $\xi \in L(\mathcal{A}')$.

Conversely, let $\xi \in L(\mathcal{A}')$, i.e., let $\xi \in L(\mathcal{A}', f'_{\text{in}}(\xi(\varepsilon))$. As φ is a homomorphism, $\varphi(f_{\text{in}}(\xi(\varepsilon))) = f'_{\text{in}}(\xi(\varepsilon))$. Then by (i), $\xi \in L(\mathcal{A}, f_{\text{in}}(\xi(\varepsilon))$ which proves that $\xi \in L(\mathcal{A})$. □

Lemma 13. If τ is a congruence of $\mathcal{A}$, then $\mathcal{A}/\tau$ is a homomorphic image of $\mathcal{A}$.

Proof. It is easy to check that the mapping $\varphi : Q \to Q/\tau$ defined by $\varphi(q) = q/\tau$ is a surjective homomorphism from $\mathcal{A}$ to $\mathcal{A}/\tau$. □

Lemma 14. If τ is a congruence of $\mathcal{A}$, then $L(\mathcal{A}, q) = L(\mathcal{A}/\tau, q/\tau)$ for every $q \in Q$. Moreover, $L(\mathcal{A}) = L(\mathcal{A}/\tau)$.

Proof. It follows from Lemmas 12 and 13. □

4.2 Minimization of DTTA

The DTTA $\mathcal{A}$ is called *minimal* if

$$|Q| \leq |Q'|, \sum_{q \in Q} |S_q^{\rightarrow}| \leq \sum_{q \in Q'} |T_q^{\rightarrow}|, \text{ and } \sum_{q \in Q} |S_q^{\leftarrow}| \leq \sum_{q \in Q'} |T_q^{\leftarrow}|$$

for any DTTA $\mathcal{A}'$ which is equivalent to $\mathcal{A}$. Moreover, $\mathcal{A}$ is *state-separated* if

- $\|\mathcal{B}_q\|(x) \in Q_+^* \cup Q_e^*$ for every $q \in Q_+$ and
- $\|\mathcal{B}_q\|(x) \in Q_e^*$ for every $q \in Q_e$

for every $x \in \Sigma^*$.

Lemma 15. For the DTTA $\mathcal{A}$ the following two statements are equivalent.

(i) $\mathcal{A}$ is state-separated.
(ii) If $\|\mathcal{B}_q\|(x) \in Q^* Q_e Q^*$, then $\|\mathcal{B}_q\|(x) \in Q_e^*$ for every $q \in Q$ and $x \in \Sigma^*$.

Proof. It is clear that (i) implies (ii). Now assume that (ii) holds. Let $x \in \Sigma^*$ and $q \in Q$. If $q \in Q_e$, then obviously $\|\mathcal{B}_q\|(x) \in Q^*Q_eQ^*$. Hence by (ii), $\|\mathcal{B}_q\|(x) \in Q_e^*$. Now let $q \in Q_+$. If $\|\mathcal{B}_q\|(x) \in Q^*Q_eQ^*$, then by (ii), $\|\mathcal{B}_q\|(x) \in Q_e^*$. Otherwise, $\|\mathcal{B}_q\|(x) \in Q_+^*$. Hence (i) holds. □

Lemma 16. The DTTA $\mathcal{A}$ is state-separated if and only if for all state $q \in Q$, reachable states $s^\rightarrow \in \mathcal{S}_q^\rightarrow$ and $s^\leftarrow \in \mathcal{S}_q^\leftarrow$, and $a, b \in \Sigma$,
$f_q(s^\rightarrow, a, \delta_q^\leftarrow(s^\leftarrow, b)) \in Q_e$ if and only if $f_q(\delta_q^\rightarrow(s^\rightarrow, a), b, s^\leftarrow) \in Q_e$.

Proof. ($\Rightarrow$) Assume that $\mathcal{A}$ is state-separated and let $q \in Q$. Moreover, let $s^\rightarrow \in \mathcal{S}_q^\rightarrow$ and $s^\leftarrow \in \mathcal{S}_q^\leftarrow$ be reachable states, and $a, b \in \Sigma$. Then there are $j \geq 0$ and $a_1 \ldots a_j \in \Sigma^*$ such that $s_{q,0}^\rightarrow a_1 \ldots a_j = s^\rightarrow$, and there are $k \geq j+3$ and $a_{j+3} \ldots a_k \in \Sigma^*$ such that $s_{q,0}^\leftarrow a_k \ldots a_{j+3} = s^\leftarrow$. Let $a_{j+1} = a$ and $a_{j+2} = b$, and $x = a_1 \ldots a_k$.

Then $\|\mathcal{B}_q\|_{(s_{q,0}^\rightarrow, s_{q,0}^\leftarrow)}(x) = q_1 \ldots q_k$, where $q_1, \ldots, q_k$ are obtained as follows. Let

- $t_0^\rightarrow t_1^\rightarrow \ldots t_{k-1}^\rightarrow t_k^\rightarrow$ be the $s_{q,0}^\rightarrow$-run of $\mathcal{S}_q^\rightarrow$ on $a_1 \ldots a_k$,
- $t_0^\leftarrow t_1^\leftarrow \ldots t_{k-1}^\leftarrow t_k^\leftarrow$ the $s_{q,0}^\leftarrow$-run of $\mathcal{S}_q^\leftarrow$ on the reversed input $a_k \ldots a_1$, and
- let $q_i = f_q(t_{i-1}^\rightarrow, a_i, t_{k-i}^\leftarrow)$ for $1 \leq i \leq k$.

Here $t_j^\rightarrow = s^\rightarrow$, $t_{j+1}^\rightarrow = \delta_q^\rightarrow(s^\rightarrow, a)$, $t_{k-j-2}^\leftarrow = s^\leftarrow$, $t_{k-j-1}^\leftarrow = \delta_q^\leftarrow(s^\leftarrow, b)$, $f_q(s^\rightarrow, a, \delta_q^\leftarrow(s^\leftarrow, b)) = q_j$, and $f_q(\delta_q^\rightarrow(s^\rightarrow, a), b, s^\leftarrow) = q_{j+1}$. If $q_{j+1} \in Q_e$, then by Lemma 15, $\|\mathcal{B}_q\|(x) \in Q_e^*$. Therefore $q_{j+2} \in Q_e$. Conversely, if $q_{j+2} \in Q_e$, then by Lemma 15, $q_{j+1} \in Q_e$.

($\Leftarrow$) By Lemma 15 it is sufficient to show that $\|\mathcal{B}_q\|(x) \in Q^*Q_eQ^*$ implies $\|\mathcal{B}_q\|(x) \in Q_e^*$ for every $q \in Q$ and $x \in \Sigma^*$.

Let $x = a_1 \ldots a_k$, $k \geq 1$, be arbitrary, and let $\|\mathcal{B}_q\|_{(s_{q,0}^\rightarrow, s_{q,0}^\leftarrow)}(x)$ be as in the first part of the proof. Assume that $q_i \in Q_e$ for some $1 \leq i \leq k$. If $i < k$, then by our assumption, $q_{i+1} = q_e$ as well. Iterating this reasoning, we get that $q_j = q_e$ for each $i \leq j \leq k$. If $i > 1$, then by our assumption, $q_{i-1} = q_e$ as well. As before, we get that $q_j = q_e$ for each $1 \leq j \leq i$. Hence $q_i = q_e$ for each $1 \leq i \leq k$. □

Lemma 17. It is decidable whether $\mathcal{A}$ is state-separated or not.

Proof. The sets Q_+ and $Q_e = Q \setminus Q_+$ are effectively computable (cf. Lemma 10). Then, by direct inspection of $\mathcal{A}$, we can decide whether the condition of Lemma 16 holds. □

In the rest of this section we assume that $\mathcal{A}$ and $\mathcal{A}'$ are state-separated with $Q = Q_+ \cup Q_e$ and $Q' = Q'_+ \cup Q'_e$, respectively. In fact, our minimization algorithm works only for state-separated DTTA.

We introduce the equivalence relation $\tau_\mathcal{A} \subseteq Q \times Q$ as follows: for all $p, q \in Q$,

let $p\tau_\mathcal{A}q$ if and only if $L(\mathcal{A}, p) = L(\mathcal{A}, q)$.

The DTTA $\mathcal{A}$ is *reduced* if $\tau_\mathcal{A}$ is the identity relation.

Lemma 18. Let $q \in Q$ and $q' \in Q'$ such that $L(\mathcal{A}, q) = L(\mathcal{A}', q')$. Moreover, let $k \geq 1$, $a_1 \ldots a_k \in \Sigma^*$, and let $\|\mathcal{B}_q\|(a_1 \ldots a_k) = q_1 \ldots q_k$ and $\|\mathcal{B}_{q'}\|(a_1 \ldots a_k) = q'_1 \ldots q'_k$. Then $L(\mathcal{A}, q_i) = L(\mathcal{A}', q'_i)$ for all $i = 1, \ldots, k$.

Proof. Since $L(\mathcal{A},q) = L(\mathcal{A}',q')$, we have either (1) $q \in Q_e$ and $q' \in Q'_e$ or (2) $q \in Q_+$ and $q' \in Q'_+$. Let us recall that $\mathcal{A}$ and $\mathcal{A}'$ are state-separated.

In case (1) we have $q_1 \dots q_k \in Q_e^*$ and $q'_1 \dots q'_k \in Q_e'^*$, hence the statement holds.

In case (2) either (2a) $q_1 \dots q_k \in Q_e^*$ and $q'_1 \dots q'_k \in Q_e'^*$ or (2b) $q_1 \dots q_k \in Q_+^*$ and $q'_1 \dots q'_k \in Q_+'^*$. (The other two cases are excluded because $L(\mathcal{A},q) = L(\mathcal{A}',q')$.)

In case (2a) the statement again holds, so let us assume that (2b) holds. Arguing by contradiction, assume that $L(\mathcal{A},q_i) \neq L(\mathcal{A}',q'_i)$ for some $1 \leq i \leq k$. Then there exists a tree $\xi \in (L(\mathcal{A},q_i) \setminus L(\mathcal{A}',q'_i)) \cup (L(\mathcal{A}',q'_i) \setminus (L(\mathcal{A},q_i))$ and there are trees $\eta_j \in L(\mathcal{A},q_j)$ and $\theta_j \in L(\mathcal{A}',q'_j)$ for each $j = 1,\dots,i-1,i+1,\dots k$. Hence $a(\eta_1,\dots,\eta_{i-1},\xi,\eta_{i+1},\dots,\eta_k) \in (L(\mathcal{A},q) \setminus L(\mathcal{A}',q'))$ or $a(\theta_1,\dots,\theta_{i-1},\xi,\theta_{i+1},\dots,\theta_k) \in (L(\mathcal{A}',q') \setminus (L(\mathcal{A},q))$. Thus $L(\mathcal{A},q) \neq L(\mathcal{A}',q')$, which is a contradiction. □

Lemma 19. The relation $\tau_{\mathcal{A}}$ is a congruence of $\mathcal{A}$.

Proof. Let $p,q \in Q$ such that $p\tau_{\mathcal{A}}q$. For showing property (i), let $a_1 \dots a_k \in \Sigma^*$ with $\|\mathcal{B}_p\|(a_1 \dots a_k) = p_1 \dots p_k$ and $\|\mathcal{B}_q\|(a_1 \dots a_k) = q_1 \dots q_k$. Then by Lemma 18 with $\mathcal{A} = \mathcal{A}'$, we have $L(\mathcal{A},p_i) = L(\mathcal{A},q_i)$ for all $i = 1,\dots,k$. Hence by the definition of $\tau_{\mathcal{A}}$, $p_i\tau_{\mathcal{A}}q_i$ for every $1 \leq i \leq k$.

Finally, we show that (ii) holds by contradiction as follows: if $p \in F$ and $q \notin F$, then $a \in (L(\mathcal{A},p) \setminus L(\mathcal{A},q))$ for every $a \in \Sigma$ which contradicts to $p\tau_{\mathcal{A}}q$. □

Lemma 20. The DTTA $\mathcal{A}/\tau_{\mathcal{A}}$ is reduced.

Proof. Assume that $L(\mathcal{A}/\tau_{\mathcal{A}},p/\tau_{\mathcal{A}}) = L(\mathcal{A}/\tau_{\mathcal{A}},q/\tau_{\mathcal{A}})$ for some $p,q \in Q$. Then by Lemma 14 and Lemma 19, $L(\mathcal{A},p) = L(\mathcal{A}/\tau_{\mathcal{A}},p/\tau_{\mathcal{A}}) = L(\mathcal{A}/\tau_{\mathcal{A}},q/\tau_{\mathcal{A}}) = L(\mathcal{A},q)$. Hence $p\tau_{\mathcal{A}}q$, i.e., $p/\tau_{\mathcal{A}} = q/\tau_{\mathcal{A}}$. □

Theorem 2. Assume that $\mathcal{A}$ and $\mathcal{A}'$ are connected and reduced. Then

$$L(\mathcal{A}) = L(\mathcal{A}') \text{ if and only if } \mathcal{A} \cong \mathcal{A}'.$$

Proof. We prove the implication from left to right, because the proof of the other direction is obvious. Assume that $L(\mathcal{A}) = L(\mathcal{A}')$. Let us define the relation $\varphi \subseteq Q \times Q'$ as follows: $\varphi = \{(q,q') \mid L(\mathcal{A},q) = L(\mathcal{A}',q')\}$. For convenience, we divide the proof in five steps.

(i) We show that for each $q \in Q$, there exists $q' \in Q'$ such that $(q,q') \in \varphi$, i.e., the domain of φ is Q. As $\mathcal{A}$ is connected, we have

$$f_{\text{in}}(a) \to_{\mathcal{A}} q_1 \to_{\mathcal{A}} \cdots \to_{\mathcal{A}} q_n = q$$

for some $a \in \Sigma$, $n \geq 0$, and $q_1,\dots,q_n \in Q$. If $n = 0$, then $q = f_{\text{in}}(a)$. Since $L(\mathcal{A}) = L(\mathcal{A}')$, we have $L(\mathcal{A},f_{\text{in}}(a)) = L(\mathcal{A}',f'_{\text{in}}(a))$, hence $(q,f'_{\text{in}}(a)) \in \varphi$. If $n \geq 1$, then by Lemma 18 there exists $q'_1,\dots,q'_n \in Q'$ such that

$$f'_{\text{in}}(a) \to_{\mathcal{A}'} q'_1 \to_{\mathcal{A}'} \cdots \to_{\mathcal{A}'} q'_n$$

and $L(\mathcal{A},q_i) = L(\mathcal{A}',q'_i)$ for each $i = 1,\dots,n$. Thus $(q,q'_n) \in \varphi$.

(ii) We show that φ is a mapping. For any $q \in Q$ and $q_1', q_2' \in Q'$, if $(q, q_1') \in \varphi$ and $(q, q_2') \in \varphi$, then $L(\mathcal{A}', q_1') = L(\mathcal{A}, q) = L(\mathcal{A}', q_2')$, and hence $q_1' = q_2'$.

(iii) We show that φ is injective. For any $q_1, q_2 \in Q$ and $q' \in Q'$, if $\varphi(q_1) = q'$ and $\varphi(q_2) = q'$, then $L(\mathcal{A}, q_1) = L(\mathcal{A}, q') = L(\mathcal{A}, q_2)$, and hence $q_1 = q_2$.

(iv) We show that φ is surjective. Repeating the argument used in (i) with the roles of $\mathcal{A}$ and $\mathcal{A}'$ reversed we see that for every $q' \in Q'$ there exists a $q \in Q$ such that $L(\mathcal{A}, q) = L(\mathcal{A}', q')$.

(v) We show that φ is a homomorphism.

First we show that $f'_{\text{in}} = f_{\text{in}} \circ \varphi$. As $L(\mathcal{A}) = L(\mathcal{A}')$, we have $L(\mathcal{A}, f_{\text{in}}(a)) = L(\mathcal{A}', f'_{\text{in}}(a))$ for each $a \in \Sigma$. Hence, by the definition of φ, $\varphi(f_{\text{in}}(a)) = f'_{\text{in}}(a)$ for each $a \in \Sigma$. Thus we have $f'_{\text{in}} = f_{\text{in}} \circ \varphi$.

Second, we show that $\|\mathcal{B}'_{\varphi(q)}\| = \|\mathcal{B}_q\| \circ \varphi^*$ for every $q \in Q$. Let $q \in Q$, $q' \in Q'$ and $a_1 \dots a_k \in \Sigma^*$, $k \geq 1$, with $\|\mathcal{B}_q\|(a_1 \dots a_k) = q_1 \dots q_k$ and $\|\mathcal{B}_{q'}\|(a_1 \dots a_k) = q_1' \dots q_k'$. Then by Lemma 18, $\varphi(q_i) = q_i'$ for each $i = 1, \dots, k$. Hence $\|\mathcal{B}'_{\varphi(q)}\| = \|\mathcal{B}_q\| \circ \varphi^*$ for every $q \in Q$.

Third, we show that $q \in F \iff \varphi(q) \in F'$ for every $q \in Q$. We proceed by contradiction. Assume that $q \in F$ and $\varphi(q) \notin F$ for some $q \in Q$. Then for each $a \in \Sigma$, $a \in L(\mathcal{A}, q)$ and $a \notin L(\mathcal{A}, \varphi(q))$. This is a contradiction. The case $q \notin F$ and $\varphi(q) \in F$ is analogous to the previous case. Thus $\mathcal{A}$ and $\mathcal{A}'$ are isomorphic. $\square$

By Theorem 2, we have the following result.

Corollary 1. Assume that $\mathcal{A}$ and $\mathcal{A}'$ are connected. Then $L(\mathcal{A}) = L(\mathcal{A}')$ if and only if $\mathcal{A}/\tau_{\mathcal{A}} \cong \mathcal{A}'/\tau_{\mathcal{A}'}$.

Proof. Assume that $L(\mathcal{A}) = L(\mathcal{A}')$. Then by Lemmas 14 and 19, we have $L(\mathcal{A}/\tau_{\mathcal{A}}) = L(\mathcal{A}) = L(\mathcal{A}') = L(\mathcal{A}'/\tau_{\mathcal{A}'})$. By Lemma 20, $\mathcal{A}/\tau_{\mathcal{A}}$ and $\mathcal{A}'/\tau_{\mathcal{A}'}$ are connected and reduced. Hence, by Theorem 2 we obtain $\mathcal{A}/\tau_{\mathcal{A}} \cong \mathcal{A}'/\tau_{\mathcal{A}'}$.

Conversely, assume that $\mathcal{A}/\tau_{\mathcal{A}} \cong \mathcal{A}'/\tau_{\mathcal{A}'}$. Then by Lemmas 14 and 19, we have $L(\mathcal{A}) = L(\mathcal{A}/\tau_{\mathcal{A}}) = L(\mathcal{A}'/\tau_{\mathcal{A}'}) = L(\mathcal{A}')$. $\square$

Lemma 21. Let $\varphi : Q \to Q'$ be a homomorphism from $\mathcal{A}$ to $\mathcal{A}'$. Moreover, assume that $\mathcal{B}_q$ is connected for each $q \in Q$ and $\mathcal{B}'_{q'}$ is connected and reduced for each $q' \in Q'$. For every $q \in Q$ and $q' \in Q'$ with $\varphi(q) = q'$, the bimachine $\mathcal{B}'_{q'}$ is a homomorphic image of $\mathcal{B}_q$.

Proof. Let $q \in Q$ and $q' \in Q'$ with $\varphi(q) = q'$. First we show that $\mathcal{T}^{\rightarrow}_{q'}$ is a homomorphic image of $\mathcal{S}^{\rightarrow}_q$. For this, let us define the relation $\psi^{\rightarrow}_{q,q'} \subseteq S^{\rightarrow}_q \times T^{\rightarrow}_{q'}$ by

$$\psi^{\rightarrow}_{q,q'} = \{(s^{\rightarrow}_{q,0}x, t^{\rightarrow}_{q',0}x) \mid x \in \Sigma^*\}.$$

We note that the domain of $\psi^{\rightarrow}_{q,q'}$ is $S^{\rightarrow}_q$ because $\mathcal{B}_q$ is connected. Next we show by contradiction that $\psi^{\rightarrow}_{q,q'}$ is a mapping. For this, let us assume that there are $x, y \in \Sigma^*$ such that $s^{\rightarrow}_{q,0}x = s^{\rightarrow}_{q,0}y$ and $t^{\rightarrow}_{q',0}x \neq t^{\rightarrow}_{q',0}y$. Since $\mathcal{B}'_{q'}$ is reduced, there are $u, z \in \Sigma^*$ such that $\|\mathcal{B}'_{q'}\|_{(t^{\rightarrow}_{q',0}x, t^{\leftarrow}_{q',0}z)}(u) \neq \|\mathcal{B}'_{q'}\|_{(t^{\rightarrow}_{q',0}y, t^{\leftarrow}_{q',0}z)}(u)$, i.e.,

$$\|\mathcal{B}'_{q'}\|_{(t^{\rightarrow}_{q',0}x, t^{\leftarrow}_{q',0})}(uz^{-1}) \neq \|\mathcal{B}'_{q'}\|_{(t^{\rightarrow}_{q',0}y, t^{\leftarrow}_{q',0})}(uz^{-1}).$$

On the other hand, by $\|\mathcal{B}_q\| \circ \varphi^* = \|\mathcal{B}'_{q'}\|$, we have

$$\varphi^*(\|\mathcal{B}_q\|(xuz^{-1})) = \|\mathcal{B}'_{q'}\|(xuz^{-1}) \text{ and } \varphi^*(\|\mathcal{B}_q\|(yuz^{-1})) = \|\mathcal{B}'_{q'}\|(yuz^{-1}).$$

Thus

$$\varphi^*(\|\mathcal{B}_q\|_{(s^{\rightarrow}_{q,0}x, s^{\leftarrow}_{q,0})}(uz^{-1})) = \|\mathcal{B}'_{q'}\|_{(t^{\rightarrow}_{q',0}x, t^{\leftarrow}_{q',0})}(uz^{-1}) \text{ and}$$
$$\varphi^*(\|\mathcal{B}_q\|_{(s^{\rightarrow}_{q,0}y, s^{\leftarrow}_{q,0})}(uz^{-1})) = \|\mathcal{B}'_{q'}\|_{(t^{\rightarrow}_{q',0}y, t^{\leftarrow}_{q',0})}(uz^{-1}).$$

Hence $\varphi^*(\|\mathcal{B}_q\|_{(s^{\rightarrow}_{q,0}x, s^{\leftarrow}_{q,0})}(uz^{-1})) \neq \varphi^*(\|\mathcal{B}_q\|_{(s^{\rightarrow}_{q,0}y, s^{\leftarrow}_{q,0})}(uz^{-1}))$ and thus $\|\mathcal{B}_q\|_{(s^{\rightarrow}_{q,0}x, s^{\leftarrow}_{q,0})}(uz^{-1}) \neq \|\mathcal{B}_q\|_{(s^{\rightarrow}_{q,0}y, s^{\leftarrow}_{q,0})}(uz^{-1})$. This is a contradiction by our assumption $s^{\rightarrow}_{q,0}x = s^{\rightarrow}_{q,0}y$.

Since $\mathcal{B}'_{q'}$ is connected, the mapping $\psi^{\rightarrow}_{q,q'}$ is surjective. Finally we show that $\psi^{\rightarrow}_{q,q'}$ is a homomorphism. Obviously, $\psi^{\rightarrow}_{q,q'}(s^{\rightarrow}_{q,0}) = t^{\rightarrow}_{q',0}$. Moreover, for every $x \in \Sigma^*$ and $a \in \Sigma$, we have

$$\psi^{\rightarrow}_{q,q'}(\delta^{\rightarrow}_{q,0}(s^{\rightarrow}_{q,0}x, a)) = \psi^{\rightarrow}_{q,q'}(s^{\rightarrow}_{q,0}xa) = t^{\rightarrow}_{q',0}xa = \gamma^{\rightarrow}_{q',0}(t^{\rightarrow}_{q',0}x, a) = \gamma^{\rightarrow}_{q',0}(\psi^{\rightarrow}_{q,q'}(s^{\rightarrow}_{q,0}x), a).$$

Analogously, we can define the relation $\psi^{\leftarrow}_{q,q'} \subseteq S^{\leftarrow}_q \times T^{\leftarrow}_{q'}$ and show that it is a homomorphism from $\mathcal{S}^{\leftarrow}_q$ onto $\mathcal{T}^{\leftarrow}_{q'}$. Hence $\mathcal{B}'_{q'}$ is a homomorphic image of $\mathcal{B}_q$ via $(\psi^{\rightarrow}_{q,q'}, \psi^{\leftarrow}_{q,q'})$. □

Lemma 22. Assume that $\mathcal{A}'$ is a homomorphic image of $\mathcal{A}$, that $\mathcal{B}_q$ is connected for each $q \in Q$, and that $\mathcal{B}'_{q'}$ is connected and reduced for each $q' \in Q'$. Then

$$|Q'| \leq |Q|, \sum_{q \in Q'} |T^{\rightarrow}_q| \leq \sum_{q \in Q} |S^{\rightarrow}_q|, \text{ and } \sum_{q \in Q'} |T^{\leftarrow}_q| \leq \sum_{q \in Q} |S^{\leftarrow}_q|.$$

Proof. Let $\varphi : Q \to Q'$ be a surjective homomorphism from $\mathcal{A}$ to $\mathcal{A}'$. By Lemma 21, for every $q \in Q$, the bimachine $\mathcal{B}'_{\varphi(q)}$ is a homomorphic image of $\mathcal{B}_q$. Thus, by Lemma 3, $|T^{\rightarrow}_{\varphi(q)}| \leq |S^{\rightarrow}_q|$ and $|T^{\leftarrow}_{\varphi(q)}| \leq |S^{\leftarrow}_q|$ for every $q \in Q$. Consequently, as φ is a surjective mapping, the statement of the lemma holds. □

Lemma 23. Assume that $\mathcal{A}'$ is a homomorphic image of $\mathcal{A}$ and that $\mathcal{B}'_{q'}$ is connected and reduced for each $q' \in Q'$. Then

$$|Q'| \leq |Q|, \sum_{q \in Q'} |T^{\rightarrow}_q| \leq \sum_{q \in Q} |S^{\rightarrow}_q|, \text{ and } \sum_{q \in Q'} |T^{\leftarrow}_q| \leq \sum_{q \in Q} |S^{\leftarrow}_q|.$$

Proof. Let $\mathcal{B}^c_q$ be the connected part of $\mathcal{B}_q$ for each $q \in Q$. As mentioned, the bimachine $\mathcal{B}^c_q$ is equivalent to $\mathcal{B}_q$ for each $q \in Q$. Hence the DTTA $\big(Q, \Sigma, f, (\mathcal{B}^c_q \mid q \in Q), F\big)$ is equivalent to $\mathcal{A}$ and, obviously,

$$\sum_{q \in Q} |S^{\rightarrow c}_q| \leq \sum_{q \in Q} |S^{\rightarrow}_q|, \text{ and } \sum_{q \in Q} |S^{\leftarrow c}_q| \leq \sum_{q \in Q} |S^{\leftarrow}_q|.$$

Moreover, $\mathcal{A}'$ is a homomorphic image of $\big(Q, \Sigma, f, (\mathcal{B}^c_q \mid q \in Q), F\big)$. Hence by Lemma 22,

$$|Q'| \le |Q|, \sum_{q\in Q'} |T^{\rightarrow}_q| \le \sum_{q\in Q} |S^{\rightarrow c}_q|, \text{ and } \sum_{q\in Q'} |T^{\leftarrow}_q| \le \sum_{q\in Q} |S^{\leftarrow c}_q|.$$

These and the above inequalities imply the lemma. □

Lemma 24. Assume that $\mathcal{A}$ is connected and consider $\mathcal{A}/\tau_{\mathcal{A}} = \big(Q/\tau_{\mathcal{A}}, \Sigma, f_{\mathrm{in},\tau_{\mathcal{A}}}, (\mathcal{B}_{q/\tau_{\mathcal{A}}} \mid q/\tau_{\mathcal{A}} \in Q/\tau_{\mathcal{A}}), F/\tau_{\mathcal{A}}\big)$. For each $q/\tau_{\mathcal{A}} \in Q/\tau_{\mathcal{A}}$, let $\mathcal{B}^c_{q/\tau_{\mathcal{A}}}$ be the connected part of $\mathcal{B}_{q/\tau_{\mathcal{A}}}$ and let

$$\mathcal{M} = \big(Q/\tau_{\mathcal{A}}, \Sigma, f_{\mathrm{in},\tau_{\mathcal{A}}}, (\mathcal{B}^c_{q/\tau_{\mathcal{A}}}/\rho_{\mathcal{B}^c_{q/\tau_{\mathcal{A}}}} \mid q/\tau_{\mathcal{A}} \in Q/\tau_{\mathcal{A}}), F/\tau_{\mathcal{A}}\big).$$

Then $\mathcal{M}$ is a minimal DTTA and equivalent to $\mathcal{A}$.

Proof. Let $L(\mathcal{A}) = L(\mathcal{A}')$. By Propositions 4 and 5, we may assume that $\mathcal{A}'$ is connected. Then, by Corollary 1, $\mathcal{A}'/\tau_{\mathcal{A}'} \cong \mathcal{A}/\tau_{\mathcal{A}}$. Hence, by Lemmas 4 and 19, there is a surjective homomorphism $\varphi : Q' \to Q/\tau_{\mathcal{A}}$ from $\mathcal{A}'$ to $\mathcal{A}/\tau_{\mathcal{A}}$. Therefore, φ is a surjective homomorphism from $\mathcal{A}'$ to $\mathcal{M}$. Consequently, by Lemma 23,

- $|Q/\tau_{\mathcal{A}}| \le |Q'|$,
- $\sum_{q/\tau_{\mathcal{A}}\in Q/\tau_{\mathcal{A}}} |S^{\rightarrow c}_{q/\tau_{\mathcal{A}}}/\rho_{\mathcal{B}^c_{q/\tau_{\mathcal{A}}}}| \le \sum_{q\in Q'} |T^{\rightarrow}_q|$, and
- $\sum_{q/\tau_{\mathcal{A}}\in Q/\tau_{\mathcal{A}}} |S^{\leftarrow c}_{q/\tau_{\mathcal{A}}}/\rho_{\mathcal{B}^c_{q/\tau_{\mathcal{A}}}}| \le \sum_{q\in Q'} |T^{\leftarrow}_q|$.

Therefore, $\mathcal{M}$ is a minimal DTTA. By Lemma 14, $\mathcal{A}/\tau_{\mathcal{A}}$ is equivalent to $\mathcal{A}$. Hence, by Lemma 5, $\mathcal{M}$ is equivalent to $\mathcal{A}$ as well. □

In the rest of the paper we give an algorithm which computes the minimal DTTA which is equivalent to $\mathcal{A}$. For this we will need the concept of the direct product of bimachines. The *direct product of the semi-automata $\mathcal{S}$ and $\mathcal{T}$* is the semi-automaton $\mathcal{S}\times\mathcal{T} = (S\times T, \Sigma, (s_0, t_0), \delta'')$, where $\delta''((s,t), a) = (\delta(s,a), \delta'(t,a))$ for every $(s,t) \in S \times T$ and $a \in \Sigma$. The *direct product of the bimachines $\mathcal{B}$ and $\mathcal{B}'$* is the bimachine

$$\mathcal{B} \times \mathcal{B}' = (\Sigma, \Gamma \times \Gamma, \mathcal{S}^{\rightarrow} \times \mathcal{T}^{\rightarrow}, \mathcal{S}^{\leftarrow} \times \mathcal{T}^{\leftarrow}, f''),$$

where $f''((s^{\rightarrow}, t^{\rightarrow}), a, (s^{\leftarrow}, t^{\leftarrow})) = (f(s^{\rightarrow}, a, s^{\leftarrow}), f'(t^{\rightarrow}, a, t^{\leftarrow}))$ for all $(s^{\rightarrow}, t^{\rightarrow}) \in S^{\rightarrow} \times T^{\rightarrow}$, $(s^{\leftarrow}, t^{\leftarrow}) \in S^{\leftarrow} \times T^{\leftarrow}$, and $a \in \Sigma$.

To give an algorithm which computes the minimal automaton equivalent to $\mathcal{A}$, we define the relation $\tau_n \subseteq Q \times Q$ for every $n \ge 0$, by induction on n.

Base of induction: For each $p, q \in Q$, let $p\tau_0 q$ if and only if ($p \in F \iff q \in F$).

Induction step: Let $n \ge 0$ and assume that we have defined τ_n. For each $p, q \in Q$, let $p\tau_{n+1}q$ if and only if

- $p\tau_n q$ and

- for the bimachine $\mathcal{B}_p \times \mathcal{B}_q = (\Sigma, Q \times Q, \mathcal{S}_p^{\rightharpoondown} \times \mathcal{S}_q^{\rightharpoondown}, \mathcal{S}_p^{\leftharpoondown} \times \mathcal{S}_q^{\leftharpoondown}, f_{(p,q)})$ and for any reachable pair $((s^{\rightharpoondown}, t^{\rightharpoondown}), (s^{\leftharpoondown}, t^{\leftharpoondown}))$ in $\mathcal{B}_p \times \mathcal{B}_q$ and $a \in \Sigma$, if $f_{(p,q)}((s^{\rightharpoondown}, t^{\rightharpoondown}), a, (s^{\leftharpoondown}, t^{\leftharpoondown})) = (r_1, r_2)$, then we have $r_1 \tau_n r_2$.

Lemma 25. For each $n \geq 0$, τ_n is an equivalence relation.

Proof. We proceed by induction on n.

Base of induction: $n = 0$. By definition, τ_0 is an equivalence relation.

Induction step: We assume that the lemma holds for $n \geq 0$, and show that it also holds for $n+1$. By definition and the induction hypothesis, τ_{n+1} is reflexive and symmetric. We will show that τ_{n+1} is transitive. To this end, let $p, q, r \in Q$, and assume that $p\tau_{n+1}q$ and $q\tau_{n+1}r$. Since $p\tau_{n+1}q$ and $q\tau_{n+1}r$, we have $p\tau_n q$ and $q\tau_n r$. By the induction hypothesis, $p\tau_n r$. All is left to show is that for the bimachine $\mathcal{B}_p \times \mathcal{B}_r = (\Sigma, Q \times Q, \mathcal{S}_p^{\rightharpoondown} \times \mathcal{S}_r^{\rightharpoondown}, \mathcal{S}_p^{\leftharpoondown} \times \mathcal{S}_r^{\leftharpoondown}, f_{(p,r)})$ and for any reachable pair $((s^{\rightharpoondown}, t^{\rightharpoondown}), (s^{\leftharpoondown}, t^{\leftharpoondown}))$ in $\mathcal{B}_p \times \mathcal{B}_r$ and $a \in \Sigma$, if $f_{(p,r)}((s^{\rightharpoondown}, t^{\rightharpoondown}), a, (s^{\leftharpoondown}, t^{\leftharpoondown})) = (p', r')$, then we have $p'\tau_n r'$. To this end, take a word $w = a_1 \dots a_k \in \Sigma^*$, $k \geq 1$, such that

- $(s_0^{\rightharpoondown}, t_0^{\rightharpoondown})(s_1^{\rightharpoondown}, t_1^{\rightharpoondown}) \dots (s_{k-1}^{\rightharpoondown}, t_{k-1}^{\rightharpoondown})(s_k^{\rightharpoondown}, t_k^{\rightharpoondown})$ is the run of $\mathcal{S}_p^{\rightharpoondown} \times \mathcal{S}_r^{\rightharpoondown}$ on $a_1 \dots a_k$,
- $(s_0^{\leftharpoondown}, t_0^{\leftharpoondown})(s_1^{\leftharpoondown}, t_1^{\leftharpoondown}) \dots (s_{k-1}^{\leftharpoondown}, t_{k-1}^{\leftharpoondown})(s_k^{\leftharpoondown}, t_k^{\leftharpoondown})$ is the run of $\mathcal{S}_p^{\leftharpoondown} \times \mathcal{S}_r^{\leftharpoondown}$ on $a_k \dots a_1$,
- $((s^{\rightharpoondown}, t^{\rightharpoondown}), a, (s^{\leftharpoondown}, t^{\leftharpoondown})) = ((s_{j-1}^{\rightharpoondown}, t_{j-1}^{\rightharpoondown}), a_j, (s_{k-j}^{\leftharpoondown} t_{k-j}^{\leftharpoondown}))$ for some $1 \leq j \leq k$ and
- $(p_i, r_i) = f_{(p,r)}((s_{i-1}^{\rightharpoondown}, t_{i-1}^{\rightharpoondown}), a_i, (s_{k-i}^{\leftharpoondown}, t_{k-i}^{\leftharpoondown}))$ for $1 \leq i \leq k$.

Then $\|\mathcal{B}_p \times \mathcal{B}_r\|(w) = (p_1, r_1) \dots (p_k, r_k)$ and $(p', r') = (p_j, r_j)$.

Let $y_0^{\rightharpoondown} y_1^{\rightharpoondown} \dots y_{k-1}^{\rightharpoondown} y_k^{\rightharpoondown}$ be the run of $\mathcal{S}_q^{\rightharpoondown}$ on $a_1 \dots a_k$, and $y_0^{\leftharpoondown} y_1^{\leftharpoondown} \dots y_{k-1}^{\leftharpoondown} y_k^{\leftharpoondown}$ the run of $\mathcal{S}_q^{\leftharpoondown}$ on the reversed input $a_k \dots a_1$, and let $q_i = f(y_{i-1}^{\rightharpoondown}, a_i, y_{k-i}^{\leftharpoondown})$ for $1 \leq i \leq k$. Then $\|\mathcal{B}_q\|(w) = q_1 \dots q_k$ and $\|\mathcal{B}_p \times \mathcal{B}_q\|(w) = (p_1, q_1) \dots (p_k, q_k)$ and $\|\mathcal{B}_q \times \mathcal{B}_r\|(w) = (q_1, r_1) \dots (q_k, r_k)$. Since $p\tau_{n+1}q$ and $q\tau_{n+1}r$, we have $p_j \tau_n q_j$ and $q_j \tau_n r_j$. By the induction hypothesis, τ_n is an equivalence relation, hence $p_j \tau_n r_j$. Since $(p', r') = (p_j, r_j)$, we have $p'\tau_n r'$. Therefore $p\tau_{n+1}r$, and hence τ_{n+1} is transitive. □

Obviously, we have

$$\tau_0 \supseteq \tau_1 \supseteq \tau_2 \supseteq \cdots$$

and thus there is an integer $n_0 \geq 0$ such that $\tau_{n_0} = \tau_{n_0+1}$. Moreover, we can prove that $\tau_{n_0} = \tau_{n_0+1}$ implies $\tau_{n_0+1} = \tau_{n_0+2} = \cdots$ for every $n_0 \geq 0$.

Lemma 26. For all $n, l \geq 0$, $p, q \in Q$, $\xi \in T_\Sigma$ with $\text{height}(\xi) \geq l$, $x \in \text{dom}(\xi)$ with $|x| = l$, p-run r_p of $\mathcal{A}$ on ξ and q-run r_q of $\mathcal{A}$ on ξ, if $p\tau_{n+l}q$, then $r_p(x)\tau_n r_q(x)$.

Proof. We proceed by induction on l. If $l = 0$, then $p = r_p(x)$ and $q = r_q(x)$. By our assumption $p\tau_{n+0}q$, we have $r_p(x)\tau_n r_q(x)$.

Induction step: We assume that the lemma holds for $l \geq 0$, and show that it also holds for $l+1$. To this end, let $\xi \in T_\Sigma$ with $\xi = a(\xi_1 \dots \xi_k)$, $\text{height}(\xi) \geq l+1$, and let $x = iy$, where $0 \leq i \leq k$, $|x| = l+1$ and hence $|y| = l$, and assume that $p\tau_{n+l+1}q$. Consider an arbitrary p-run r_p of $\mathcal{A}$ on ξ and an arbitrary q-run r_q of $\mathcal{A}$ on ξ. If $r_p(i) = p'$ and $r_q(i) = q'$, then by the definition of τ_{n+l+1}, $p'\tau_{n+l}q'$. Hence, by the induction hypothesis, for the p'-run $r_{p'}$ of $\mathcal{A}$ on ξ_i and for the q'-run $r_{q'}$ of $\mathcal{A}$ on ξ_i, we have $r_{p'}(y)\tau_n r_{q'}(y)$. Observe that $r_{p'}(y) = r_p(x)$ and $r_{q'}(y) = r_q(x)$. Consequently, $r_p(x)\tau_n r_q(x)$. □

Lemma 27. Let n_0 be the least integer with $\tau_{n_0} = \tau_{n_0+1}$. Then $\tau_{n_0} = \tau_{\mathcal{A}}$.

Proof. First we show that $\tau_{n_0} \subseteq \tau_{\mathcal{A}}$. Let $p\tau_{n_0}q$. Then $p\tau_{n_0+l}q$ for each $l \geq 0$, hence by Lemma 26, for all $l \geq 0$, $\xi \in T_\Sigma$ with height$(\xi) \geq l$, $x \in \mathrm{dom}(\xi)$ with $|x| = l$, p-run r_p of $\mathcal{A}$ on ξ and q-run r_q of $\mathcal{A}$ on ξ, we have $r_p(x)\tau_n r_q(x)$. By the inclusion $\tau_0 \supseteq \tau_{n_0}$, we have $r_p(x)\tau_0 r_q(x)$. Hence, by the definition of ρ_0, we have $(r_p(x) \in F$ if and only if $r_q(x) \in F)$. Since $l \geq 0$, $\xi \in T_\Sigma$, and $x \in \mathrm{dom}(\xi)$ are arbitrary, $L(\mathcal{A}, p) = L(\mathcal{A}, q)$.

We now show that $\tau_{\mathcal{A}} \subseteq \tau_{n_0}$. To this end we show that for all $p, q \in Q$, $n \geq 0$, if $(p, q) \notin \tau_n$, then $(p, q) \notin \tau_{\mathcal{A}}$. We proceed by induction on n.

Base of induction: $n = 0$. If $(p, q) \notin \tau_0$, then ($p \in F$ if and only if $q \notin F$). Hence $L(\mathcal{A}, p) \neq L(\mathcal{A}, q)$ and thus $p\tau_A q$ does not hold.

Induction step. Assume that $p\tau_{n+1}q$ does not hold. Then $p\tau_n q$ does not hold or $p\tau_n q$ and there is a word $z \in \Sigma^*$ such that $\|\mathcal{B}_p\|(z) = p_1 \dots p_k$ and $\|\mathcal{B}_q\|(z) = q_1 \dots q_k$ and $(p_i, q_i) \notin \tau_n$ for some $1 \leq i \leq k$. In the first case, by the induction hypothesis, $(p, q) \notin \tau_{\mathcal{A}}$. In the second case, $L(\mathcal{A}, p_i) \setminus L(\mathcal{A}, q_i) \neq \emptyset$ or $L(\mathcal{A}, q_i) \setminus L(\mathcal{A}, p_i) \neq \emptyset$. If $L(\mathcal{A}, p_i) \setminus L(\mathcal{A}, q_i) \neq \emptyset$, then let $\xi_i \in (L(\mathcal{A}, p_i) \setminus L(\mathcal{A}, q_i))$, otherwise let $\xi_i \in L(\mathcal{A}, p_i)$. If $L(\mathcal{A}, q_i) \setminus L(\mathcal{A}, p_i) \neq \emptyset$, then let $\zeta_i \in (L(\mathcal{A}, q_i) \setminus L(\mathcal{A}, p_i))$, otherwise let $\zeta_i \in L(\mathcal{A}, q_i)$. For each $1 \leq j \leq k$ with $j \neq i$, let $\xi_j \in L(\mathcal{A}, p_j)$ and $\zeta_j \in L(\mathcal{A}, q_j)$. Then let $\xi = a(\xi_1 \dots \xi_k)$ and $\zeta = a(\zeta_1 \dots \zeta_k)$. Consequently, $\xi \in (L(\mathcal{A}, p) \setminus L(\mathcal{A}, q))$ or $\zeta \in (L(\mathcal{A}, q) \setminus L(\mathcal{A}, p))$. Hence $L(\mathcal{A}, p) \neq L(\mathcal{A}, q)$ and thus $p\tau_A q$ does not hold. □

Proposition 7. There is a polynomial time algorithm which constructs $\mathcal{A}/\tau_{\mathcal{A}}$ for a given $\mathcal{A}$.

Proof. We compute τ_1 in $\mathcal{O}(|Q|^2)$ time. For every $1 < n \leq n_0$, the relation τ_n can be computed in $\mathcal{O}(|Q|^2(N^{\rightarrow})^2|\Sigma|(N^{\leftarrow})^2)$ time, where $N^{\rightarrow} = \max\{|S_p^{\rightarrow}| \mid p \in Q\}$ and $N^{\leftarrow} = \max\{|S_p^{\leftarrow}| \mid p \in Q\}$. Since there are at most $|Q|$ steps, the relation τ_{n_0} can be computed in $\mathcal{O}(|Q|^3(N^{\rightarrow})^2|\Sigma|(N^{\leftarrow})^2)$ time. □

Theorem 3. There is a polynomial time algorithm which constructs for $\mathcal{A}$ an equivalent minimal DTTA.

Proof. By Propositions 6, 7, 2, and 3, respectively, we compute the following sequence of DTTAs in polynomial time.

1) The connected part $\mathcal{A}^c = \big(Q^c, \Sigma, f^c_{\mathrm{in}}, (\mathcal{B}_q \mid q \in Q^c), F^c\big)$ of $\mathcal{A}$.

2) The congruence $\tau_{\mathcal{A}^c}$ and the DTTA

$$\mathcal{A}^c/\tau_{\mathcal{A}^c} = \big(Q^c/\tau_{\mathcal{A}^c}, \Sigma, f^c_{\mathrm{in},\tau^c_{\mathcal{A}}}, (\mathcal{B}_{q/\tau_{\mathcal{A}^c}} \mid q/\tau_{\mathcal{A}^c} \in Q^c/\tau_{\mathcal{A}^c}), F^c/\tau_{\mathcal{A}^c}\big).$$

3) For each $q/\tau_{\mathcal{A}^c} \in Q^c/\tau_{\mathcal{A}^c}$, the connected part $\mathcal{B}^c_{q/\tau_{\mathcal{A}^c}}$ of $\mathcal{B}_{q/\tau_{\mathcal{A}^c}}$.

4) The DTTA

$$\big(Q^c/\tau_{\mathcal{A}^c}, \Sigma, f^c_{\mathrm{in},\tau_{\mathcal{A}^c}}, (\mathcal{B}^c_{q/\tau_{\mathcal{A}^c}}/\rho_{\mathcal{B}^c_{q/\tau_{\mathcal{A}^c}}} \mid q/\tau_{\mathcal{A}^c} \in Q^c/\tau_{\mathcal{A}^c}), F^c/\tau_{\mathcal{A}^c}\big).$$

By Lemma 24, the latter one is a minimal DTTA which is equivalent to $\mathcal{A}$. □

References

[1] Björklund J. and Cleophas L. A Taxonomy of Minimisation Algorithms for Deterministic Tree Automata. Journal of Universal Computer Science vol. 22(2): 180–196, 2016.

[2] Brüggemann-Klein A., Murata M., and Wood D., *Regular tree and regular hedge languages over unranked trees.* Technical Report HKUST-TCSC-2001-0, The Hong Kong University of Science and Technology, Hong Kong, China, 2001.

[3] Comon H., Dauchet M., Gilleron R., Löding C., Jacquemard F., Lugiez D., Tison S., and Tommasi M. *Tree Automata Techniques and Applications.* http://www.grappa.univ-lille3.fr/tata, 2007.

[4] Cristau J., Löding C., and Thomas W. Deterministic Automata on Unranked Trees. In Liskiewicz M. and Reischuk R. editors, *Fundamentals of Computation Theory, 15th International Symposium, FCT 2005, Proceedings*, Lecture Notes in Computer Science 3623, pages 68–79. Springer-Verlag, Berlin, 2005.

[5] Gécseg F. and Steinby M. Minimal ascending tree automata. *Acta Cybernetica* 4(1): 37–44, 1978.

[6] Jiang T. and Ravikumar B. Minimal NFA problems are hard. *SIAM Journal on Computing* 22(6): 1117–1141, 1993.

[7] Martens W. and Niehren J. On the minimization of XML Schemas and tree automata for unranked trees. *Journal of Computer and Systems Sciences* 73(4): 550–583, 2007.

[8] Martens W., Neven F., and Schwentick T. Deterministic Top-down Tree Automata: Past, Present, and Future. In Flum J., Grädel E, and Wilke T. editors, *Logic and Automata – History and Perspectives*, pages 505–530. Amsterdam University Press, 2008.

[9] Martens W., Neven F., Schwentick T., and Bex G. J. Expressiveness and complexity of XML Schema, *Journal ACM Transactions on Database Systems (TODS)* 31(3): 770–813, 2006.

[10] Neven F. Automata theory for XML researchers. *ACM Sigmod Record* 31(3): 39–46, 2002.

[11] Piao X. and Salomaa K. Operational State Complexity of Deterministic Unranked Tree Automata, In McQuillan I. and Pighizzini G. editors, *Proceedings Twelfth Annual Workshop on Descriptional Complexity of Formal Systems, DCFS 2010, Electronic Proceedings in Theoretical Computer Science* 31: 149–158, 2010.

[12] Piao X. and Salomaa K. Lower bounds for the size of deterministic unranked tree automata, *Theoretical Computer Science* 454: 231–239, 2012.

Curriculum Vitae of Zoltán Ésik

Education and Academic Degrees

1. Doctor of the Hungarian Academy of Science (DSc), 1996. Thesis: Iteration Theories.
2. Habilitation in Computer Science: Attila József University, Szeged, 1995. Thesis: Iteration Theories.
3. Candidate of Mathematical Sciences: Hungarian Academy of Sciences, 1985. Thesis: Top-down tree transformations.
4. University Doctor (PhD): Attila József University, Szeged, 1979. Thesis: Decidability results concerning tree transformations.
5. University Diploma (MSc) in Mathematics: Attila József University, Szeged, 1974.

Employment and related activities

1. 2003–2016 : Head of the Department of Foundations of Computer Science.
2. 1997–2016 : Full Professor, Department of Foundations of Computer Science, Institute of Informatics, University of Szeged (formerly: Attila József University), Hungary.
3. 1996–1997: Part-time Associate Professor, Institute of Mathematics and Informatics, Lajos Kossuth University, Debrecen.
4. 1990–1996: Associate Professor, Department of Foundations of Computer Science, Institute of Informatics, Attila József University, Szeged, Hungary.
5. 1987–1989: Associate Professor, Department of Computer Science, Institute of Mathematics, Attila József University, Szeged, Hungary.
6. 1979–1986: Assistant Professor, Department of Computer Science, Institute of Mathematics, Attila József University, Szeged, Hungary.
7. 1974–1978: Assistant Lecturer, Department of Computer Science, Institute of Mathematics, Attila József University, Szeged, Hungary.

DOI: 10.14232/actacyb.23.1.2017.22

Visiting Positions and Fellowships

1. Visiting Scientist, Kyoto Sangyo University, Kyoto, Japan: September 2008, September 2007, October 2006, June 2005, August 2003, September 2002, January 2001 and March 2000.
2. Visiting Professor, University of Bordeaux, France: May 2003.
3. Visiting Professor, TU Dresden, Germany: June–July 2002.
4. Visiting Professor, LIAFA, Denis Diderot University, Paris: May 2002.
5. Visiting Professor, University of Aalborg, Denmark: May 2001–April 2002.
6. Visiting Professor, University of Waterloo, Canada: June 2000.
7. Visiting Professor, University of Aizu, Japan: January–June 1999.
8. Fulbright Research Fellow, Stevens Institute of Technology, Hoboken, USA: June–September 1997.
9. JSPS Research Fellow, Kyoto Sangyo University, Japan: February – April, 1997.
10. Visiting Professor, LIAFA, Université Paris 6 and 7, France: May 1997.
11. Alexander von Humboldt Research Fellow, Institute of Informatics, University of Stuttgart, Germany: September–November 1994.
12. COST Research Fellow, LFCS, Department of Computer Science, The University of Edinburgh, UK: September–November 1993.
13. Visiting Professor, Department of Computer Science, Stevens Institute of Technology, New Jersey, USA: September–November 1990.
14. Alexander von Humboldt Research Fellow, Institute of Informatics, TU Munich, Germany: 1988–1989.
15. Visiting Professor, Department of Pure and Applied Mathematics, Stevens Institute of Technology, New Jersey, USA: September 1983–August 1984.

Courses Conducted

Mathematical foundations of logic and functional programming. Process algebra. Finite transition systems. Finite model theory. Logic in computer science. Foundations of computer science. Category theory in computer science. Universal algebra in computer science. Complexity theory. Computability theory. Iteration theories. Automata and formal languages. Introduction to analysis. Linear algebra. Automata and formal logic. Compilers. Mathematical logic. Algebraic and graph theoretic properties of block schemes. Formal semantics. Algorithm theory. Mathematical foundations of software. Mathematical foundations of programming. System programming.

Research Interest

Automata and formal language theory. Algebra, categories and logic in computer science. Fixed point theory. Iteration theories. Temporal logics. Concurrency. Semantics.

Publications

Two books, four book chapters, 32 edited volumes, approx. 250 research articles.

Editorial Work

Member of the Editorial Board of: Journal of Automata, Languages and Combinatorics (2016), Journal of Mathematics and Computer Science (2009–2016), Alkalmazott Matematikai Lapok (Journal of Applied Mathematics, in Hungarian, 2004–2016), Theoretical Computer Science (1999–2015), Theoretical Informatics and Applications (1998–2016), Discrete Mathematics and Theoretical Computer Science (1996–2001), Acta Cybernetica (1987–2016), Acta Scientiarium Mathematica (1986–2000), Algebra (Hindawi Publishers).

Membership in International Learned Bodies and Professional Associations

1. Fellow of the EATCS, 2016.
2. Member of the Presburger Award Committee, 2015–2016.
3. Member of the Academia Europaea, 2010–2016.
4. Member of the Steering Committee of the FICS workshop series, 2009–2012.
5. Member of the Steering Committee of the Algebraic Informatics conference series, 2007–2016.
6. Member of the Board of the European Association for Computer Science Logic, 2005–2016.
7. Member of the WG 1.8 (Concurrency) of IFIP TC1.
8. Member of the Council of the European Association for Theoretical Computer Science (EATCS), 2004–2016.

9. Member of the IFIP Technical Committee 1 (Theoretical Computer Science), 2001–2016. (Hungarian representative)
10. Member of the Steering Committee of the Fundamentals of Computation Science conferences, 1998–2016.
11. Member of the EATCS, 1985–2016.

Membership in Learned Bodies, Professional Associations and Committees in Hungary

1. Member of the Scientific and Habilitation Committee, University of Debrecen, Faculty of Informatics, 2010–2016.
2. Member of the Board of Experts, National Fellowship Committee of Hungary, 2008–2016.
3. Member of the Mathematics Committee of the János Bolyai Fellowship Award, Hungarian Academy of Sciences, 2006–2015.
4. Member of the Computer Science Committee of the Hungarian Academy of Science, 2005–2007, 2002–2004, 1994–1996.
5. Member of the Board for Natural Sciences and Mathematics, National Foundation for Scientific Research of Hungary, 2003–2005.
6. Leader of the PhD Program in Informatics of the University of Szeged, 2001–2004.
7. Deputy director of the PhD School in Mathematics and Computer Science of the University of Szeged, 2001–2004.
8. Member of the Computer Science Habilitation Committee, University of Szeged, 1997–2016.
9. Member of the Doctoral Committee of the Faculty of Science of Lajos Kossuth University, Debrecen, 1996–1998.
10. Member of the Mathematical Jury of the Hungarian National Foundation for Scientific Research, 1995–1998.
11. Chairman of the Doctoral Committee of the Institute of Informatics of the Faculty of Science of University of Szeged, 1993–2004.
12. Member of the Doctoral Committee of the Faculty of Science of University of Szeged, 1992–2004.

Invited Lectures at Conferences, Workshops and Summer Schools

Trends in Tree Automata and Tree Transducers, Seoul, 2016. Mathematical Foundations of Computer Science, Milan, 2015. Automata, Logic, Formal languages and Algebra, Bordeaux, 2015. The Role of Theory in Computer Science, Waterloo, 2015. Semigroups, Languages and Algebras, Akita, 2014. Weighted Automata: Theory and Applications, Leipzig, 2014. Automata, Logic, Formal languages and Algebra, Stellenbosch, 2013. Mathematics and Informatics, Targu Mures, 2013. Weighted Automata: Theory and Applications, Dresden, 2012. Lattices and Relations, Amsterdam, 2012. Algebras, Languages, Algorithms and Computation, Kyoto, 2011. Highlights of AUTOMATHA, Vienna, 2010. Weighted Automata: Theory and Applications, Leipzig, 2010. Dagstuhl seminar on Quantitative Models, 2010. AUTOMATHA, Liege, 2009. Summer School on Algebraic Theory of Automata, Lisbon, 2008. Developments in Language Theory, Kyoto, 2008. Weighted Automata: Theory and Applications, Dresden, 2008. Algebraic Informatics, Thessaloniki, 2005. Mathematical Foundations of Computer Science, Palics, 2005. Novi Sad Algebraic Conference, 2005. International Ph. D. School of Formal Language Theory and Applications, Tarragona, 2004. Joint Mathematics Meeting, ASM Special Session on Fixed Points, Phoenix, 2004. Weighted Automata: Theory and Applications, Dresden, 2004. Categorical Methods in Computer Science, Warsaw, 2003. International School of Formal Language Theory, Tarragona, 2003. Developments in Language Theory, Kyoto, 2002. Weighted Automata and Applications, Dresden, 2002. Int. Conf. Discrete Mathematics and Applications, Blagoevgrad, 2001. Dagstuhl seminar on Applications of Kleene Algebra, 2001. Expressiveness in Concurrency, Aalborg, 2001. Workshop on Max-Plus Algebras, Prague, 2001. Fixed Points in Computer Science, Florence, 2001. Developments in Language Theory, Vienna, 2001. Dagstuhl Seminar on Logic, Algebra, and Formal Verification, 2000. Category Theory, Como, 2000. Words, Languages and Combinatorics, Kyoto, 2000. Algebraic Engineering, Aizu, 1997. Logic in Computer Science, Novi Sad, 1995. Mathematical Foundations of Computer Science, Kosice, 1994. IMYCS, Smolenice, 1988.

Other conference and seminar presentations

Talks at cca. 80 international conferences and workshops. Seminar and/or colloquium presentations at the following universities and research centers (several talks at some places): University of Aalborg, BRICS, Aarhus, University of Aizu, CWI, Amsterdam, Autonomous University of Barcelona, LaBRI, University of Bordeaux, University of Bremen, University of Brno, University of Edinburgh, University of Florence, University of Hamburg, University of Hanover, University of Kassel, RIMS, Kyoto, Kyoto Sangyo University, Unversity of Leipzig, University of Linz, University of Magdeburg, University of Matsue, University of Metz, Technical University of Munich, City University of New York, LIAFA, University of Paris 6 and 7, University of Pisa, University of Reykjavik, University of Rome, University of Saarbrücken, University of Sao Paulo, Stanford University, Sydney Category Seminar, UCLA, Shimane University, University of Stuttgart, Stevens Institute of Technology, University of Tarragona, University of Tsukuba, University of Turku, TU Vienna, University of Waterloo.

Other Professional Activities

Member of the Program Committee of DCFS 2016, Bucharest, Highlights of Logic, Games and Automata 2015, Prague, CAI 2015, Stuttgart, AutoMathA 2015, Leipzig, ICALP 2015, Kyoto, Highlights of Logic, Games and Automata 2014, Paris, MFCS 2014, Budapest (co-chair), AFL 2014, Szeged (co-chair), CSR 2014, Moscow, DLT 2013, Paris, MFCS 2013, Wien, CAI 2013, Marseilles, FICS 2012, Tallinn, FSTTCS 2011, Mumbai, DLT 2011, Milan, ICALP 2011, Zurich, AFL 2011, Debrecen, DCFS 2011, Giessen, FSTTCS 2010, Chennai, FICS 2010, Brno, FICS 2009, Coimbra, MEMICS 2009, Brno, DLT 2009, Stuttgart, QUANTLOG 2009, Rhodes (chair), STACS 2009, Freiburg, MEMICS 2008, Brno, LATA 2008, Tarragona, AFL 2008, Balatonfüred (co-chair), DLT 2008, Kyoto, FOSSACS 2008, Budapest, FCT 2007, Budapest (co-chair), LATA 2007, Tarragona, MEMICS 2007, Brno, CAI 2007, Thessaloniki, Algebraic Theory of Automata and Logic, 2006, Szeged, Logic, Models and Computer Science, 2006, Camerino, DLT 2006, Santa Barbara, RELMICS /AKA 2006, Manchester, CSL 2006, Szeged (chair), FOSSACS 2006, Vienna, CSL 2005, Oxford, ICALP 2005, Lisboa, AFL 2005, Dobogókő (co-chair), DLT 2005, Palermo, DLT 2004, Auckland, Process Algebra: Open Problems and Future Directions, Bertinoro, 2003, EXPRESS 2003, Marseilles, FCT 2003, Malmö, FICS 2003, Warsaw (co-chair), DLT 2003, Szeged (co-chair), AFL 2002, Debrecen, FICS 2002, Copenhagen (co-chair), FICS 2000, Paris, AFL 1999, Vasszécsény, FICS 1998, Brno (chair), FOSSACS 1998, Lisbon, Universal Machines and Computations 1998, Metz, FCT 1997, Krakow, Logic in Computer Science LIRA 1997, Novi Sad, AFL 1996, Salgótarján, FCT 1995, Dresden, STACS 1995, Munich, FCT 1993, Szeged (chair).

Professional Awards

1. Master teacher, Ministry of Education of Hungary, 2005.

2. Széchenyi Professor Award, 1997.

3. Winner of the Gyula Farkas Research Award, János Bolyai Mathematical Society, 1980.

4. Winner of the Kató Rényi Research Award, János Bolyai Mathematical Society, 1974.

Research Grants

1. 2014–2016: Extensions of the Theory of Automata and Languages, National Foundation of Hungary for Scientific Research, NKFI K 108448, principal investigator.
2. 2014–2016: Algebraic Structures and Fixed Point Operations in Computer Science, National Foundation of Hungary for Scientific Research, NKFI K 110883, principal investigator.
3. 2012–2014: Extensions and Applications of Fixed Point Theory for Non-Monotonic Formalisms. TÉT Greek-Hungarian Bilateral Cooperation. TÉT10-1-2011-0548 (Greek partner: Panos Rondogiannis).
4. 2010: Automata, Languages and Fixed Points. Austrian-Hungarian Action Foundation, 77öu9 (Austrian partner: Werner Kuich).
5. 2008–2012: Automata, Fixed Points, and Logic, National Foundation of Hungary for Scientific Research, OTKA K 75249, principal investigator.
6. 2005–2010: Member of the Executive Board and Steering Committee of the ESF project AUTOMATHA.
7. 2007–2009: Algebraic Theory of Automata, Hungarian Academy of Science and CNRS, principal co-investigator (French partner: Jean-Eric Pin).
8. 2006–2008: Automata and Formal Languages, Hungarian Academy of Science and Japan Society for the Promotion of Science, MTA-JSPS 101, principal co-investigator. (Japanese partner: Masami Ito).
9. 2001–2004: Iteration theories, Principal Investigator, National Foundation of Hungary for Scientific Research, OTKA T35163.
10. 2001–2002: Algebraic Structures in Automata and Language Theory, Austrian-Hungarian Action Foundation, 47öu1.
11. 2000–2001: Fixed Points in Language Theory, Principal Investigator, Austrian-Hungarian cooperative research grant, A-4/1999.
12. 1999–2002: Concurrent Processes and Formal Languages, Principal Investigator, National Foundation of Hungary for Scientific Research, OTKA, T30511.
13. 1999–2000: Fixed Points in Computer Science, Principal Investigator, Ministry of Education of Hungary, FKFP 247/1999.
14. 1997: The Shuffle Operation on Languages and Posets, Japan Society for the Promotion of Science, principal co-investigator. (Japanese partner: Masami Ito).

15. 1997–2000: Compositions of Tree Automata and Varieties of Tree Languages, Ministry of Education of Hungary, FKFP 704.
16. 1997–99, Algebraic Structures in the Theory of Automata and Formal Languages, Principal Investigator, Austrian-Hungarian Action Foundation.
17. 1997–99, Algebraic Aspects of Automata and Formal Languages, Principal Investigator, French–Hungarian Joint Project, BALATON F28/96.
18. 1997–2000: Iteration Theories, Principal Investigator, Hungarian National Foundation for Scientific Research, T22423.
19. 1996–97: Iteration Theories, Principal Investigator, Ministry of Education of Hungary, 7/1996.
20. 1996–97: Computational Models for Trees, Ministry of Education of Hungary, 665/96.
21. 1995–97: Iteration Theories, Principal Investigator, US-Hungarian Joint Fund, J.F.No. 351.
22. 1995–97: Iteration Theories, Principal Investigator, Hungarian National Foundation for Scientific Research, T16344.
23. 1993: Iteration Theories and Concurrency, Commission of the European Community, CIPA 3511CT920168.
24. 1993–1996: Structural Theory of Automata, Principal Investigator, Hungarian National Foundation for Scientific Research, T7383.
25. 1991–1994: Iteration Theories, Principal Investigator, Hungarian National Foundation for Scientific Research, 2037.
26. 1991–1993: Applications of Iteration Theories, Joint grant with Stephen L. Bloom, Hungarian Academy of Sciences and NSF (USA), INT-90 16123.
27. 1986–1990: Structural Theory of Automata, Principal Investigator, Hungarian National Foundation for Scientific Research, 1144.

Doctoral Students

László Bernátsky, 2000, Szabolcs Iván, 2008, Zoltán L. Németh, 2008, Tamás Hajgató, 2014.

Acta Cybernetica **23** (2017) 411–435.

Publication List of Zoltán Ésik

Books

[1] Zoltán Ésik. *Modern Automata Theory, Russian translation: Covremennaja Teoria Avtomatovm Kaliningrad*. Technische Universität Wien, 2007.

[2] Stephen L. Bloom and Zoltán Ésik. *Iteration Theories - The Equational Logic of Iterative Processes*. EATCS Monographs on Theoretical Computer Science. Springer, 1993.

Book chapters

[1] Equational theories for automata. In *Handbook of Automata*. EMS Publishing House, to appear.

[2] Zoltán Ésik. Fixed point theory. In *Handbook of Weighted Automata*, pages 29–65. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[3] Zoltán Ésik and Werner Kuich. Finite automata. In *Handbook of Weighted Automata*, pages 69–104. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[4] Zoltán Ésik. Automata theory. In *Encyclopedia of Computer Science and Technology*, pages 9–36. Marcel Dekker Inc., New York, 1991.

Edited volumes

[1] Zoltán Ésik and Martin Dietzfelbinger, editors. *Mathematical Foundations of Computer Science 2014, Special Issue*, Information and Computation, to appear.

[2] Zoltán Ésik and Zoltán Fülöp, editors. *Automata and Formal Languages: Special Issue AFL 2014*, volume 26(8) of *International Journal of Foundations of Computer Science*. World Scientific, 2015.

DOI: 10.14232/actacyb.23.1.2017.23

[3] Erzsébet Csuhaj-Varjú, Martin Dietzfelbinger, and Zoltán Ésik, editors. *Mathematical Foundations of Computer Science 2014 - 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part I*, volume 8634 of *Lecture Notes in Computer Science*. Springer, 2014.

[4] Erzsébet Csuhaj-Varjú, Martin Dietzfelbinger, and Zoltán Ésik, editors. *Mathematical Foundations of Computer Science 2014 - 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part II*, volume 8635 of *Lecture Notes in Computer Science*. Springer, 2014.

[5] Zoltán Ésik and Zoltán Fülöp, editors. *Proceedings 14th International Conference on Automata and Formal Languages, AFL 2014, Szeged, Hungary, May 27-29, 2014*, volume 151 of *EPTCS*, 2014.

[6] Dale Miller and Zoltán Ésik, editors. *Proceedings 8th Workshop on Fixed Points in Computer Science, FICS 2012, Tallinn, Estonia, 24th March 2012*, volume 77 of *EPTCS*, 2012.

[7] Pál Dömösi and Zoltán Ésik, editors. *Proceedings 13th International Conference on Automata and Formal Languages, AFL 2011*, volume 23 of *Int. J. Found. Comput. Sci.*, 2012.

[8] Erzsébet Csuhaj-Varjú and Zoltán Ésik, editors. *Fundamentals of Computation Theory. Special issue devoted to papers presented at FCT 07*, volume 411 of *Theor. Comput. Sci.*, 2010.

[9] Erzsébet Csuhaj-Varjú and Zoltán Ésik, editors. *Automata and Formal Languages: Special Issue AFL 2008*, volume 21 of *Int. J. Found. Comput. Sci.*, 2010.

[10] Erzsébet Csuhaj-Varjú and Zoltán Ésik, editors. *Automata and Formal Languages*, volume 19 of *Acta Cybern.*, 2009.

[11] Zoltán Ésik and Zoltán Fülöp, editors. *Automata, Formal Languages, and Related Topics - Dedicated to Ferenc Gécseg on the occasion of his 70th birthday.* Institute of Informatics, University of Szeged, Hungary, 2009.

[12] Erzsébet Csuhaj-Varjú and Zoltán Ésik, editors. *Automata and Formal Languages, 12th International Conference, AFL 2008, Balatonfüred, Hungary, May 27-30, 2008, Proceedings*, 2008.

[13] Erzsébet Csuhaj-Varjú and Zoltán Ésik, editors. *Fundamentals of Computation Theory, 16th International Symposium, FCT 2007, Budapest, Hungary, August 27-30, 2007, Proceedings*, volume 4639 of *Lecture Notes in Computer Science*. Springer, 2007.

[14] Zoltán Ésik and R. Ramanujam, editors. *Selected Papers of the Conference "Computer Science Logic 2006"*, number 4 in LMCS, 2006.

[15] Zoltán Ésik, editor. *Automata and Formal Languages: Special Issue of AFL 2005*, volume 17(4) of *Acta Cybern.*, 2006.

[16] Zoltán Ésik, editor. *Automata and Formal Languages: Special Issue of AFL 2005*, volume 366(3) of *Theor. Comput. Sci.*, 2006.

[17] Zoltán Ésik, editor. *Computer Science Logic, 20th International Workshop, CSL 2006, 15th Annual Conference of the EACSL, Szeged, Hungary, September 25-29, 2006, Proceedings*, volume 4207 of *Lecture Notes in Computer Science*. Springer, 2006.

[18] Zoltán Ésik, Carlos Martín-Vide, and Victor Mitrana, editors. *Recent Advances in Formal Languages and Applications*, volume 25 of *Studies in Computational Intelligence*. Springer, 2006.

[19] Alberto Bertoni, Zoltán Ésik, and Juhani Karhumäki, editors. *The art of rationality: In honour of Professor Christian Choffrut on the occasion of his 60th birthday*. Theor. Comput. Sci. Elsevier, 2006.

[20] Zoltán Ésik, editor. *Automata and Formal Languages (AFL '05)*. Acta Cybern. University of Szeged, 2006.

[21] Luca Aceto, Wan Fokkink, Anna Ingólfsdóttir, and Zoltán Ésik, editors. *Process Algebra, Special Issue*, volume 335(2-3) of *Theor. Comput. Sci.*, 2005.

[22] Zoltán Ésik and Zoltán Fülöp, editors. *Automata and Formal Languages: 11th International Conference, AFL 2005*. University of Szeged, 2005.

[23] Zoltán Ésik and Igor Walukiewicz, editors. *Fixed Points in Computer Science 03, Warsaw, Special issue*, volume 38(4) of *ITA*, 2004.

[24] Luca Aceto, Wan Fokkink, Anna Ingólfsdóttir, and Zoltán Ésik, editors. *Workshop on Process Algebra: Open Problems and Future Directions, PA '03, (Bologna, Italy, 21-25 July, 2003)*, volume NS 03-3 of *BRICS Notes Series*, 2003.

[25] Zoltán Ésik and Anna Ingólfsdóttir, editors. *Fixed Points in Computer Science 02, Copenhagen, Special issue*, volume 37 of *ITA*, 2003.

[26] Zoltán Ésik and Zoltán Fülöp, editors. *Developments in Language Theory, 7th International Conference, DLT 2003, Szeged, Hungary, July 7-11, 2003, Proceedings*, volume 2710 of *Lecture Notes in Computer Science*. Springer, 2003.

[27] Zoltán Ésik and Anna Ingólfsdóttir, editors. *Fixed Points in Computer Science, FICS 2002, Copenhagen, Denmark, 20-21 July 2002, Preliminary Proceedings*, volume NS-02-2 of *BRICS Notes Series*. University of Aarhus, 2002.

[28] János Csirik, Zoltán Ésik, Zoltán Fülöp, and Balázs Imreh, editors. *Special issue dedicated to the 60th birthday of Prof. Ferenc Gecseg*, volume 14 of *Acta Cybern.*, 1999.

[29] Zoltán Ésik, editor. *Proceedings Workshop on Fixed Points in Computer Science, FICS*, volume 33 of *Theoretical Infomatics and Applications*, 1999.

[30] Zoltán Ésik (editor). *Számítási bonyolultság.* Novadat, 1999.

[31] Zoltán Ésik, editor. *Fundamentals of Computation Theory, 9th International Symposium, FCT '93, Szeged, Hungary, August 23-27, 1993, Proceedings*, volume 710 of *Lecture Notes in Computer Science.* Springer, 1993.

[32] Zoltán Ésik (editor). *Számok valóson innen és túl.* Gondolat, Budapest, 1987.

Refereed journal articles

[1] Zoltán Ésik. Equational axioms associated with finite automata for fixed point operations in Cartesian categories. *Mathematical Structures in Computer Science*, 27(1):54–69, 2017.

[2] Arnaud Carayol and Zoltán Ésik. An analysis of the equational properties of the well-founded fixed point. *J. Log. Algebr. Meth. Program.*, 86(1):308–318, 2017.

[3] Zoltán Ésik. Residuated Park theories. *J. Log. Comput.*, 25(2):453–471, 2015.

[4] Zoltán Ésik and Panos Rondogiannis. A fixed point theorem for non-monotonic functions. *Theor. Comput. Sci.*, 574:18–38, 2015.

[5] Manfred Droste, Zoltán Ésik, and Werner Kuich. Conway and iteration hemirings Part 1. *IJAC*, 24(4):461–482, 2014.

[6] Manfred Droste, Zoltán Ésik, and Werner Kuich. Conway and iteration hemirings Part 2. *IJAC*, 24(4):483–514, 2014.

[7] Zoltán Ésik and Szabolcs Iván. Operational characterization of scattered MCFLs. *Int. J. Found. Comput. Sci.*, 25(8):1001–1016, 2014.

[8] Zoltán Ésik and Tamás Hajgató. On the structure of free iteration semirings. *Journal of Automata, Languages and Combinatorics*, 19(1-4):57–66, 2014.

[9] Zoltán Ésik. Axiomatizing weighted synchronization trees and weighted bisimilarity. *Theor. Comput. Sci.*, 534:2–23, 2014.

[10] Angelos Charalambidis, Zoltán Ésik, and Panos Rondogiannis. Minimum model semantics for extensional higher-order logic programming with negation. *TPLP*, 14(4-5):725–737, 2014.

[11] Zoltán Ésik and Satoshi Okawa. On context-free languages of scattered words. *Int. J. Found. Comput. Sci.*, 24(7):1029–1048, 2013.

[12] Arnaud Carayol and Zoltán Ésik. The FC-rank of a context-free language. *Inf. Process. Lett.*, 113(8):285–287, 2013.

[13] Zoltán Ésik and Werner Kuich. Free inductive K-semialgebras. *J. Log. Algebr. Program.*, 82(3-4):111–122, 2013.

[14] Zoltán Ésik. Ordinal automata and Cantor normal form. *Int. J. Found. Comput. Sci.*, 23(1):87–98, 2012.

[15] Zoltán Ésik and Szabolcs Iván. On Müller context-free grammars. *Theor. Comput. Sci.*, 416:17–32, 2012.

[16] Zoltán Ésik and Werner Kuich. Free iterative and iteration K-semialgebras. *Algebra Universalis*, 67:141–162, 2012.

[17] Stephen L. Bloom and Zoltán Ésik. Algebraic linear orderings. *Int. J. Found. Comput. Sci.*, 22(2):491–515, 2011.

[18] Zoltán Ésik and Andreas Maletti. The category of simulations for weighted tree automata. *Int. J. Found. Comput. Sci.*, 22(8):1845–1859, 2011.

[19] Zoltán Ésik. An undecidable property of context-free linear orders. *Inf. Process. Lett.*, 111(3):107–109, 2011.

[20] Zoltán Ésik and Tamás Hajgató. Dagger extension theorem. *Mathematical Structures in Computer Science*, 21(5):1035–1066, 2011.

[21] Zoltán Ésik and Szabolcs Iván. Büchi context-free languages. *Theor. Comput. Sci.*, 412(8-10):805–821, 2011.

[22] Stephen L. Bloom and Zoltán Ésik. Algebraic ordinals. *Fundam. Inform.*, 99(4):383–407, 2010.

[23] Zoltán Ésik and Pascal Weil. Algebraic characterization of logically defined tree languages. *IJAC*, 20(2):195–239, 2010.

[24] Zoltán Ésik. Axiomatizing the equational theory of regular tree languages. *J. Log. Algebr. Program.*, 79(2):189–213, 2010.

[25] Stephen L. Bloom and Zoltán Ésik. A Mezei-Wright theorem for categorical algebras. *Theor. Comput. Sci.*, 411(2):341–359, 2010.

[26] Zoltán Ésik and Szabolcs Iván. A family of temporal logics on finite trees. *Publ. Math. Debrecen*, 77(3–4):277–297, 2010.

[27] Stephen L. Bloom and Zoltán Ésik. Axiomatizing rational power series over natural numbers. *Inf. Comput.*, 207(7):793–811, 2009.

[28] Zoltán Ésik, Yuan Gao, Guangwu Liu, and Sheng Yu. Estimation of state complexity of combined operations. *Theor. Comput. Sci.*, 410(35):3272–3280, 2009.

[29] Zoltán Ésik and Szabolcs Iván. Products of tree automata with an application to temporal logic. *Fundam. Inform.*, 82(1-2):61–78, 2008.

[30] Zoltán Ésik and Szabolcs Iván. Some varieties of finite tree automata related to restricted temporal logics. *Fundam. Inform.*, 82(1-2):79–103, 2008.

[31] Stephen L. Bloom, Zoltán Ésik, and Werner Kuich. Partial Conway and iteration semirings. *Fundam. Inform.*, 86(1-2):19–40, 2008.

[32] Zoltán Ésik and Guangwu Liu. Fuzzy tree automata. *Fuzzy Sets and Systems*, 158(13):1450–1460, 2007.

[33] Zoltán Ésik and Werner Kuich. Boolean fuzzy sets. *Int. J. Found. Comput. Sci.*, 18(6):1197–1207, 2007.

[34] Zoltán Ésik and Werner Kuich. A semiring-semimodule generalization of transducers and abstract ω-families of power series. *Journal of Automata, Languages and Combinatorics*, 12(4):435–454, 2007.

[35] Zoltán Ésik and Werner Kuich. On iteration semiring-semimodule pairs. *Semigroup Forum*, 75:129–159, 2007.

[36] Zoltán Ésik. Characterizing CTL-like logics on finite trees. *Theor. Comput. Sci.*, 356(1-2):136–152, 2006.

[37] Zoltán Ésik and Hans Leiß. Algebraically complete semirings and Greibach normal form. *Ann. Pure Appl. Logic*, 133(1-3):173–203, 2005.

[38] Stephen L. Bloom and Zoltán Ésik. The equational theory of regular words. *Inf. Comput.*, 197(1-2):55–89, 2005.

[39] Zoltán Ésik and Zoltán L. Németh. Algebraic and graph-theoretic properties of infinite n-posets. *ITA*, 39(1):305–322, 2005.

[40] Zoltán Ésik and Werner Kuich. A semiring-semimodule generalization of ω-regular languages I. *Journal of Automata, Languages and Combinatorics*, 10(2/3):203–242, 2005.

[41] Zoltán Ésik and Werner Kuich. A semiring-semimodule generalization of ω-regular languages II. *Journal of Automata, Languages and Combinatorics*, 10(2/3):243–264, 2005.

[42] Zoltán Ésik and Pascal Weil. Algebraic recognizability of regular tree languages. *Theor. Comput. Sci.*, 340(1):291–321, 2005.

[43] Stephen L. Bloom and Zoltán Ésik. Axiomatizing ω and ω-op powers of words. *ITA*, 38(1):3–17, 2004.

[44] Zoltán Ésik and Zoltán L. Németh. Higher dimensional automata. *Journal of Automata, Languages and Combinatorics*, 9(1):3–29, 2004.

[45] Zoltán Ésik and Werner Kuich. Inductive star-semirings. *Theor. Comput. Sci.*, 324(1):3–33, 2004.

[46] Zoltán Ésik and Masami Ito. Temporal logic with cyclic counting and the degree of aperiodicity of finite automata. *Acta Cybern.*, 16(1):1–28, 2003.

[47] Janusz A. Brzozowski and Zoltán Ésik. Hazard algebras. *Formal Methods in System Design*, 23(3):223–256, 2003.

[48] Stephen L. Bloom and Zoltán Ésik. Deciding whether the frontier of a regular tree is scattered. *Fundam. Inform.*, 55(1):1–21, 2003.

[49] Zoltán Ésik and Kim Guldstrand Larsen. Regular languages definable by Lindström quantifiers. *ITA*, 37(3):179–241, 2003.

[50] Stephen L. Bloom and Zoltán Ésik. An extension theorem with an application to formal tree series. *Journal of Automata, Languages and Combinatorics*, 8(2):145–185, 2003.

[51] Zoltán Ésik and Werner Kuich. Formal tree series. *Journal of Automata, Languages and Combinatorics*, 8(2):219–285, 2003.

[52] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. Equational theories of tropical semirings. *Theor. Comput. Sci.*, 3(298):417–469, 2003.

[53] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. The max-plus algebra of the natural numbers has no finite equational basis. *Theor. Comput. Sci.*, 293(1):169–188, 2003.

[54] Zoltán Ésik. Free De Morgan bisemigroups and bisemilattices. *Algebra Colloquium*, 10:23–32, 2003.

[55] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. A fully equational proof of Parikh's theorem. *ITA*, 36(2):129–153, 2002.

[56] Zoltán Ésik and Werner Kuich. Rationally additive semirings. *J. UCS*, 8(2):173–183, 2002.

[57] Zoltán Ésik. Continuous additive algebras and injective simulations of synchronization trees. *J. Log. Comput.*, 12(2):271–300, 2002.

[58] Zoltán Ésik. Axiomatizing the subsumption and subword preorders on finite and infinite partial words. *Theor. Comput. Sci.*, 273(1-2):225–248, 2002.

[59] Z. Ésik and W. Kuich. Locally closed semirings. *Monatshefte für Mathematik*, 137(1):21–29, 2002.

[60] Pál Dömösi and Zoltán Ésik. A note on completeness of the ν_3-product. *Publ. Math. Debrecen*, 60:539–550, 2002.

[61] Stephen L. Bloom, Zoltán Ésik, Anna Labella, and Ernest G. Manes. Iteration 2-theories. *Applied Categorical Structures*, 9(2):173–216, 2001.

[62] Sinisa Crvenkovic, Igor Dolinka, and Zoltán Ésik. On equations for union-free regular languages. *Inf. Comput.*, 164(1):152–172, 2001.

[63] Pál Dömösi and Zoltán Ésik. Homomorphic simulation and Letichevsky's criterion. *Journal of Automata, Languages and Combinatorics*, 6(4):427–436, 2001.

[64] Zoltán Ésik and Werner Kuich. A Kleene theorem for Lindenmayerian algebraic power series. *Journal of Automata, Languages and Combinatorics*, 5(2):109–122, 2000.

[65] Sinisa Crvenkovic, Igor Dolinka, and Zoltán Ésik. The variety of Kleene algebras with conversion is not finitely based. *Theor. Comput. Sci.*, 230(1-2):235–245, 2000.

[66] Zoltán Ésik. A proof of the Krohn-Rhodes decomposition theorem. *Theor. Comput. Sci.*, 234(1-2):287–300, 2000.

[67] Zoltán Ésik. The power of the group-identities for iteration. *IJAC*, 10(3):349–374, 2000.

[68] Zoltán Ésik. Axiomatizing iteration categories. *Acta Cybern.*, 14(1):65–82, 1999.

[69] Zoltán Ésik. Group axioms for iteration. *Inf. Comput.*, 148(2):131–180, 1999.

[70] Sinisa Crvenkovic, Igor Dolinka, and Zoltán Ésik. A note on equations for commutative regular languages. *Inf. Process. Lett.*, 70(6):265–267, 1999.

[71] Zoltán Ésik. A variety theorem for trees and theories. *Publ. Math. Debrecen*, 54:711–762, 1999.

[72] Zoltán Ésik and Michael Bertol. Nonfinite axiomatizability of the equational theory of shuffle. *Acta Inf.*, 35(6):505–539, 1998.

[73] Zoltán Ésik. A Cayley theorem for ternary algebras. *IJAC*, 8(3):311–316, 1998.

[74] László Bernátsky and Zoltán Ésik. Semantics on flowchart programs and the free Conway theories. *ITA*, 32(1-3):35–78, 1998.

[75] Zoltán Ésik and Anna Labella. Equational properties of iteration in algebraically complete categories. *Theor. Comput. Sci.*, 195(1):61–89, 1998.

[76] Stephen L. Bloom and Zoltán Ésik. Shuffle binoids. *RAIRO - Theoretical Informatics and Applications - Informatique Théorique et Applications*, 32(4-6):175–198, 1998.

[77] Zoltán Ésik and Imre Simon. Modeling literal morphisms by shuffle. *Semigroup Forum*, 56(2):225–227, 1998.

[78] Stephen L. Bloom and Zoltán Ésik. Axiomatizing shuffle and concatenation in languages. *Inf. Comput.*, 139(1):62–91, 1997.

[79] Stephen L. Bloom and Zoltán Ésik. Varieties generated by languages with poset operations. *Mathematical Structures in Computer Science*, 7(6):701–713, 1997.

[80] Zoltán Ésik. Completeness of Park induction. *Theor. Comput. Sci.*, 177(1):217–283, 1997.

[81] Stephen L. Bloom and Zoltán Ésik. The equational logic of fixed points (tutorial). *Theor. Comput. Sci.*, 179(1-2):1–60, 1997.

[82] Stephen L. Bloom and Zoltán Ésik. Fixed-point operations on CCC's. part I. *Theor. Comput. Sci.*, 155(1):1–38, 1996.

[83] Stephen L. Bloom and Zoltán Ésik. Free shuffle algebras in language varieties. *Theor. Comput. Sci.*, 163(1&2):55–98, 1996.

[84] Zoltán Ésik. Definite tree automata and their cascade compositions. *Publ. Math. Debrecen*, 48:243–261, 1996.

[85] Stephen L. Bloom and Zoltán Ésik. Some equational laws of initiality in 2CCC's. *Int. J. Found. Comput. Sci.*, 6(2):95–118, 1995.

[86] Zoltán Ésik and László Bernátsky. Equational properties of Kleene algebras of relations with conversion. *Theor. Comput. Sci.*, 137(2):237–251, 1995.

[87] Stephen L. Bloom, Zoltán Ésik, and Gheorghe Stefanescu. Notes on equational theories of relations. *Algebra Universalis*, 33(1):98–126, 1995.

[88] Stephen L. Bloom, Zoltán Ésik, and Dirk Taubner. Iteration theories of synchronization trees. *Inf. Comput.*, 102(1):1–55, 1993.

[89] Stephen L. Bloom and Zoltán Ésik. Erratum: Iteration algebras. *Int. J. Found. Comput. Sci.*, 4(1):99, 1993.

[90] Stephen L. Bloom and Zoltán Ésik. Matrix and matricial iteration theories, part I. *J. Comput. Syst. Sci.*, 46(3):381–408, 1993.

[91] Stephen L. Bloom and Zoltán Ésik. Matrix and matricial iteration theories, part II. *J. Comput. Syst. Sci.*, 46(3):409–439, 1993.

[92] Stephen L. Bloom and Zoltán Ésik. Equational axioms for regular sets. *Mathematical Structures in Computer Science*, 3(1):1–24, 1993.

[93] Stephen L. Bloom and Zoltán Ésik. Iteration algebras. *Int. J. Found. Comput. Sci.*, 3(3):245–302, 1992.

[94] Zoltán Ésik. Varieties of automata and transformation semigroups. *Acta Math. Hung.*, 59:59–74, 1992.

[95] Zoltán Ésik. A note on isomorphic simulation of automata by networks of two-state automata. *Discrete Applied Mathematics*, 30(1):77–82, 1991.

[96] Stephen L. Bloom and Zoltán Ésik. Floyd-Hoare logic in iteration theories. *J. ACM*, 38(4):887–934, 1991.

[97] Zoltán Ésik. Results on homomorphic realization of automata by α_0-products. *Theor. Comput. Sci.*, 87(2):229–249, 1991.

[98] Pál Dömösi and Zoltán Ésik. Product hierarchies of automata and homomorphic simulation. *Acta Cybern.*, 9(4):371–373, 1990.

[99] Zoltán Ésik. A note on the axiomatization of iteration theories. *Acta Cybern.*, 9(4):375–384, 1990.

[100] Stephen L. Bloom, Zoltán Ésik, and Ernest G. Manes. A Cayley theorem for Boolean algebras. *The American Mathematical Monthly*, 97(9):831–833, 1990.

[101] Stephen L. Bloom and Zoltán Ésik. Equational logic of circular data type specification. *Theor. Comput. Sci.*, 63(3):303–331, 1989.

[102] Zoltán Ésik and Ferenc Gécseg. A decidability result for homomorphic representation of automata by α_0-products. *Acta Math. Hung.*, 53(1–2):205–212, 1989.

[103] Zoltán Ésik and Ferenc Gécseg. On α_1^λ-products of automata. *Acta Sci. Math. Szeged*, 53:245–253, 1989.

[104] Pál Dömösi and Zoltán Ésik. On the hierarchy of ν_i-product. *Acta Cybern.*, 8(3):253–257, 1988.

[105] Pál Dömösi and Zoltán Ésik. On homomorphic simulation of automata by α_0-products. *Acta Cybern.*, 8(4):315–323, 1988.

[106] Zoltán Ésik. Independence of the equational axioms for iteration theories. *J. Comput. Syst. Sci.*, 36(1):66–76, 1988.

[107] Stephen L. Bloom and Zoltán Ésik. Varieties of iteration theories. *SIAM J. Comput.*, 17(5):939–966, 1988.

[108] Pál Dömösi and Zoltán Ésik. Critical classes for the α_0-product. *Theor. Comput. Sci.*, 61:17–24, 1988.

[109] Zoltán Ésik. On cycles of directed graphs. *Periodica Math. Hung.*, 19:19–23, 1988.

[110] Zoltán Ésik and J. Virágh. A note on α_0^*-products of aperiodic automata. *Acta Cybern.*, 8(1):41–43, 1987.

[111] Zoltán Ésik. Loop products and loop-free products. *Acta Cybern.*, 8(1):45–48, 1987.

[112] Zoltán Ésik. On isomorphic realization of automata with α_0-products. *Acta Cybern.*, 8(2):119–127, 1987.

[113] Zoltán Ésik and Ferenc Gécseg. On a representation of tree automata. *Theor. Comput. Sci.*, 53:243–255, 1987.

[114] Pál Dömösi and Zoltán Ésik. On homomorphic simulation of automata by ν_1-products. *Papers on Automata Theory*, IX:91–112, 1987.

[115] Zoltán Ésik. Varieties and general products of top-down algebras. *Acta Cybern.*, 7(3):293–298, 1986.

[116] Zoltán Ésik and J. Virágh. On products of automata with identity. *Acta Cybern.*, 7(3):299–311, 1986.

[117] Zoltán Ésik and Pál Dömösi. Complete classes of automata for the α_0-product. *Theor. Comput. Sci.*, 47(3):1–14, 1986.

[118] Zoltán Ésik and Ferenc Gécseg. On α_0-products and α_2-products. *Theor. Comput. Sci.*, 48(3):1–8, 1986.

[119] Zoltán Ésik. Complete classes of automata for the α_i-product. *Found. Control Engrg.*, 11:95–107, 1986.

[120] Pál Dömösi and Zoltán Ésik. On homomorphic realization of automata with α_0-products. *Papers on Automata Theory*, 8:63–97, 1986.

[121] Zoltán Ésik and Ferenc Gécseg. Type independent varieties and metric equivalence of tree automata. *Fundam. Inform.*, 9:205–216, 1986.

[122] Zoltán Ésik. On the weak equivalence of Elgot's flow-chart schemata. *Acta Cybern.*, 7(2):147–154, 1985.

[123] Stephen L. Bloom and Zoltán Ésik. Axiomatizing schemes and their behaviors. *J. Comput. Syst. Sci.*, 31(3):375–393, 1985.

[124] Zoltán Ésik. Homomorphically complete classes of automata with respect to the α_2-product. *Acta Sci. Math. Szeged*, 48:135–141, 1985.

[125] Zoltán Ésik and Gyula Horváth. Pseudo varieties and α_0-products. *Papers on Automata Theory*, 6:47–76, 1984.

[126] Zoltán Ésik. A note on kernel languages of programs (in hungarian). *Alkalmazott Matematikai Lapok*, 10:61–63, 1984.

[127] Zoltán Ésik and Ferenc Gécseg. General products and equational classes of automata. *Acta Cybern.*, 6(3):281–284, 1983.

[128] Zoltán Ésik. On identities preserved by general products of algebras,. *Acta Cybern.*, 6:285–289, 1983.

[129] Zoltán Ésik. Decidability results concerning tree transducers II. *Acta Cybern.*, 6(3):303–314, 1983.

[130] Zoltán Ésik. Algebras of iteration theories. *J. Comput. Syst. Sci.*, 27(2):291–303, 1983.

[131] Zoltán Ésik and Gyula Horváth. The α_2-product is homomorphically general. *Papers on Automata Theory*, 5:49–62, 1983.

[132] Zoltán Ésik. On homomorphic realization of monotone automata. *Papers on Automata Theory*, 5:63–76, 1983.

[133] Zoltán Ésik. On generalized iterative algebraic theories. *Computational Linguistics and Computer Languages*, 15:95–110, 1982.

[134] Zoltán Ésik and Balázs Imreh. Remarks on finite commutative automata. *Acta Cybern.*, 5(2):143–146, 1981.

[135] Zoltán Ésik and Balázs Imreh. Subdirectly irreducible commutative automata. *Acta Cybern.*, 5(3):251–260, 1981.

[136] Zoltán Ésik. Decidability results concerning tree transducers I. *Acta Cybern.*, 5(1):1–20, 1980.

[137] Zoltán Ésik. Identities in iterative and rational algebraic theories. *Computational Linguistics and Computer Languages*, 14:183–207, 1980.

[138] Zoltán Ésik. On two problems of A. Salomaa. *Acta Cybern.*, 2(4):299–306, 1976.

Refereed conference articles

[1] Arnaud Carayol and Zoltán Ésik. An analysis of the equational properties of the well-founded fixed point. In Chitta Baral, James P. Delgrande, and Frank Wolter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016.*, pages 533–536. AAAI Press, 2016.

[2] Zoltán Ésik. Ternary equational languages. In Yo-Sub Han and Kai Salomaa, editors, *Implementation and Application of Automata - 21st International Conference, CIAA 2016, Seoul, South Korea, July 19-22, 2016, Proceedings*, volume 9705 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2016.

[3] Zoltán Ésik and Szabolcs Iván. MSO-definable properties of Muller context-free languages are decidable. In Cezar Câmpeanu, Florin Manea, and Jeffrey Shallit, editors, *Descriptional Complexity of Formal Systems - 18th IFIP WG 1.2 International Conference, DCFS 2016, Bucharest, Romania, July 5-8, 2016. Proceedings*, volume 9777 of *Lecture Notes in Computer Science*, pages 87–97. Springer, 2016.

[4] Zoltán Ésik, Uli Fahrenberg, and Axel Legay. *-continuous Kleene ω-algebras. In Igor Potapov, editor, *Developments in Language Theory - 19th International Conference, DLT 2015, Liverpool, UK, July 27-30, 2015, Proceedings.*, volume 9168 of *Lecture Notes in Computer Science*, pages 240–251. Springer, 2015.

[5] Zoltán Ésik. Equational properties of fixed point operations in Cartesian categories: An overview. In Giuseppe F. Italiano, Giovanni Pighizzini, and Donald Sannella, editors, *Mathematical Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part I*, volume 9234 of *Lecture Notes in Computer Science*, pages 18–37. Springer, 2015.

[6] Zoltán Ésik. Equational properties of stratified least fixed points (extended abstract). In Valeria de Paiva, Ruy J. G. B. de Queiroz, Lawrence S. Moss, Daniel Leivant, and Anjolina Grisi de Oliveira, editors, *Logic, Language, Information, and Computation - 22nd International Workshop, WoLLIC 2015, Bloomington, IN, USA, July 20-23, 2015, Proceedings*, volume 9160 of *Lecture Notes in Computer Science*, pages 174–188. Springer, 2015.

[7] Zoltán Ésik, Uli Fahrenberg, and Axel Legay. *-continuous Kleene ω-algebras for energy problems. In Ralph Matthes and Matteo Mio, editors, *Proceedings Tenth International Workshop on Fixed Points in Computer Science, FICS 2015, Berlin, Germany, September 11-12, 2015.*, volume 191 of *EPTCS*, pages 48–59, 2015.

[8] Zoltán Ésik. A representation theorem for stratified complete lattices. In *11th Tbilisi Symp. Language, Logic and Computation, 21–26 Sept. 2015*, pages 56–58. Georgian Academy of Science, 2015.

[9] Zoltán Ésik and Panos Rondogiannis. Theorems on pre-fixed points of non-monotonic functions with applications in logic programming and formal grammars. In Ulrich Kohlenbach, Pablo Barceló, and Ruy J. G. B. de Queiroz, editors, *Logic, Language, Information, and Computation - 21st International*

Workshop, WoLLIC 2014, Valparaíso, Chile, September 1-4, 2014. Proceedings, volume 8652 of *Lecture Notes in Computer Science*, pages 166–180. Springer, 2014.

[10] Zoltán Ésik, Uli Fahrenberg, Axel Legay, and Karin Quaas. Kleene algebras and semimodules for energy problems. In Dang Van Hung and Mizuhito Ogawa, editors, *Automated Technology for Verification and Analysis - 11th International Symposium, ATVA 2013, Hanoi, Vietnam, October 15-18, 2013. Proceedings*, volume 8172 of *Lecture Notes in Computer Science*, pages 102–117. Springer, 2013.

[11] Zoltán Ésik and Szabolcs Iván. Operational characterization of scattered MCFLs. In Marie-Pierre Béal and Olivier Carton, editors, *Developments in Language Theory - 17th International Conference, DLT 2013, Marne-la-Vallée, France, June 18-21, 2013. Proceedings*, volume 7907 of *Lecture Notes in Computer Science*, pages 215–226. Springer, 2013.

[12] Zoltán Ésik and Panos Rondogiannis. A fixed-point theorem for non-monotonic functions. In *Panhellenic Logic - 9th Symposium, Athens, Greece, July 15-18, 2013. Proceedings*, pages 43–48, 2013.

[13] Zoltán Ésik. On a connection between concurrency and formal languages. In *Mathematical Foundation of Programming Semantics, MFPS29, New Orleans, 2013*, volume 298 of *ENTCS*, page 143–164, 2013.

[14] Zoltán Ésik and Satoshi Okawa. On context-free languages of scattered words. In Hsu-Chun Yen and Oscar H. Ibarra, editors, *Developments in Language Theory - 16th International Conference, DLT 2012, Taipei, Taiwan, August 14-17, 2012. Proceedings*, volume 7410 of *Lecture Notes in Computer Science*, pages 142–153. Springer, 2012.

[15] Luca Aceto, Arnaud Carayol, Zoltán Ésik, and Anna Ingólfsdóttir. Algebraic synchronization trees and processes. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part II*, volume 7392 of *Lecture Notes in Computer Science*, pages 30–41. Springer, 2012.

[16] Arnaud Carayol and Zoltán Ésik. A context-free linear ordering with an undecidable first-order theory. In Jos C. M. Baeten, Thomas Ball, and Frank S. de Boer, editors, *Theoretical Computer Science - 7th IFIP TC 1/WG 2.2 International Conference, TCS 2012, Amsterdam, The Netherlands, September 26-28, 2012. Proceedings*, volume 7604 of *Lecture Notes in Computer Science*, pages 104–118. Springer, 2012.

[17] Zoltán Ésik and Szabolcs Iván. Hausdorff rank of scattered context-free linear orders. In David Fernández-Baca, editor, *LATIN 2012: Theoretical Informatics - 10th Latin American Symposium, Arequipa, Peru, April 16-20, 2012.*

Proceedings, volume 7256 of *Lecture Notes in Computer Science*, pages 291–302. Springer, 2012.

[18] Zoltán Ésik. Scattered context-free linear orderings. In Giancarlo Mauri and Alberto Leporati, editors, *Developments in Language Theory - 15th International Conference, DLT 2011, Milan, Italy, July 19-22, 2011. Proceedings*, volume 6795 of *Lecture Notes in Computer Science*, pages 216–227. Springer, 2011.

[19] Zoltán Ésik. Multi-linear iterative K-Σ-semialgebras. In *Proceedings 27th Int. Conference on Mathematical Foundations of Programming Semantics, Carnegie Mellon University, May 2011*, volume 276 of *Electr. Notes Theor. Comput. Sci.*, pages 159–170, 2011.

[20] Zoltán Ésik. Residuated Park theories. In *Proceedings of the Fifth International Conference on Topology, Algebra and Categories in Logic, Marseille, France*, pages 37–40, 2011.

[21] Zoltán Ésik and Andreas Maletti. Simulations of weighted tree automata. In *Proceedings of the 15th International Conference on Implementation and Application of Automata. Winnipeg, Canada*, volume 6482 of *Lecture Notes in Computer Science*, pages 321–330. Springer, 2011.

[22] Zoltán Ésik and Werner Kuich. Axiomatizing rational series. In *Proceedings of the 8th Panhellenic Logic Symposium. Ioannina, Greece*, pages 30–34, 2011.

[23] Zoltán Ésik and Szabolcs Iván. On Müller context-free grammars. In Yuan Gao, Hanlin Lu, Shinnosuke Seki, and Sheng Yu, editors, *Developments in Language Theory, 14th International Conference, DLT 2010, London, ON, Canada, August 17-20, 2010. Proceedings*, volume 6224 of *Lecture Notes in Computer Science*, pages 173–184. Springer, 2010.

[24] Zoltán Ésik and Andreas Maletti. Simulation vs. equivalence. In Hamid R. Arabnia, George A. Gravvanis, and Ashu M. G. Solo, editors, *Proceedings of the 2010 International Conference on Foundations of Computer Science, FCS 2010, July 12-15, 2010, Las Vegas, Nevada, USA*, pages 119–124. CSREA Press, 2010.

[25] Zoltán Ésik and Andreas Maletti. Simulations of weighted tree automata. In Michael Domaratzki and Kai Salomaa, editors, *Implementation and Application of Automata - 15th International Conference, CIAA 2010, Winnipeg, MB, Canada, August 12-15, 2010. Revised Selected Papers*, volume 6482 of *Lecture Notes in Computer Science*, pages 321–330. Springer, 2010.

[26] Zoltán Ésik and Szabolcs Iván. Extended temporal logics on finite trees. In *Masami Ito, Yuji Kobayashi, Kunitaka Shoji (eds.) Automata, Formal Languages, and Algebraic Systems. Kyoto, Japan*, pages 47–62. World Scientific, 2010.

[27] Zoltán Ésik. Representing small ordinals by finite automata. In Ian McQuillan and Giovanni Pighizzini, editors, *Proceedings Twelfth Annual Workshop on Descriptional Complexity of Formal Systems, DCFS 2010, Saskatoon, Canada, 8-10th August 2010.*, volume 31 of *EPTCS*, pages 78–87, 2010.

[28] Zoltán Ésik, Werner Kuich, and Masami Ito. Linear languages of finite and infinite words. In *Masami Ito, Yuji Kobayashi, Kunitaka Shoji (eds.) Automata, Formal Languages, and Algebraic Systems. Kyoto, Japan*, pages 33–46. World Scientific, 2010.

[29] Stephen L. Bloom, Zoltán Ésik, and Werner Kuich. Cycle-free finite automata in partial iterative semirings. In Symeon Bozapalidis and George Rahonis, editors, *Algebraic Informatics, Third International Conference, CAI 2009, Thessaloniki, Greece, May 19-22, 2009, Proceedings*, volume 5725 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2009.

[30] Zoltán Ésik and Tamás Hajgató. Iteration Grove theories with applications. In Symeon Bozapalidis and George Rahonis, editors, *Algebraic Informatics, Third International Conference, CAI 2009, Thessaloniki, Greece, May 19-22, 2009, Proceedings*, volume 5725 of *Lecture Notes in Computer Science*, pages 227–249. Springer, 2009.

[31] Zoltán Ésik and Szabolcs Iván. Context-free languages of countable words. In Martin Leucker and Carroll Morgan, editors, *Theoretical Aspects of Computing - ICTAC 2009, 6th International Colloquium, Kuala Lumpur, Malaysia, August 16-20, 2009. Proceedings*, volume 5684 of *Lecture Notes in Computer Science*, pages 185–199. Springer, 2009.

[32] Stephen L. Bloom and Zoltán Ésik. Scattered algebraic linear orderings. In Ralph Matthes and Tarmo Uustalu, editors, *6th Workshop on Fixed Points in Computer Science: FICS 2009*, pages 25–30, 2009.

[33] Zoltán Ésik, Yuan Gao, Guangwu Liu, and Sheng Yu. Estimation of state complexity of combined operations. In Cezar Câmpeanu and Giovanni Pighizzini, editors, *10th International Workshop on Descriptional Complexity of Formal Systems, DCFS 2008, Charlottetown, Prince Edward Island, Canada, July 16-18, 2008.*, pages 168–181. University of Prince Edward Island, 2008.

[34] Zoltán Ésik. Iteration semirings. In Masami Ito and Masafumi Toyama, editors, *Developments in Language Theory, 12th International Conference, DLT 2008, Kyoto, Japan, September 16-19, 2008. Proceedings*, volume 5257 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2008.

[35] Zoltán Ésik and Szabolcs Iván. Games for temporal logics on trees. In Oscar H. Ibarra and Bala Ravikumar, editors, *Implementation and Applications of Automata, 13th International Conference, CIAA 2008, San Francisco, California, USA, July 21-24, 2008. Proceedings*, volume 5148 of *Lecture Notes in Computer Science*, pages 191–200. Springer, 2008.

[36] Zoltán Ésik and Szabolcs Iván. Aperiodicity in tree automata. In Symeon Bozapalidis and George Rahonis, editors, *Algebraic Informatics, Second International Conference, CAI 2007, Thessaloniki, Greece, May 21-25, 2007, Revised Selected and Invited Papers*, volume 4728 of *Lecture Notes in Computer Science*, pages 189–207. Springer, 2007.

[37] Stephen L. Bloom and Zoltán Ésik. Regular and algebraic words and ordinals. In Till Mossakowski, Ugo Montanari, and Magne Haveraaen, editors, *Algebra and Coalgebra in Computer Science, Second International Conference, CALCO 2007, Bergen, Norway, August 20-24, 2007, Proceedings*, volume 4624 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2007.

[38] Zoltán Ésik and Gabriela Martín. An algebraic characterization of Wolper's logic. In Hamid R. Arabnia and Pei Li Zhou, editors, *Proceedings of the 2007 International Conference on Foundations of Computer Science, FCS 2007, June 25-28, 2007, Las Vegas, Nevada, USA*, pages 139–143. CSREA Press, 2007.

[39] Zoltán Ésik and Werner Kuich. Fixed points in semiring theory. In Michael Kunc and Alexander Okhotin, editors, *Theory and Applications of Language Equations: Proceedings of the 1st International Workshop*, pages 5–13, 2007.

[40] Stephen L. Bloom and Zoltán Ésik. Completing categorical algebras. In Gonzalo Navarro, Leopoldo E. Bertossi, and Yoshiharu Kohayakawa, editors, *Fourth IFIP International Conference on Theoretical Computer Science (TCS 2006), IFIP 19th World Computer Congress, TC-1 Foundations of Computer Science, August 23-24, 2006, Santiago, Chile*, volume 209 of *IFIP*, pages 231–249. Springer, 2006.

[41] Zoltán Ésik. Cascade products and temporal logics on finite trees. In *Proceedings of the Workshop on Algebraic Process Calculi: The First Twenty Five Years and Beyond*, volume 162 of *Electr. Notes Theor. Comput. Sci.*, pages 163–166, 2006.

[42] Zoltán Ésik. An algebraic characterization of the expressive power of temporal logics on finite trees, Part 1. In *1st International Conference on Algebraic Informatics*, pages 53–78, 2005.

[43] Zoltán Ésik. An algebraic characterization of the expressive power of temporal logics on finite trees, Part 2. In *1st International Conference on Algebraic Informatics*, pages 79–100, 2005.

[44] Zoltán Ésik. An algebraic characterization of the expressive power of temporal logics on finite trees, Part 3. In *1st International Conference on Algebraic Informatics*, pages 101–110, 2005.

[45] Zoltán Ésik and Gabriela Martin. A note on Wolper's logic. In *Workshop on Semigroups and Automata*, volume 3580 of *Lecture Notes in Computer Science*, pages 61–68, 2005.

[46] Zoltán Ésik and Werner Kuich. An algebraic generalization of ω-regular languages. In Jirí Fiala, Václav Koubek, and Jan Kratochvíl, editors, *Mathematical Foundations of Computer Science 2004, 29th International Symposium, MFCS 2004, Prague, Czech Republic, August 22-27, 2004, Proceedings*, volume 3153 of *Lecture Notes in Computer Science*, pages 648–659. Springer, 2004.

[47] Stephen L. Bloom and Zoltán Ésik. Axioms for regular words: Extended abstract. In Paritosh K. Pandya and Jaikumar Radhakrishnan, editors, *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference, Mumbai, India, December 15-17, 2003, Proceedings*, volume 2914 of *Lecture Notes in Computer Science*, pages 50–61. Springer, 2003.

[48] Zoltán Ésik and Pascal Weil. On logically defined recognizable tree languages. In Paritosh K. Pandya and Jaikumar Radhakrishnan, editors, *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference, Mumbai, India, December 15-17, 2003, Proceedings*, volume 2914 of *Lecture Notes in Computer Science*, pages 195–207. Springer, 2003.

[49] Zoltán Ésik. Extended temporal logic on finite words and wreath products of monoids with distinguished generators. In Masami Ito and Masafumi Toyama, editors, *Developments in Language Theory, 6th International Conference, DLT 2002*, volume 2450 of *Lecture Notes in Computer Science*, pages 43–58, 2003.

[50] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. Equational axioms for probabilistic bisimilarity. In Hélène Kirchner and Christophe Ringeissen, editors, *Algebraic Methodology and Software Technology, 9th International Conference, AMAST 2002, Saint-Gilles-les-Bains, Reunion Island, France, September 9-13, 2002, Proceedings*, volume 2422 of *Lecture Notes in Computer Science*, pages 239–253. Springer, 2002.

[51] Zoltán Ésik and Hans Leiß. Greibach normal form in algebraically complete semirings. In Julian C. Bradfield, editor, *Computer Science Logic, 16th International Workshop, CSL 2002, 11th Annual Conference of the EACSL, Edinburgh, Scotland, UK, September 22-25, 2002, Proceedings*, volume 2471 of *Lecture Notes in Computer Science*, pages 135–150. Springer, 2002.

[52] Zoltán Ésik. Extended temporal logic on finite words and wreath product of monoids with distinguished generators. In Masami Ito and Masafumi Toyama, editors, *Developments in Language Theory, 6th International Conference, DLT 2002, Kyoto, Japan, September 18-21, 2002, Revised Papers*, volume 2450 of *Lecture Notes in Computer Science*, pages 43–58. Springer, 2002.

[53] Z. Ésik. The equational theory of fixed points with applications to generalized language theory. In Werner Kuich, Grzegorz Rozenberg, and Arto Salomaa, editors, *Developments in Language Theory: 5th International Conference, DLT*

2001 Wien, Austria, July 16–21, 2001 Revised Papers, pages 21–36, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[54] Stephen L. Bloom and Zoltán Ésik. Unique, guarded fixed points in an additive setting (extended abstract). In *Proceedings Category Theory and Computer Science, CTCS 2002, Ottawa*, volume 69 of *Electr. Notes Theor. Comput. Sci.*, pages 47–61, 2002.

[55] Zoltán Ésik and Werner Kuich. Conway-halbringe als grundlage für eine mathematische automatentheorie. In Ju.I. Schevtschenko S.I. Aleschnikov, S.Ju. Piljugin, editor, *Doklady meschdunarodnogo matematitscheskogo seminara k 140-letiju sodnja roschdenija Davida Gilberta iz Kenigsberga i 25-letiju matematitscheskogo fakulteta (Vorträge des internationalen mathematischen Seminars zum 140. Geburtstag David Hilberts aus Königsberg und zum 25 - jährigen Jubiläum der mathematischen Fakultät*, pages 240–246. Universität in Königsberg, 2002.

[56] Zoltán Ésik. The equational theory of fixed points with applications to generalized language theory. In Werner Kuich, Grzegorz Rozenberg, and Arto Salomaa, editors, *Developments in Language Theory, 5th International Conference, DLT 2001, Vienna, Austria, July 16-21, 2001, Revised Papers*, volume 2295 of *Lecture Notes in Computer Science*, pages 21–36. Springer, 2001.

[57] Zoltán Ésik and Zoltán L. Németh. Automata on series-parallel biposets. In Werner Kuich, Grzegorz Rozenberg, and Arto Salomaa, editors, *Developments in Language Theory, 5th International Conference, DLT 2001, Vienna, Austria, July 16-21, 2001, Revised Papers*, volume 2295 of *Lecture Notes in Computer Science*, pages 217–227. Springer, 2001.

[58] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. Axiomatizing tropical semirings. In Furio Honsell and Marino Miculan, editors, *Foundations of Software Science and Computation Structures, 4th International Conference, FOSSACS 2001 Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2001 Genova, Italy, April 2-6, 2001, Proceedings*, volume 2030 of *Lecture Notes in Computer Science*, pages 42–56. Springer, 2001.

[59] Janusz A. Brzozowski, Zoltán Ésik, and Y. Iland. Algebras for hazard detection. In *31st IEEE International Symposium on Multiple-Valued Logic, ISMVL 2001, Warsaw, Poland, May 22-24, 2001, Proceedings*, pages 3–14. IEEE Computer Society, 2001.

[60] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdottir. Nonfinitely based tropical semirings. In Gaubert, S. : Loiseau, J. J. (eds.), editor, *Proceedings of the Workshop on Max-Plus Algebra and Their Applications to Discrete-Event Systems : Theoretical Computer Science, and Optimization, August 27-29 2001, Prague, Czech Republic*, pages 29–34. Elsevier Science, 2001.

[61] Zoltán Ésik. Axiomatizing the least fixed point operation and binary supremum. In Peter Clote and Helmut Schwichtenberg, editors, *Computer Science Logic, 14th Annual Conference of the EACSL, Fischbachau, Germany, August 21-26, 2000, Proceedings*, volume 1862 of *Lecture Notes in Computer Science*, pages 302–316. Springer, 2000.

[62] Janusz A. Brzozowski and Zoltán Ésik. Hazard algebras (extended abstract). In Arto Salomaa, Derick Wood, and Sheng Yu, editors, *A Half-Century of Automata Theory: Celebration and Inspiration*, pages 1–19. World Scientific, 2000.

[63] Stephen L. Bloom and Zoltán Ésik. Iteration algebras are not finitely axiomatizable (extended abstract). In Gaston H. Gonnet, Daniel Panario, and Alfredo Viola, editors, *LATIN 2000: Theoretical Informatics, 4th Latin American Symposium, Punta del Este, Uruguay, April 10-14, 2000, Proceedings*, volume 1776 of *Lecture Notes in Computer Science*, pages 367–376. Springer, 2000.

[64] Zoltán Ésik. Iteration theories of Boolean functions. In Mogens Nielsen and Branislav Rovan, editors, *Mathematical Foundations of Computer Science 2000, 25th International Symposium, MFCS 2000, Bratislava, Slovakia, August 28 - September 1, 2000, Proceedings*, volume 1893 of *Lecture Notes in Computer Science*, pages 343–352. Springer, 2000.

[65] Luca Aceto, Zoltán Ésik, and Anna Ingólfsdóttir. On the two-variable fragment of the equational theory of the max-sum algebra of the natural numbers. In Horst Reichel and Sophie Tison, editors, *STACS 2000, 17th Annual Symposium on Theoretical Aspects of Computer Science, Lille, France, February 2000, Proceedings*, volume 1770 of *Lecture Notes in Computer Science*, pages 267–278. Springer, 2000.

[66] Zoltán Ésik. Free algebras for generalized automata and language theory. In *Proceedings of Algebraic Systems : Formal Languages and Computation, RIMS Kokyuroku 1166, Kyoto University, 2000*, pages 52–58, 2000.

[67] Stephen L. Bloom and Zoltán Ésik. There is no finite axiomatization of iteration theories. In *Proceedings LATIN 2000, Punta del Este, Uruguay*, volume 1776 of *Lecture Notes in Computer Science*, page 367–376. Springer-Verlag, 2000.

[68] Zoltán Ésik and Satoshi Okawa. Series and parallel operations on pomsets. In C. Pandu Rangan, Venkatesh Raman, and Ramaswamy Ramanujam, editors, *Foundations of Software Technology and Theoretical Computer Science, 19th Conference, Chennai, India, December 13-15, 1999, Proceedings*, volume 1738 of *Lecture Notes in Computer Science*, pages 316–328. Springer, 1999.

[69] Zoltán Ésik, Masami Ito, and Masashi Katsura. The equational theory of reversal. In Gaubert, S. : Loiseau, J. J. (eds.), editor, *Proceedings of the*

International Workshop on Formal Languages and Computer Systems, Kyoto, March 18 – 21 1997 and the First International Conference on Semigroups and Algebraic Engineering, Aizu, 24 – 28 March 1997, pages 502–521. World Scientific, 1999.

[70] Zoltán Ésik. Axiomatizing the equational theory of regular tree languages (extended abstract). In Michel Morvan, Christoph Meinel, and Daniel Krob, editors, *STACS 98, 15th Annual Symposium on Theoretical Aspects of Computer Science, Paris, France, February 25-27, 1998, Proceedings*, volume 1373 of *Lecture Notes in Computer Science*, pages 455–465. Springer, 1998.

[71] Stephen L. Bloom, Anna Labella, Zoltán Ésik, and Ernest G. Manes. Iteration 2-theories: Extended abstract. In Michael Johnson, editor, *Algebraic Methodology and Software Technology, 6th International Conference, AMAST '97, Sydney, Australia, December 13-17, 1997, Proceedings*, volume 1349 of *Lecture Notes in Computer Science*, pages 30–44. Springer, 1997.

[72] Stephen L. Bloom and Zoltán Ésik. Research project, axiomatizing shuffle. In *Trabajos seleccionados WAIT '97, Buenos Aires*, pages 47–54. Sociedad Argentina de Informatica e Investogacion Operativa, 1997.

[73] Zoltán Ésik and Anna Labella. Equational properties of iteration in algebraically complete categories. In Wojciech Penczek and Andrzej Szalas, editors, *Mathematical Foundations of Computer Science 1996, 21st International Symposium, MFCS'96, Cracow, Poland, September 2-6, 1996, Proceedings*, volume 1113 of *Lecture Notes in Computer Science*, pages 336–347. Springer, 1996.

[74] Zoltán Ésik and Michael Bertol. Nonfinite axiomatizability of the equational theory of shuffle. In Zoltán Fülöp and Ferenc Gécseg, editors, *Automata, Languages and Programming, 22nd International Colloquium, ICALP95, Szeged, Hungary, July 10-14, 1995, Proceedings*, volume 944 of *Lecture Notes in Computer Science*, pages 27–38. Springer, 1995.

[75] Stephen L. Bloom and Zoltán Ésik. Free shuffle algebras in language varieties (extended abstract). In Ricardo A. Baeza-Yates, Eric Goles Ch., and Patricio V. Poblete, editors, *LATIN '95: Theoretical Informatics, Second Latin American Symposium, Valparaíso, Chile, April 3-7, 1995, Proceedings*, volume 911 of *Lecture Notes in Computer Science*, pages 99–111. Springer, 1995.

[76] Stephen L. Bloom and Zoltán Ésik. Nonfinite axiomatizability of shuffle inequalities. In Peter D. Mosses, Mogens Nielsen, and Michael I. Schwartzbach, editors, *TAPSOFT'95: Theory and Practice of Software Development, 6th International Joint Conference CAAP/FASE, Aarhus, Denmark, May 22-26, 1995, Proceedings*, volume 915 of *Lecture Notes in Computer Science*, pages 318–333. Springer, 1995.

[77] Zoltán Ésik and László Bernátsky. Scott induction and equational proofs. In *Mathematical Foundations of Programming Semantics '95, New Orleans*, volume 1 of *Electr. Notes Theor. Comput. Sci.*, pages 154–181, 1995.

[78] Stephen L. Bloom and Zoltán Ésik. Solving polynomial fixed point equations. In Igor Prívara, Branislav Rovan, and Peter Ruzicka, editors, *Mathematical Foundations of Computer Science 1994, 19th International Symposium, MFCS'94, Kosice, Slovakia, August 22 - 26, 1994, Proceedings*, volume 841 of *Lecture Notes in Computer Science*, pages 52–67. Springer, 1994.

[79] László Bernátsky, Stephen L. Bloom, Zoltán Ésik, and Gheorghe Stefanescu. Equational theories of relations and regular sets. In Masami Ito and Helmut Jürgensen, editor, *Proceedings of the International Conference on Words, Languages and Combinatorics, II.*, pages 40–48. World Scientific, 1994.

[80] Stephen L. Bloom and Zoltán Ésik. Some quasi-varieties of iteration theories. In Stephen D. Brookes, Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt, editors, *Mathematical Foundations of Programming Semantics, 9th International Conference, New Orleans, LA, USA, April 7-10, 1993, Proceedings*, volume 802 of *Lecture Notes in Computer Science*, pages 378–409. Springer, 1993.

[81] Stephen L. Bloom and Zoltán Ésik. Program correctness and matricial iteration theories. In Stephen D. Brookes, Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt, editors, *Mathematical Foundations of Programming Semantics, 7th International Conference, Pittsburgh, PA, USA, March 25-28, 1991, Proceedings*, volume 598 of *Lecture Notes in Computer Science*, pages 457–476. Springer, 1991.

[82] Stephen L. Bloom and Zoltán Ésik. Iteration algebras (extended abstract). In Samson Abramsky and T. S. E. Maibaum, editors, *TAPSOFT'91: Proceedings of the International Joint Conference on Theory and Practice of Software Development, Brighton, UK, April 8-12, 1991, Volume 1: Colloquium on Trees in Algebra and Programming (CAAP'91)*, volume 493 of *Lecture Notes in Computer Science*, pages 264–274. Springer, 1991.

[83] Stephen L. Bloom, Zoltán Ésik, and Dirk Taubner. Iteration theories of synchronization trees. In *Semantics for Concurrency: Proceedings of the International BCS-FACS Workshop, Sponsored by Logic for IT (S.E.R.C.), 23–25 July 1990, University of Leicester, UK*, pages 96–115, London, 1990. Springer London.

[84] Pál Dömösi, Zoltán Ésik, and Balázs Imreh. On product hierarchies of automata. In János Csirik, János Demetrovics, and Ferenc Gécseg, editors, *Fundamentals of Computation Theory, International Conference FCT'89, Szeged, Hungary, August 21-25, 1989, Proceedings*, volume 380 of *Lecture Notes in Computer Science*, pages 137–144. Springer, 1989.

[85] Zoltán Ésik. An extension of the Krohn-Rhodes decomposition of automata. In Jürgen Dassow and Jozef Kelemen, editors, *Machines, Languages, and Complexity, 5th International Meeting of Young Computer Scientists, Smolenice, Czechoslovakia, November 14-18, 1988, Proceedings*, volume 381 of *Lecture Notes in Computer Science*, pages 66–71. Springer, 1988.

[86] Pál Dömösi and Zoltán Ésik. On homomorphic realization and homomorphic simulation of automata by α_0-products. In *Proceedings Conference on Automata, Languages and Programming Systems, Salgótarján*, pages 89–98, 1988.

[87] Zoltán Ésik, Pál Dömösi, Ferenc Gécseg, and J. Virágh. Homomorphic realizations of automata with compositions. In Jozef Gruska, Branislav Rovan, and Juraj Wiedermann, editors, *Mathematical Foundations of Computer Science 1986, Bratislava, Czechoslovakia, August 25-29, 1996, Proceedings*, volume 233 of *Lecture Notes in Computer Science*, pages 299–307. Springer, 1986.

[88] Zoltán Ésik. Completeness results in automata theory. In *Proceedings Conference on Automata, Languages and Programming Systems, Salgótarján*, pages 110–122. Karl Marx Univ. of Economics, 1986.

[89] Zoltán Ésik and János Virágh. On λ-products of automata. In *Proceedings 4th Hungarian Computer Sci. Conf*, pages 79–89. Akadémiai Kiadó, 1986.

[90] Zoltán Ésik. On Elgot's flowchart schemes. In *System Theoretical Aspects in Computer Science, Salgótarján*, pages 99–102, 1982.

[91] Zoltán Ésik. An axiomatization of regular forests in the language of algebraic theories with iteration. In Ferenc Gécseg, editor, *Fundamentals of Computation Theory, FCT'81, Proceedings of the 1981 International FCT-Conference, Szeged, Hungary, August 24-28, 1981*, volume 117 of *Lecture Notes in Computer Science*, pages 130–136. Springer, 1981.

[92] Zoltán Ésik. On functional tree transducers. In *Fundamentals of Computation Theory*, pages 121–127, 1979.

[93] Zoltán Ésik. On decidability of injectivity of tree transformations. In *Les arbres en algebre et en programmation, Lille*, pages 107–133, 1978.

Refereed articles in other edited volumes

[1] Zoltán Ésik and Werner Kuich. On power series over a graded monoid. In Cristian S. Calude, Rusins Freivalds, and Kazuo Iwama, editors, *Computing with New Resources - Essays Dedicated to Jozef Gruska on the Occasion of His 80th Birthday*, volume 8808 of *Lecture Notes in Computer Science*, pages 49–55. Springer, 2014.

[2] Zoltán Ésik. Partial Conway and iteration semiring-semimodule pairs. In Werner Kuich and George Rahonis, editors, *Algebraic Foundations in Computer Science - Essays Dedicated to Symeon Bozapalidis on the Occasion of His Retirement*, volume 7020 of *Lecture Notes in Computer Science*, pages 56–71. Springer, 2011.

[3] Zoltán Ésik and Tamás Hajgató. Kleene theorem in partial Conway theories with applications. In Werner Kuich and George Rahonis, editors, *Algebraic Foundations in Computer Science - Essays Dedicated to Symeon Bozapalidis on the Occasion of His Retirement*, volume 7020 of *Lecture Notes in Computer Science*, pages 72–93. Springer, 2011.

[4] Zoltán Ésik and Werner Kuich. A unifying Kleene theorem for weighted finite automata. In Cristian S. Calude, Grzegorz Rozenberg, and Arto Salomaa, editors, *Rainbow of Computer Science - Dedicated to Hermann Maurer on the Occasion of His 70th Birthday*, volume 6570 of *Lecture Notes in Computer Science*, pages 76–89. Springer, 2011.

[5] Zoltán Ésik and Werner Kuich. A semiring-semimodule generalization of ω-context-free languages. In Juhani Karhumäki, Hermann A. Maurer, Gheorghe Paun, and Grzegorz Rozenberg, editors, *Theory Is Forever, Essays Dedicated to Arto Salomaa on the Occasion of His 70th Birthday*, volume 3113 of *Lecture Notes in Computer Science*, pages 68–80. Springer, 2004.

[6] Zoltán Ésik and Werner Kuich. Equational axioms for a theory of automata. In Carlos Martin-Vide, Victor Mitrana, and Gheorge Paun, editors, *Formal Languages and Applications. (Studies in Fuzziness And Soft Computing 148)*, pages 183–196, 2004.

[7] Zoltán Ésik and Werner Kuich. A generation of Kozen's axiomatization of the equational theory of the regular sets. In Masami Ito, Gheorghe Paun, and Sheng Yu, editors, *Words, Semigroups, and Transductions - Festschrift in Honor of Gabriel Thierrin*, pages 99–114. World Scientific, 2001.

Other scientific papers

[1] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formal languages and automata VII: Formal tree series, Part 2. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, 10:7–49, 2012.

[2] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formal languages and automata VII: Formal tree series, Part 1. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, pages 5–32, 2012.

[3] Zoltán Ésik and Klaus Sutner. Stephen L. Bloom 1940-2010. *Fundam. Inform.*, 109(4):369–381, 2011.

[4] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formal languages and automata VI: ω-algebraic systems and transducers. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, 10:8–32, 2010.

[5] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formal languages and automata V: Conway semiring-semimodule pairs and finite automata. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, pages 6–41, 2009.

[6] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formaljnyje jasyki i avtomaty IV: Transduktory i abstraktnyje semejstva. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, pages 6–23, 2008.

[7] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formalnyje jasyki i avtomaty III.: Magazinnyje avtomaty i formalnyje stepennyje rjady. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, pages 8–27, 2006.

[8] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formal languages and automata, part II: Continuous semirings and algebraic systems. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, pages 19–45, 2004.

[9] S. Aleshnikov, J. Boltnev, Z. Ésik, S. Ishanov, and W. Kuich. Formal languages and automata I: Conway semirings and finite automata. *VESTNIK KALININGRADSKOGO GOSUDARSTVENNOGO UNIVERSITETA*, pages 7–38, 2003.

[10] Stephen L. Bloom and Zoltán Ésik. Two axiomatizations of a star semiring quasi-variety. *Bulletin of the EATCS*, 59, 1996.

[11] Stephen L. Bloom and Zoltán Ésik. Cayley iff Stone. *Bulletin of the EATCS*, 43:159–161, 1991.

[12] Stephen L. Bloom and Zoltán Ésik. Some varieties of iteration theories. *Bulletin of the EATCS*, 24:53–65, 1984.

Contents continued from outside back cover

CONTENTS

Contents continued on inside back cover

ISSN 0324—721 X

Felelős szerkesztő és kiadó: Csirik János