Tomus 3. Fasciculus 4.

ACTA CYBERNETICA

FORUM CENTRALE PUBLICATIONUM CYBERNETICARUM HUNGARICUM

Szeged, 1977

Curat: Universitas Szegediensis de Attila József nominata

3. kötet 4. füzet

ACTA CYBERNETICA

A HAZAI KIBERNETIKAI KUTATÁSOK KÖZPONTI PUBLIKÁCIÓS FÓRUMA

Szeged, 1977. december

A Szegedi József Attila Tudományegyetem gondozásában

Tomus 3.

ACTA CYBERNETICA

FORUM CENTRALE PUBLICATIONUM CYBERNETICARUM HUNGARICUM

Szeged, 1977

Curat: Universitas Szegediensis de Attila József nominata

INDEX

Tomus 3.

77-5590 — Szegedi Nyomda — F. v.: Dobó József

A survey of grammar forms — 1977*

By S. Ginsburg

Introduction

In [3] the notion of a grammar form was abstracted to consider the situation when a master grammar[1] is given and one wishes to discuss grammars which "look like" the master one. Since then, research into grammar forms has continued at a rapid pace.[2] Moreover, other researchers have picked up on the form notion and have written extensively on L-forms (grammar forms applied to L-systems), e.g. [L1—L10]. In the present talk, I shall restrict myself almost entirely to *grammar* forms, and give a brief overview of those portions with which I am most familiar.

Throughout, I assume a general knowledge of language theory.

§1. Preliminaries

By way of motivation for "looks like" in grammar forms, consider the three context-free rules:

(1) $\xi \to a_1 \alpha a_2 \beta$,

(2) $\xi' \to w_1 \alpha' w_2 \beta'$, and

(3) $\xi' \to w_1 \alpha' w_2 \beta' w_3$,

where the Greek letters are nonterminals, the a_i are terminal symbols, and the w_j are terminal words. From an intuitive point of view, would you agree that rule 2 looks like rule 1 (because the primed nonterminals correspond to the unprimed

* The contents of this survey are a combination of two distinct talks. The first was at the Colloquium on Automata and Formal Languages, in Szeged, Hungary, August 30—September 2, 1977. The second was at the 6th International Symposium on Mathematical Foundations of Computer Science, in Tatranska Lomnica, Czechoslovakia, September 5—9, 1977.

[1] Unless otherwise stated, grammar is to mean context-free grammar.

[2] The pace can be determined by comparing the present survey with that given $2\frac{1}{2}$ years ago in [5].

nonterminals, and the terminal words correspond to the terminal symbols)? Would you also agree that (3) does *not* look like (1) (because while β' corresponds to β, w_3 does not correspond to anything to the right of β)? If your answers were yes to both questions, then you should have no difficulty in agreeing with the reasonableness of the abstraction of when one grammar looks like another.

We now formalize our ideas.

Definition. A *grammar form* is a grammar[3] $G=(V, \Sigma, P, \sigma)$, together with underlying infinite alphabets V_∞ and Σ_∞, such that $\Sigma_\infty \subseteqq V_\infty$, $V_\infty - \Sigma_\infty$ is infinite, $\Sigma \subseteqq \Sigma_\infty$, and $V-\Sigma \subseteqq V_\infty - \Sigma_\infty$.

The underlying alphabets V_∞ and Σ_∞ will always be understood. Hence we shall usually omit them and identify a grammar form with a grammar. The term "grammar form" will be employed when we wish to emphasize that the grammar G is conceived as a master grammar for describing a family of grammars, each of which looks like G. The term "grammar" will be used to indicate that the grammar G is to be considered primarily as a device generating a set of strings, i.e., generating a language.

We now specify when one grammar is to "look like" another. The mechanism for accomplishing this is an "interpretation".

Definition. An *interpretation* of a grammar form $G=(V, \Sigma, P, \sigma)$ is a 5-tuple $I=(\mu, V_I, \Sigma_I, P_I, S_I)$, where μ is a substitution on V^* satisfying

(1) $\mu(a)$ is a finite subset of Σ_∞^* for each element a in Σ, $\mu(\xi)$ is a finite subset of $V_\infty - \Sigma_\infty$ for each ξ in $V-\Sigma$, and $\mu(\alpha) \cap \mu(\beta) = \emptyset$ for all $\alpha \neq \beta$ in $V-\Sigma$.

(2) $P_I \subseteqq \bigcup_{\pi \text{ in } P} \mu(\pi)$, where $\mu(\xi \to w) = \{\alpha \to y/\alpha \text{ in } \mu(\xi),\ y \text{ in } \mu(w)\}$.

(3) S_I is in $\mu(\sigma)$.

(4) $V_I(\Sigma_I)$ contains the set of all symbols (terminals) occurring in the rules of P_I.

$G_I=(V_I, \Sigma_I, P_I, S_I)$ is called the *grammar of the interpretation.*

The grammar G_I is context free and is supposed to look like the master grammar G. The substitution μ indicates what symbols in the original grammar can be replaced by what strings, i.e., which words look like what symbols. In particular, terminals are to be replaced by strings of terminals, but nonterminals are only to be replaced by nonterminals. The condition $\mu(\alpha) \cap \mu(\beta) = \emptyset$ for all $\alpha \neq \beta$ in $V-\Sigma$ means that replacement of distinct variables must be by distinct variables. Condition 2 asserts that each rule in P_I must resemble some rule in P. Note that we do not require all rules looking like those in P to appear in G_I. Condition 3 merely says that the start variables must correspond. Condition 4 is strictly technical and asserts that the terminals in G_I come from the universal variable alphabet $V_\infty - \Sigma_\infty$.

Notation. For each grammar form G let $\mathscr{G}(G) = \{G_I/I \text{ an interpretation of } G\}$ and let $\mathscr{L}(G) = \{L(G_I)/G_I \text{ in } \mathscr{G}(G)\}$. $\mathscr{L}(G)$ is called the *grammatical family of G.*

Thus the grammar form G acts as a master grammar for all grammars in $\mathscr{G}(G)$.

---

[3] We assume the reader is familiar, to some extent, with context-free grammars. Here Σ is the finite set of terminals, V is the finite set of both terminals and nonterminals, P is the finite set of rules each of the shape $\xi \to w$, where ξ is a nonterminal and w is in V^*, and σ is in $V-\Sigma$.

We now illustrate the above concepts with some specific grammar forms G. The resulting $\mathscr{G}(G)$ and $\mathscr{L}(G)$ will turn out to be well-known families of grammars and languages.

Examples. (*a*) Let $G=(\{\sigma, a\}, \{a\}, P, \sigma)$, where $P=\{\sigma\to a\sigma, \sigma\to a\}$. Each rule resembling $\sigma\to a\sigma$ is of the kind $\xi\to wv$, where ξ, v are variables and w is a terminal word. The rule $\sigma\to a$ gives rise to rules $\xi\to w$, where w is a terminal word. Then $\mathscr{G}(G)$ is the family of all right-linear grammars and $\mathscr{L}(G)$ is the family of regular sets.

(*b*) Let $G=(\{\sigma, a, b, c\}, \{a, b, c\}, P, \sigma)$, with $P=\{\sigma\to a\sigma b, \sigma\to c\}$. Then $\mathscr{G}(G)$ is the family of all linear grammars and $\mathscr{L}(G)$ is the family of all languages.

(*c*) Let $G=(\{\sigma, a\}, \{a\}, P, \sigma)$, with $P=\{\sigma\to\sigma\sigma, \sigma\to a\}$. Then $\mathscr{G}(G)$ is the family of all grammar in Chomsky binary normal type and $\mathscr{L}(G)$ is the family of all context-free languages.

Results involving just $\mathscr{G}(G)$ or relations between grammars, such as "is an interpretation of", may be viewed as grammar theory. Results concerned with grammatical families may be either grammar theory or language theory, depending on the emphasis.

Finally we have:

Definition. Grammar forms G_1 and G_2 are said to be *strongly equivalent* if $\mathscr{G}(G_1)=\mathscr{G}(G_2)$, and *(weakly) equivalent* if $\mathscr{L}(G_1)=\mathscr{L}(G_2)$.

Thus strong equivalence is a grammar concept, while equivalence may be either a grammar or language concept.

The notion of interpretation given above is the most general that has been seriously considered. On the other hand, there are numerous restrictions on interpretations, leading to such kinds as nondecreasing,[4] length preserving,[5] strict,[6] etc. For each such kind of interpretation x, one may speak of *strong x-equivalence* and *(weak) x-equivalence*, meaning that $\mathscr{G}_x(G_1)=\mathscr{G}_x(G_2)$ and $\mathscr{L}_x(G_1)=\mathscr{L}_x(G_2)$, respectively, $\mathscr{G}_x(G_1)$ being the family of grammars obtained from x-interpretations of G_1 and $\mathscr{L}_x(G_1)$ being the family of languages $\{L(G)/G \text{ in } \mathscr{G}_x(G_1)\}$.

In presenting our survey of grammar form theory, we shall divide the results into five categories. These are grammar, language, decidability, complexity, and applications. As will be noted, some of the results fit into more than one category. In view of the nonmathematical nature of the applications and the mathematical nature of this audience, I shall not report on applications.

§ 2. Grammar theory

The results here are essentially of two kinds. The first involves the notion of "is an interpretation of", while the second concerns normalization theorems, i.e., results such as: For each grammar form with properties $A, B, \ldots$ there exists an equivalent grammar form with properties $P, Q, \ldots$

In [3] it was shown that the relation "is an interpretation of" is transitive. In [10] it was proved that modulo strong equivalence, all grammar forms under "is an

[4] For each element a in Σ, $\mu(a)$ is ε-free.

[5] For each element a in Σ, $\mu(a)$ is a finite subset of Σ_∞.

[6] μ is length preserving, and $\mu(a)\cap\mu(b)=\emptyset$ for all $a\neq b$ in Σ.

interpretation of" form a distributive lattice. Indeed, the existence of a glb for two grammar forms has an interesting restatement as: For all grammar forms G_1 and G_2, there exists a grammar form G_3 such that $\mathcal{G}(G_1)\cap\mathcal{G}(G_2)=\mathcal{G}(G_3)$. In [11], a new operator Q on a grammar form G is defined, yielding a family of grammars. Specifically, $Q(G)=\{G_I/I$ a quasi-interpretation of $G\}$, where a *quasi-interpretation* of a grammar form $G=(V, \Sigma, P, \sigma)$ is a 5-tuple $I=(\mu, V_I, \Sigma_I, P_I, S_I)$ satisfying

(i) μ is a substitution on V^* such that $\mu(a)$ is a finite subset of Σ_∞^* for each element a in Σ and $\mu(\xi)$ is a finite subset of $V_\infty-\Sigma_\infty$ for each ξ in $V-\Sigma$;

(ii) $P_I=\mu(P)$;

(iii) S_I is in $\mu(\sigma)$; and

(iv) $G_I=(V_I, \Sigma_I, P_I, S_I)$ is a grammar for which $V_I(\Sigma_I)$ contains each symbol (terminal) occurring in P_I.

Two results [11] involving $Q(G)$ are: For each grammar form G, $\mathcal{G}Q(G)=Q\mathcal{G}(G)$, and the collection of all families $\mathcal{G}(G')$, G' in $Q(G)$, is finite.

An outstanding open question is the following: Let G be a grammar form and $\mathcal{L}\subseteq\mathcal{L}(G)$ a grammatical family. Is $\mathcal{L}$ in the class $\{\mathcal{L}(G_I)/I$ an interpretation of $G\}$? In other words, do all interpretation grammars of G, when viewed as grammar forms, yield all grammatical subfamilies of $\mathcal{L}(G)$? Analogous questions hold if interpretation is replaced by x-interpretation, x some "reasonable" kind of interpretation.

An open topic suggested by the Q operator is the following: Find different operators $\mathcal{U}$ on grammar forms G so that

(i) $\mathcal{U}(G)$ is a family of grammars, and

(ii) $\mathcal{U}$ has nice properties vis-a-vis operators already specified, e.g., with $\mathcal{G}$ and Q.

Onc would hope that there are a whole host of different operators yielding a variety of new relations and insights. Of special interest would be operators suggested by well-known transformations of grammars in, say compiler theory.

Turning to normalization results we have the following, proved in [3]: Each grammar form has an equivalent, completely reduced[7] sequential grammar form.

Indeed, one might think of a large class of normalization problems thusly: Let P be a property about grammars, e.g., unambiguity. Find grammar forms G with the property: There exists a grammar form G' so that $\mathcal{L}(G)=\{L(G_I)/G_I$ in $\mathcal{G}(G')$, G_I has property $P\}$.

There are many variations to the above stated canonical type problem. Consider this result [7]. If G is an unambiguous grammar form, then $\mathcal{L}_{\text{strict}}(G)=\{L(G_I)/G_I$ in $\mathcal{G}_{\text{strict}}(G)$, G_I unambiguous$\}$. Thus, there are "sufficiently many" unambiguous strict interpretations of an unambiguous grammar form to yield all strict interpretation languages.

Finally, in [14] various kinds, x, of interpretations of a form are studied from the viewpoint of conflict freeness (as used in compiling). For example, let $G=(V, \Sigma, P, \sigma)$ be a grammar form with the property that for each variable ξ there is a non ε terminal word w such that $\xi\overset{+}{\Rightarrow}w$. Then the following three conditions occur simultaneously:

(1) $\mathcal{G}(G)$ is conflict free (i.e., each grammar in $\mathcal{G}(G)$ is conflict free).

---

[7] A grammar form $G=(V, \Sigma, P, \sigma)$ is *completely reduced* if (i) G is reduced, (ii) there are no variables α and β such that $\alpha\to\beta$ is in P, and (iii) for each variable α in $V-(\Sigma\cup\{\sigma\})$ there exist x and y in Σ^*, $xy\neq\varepsilon$, such that $\alpha\to x\alpha y$ is in P.

(2) $\mathscr{G}_{\text{nondecreasing}}(G)$ is conflict free.

(3) G is separated (that is, for each rule $\xi \to w$ in P, w is in $(V-\Sigma)^* \cup \Sigma^*$) and whenever a rule $\xi \to \gamma$ is in P, with γ in $(V-\Sigma)^+$, then γ is in $V-\Sigma$.

Given a grammar form G, $\mathscr{G}_{\text{strict}}(G)$ is conflict free if and only if G is conflict free. Characterization results are presented on a grammar form in order for it to have a strongly ($x-$) equivalent conflict free grammar form, where x is strict, length preserving, and nondecreasing, respectively. It is also shown that every grammar form has an equivalent conflict free grammar form.

§ 3. Language theory

We now review some language theory results. Since language theory itself is so vast, this section could easily dominate all the others. In addition, it is very easy, considering our experience, to phrase innumerable questions about grammar forms which have a language theory flavor. While one cannot stop "progress", I personally believe it is not in the best interests of grammar form theory to exploit grammar forms for the purpose of language interests. The real aim of grammar form theory should be to develop new ideas, insights, questions, etc. about *grammar* concepts.

In § 1, examples were given to show that the regular sets, the linear languages, and the context-free languages are grammatical families. In [3], characterizations on G were given in order that $\mathscr{L}(G)$ be

(1) $\mathscr{R}$, the family of regular sets,

(2) $\mathscr{L}_{\text{lin}}$, the family of linear languages, and

(3) $\mathscr{L}_{CF}$ the family of context free languages.

For (3), the if and only if is quite interesting, namely that G be an expansive grammar in the classical language theory sense. From this it follows that each grammatical family $\mathscr{L}(G) \neq \mathscr{L}_{CF}$ contains only derivation bounded languages. Thus, the one-counter languages are not a grammatical family. This might explain why no "simple" type of context-free-like grammar is around to describe these languages.

Whenever one has a family of languages, it makes sense to investigate its closure properties. For grammar forms we have the surprising result [3] that if G is nontrivial, i.e., $L(G)$ is infinite, then $\mathscr{L}(G)$ is a full principal semi-AFL. The converse, of course, is not true. As mentioned above, the one-counter languages are not a grammatical family. Neither is the full principal semi-AFL generated by $\{a^n b^n / n \geqq 1\}$. In connection with the above semi-AFL result there is a cluster of open questions concerning grammars G such that $L(G)$ is a full generator for $\mathscr{L}(G)$. For example, what are some necessary and sufficient conditions on G, or what are just some useful sufficient conditions? The reader is cautioned to be careful. There are many pitfalls. My favorite is this: $G = (\{\sigma, a, b\}, \{a, b\}, \{\sigma \to a\sigma b, \sigma \to ab\}, \sigma)$ is a form for which $\mathscr{L}(G) = \mathscr{L}_{\text{lin}}$. On the other hand, $L(G) = \{a^n b^n / n \geqq 1\}$, which is not a full generator for $\mathscr{L}_{\text{lin}}$.

One of the major operations in language theory is that of substitution. It is thus natural to try to define the substitution of one grammar form into another. This can be done as follows: For grammar forms G and G', let $\hat{\text{Sub}}\,(G, G')$ be the form obtained by substituting the start variable of G' for every occurrence of a terminal in the productions of G. This yields [13] the obvious result desired, namely, if

G is nontrivial then for every grammar form[8] G', $\mathscr{L}(\hat{\mathrm{Sub}}(G,G'))=\hat{\mathrm{Sub}}(\mathscr{L}(G),\mathscr{L}(G'))$. Now it is known that if $\mathscr{L}$ is a full semi-AFL, then $\hat{\mathrm{Sub}}(\mathscr{R},\mathscr{L})$ is a full AFL. Since the grammar form with rules $\sigma\to a\sigma$, $\sigma\to a$ yields $\mathscr{R}$, it follows that for each grammar form $G'=(V',\Sigma',P',\sigma')$, the form $\hat{\mathrm{Sub}}(G,G')=(V'',\Sigma',P'',\sigma')$ where $P''=P'\cup\{\sigma\to\sigma'\sigma,\sigma\to\sigma'\}$ yields the full AFL generated by $\mathscr{L}(G')$.

Earlier, we noted that for each nontrivial grammar form G, $\mathscr{L}(G)$ is a full principal semi-AFL. It remains an open problem to characterize "internally" those full semi-AFL which are grammatical families. However, we can given "external" characterizations of such semi-AFL. These characterizations are similar in spirit to the Kleene theorem for regular sets, in that they describe the collection of almost all grammatical families in terms of a few elementary ones and composition under some basic operations. We elaborate. For sets $\mathscr{L}_1$ and $\mathscr{L}_2$ of languages, let

$$\mathscr{L}_1\vee\mathscr{L}_2=\{L_1\cup L_2/L_1 \text{ in } \mathscr{L}_1, L_2 \text{ in } \mathscr{L}_2\}$$

and

$$\mathscr{L}_1\odot\mathscr{L}_2=\left\{\bigcup_{i=1}^{k} L_{1i}L_{2i}/k\geqq 1, \text{ each } L_{1i} \text{ in } \mathscr{L}_1, \text{ each } L_{2i} \text{ in } \mathscr{L}_2\right\}.$$

Let $\hat{\mathscr{F}}$ be the full AFL operator, i.e., for each family $\mathscr{L}$ of languages let $\hat{\mathscr{F}}(\mathscr{L})$ be the smallest full AFL containing $\mathscr{L}$. Finally, for all sets $\mathscr{L}_a$, $\mathscr{L}_b$, $\mathscr{L}_c$ of languages, let $\mathscr{T}(\mathscr{L}_a,\mathscr{L}_b,\mathscr{L}_c)=\{\tau(L)/L=L(G),\ G=(V_1,A\cup B\cup C,P,\sigma)$ is a split linear grammar,[9] τ is a substitution on $(A\cup B\cup C)^*$ such that $\tau(x)$ is in $\mathscr{L}_a$ if x is in A, $\tau(x)$ is in $\mathscr{L}_b$ if x is in B, and $\tau(x)$ is in $\mathscr{L}_c$ if x is in $C\}$. There are two characterization results about the grammatical families [4]. The first is: The collection of all grammatical families not $\{\emptyset\}$ and not $\mathscr{L}_{CF}$ is the smallest collection of sets of languages containing $\mathscr{L}_\varepsilon=\{\{\varepsilon\}\}$ and $\mathscr{L}_{\mathrm{fin}}=\{$all finite languages$\}$ and closed under $\vee$, $\odot$, and $\mathscr{T}$. The second is: The collection of all nontrivial grammatical families not $\mathscr{L}_{CF}$ is the smallest collection of sets of languages containing $\mathscr{R}$ and closed under $\vee$, $\odot$, $\mathscr{T}$, and $\hat{\mathscr{F}}$.

At the beginning of this section it was mentioned that each grammatical family not $\mathscr{L}_{CF}$ is a family of derivation bounded languages. As any language theorist knows, there is a close analogy between derivation bounded languages and nonterminal bounded languages. Question — are the nonterminal bounded languages lurking in the grammarform bushes? Answer — yes, if you look for them. Let us call a grammar form $G=(V,\Sigma,P,\sigma)$ *sequentially ultralinear* if

(i) it is sequential, and

(ii) whenever $\xi\to\alpha\xi\beta$ is in P, α and β in V^*, then $\alpha\beta$ is in Σ^*.

Call a grammatical family *ultralinear* if it is generated by some sequentially ultralinear grammar form. The following result has been established [6]. The three statements:

[8] For two families $\mathscr{L}_1$ and $\mathscr{L}_2$ of languages, $\hat{\mathrm{Sub}}(\mathscr{L}_1,\mathscr{L}_2)=\{\tau(L_1)/L_1$ in $\mathscr{L}_1$, τ is a substitution on L_1 such that $\tau(a)$ is in $\mathscr{L}_2$ for every symbol $a\}$.

[9] A *split linear grammar* is a linear grammar $G=(V_1,\Sigma_1,P_1,\sigma_1)$ such that there exist disjoint sets A, B, C with the following properties: (1) $\Sigma_1=A\cup B\cup C$. (2) Every terminal production is of the form $\xi\to c$ for some ξ in $V_1-\Sigma_1$ and c in C. (3) Every production which is not a terminal one is of the form $\xi\to a\xi'$ for some ξ, ξ' in $V_1-\Sigma_1$ and a in A or $\xi\to\xi'b$ for some ξ, ξ' in $V_1-\Sigma_1$ and b in B.

(a) $\mathscr{L}$ is a nontrivial ultralinear grammatical family;
(b) $\mathscr{L}$ is a nontrivial grammatical family of nonterminal bounded languages; and
(c) $\mathscr{L}$ can be built up from $\mathscr{R}$ by a finite sequence of applications of $\odot$, $\vee$, and [ ], where $[\mathscr{L}]=\mathscr{T}(\mathscr{R}, \mathscr{L}, \mathscr{R})$;

are equivalent. Thus, a relatively simple class of grammar forms gives rise to a rather natural class of families of languages.

A rather popular topic in language theory is that of control sets. In [16, 17] Greibach has presented a number of results in which control sets play a leading role. The following is a sample. Let G be a nontrivial left derivation bounded grammar form with left derivation bound k. Then there is a nontrivial equivalent grammar form $G_0=(V_0, \Sigma_0, P_0, \sigma_0)$, left derivation bounded with left derivation bound k, such that for each finite alphabet Σ, $\{L\cap\Sigma^*/L \text{ in } \mathscr{L}(G)\}$ consists of all languages obtained by using regular sets as control sets for leftmost derivations over $\tau_\Sigma(G_0)$. $[\tau_\Sigma(G_0)=(V_0, \Sigma, \tau_\Sigma(P_0), \sigma_0)$, where τ_Σ is the substitution on V_0^* defined by $\tau_\Sigma(\xi)=\{\xi\}$ for each ξ in $V_0-\Sigma_0$ and $\tau_\Sigma(a)=\Sigma\cup\{\varepsilon\}$ for all a in Σ_0.$]$

§ 4 Decidability

There are a number of different decidability results. We shall mention a fair sampling.

It is solvable [3] to determine whether or not, given an arbitrary grammar G' and grammar form G, there is an interpretation I of G such that $G'=G_I$. Also, the strong equivalence problem is solvable. One question that has been open since the beginning of grammar form theory is the decidability of (weak) equivalence. That is, can one tell for arbitrary grammar forms G_1 and G_2 whether $\mathscr{L}(G_1)=\mathscr{L}(G_2)$? Even though the problem is standard in situations of this kind, nevertheless, its solution here seems to be of importance since it seems to be related to several questions involving two or more grammatical families. For example, is $\mathscr{L}(G_1)\cap\mathscr{L}(G_2)$ always a grammatical family? Given a context-free language L, does there exist a smallest grammatical family containing L?

Research is currently underway with respect to the decidability of equivalence. The author, in conjunction with JONATHAN GOLDSTINE and EDWIN H. SPANIER, has reduced the problem to about ten inclusion problems involving the operators $\vee$, $\odot$, $\mathscr{F}$, and $\mathscr{T}$. We think we have resolved all the cases (thereby settling the decidability in the affirmative). However, until *all* the details have been written, we are making no claim. We hope to be able to announce the answer within three months (say December 1, 1977).

A special case of the equivalence problem has been resolved affirmatively. In [6] it is shown that for any two sequentially ultralinear grammar forms G_1 and G_2, it is solvable to determine if $\mathscr{L}(G_1)\subset\mathscr{L}(G_2)$, and therefore if $\mathscr{L}(G_1)=\mathscr{L}(G_2)$. The proof is quite involved, and consists of showing that the operations of $\odot$, [ ], and $\vee$ applied to $\mathscr{R}$, when suitably restricted in combination, are intimately determined by the end ultralinear grammatical family. Indeed, and this is a surprising fact, there is an essentially unique canonical representation of each nontrivial ultralinear grammatical family in terms of "semibracketed expressions", namely, certain combinations of $\mathscr{R}$, $\odot$, $\vee$, and [ ].

In [7], certain decidability results are established for strict interpretations of unambiguous grammar forms. Specifically, for each unambiguous grammar form and each positive integer k, it is decidable whether

(a) an arbitrary strict interpretation grammar is k-ambiguous;

(b) for any k languages $L_1, \ldots, L_k$ generated by[10] compatible strict interpretation grammars, (i) $\bigcap_{i=1}^{k} L_i$ is empty, (ii) $\bigcap_{i=1}^{k} L_i$ is finite, (iii) $\bigcup_{i=1}^{k} L_i$ is infinite; and

(c) for any two languages L_1 and L_2 generated by compatible strict interpretation grammars, (i) $L_1 \subseteqq L_2$ and (ii) $L_1 = L_2$.

§ 5. Complexity

While some work has been done on complexity, this essentially is an area which has received only modest attention. Indeed, the summary given below is basically the same as given in section 5 of [5], with the inclusion of some material from [7].

In [10], it is shown that for each grammar form G there exists an "essentially unique" strongly equivalent form G' with the fewest number of productions possible. Furthermore, G' can always be found with its productions a subset of those of G.

Complexity of derivations is studied in [9]. For each grammar form G and each grammar G' in $\mathscr{G}(G)$, the complexity function $\Phi_{G'}$ is defined for each word x in $L(G')$ as the number of steps in a minimal G'-derivation of x. It is proved that derivations may also be speeded up by any constant factor n, in the sense that for each positive integer n, an equivalent grammar G'' in $\mathscr{G}(G)$ can be found so that $\Phi_{G''}(x) \leqq \frac{|x|}{n}$ for all large words x.

In [10] grammar forms are compared for their efficiency in representing languages, as measured by the sizes (i.e., total number of symbols, number of variable occurrences, number of productions, and number of distinct variables) of interpretation grammars. Right- and left-linear forms are essentially equal in efficiency for every regular set. Each form for the regular sets provides at most polynomial improvement over right-linear form. Moreover, any polynomial improvement is attained by some such form, at least on certain languages. Greater improvement for some languages is possible with forms expressing larger classes of languages than the regular sets. However, there are some languages for which no improvement over right-linear form is possible. A similar set of results holds for forms expressing exactly the linear languages. On the other hand, only linear improvement can occur for forms expressing $\mathscr{L}_{CF}$.

There is one more place where complexity has been considered. This is in

[10] Strict interpretations $I_j = (\mu_{I_j}, V_{I_j}, \Sigma_{I_j}, S_{I_j})$, $j = 1, \ldots, k$, $k \geqq 2$, of a grammar form (V, Σ, P, σ) are called *compatible* if $\left(\bigcup_{j=1}^{k} \mu_{I_j}(x)\right) \cap \left(\bigcup_{i=1}^{k} \mu_{I_i}(y)\right) = \emptyset$ for all x, y in V wih $x \neq y$.

parsing. While parsing can be regarded as an application, for the present purpose I shall catalogue it under complexity. The first result is from [1]. Let G be an arbitrary unambiguous grammar form. Suppose there is a function $t(n)$, $n \geqq 0$, and a parsing procedure M_G for G which, for each word ω, in $t(|\omega|)$ steps, parses ω if in $L(G)$ and rejects ω if not in $L(G)$. Then for each strict interpretation $I=(\mu_I, G_I)$ of G, there exist a parsing procedure M_I for $G_I=(V_I, \Sigma_I, P_I, S_I)$ and a constant c with the following property: For each word w in Σ_I^*, M_I, in $c \cdot t(|w|)$ steps, accepts w if w is in $L(G_I)$ and rejects w if it is not in $L(G_I)$. This result has been generalized in [7]. Specifically, let $G=(V, \Sigma, P, \sigma)$ be an arbitrary grammar form and suppose there is a parsing method M_G for G and a function $t(n)$, $n \geqq 0$, such that for each word of length n, M_G outputs all leftmost derivations of that word in at most $t(n)$ steps. Let $I=(\mu, V_I, \Sigma_I, P_I, S_I)$ be a strict interpretation of G. Then there exists a parsing procedure M_I for G_I and a constant c such that for each word w in Σ_I^*, in $c \cdot t(|w|)$ steps, M_I accepts w if in $L(G_I)$ and rejects w if not in $L(G_I)$. Furthermore, if $p(n)$, $n \geqq 0$, is such that for each word of length n in $L(G_I)$ there are no more than $p(n)$ equally-shaped derivations[11] of that word, then M_I yields, in $c \cdot t(|w|)$ steps, all leftmost G_I-derivations of w.

§ 6. Grammar forms which are not context-free

In the present section, I shall discuss grammar forms which are not necessarily context-free. [The definitions of interpretation, $\mathscr{L}(G)$, etc. carry through in the obvious way.]

The original definition of grammar form, as given in [3], was for arbitrary phrase structure grammars. Due to the scarcity of results in such a general situation, the investigation was quickly limited to context-free grammars and has stayed that way since. At present, with the exception of the first part of [3], the only results on arbitrary grammar forms are in [18]. The basic, original question, and it is still unresolved, is this: Are there any grammar forms G such that

(*) $\mathscr{L}(G) \subseteqq \mathscr{L}_{CF}$ is false and $\mathscr{L}(G) \neq \mathscr{L}_{RE}$, $\mathscr{L}_{RE}$ being the family of recursively enumerable sets?

In 1972, I mentioned this problem to my associate DR. GENE F. ROSE. He struggled with (*), on and off, for several years, to no avail. [That means that the question is difficult.] His opinion was that the answer to (*) was probably no. This opinion is also shared by the authors of [18], as is noted in their abstract. Some progress was made in [18], since it was shown there that the answer to (*) is no when the grammar form has exactly one nonterminal.

Even if the answer to (*) turns out negative, the subject of non context-free grammar forms should be a fertile field of study. All interpretations need not be studied. One could examine appropriate subclasses. [An analogous situation arises with the family of context-sensitive languages. It is not discarded just because its closure under arbitrary homomorphism is $\mathscr{L}_{RE}$.] In fact, a start on this aspect has

[11] Two derivations are *equally shaped* if their parse trees are equally shaped. Two derivation trees are *equally shaped* if each tree can be obtained from the other by relabeling nonmaximal nodes.

already been done in [18]. A number of different, restricted types of interpretations of non context-free forms are considered, and then used to characterize several well-known language families between $\mathscr{L}_{CF}$ and $\mathscr{L}_{RE}$, such as EOL, ETOL, matrix, and scattered languages. Much remains to be done.

§ 7. Future development

The discussion up to now has been on grammar forms. I would like to speak about the general notion of a form as a method of studying when one graphlike structure looks like another.

As we all know, there is a considerable body of knowledge, under the title "*L* systems," of context-free grammars in which parallel derivation occurs, that is, at each step each symbol in the string is replaced. During the past two years the concept of an *L*-form (forms applied to *L*-systems) has been studied [L1—L10]. The results themselves are of no concern to the present discussion. What is of interest is that the notion of form has been carried over to this graphlike structure, with fruitful consequences arising.

Recently, a study was made of pushdown acceptor forms (pda forms) [14]. The aim here is to get a right definition of when one pda looks like another. If one thinks of an input symbol to a pda as a terminal and a state of a pda as a nonterminal, then input symbols are replaced by finite sets of input strings and states by finite sets of states. In addition, distinct states go into disjoint sets of states. But how should one handle replacement of symbols on the auxiliary storage? The key is to regard auxiliary symbols as additional storage. Since states (which are storage) are replaced by finite sets of states (with the disjointness property), pushdown symbols should be replaced by finite sets of pushdown symbols (with the disjointness property). The main question considered for pda forms is what are the resulting families of languages? Because context-free languages coincide with pda languages, the obvious answer would appear to be the class of all grammatical families. And indeed, this is what does happen! However, the proof is quite involved. In any case, the coincidence of the two classes of families is an indication of the "correctness" of the abstraction mode.

Currently, in conjunction with DR. E. F. SCHMEICHEL, I am working on "graph forms" and "looks like" for graphs. The idea is simple. Nodes and edges in a graph are like nonterminals. One must be careful to see that linkage corresponds. Specifically, we have:

Definition. Let $G=(N, E)$ be a (finite) graph. An *interpretation* of G is a triple $I=(\mu, N_I, E_I)$, where μ is a function on $N \cup E$ such that

(i) $\mu(v)$ is a finite set of nodes for each v in N, with $\mu(v_1)\cap\mu(v_2)=\emptyset$ for $v_1\neq v_2$,

(ii) $N_I \subseteqq \bigcup_{v \text{ in } N} \mu(v)$, and

(iii) $E_I \subseteqq \bigcup_{e \text{ in } E} \mu(e)$, with $\mu(v_1, v_2)=\mu(v_1)\times\mu(v_2)$ for each edge $e=(v_1, v_2)$.

For each graph form G let $\mathscr{G}(G)=\{G_I/I \text{ an interpretation of } G\}$.

The investigation here is in its infancy and results obtained to date are scattered. In view of the similarity between interpretations for grammar forms, *L*-forms, pda forms, and graph forms, it seems highly likely that other graphlike structures can be treated from the form perspective. Situations that readily come to mind are: Petri nets, pattern theory, data bases,[12] data types,[13] security models, various types of acceptors. The key in each instance is to determine what "looks like" (i.e., the μ function) is to mean for those features of graphlike structures which are not analogous to variables in a grammar. There does not seem to be any straightforward way of doing this. Rather, insight and trial-and-error appear to be the main techniques. The benefits to be accrued from a successful model for almost any kind of graphlike structure are a strong incentive.

Abstract

The present paper gives an overview of grammar form theory 1977. Concepts, results, and open questions are considered. In addition, general philosophy and future directions are expounded.

UNIVERSITY OF SOUTHERN CALIFORNIA
LOS ANGELES, CALIFORNIA, USA

Bibliography on grammar forms

[1] BERTSCH, E., An observation on relative parsing time, *JACM*, v. 22, 1975, pp. 493—498.
[2] BLATTNER, M. and S. GINSBURG, Position restricted grammar forms, in preparation.
[3] CREMERS, A. and S. GINSBURG, Context-free grammar forms, *JCSS*, v. 11, 1975, pp. 86—116.
[4] CREMERS, A. S. GINSBURG and E. H. SPANIER, The structure of context-free grammatical families, *JCSS*, v. 15, 1977, pp. 262—279.
[5] GINSBURG, S., A survey of context-free grammar forms, in *Formal Languages and Programming*, (R. Aguilar, ed.) North-Holland, 1976.
[6] GINSBURG, S. and J. GOLDSTINE, Ultralinear grammatical families, in preparation.
[7] GINSBURG, S., B. LEONG, O. MAYER and D. WOTSCHKE, On strict interpretations of grammar forms, in preparation.
[8] GINSBURG, S. and N. LYNCH, Size complexity in context-free grammar forms, *JACM*, v. 23, 1976, pp. 582—598.
[9] GINSBURG, S. and N. LYNCH, Derivation complexity in context-free grammar forms, *SIAM J. on Computing*, v. 6, 1976, pp. 123—138.
[10] GINSBURG, S. and H. MAURER, On strongly equivalent context-free grammar forms, *Computing*, v. 16, 1976, pp. 281—291.
[11] GINSBURG, S. and H. MAURER, On quasi-interpretations of grammar forms, *Computing* v. 19, 1977, pp. 141—147.
[12] GINSBURG, S. and E. ROUNDS, Dynamci syntax specification using grammar forms, *IEEE Trans. on Software Engineering*, v. SE—4, 1978, pp. 44—55.
[13] GINSBURG, S. and E. H. SPANIER, Substitution of grammar forms, *Acta Mathematica*, v. 5, 1975, pp. 377—386.
[14] GINSBURG, S. and E. H. SPANIER, Pushdown acceptor forms, to appear in *Theoretical Computer Science*.
[15] GINSBURG, S. and D. WOOD, Simple precedence relations in grammar forms, submitted for publication.
[16] GREIBACH, S., Control sets on context-free grammar forms, *Journal of Computer and System Sciences*, v. 15, 1977, pp. 35—98.

[12] One of my doctoral students is now investigating this.

[13] I have been looking at this, in conjunction with DR. JOHN GUTTAG. There is nothing to report on as yet.

[17] GREIBACH, S., Comments on universal and left universal grammars, context-sensitive languages and context-free grammar forms, submitted for publication.
[18] MAURER, H., M. PENTTOMEN, A. SALOMAA and D. WOOD, On non context-free grammar forms, submitted for publication.
[19] MAURER, H. and D. WOOD, On grammar forms with terminal context, *Acta Informatica*, v. 6, 1976, pp. 397—401.
[20] WALTER, H., Grammatik und Sprachfamilien I, Report F. G. Automatentheorie und Formale Sprachen, T. H. Darmstadt, 1975.
[21] WALTER, H., Grammatik und Sprachfamilien II, Report F. G. Automatentheorie und Formale Sprachen, T. H. Darmstadt, 1975.
[22] WALTER, H., Grammatik und Sprachfamilien III, Report F. G. Automatentheorie und Formale Sprachen, T. H. Darmstadt, 1976.
[23] WALTER, H., Grammatik und Sprachfamilien IV, Report F. G. Automatentheorie und Formale Sprachen, T. H. Darmstadt, 1976.
[24] WALTER, H., Grammarforms und Grammarhomomorphisms, *Acta Informatica*, v. 7, 1976, pp. 75—93.
[25] WALTER, H., Structural equivalence of context-free grammar forms, to appear.

Bibliography on L-forms

[L1] CULIK, K. II and H. A. MAURER, Propagating chain-free normal forms for EOL systems, submitted for publication.
[L2] CULIK, K. II, H. A. MAURER and T. OTTMAN, Two letter complete EOL forms, submitted for publication.
[L3] CULIK, K. II, H. A. MAURER, T. OTTMAN, K. KUOHONEN and A. SALOMAA, Isomorphism, form equivalence and sequence equivalence of pdol forms, submitted for publication.
[L4] HULE, H., H. A. MAURER and T. OTTMAN, OL forms, submitted for publication.
[L5] MAURER, H. A., A. SALOMAA and D. WOOD, EOL forms, *Acta Mathematica*, v. 8, 1977, pp. 75—96.
[L6] MAURER, H. A., A. SALOMAA and D. WOOD, ETOL forms, submitted for publication.
[L7] MAURER, H. A., A. SALOMAA and D. WOOD, Good EOL forms, submitted for publication.
[L8] MAURER, H. A., A. SALOMAA and D. WOOD, Uniform interpretations of L-forms, submitted for publication.
[L9] MAURER, H. A., T. OTTMAN and A. SALOMAA, On the form equivalence of L-forms. *Theoretical Computer Science*, v. 4, 1977, pp. 199—225.
[L10] SKYUM, S., On good ETOL forms, submitted for publication.

(Received Sept. 5, 1977)

On machines as living things*

By LE HOI

I. Introduction

As it is known Von Neumann in [9] considered environment as tessellation structure. The tessellation is a mathematical system to model a behaviour and structure of uniformly interconnected identical finite automata, processing information as the result of local functions acting simultaneously throughout the array on the states of the interconnected automata. Von Neumann [9], J. Thatcher [8] E. F. Codd [4], A. Smith [7] and M. A. Arbib [1—3] considered machines only self-reproducing in tessellation without metabolism, adaptation, evolution etc.

Here we consider environment as modular space.

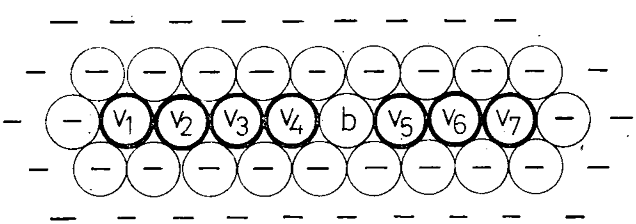

Fig. 1

In Figure 1 v_i representing a module (in state $v_i \in V$) of "solid sub-volume" is considered as a "molecule" of the solid sub-volume embedded in "fluid environment". Moreover, — representing a "raw module" is considered as a free molecule in fluid environment. Every module can change its state depending on its present state and the state of its neighbourhood. But the difference between Von Neumann's tessellation and our modular space is that positions of modules in tessellation are fixed,

* Presented at the Conference on Automata and Formal Languages, Szeged (August 30— September 2, 1977).

but modules in our modular space can move depending on their present neighbourhoods.

For one-dimensional solid volume, we denote its configuration in the environment (like in Fig. 1) by $(\overline{v_1 v_2 v_3 v_4 b v_5 v_6 v_7})$ or shortly by the word $v_1 v_2 v_3 v_4 b v_5 v_6 v_7$. The ring $(\overline{\quad\quad})$ indicates raw modules surrounding the solid volume.

II. Main problems

We define, formalise and construct some kind of universal environments enough for those machines (solid volumes embedded in them), which in the environment not only compute all of the partial recursive functions and are self-reproducing as in Von Neumann's tessellation, but also have some important other characteristics such as growth, death, adaptation and mutation.

III. Main notion and main results

It is shown that the environments which can be formalized by the so-called "parallel exchanging system" (P. E. System) are enough for our above mentioned requirements.

Definition. An RE-System is a triple $S=\langle V, F, b\rangle$ where V is a vocabulary, $b\in V$ is called the "blank", F is a finite non-empty set of productions of the form $(\alpha\bar{v}\beta, \gamma)$, $v\in V$; $\alpha, \beta, \gamma\in V^*$ (that means, v in the neighbourhood $(\alpha^-\beta)$ is replaced by γ) with the following conditions

1) If $(\alpha\bar{v}\beta, \gamma_1)\in F$ and $(\alpha\bar{v}\beta, \gamma_2)\in F$ then $\gamma_1=\gamma_2$.
2) If $(\alpha_1\bar{v}\beta_1, \gamma_1)\in F$ and $(\alpha_2\bar{v}\beta_2, \gamma_2)\in F$ then $|\alpha_1|=|\alpha_2|$ and $|\beta_1|=|\beta_2|$ ($|\alpha|$ is the length of α).
3) Productions of b are only of the forms $(\alpha\bar{b}\beta, v)$, $|v|=1$ or $(\alpha\bar{b}\beta, b^n)$ where $\underset{m}{\forall}\alpha\neq b^m$ or $\underset{m}{\forall}\beta\neq b^m$ and $n, m\in\{0, 1, 2, 3, \dots\}$.

When defining relations on V^* "x directly generates y", written $x\underset{S}{\vdash}y$, "x generates y in k-step", written $x\overset{k}{\underset{S}{\Rightarrow}}y$, and "$x$ terminally generates y" written $x\,|\!\underset{S}{\Rightarrow}y$, productions are applied simultaneously.

Theorem 1. The class of "stability function" $h\colon V^*\to V^*$ ($y=h(x)$ iff $x\,|\!\underset{S}{\Rightarrow}y$) in all PE-Systems $S=\langle V, F, b\rangle$ is a proper subclass of partial recursive functions on V^*.

We can formalise the required environments $E=\langle A, X, F, b\rangle$ as a special kind of PE-System $S=\langle V, F, b\rangle$ where $V=A\cup X$, $b\in X$ and

$$(\alpha\bar{v}\beta, \gamma)\in F \quad |\alpha|\leqq 2, \quad |\beta|\leqq 2, \quad |\gamma|\leqq 2.$$

Definitions. A modular machine (M-machine) Z in an environment $E=\langle A, X, F, b\rangle$ has the following elements:
— the signal to begin working a_0
— the signal to stop working $*$

— the body α containing a program
— an input tape x, output tape y
where a_0, $*$, α are distinguished strings of modules in states from A; x, y are strings of modules in states from X.

When beginning to work the modular machine Z has an initial configuration interpreted as the string of modules $a_0\alpha x$ surrounded by raw modules. Denote this by $(\overline{a_0\alpha x})$. Each module of the machine can change its state to a determined state; or can either become a raw module and go off the machine, or can change its state and simultaneously splice (take in) one raw module $\ominus$ above it onto its left. The behaviour of modules defined by productions F is such that the initial configuration $(\overline{a_0\alpha x})$ can enter the "terminal static configuration" $(\overline{\beta b * \gamma y})$ with $\beta, \gamma \in A^*$, $b=$blank, $y \in X^*$, $a_0 \not\subset \gamma$, $* \not\subset \gamma$ and if $\beta \vdash_E \beta'$ $(*\gamma) \vdash_E (*\gamma)'$ then $\beta b * \gamma y \vdash_E \beta' b (*\gamma)' y$. Furthermore, the machine is always surrounded by raw modules as a solid volume embedded in liquid environment E, and we write

$$a_0\alpha x \underset{E}{\Rightarrow} \beta b * \gamma y.$$

In this case we say the M-machine $Z=a_0\alpha$ or $Z=\langle \alpha, a_0, * \rangle$ in $E=\langle A, X, F, b \rangle$ (denoted by $\langle \alpha, a_0, * \rangle$ in $\langle A, X, F, b \rangle$) transforms x into y (or computes $y=F_z^b(x)$), reproduces β and modifies the program in α to the program in γ, and also write

$$a_0\alpha x \underset{F_z^b}{\Longrightarrow} \beta b * \gamma y$$

if $\beta b * \gamma y$ is the first configuration of this form derived from $a_0\alpha x$.

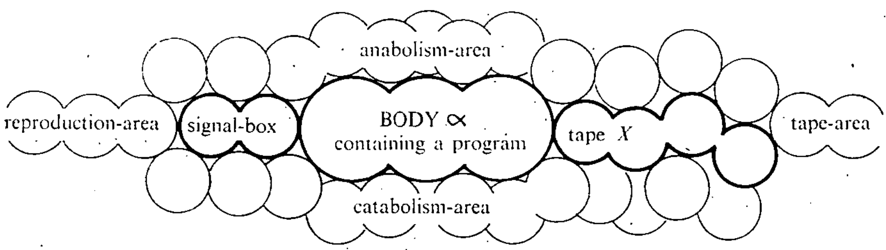

Fig. 2
M-machine Z in E

If product β also is a modular-machine then we say that machine $a_0\alpha$ is a computation-organism (C-organism). If product β equals $a_0\alpha$ or $*\alpha$ then we say that machine $a_0\alpha$ in E is *self-reproducing*. If $|\gamma| > |\alpha|$ and $a_0\alpha$ is also a C-organism in E then $\langle \alpha, a_0, * \rangle$ in E *is growing*. If $|\gamma| < |\alpha|$ then C-organism $\langle \alpha, a_0, * \rangle$ in E is *degenerating*. A C-organism $\langle \alpha, a_0, * \rangle$ in $\langle A, X, F, b \rangle$ is said *to die by* x after computing $y = F_Z^b(x)$ if $a_0\alpha x \overset{Z}{\Rightarrow} *\alpha b * \gamma y$ but $\forall x' \in X^*$: $a_0\gamma x' \,|\!\!\Rightarrow a_0\gamma x'$, that is $Z = a_0\alpha$ is no longer active after interacting with x.

If γ is a function of y (and α) such that $a_0\gamma$ is still an M-machine then $Z=a_0\alpha$ is said to be *adaptive*. If $\langle \alpha, a_0, * \rangle$ is a C-organism and $\exists x \in X^*$ such that β is also an M-machine but $\beta \neq a_0\alpha$ and $\beta \neq *\alpha$ then $\langle \alpha, a_0, * \rangle$ in E is an *M-machine with mutation*.

Theorem 2. There exists a universal environment $E_{c.u}$ in the sense that for every partial recursive function f we can construct an M-machine Z in $E_{c.u}$ to compute f.

Corollary. The class of partial recursive functions coincides with that of parallelly computable functions of modular-machines.

Some theorems show an existence and how to construct the universal environments for growing machines, for self-reproducing, for degenerating, for going to death after a number of computations or for all of them.

Notation. Let $Z=\langle\alpha_0, a_0, *\rangle$ in $\langle A, X, F, b\rangle$ be an adaptive machine and

$$a_0\alpha_0 x_1 \underset{F^b_{\alpha_0}}{\Longrightarrow} \beta_1 b * \gamma_1 y_1$$

$$a_0\gamma_1 x_2 \underset{F^b_{\gamma_1}}{\Longrightarrow} \beta_2 b * \gamma_2 y_2, \ldots, a_0\gamma_{n-1} x_n \underset{F^b_{\gamma_{n-1}}}{\Longrightarrow} \beta_n b \gamma_n y_n,$$

and $a_0\gamma_n$ be a M-machine in $\langle A, X, F, b\rangle$. Then we denote the M-machine $a_0\gamma_n$ in E by $Z(x_1, x_2, \ldots, x_n)$.

Theorem 3. There exists a universal environment for adaptive C-organisms Z's in which every $Z(x_1, x_2, \ldots, x_n)$ also is adaptive and self-reproducing if domain of $F^b_{Z(x_1, x_2, \ldots, x_n)}$ is non-empty.

Theorem 4. There exists a universal environment for adaptive C-organism with mutations Z's (i.e., Z is adaptive and also is with mutation), and if $a_0\alpha x \overset{Z}{\Rightarrow} *\beta b * \gamma y$ then $a_0\beta$ and $a_0\gamma$ also are adaptive C-organisms with mutation (if their domains, Dom F^b_Z, are non-empty) and β is a function of (α, y).

Two last theorems say that by "adaptation" and "mutation" C-organisms in evolution modify their programs in α depending on α and new situation y in the environment and then transmit the new genetic programs in γ to their offspring β.

IV. Conclusion

By tessellation structure, Von Neumann, Thatcher, Codd, Smith, Arbib were concerned with only self-reproducing machines. Professor Pawlak [5] introduced the model of stored program computer only with modification of instructions. René Thom's theory of development and morphogenesis concernes the systematic continuous-topological approach (cf. [6]). Here, by means of PE-System, we introduced a new mathematical model of computing machines not only self-reproducing but with some other essential characteristics of living things, and we showed universal environments for such machines. Since modules in tessellation can not move, self-reproductions and movements in tessellation are rather of configurations, pictures (of machines) than of machines themselves. In our modular space, self-reproduction, adaptation, movement are of modular-machines themselves.

DAI HOC THUONG NGHIEP
(Ng. th. Minh)
HANOI, VIETNAM

References

[1] ARBIB, M. A., *Brains, machines and mathematics*, McGraw-Hill Book Company, New York, 1964.
[2] ARBIB, M. A., A simple self-reproducing universal automata, *Information and Control*, v. 9, No. 2, 1966.
[3] ARBIB, M. A., *Theories of abstract automata*, 1969.
[4] CODD, E. F., Propagation, computation, construction in 2-dimensional cellular spaces, *Technical Publication*, University of Michigan, 1965; Reprinted in 1968 as Cellular Automata, New York.
[5] PAWLAK, Z., A mathematical model of digital computers, *Proc. Conf. on Theory of Automata and Form. Lang.*, Springer-Verlag, Bonn, 1973.
[6] RENE THOM, Stabilité structurelle et morphogenese, *Essai d'une théorie génerale des modeles*, W. A. Benjamin, Inc., Reading, Mass., 1972.
[7] SMITH, A. R., Simple computation-universal cellular spaces and self-reproduction, *Conf. Record. IEEE 9th Ann. Symp.* Switching and Automata Theory, 1968.
[8] THATCHER, J. W., Universality in the Von-Neumann cellular model, *Technical Report*, University of Michigan.
[9] VON NEUMANN, Theory of self-reproducing automata, A. W. Burks Ed., 1966.

(Received Oct. 31, 1977)

Local and global reversibility of finite inhomogeneous cellular automaton*

By E. KATONA

Cellular automata are highly parallel working systems, so they have high importance in computational applications (for example sorting [4], matrix operations, etc.). It seems difficult to apply the classical infinite, homogeneous cellular automata to these purposes [1], [2]. For this reason the classical definitions are modified in this work. In point 1. we introduce the notion of *finite, inhomogeneous* cellular automaton. The reason of first modification (using by many authors, e.g. [7]) is clear: only finite automaton is realisable in practice. Further the second modification (the inhomogeneity) makes the cellular automaton more flexible [11], without excluding the homogeneity in hardware [3].

In the theory of cellular automata there is a very important and interesting question, that how appear the characteristics of local maps in the global map, and conversely. This is the basic conception of present work too, having in the centre the problem of *reversibility*. This subject has been investigated by many authors (in particular by T. Toffoli [8], [9]), but always in the global sense. In this context the reversibility is equivalent to the *bijectivity* of global map.

To the contrary, we mean the reversibility in *local sense*: a cellular automaton we shall call reversible, if its local maps may be changed so, that the new global map is the inverse of the original one.

The bijectivity of global map forms necessary condition for our "strong reversibility". Therefore in point 2. a connection will be proved between the local maps and the number of eden-configurations, from which derives a necessary condition for bijectivity (it is the generalization of results in [5]).

In point 3. a necessary and sufficient condition is presented to the reversibility. With this criterion we can decide the reversibility of a given cellular automaton, and construct its reverse.

The point 4. contains concrete investigations in case of one-dimensional cellular automaton, with the result: only very simple reversible cellular automata exist in this special case.

* Presented at the Conference on Automata and Formal Languages, Szeged (August 30—September 2, 1977).

1. Basic definitions

(i) *Inhomogeneous cellular automaton* is a (C, A, N, Φ) four-tuple, where
$C=\{c_1, \ldots, c_m\}$ is the finite *set of cells*,
$A=\{0, 1, \ldots, s-1\}$ is the *set of cell-states*,
N: $c_i \mapsto (c_{i_1}, \ldots, c_{i_{n_i}})$ is the *neighbourhood function*, which assigns to each cell its neighbours. (The specification of neighbours may be different cell by cell, i.e. the cellular automaton has totally arbitrary topology.)
Φ: $c_i \mapsto f_i$ is the *function-system*, which assigns to each cell an f_i: $A^{n_i} \to A$ local map. (The local maps also may be different cell by cell.)

(ii) *Configuration* is a map α: $C \to A$, we denote it always with Greek letters.
(iii) *Neighbourhood* of cell c_i in a given configuration is the n_i-tuple of states of its neighbours.
(iv) The *global map* of a cellular automaton is a map F: $\mathcal{A} \to \mathcal{A}$ where $\mathcal{A}$ is the set of all configurations, and $F(\alpha)=\beta$, if for all i $f_i(a_{i_1}, \ldots, a_{i_{n_i}})=\beta(c_i)$ (where $(a_{i_1}, \ldots, a_{i_{n_i}})$ is the neighbourhood of c_i in α).
In further we use the abbreviation CA instead of cellular automaton.

2. Relation between the local maps and the number of eden-configurations

We consider a CA (C, A, N, Φ) with the global map F.
The following definition is well-known from the literature:
Definition. A configuration α will be called *garden-of-eden configuration* (in short eden-configuration), if there is no β, for which $F(\alpha)=\beta$.
We have an obvious equivalence:
F is bijective $\Leftrightarrow$ there is no eden-configuration.

Let be c a cell with n neighbours, and f its local map. Suppose, that there are p_a different neighbourhoods of c, where the new cell-state given by f is a. The number of all possible neighbourhoods is s^n, consequently $\sum_{a \in A} p_a = s^n$.

Definition. We say, that the local map f is balanced, if $\forall a$: $p_a=p$, where obviously $p=s^n/s=s^{n-1}$.
When f is unbalanced, the measure of this may be characterized with the quantity $q=\sum_{\substack{a \in A \\ p_a<p}} (p-p_a)$, and we say: f is q-unbalanced.

Theorem. Let be (C, A, N, Φ) an arbitrary CA, c a cell in it, and f its local map. If f is q-unbalanced, then the CA has at least $q \cdot s^{m-n}$ eden-configurations (m is the number of cells, s is the number of cell-sates).

Proof. It is clear, that there are s^{m-n} different configurations, where the neighbourhood of c is a given $(a_1, \ldots, a_n)$. So there are exactly $p_a \cdot s^{m-n}$ configurations, where the new cell-state of c is a. At the same time the number of all configurations, where the state of c is a, is $s^{m-1}=p \cdot s^{m-n}$. Consequently if $p_a<p$, then among these $p \cdot s^{m-n}$ configurations there are $(p-p_a) \cdot s^{m-n}$ eden-configurations.
We find the same situation by all state a having the property $p_a<p$, consequently the CA has at least $\sum_{p_a<p} (p-p_a) \cdot s^{m-n}$ eden-configurations. □

Corollaries. (i) If in a CA for any i the local map of c_i is q_i-unbalanced, then the CA has at least $\max\limits_{1 \leq i \leq m} (q_i \cdot s^{m-n})$ eden-configurations.

(ii) To the bijectivity of global map is necessary condition, that all local maps are balanced.

Similar results are published in works [5], [6] on classical infinite, homogeneous CA.

3. The problem of reversibility
Algorithm for decision of reversibility, and construction of the reverse

Definition. A CA (C, A, N, Φ) with a global map F is *reversible*, if there exists another function-system Φ' such, that the CA (C, A, N, Φ') generates the global map F^{-1}.

The first problem in this subject: to decide from a given CA, whether it is reversible. On this purpose we introduce a general algorithm, which is suitable for constructing the reverse, too.

Let be (C, A, N, Φ) a CA, c_i a cell in it. Let's denote with N_1 the neighbours of c_i, and with N_2 the neighbours of neigbours (with a bit incorrect notation $N_1 = N(c_i)$, $N_2 = N(N(c_i))$). It is clear, that the state of N_1 at time $t+1$ is determined by the state of N_2 at time t. If we know the local functions in N_1, we may describe this transition with a table called in following as *inverse-constructing-table* (ICT in short). In case of one-dimensional, two-state CA it is illustrated on figure 1.

If the cell c_i has an f_i' reverse local function, then this function gives back from any N_1-state of column $t+1$ of ICT the state of c_i in column t. Consequently the existence of f_i' has the following necessary condition: if two N_1-states in column $t+1$ of ICT are equal, then the corresponding c_i-states in column t also should be equal. Furthermore this condition is sufficient to the existence of f_i' reverse function, because we may construct it by the ICT.

$$\underbrace{c_{i-2} \mid \overbrace{c_{i-1} \mid c_i \mid c_{i+1}}^{N_1} \mid c_{i+2}}_{N_2}$$

The local maps in N_1:

f_{i-1}		f_i		f_{i+1}	
000	x_0	000	y_0	000	z_0
001	x_1	001	y_1	001	z_1
010	x_2	010	y_2	010	z_2
011	x_3	011	y_3	011	z_3
100	x_4	100	y_4	100	z_4
101	x_5	101	y_5	101	z_5
110	x_6	110	y_6	110	z_6
111	x_7	111	y_7	111	z_7

The ICT of c_i:

t	$t+1$	t	$t+1$
00000	$x_0 y_0 z_0$	10000	$x_4 y_0 z_0$
00001	$x_0 y_0 z_1$	10001	$x_4 y_0 z_1$
00010	$x_0 y_1 z_2$	10010	$x_4 y_1 z_2$
00011	$x_0 y_1 z_3$	10011	$x_4 y_1 z_3$
00100	$x_1 y_2 z_4$	10100	$x_5 y_2 z_4$
00101	$x_1 y_2 z_5$	10101	$x_5 y_2 z_5$
00110	$x_1 y_3 z_6$	10110	$x_5 y_3 z_6$
00111	$x_1 y_3 z_7$	10111	$x_5 y_3 z_7$
01000	$x_2 y_4 z_0$	11000	$x_6 y_4 z_0$
01001	$x_2 y_4 z_1$	11001	$x_6 y_4 z_1$
01010	$x_2 y_5 z_2$	11010	$x_6 y_5 z_2$
01011	$x_2 y_5 z_3$	11011	$x_6 y_5 z_3$
01100	$x_3 y_6 z_4$	11100	$x_7 y_6 z_4$
01101	$x_3 y_6 z_5$	11101	$x_7 y_6 z_5$
01110	$x_3 y_7 z_6$	11110	$x_7 y_7 z_6$
01111	$x_3 y_7 z_7$	11111	$x_7 y_7 z_7$

Fig. 1
The construction of ICT in case of one-dimensional two-state CA.

So the following in obtained:

Proposition. A CA (C, A, N, Φ) is reversible $\Leftrightarrow$ for each cell c_i, its ICT satisfies: if two N_1-states in column $t+1$ agree, then the corresponding c_i-states in column t must agree too.

If this condition is satisfied, then we can construct the reverse function-system.

4 The reversibility of one-dimensional two-state cellular automaton

The preceding algorithm decides only about a given Φ whether it is reversible, but does not help to find concrete reversible function-systems. It is clear, that there exist trivial ones, for example the identical function-system (where each cell keeps its state, independently of neighbours), or the shift function-system, (where each cell receives the state of the same neighbour).

Nontrivial reversible function-systems have high importance in practice, but to construct them is very difficult. In further we give a necessary condition to the reversibility of one-dimensional two-state CA, from which we shall see, that in one-dimension only very special function-systems are reversible, consequently it is easy to construct them.

So in following the CA (C, A_0, N_0, Φ) will be investigated, where

$C=\{c_1, \ldots, c_m\}$, $m \geqq 5$ is supposed (this assumption makes easier the investigation),

$A_0 = \{0, 1\}$,

N_0: $c_i \mapsto (c_{i-1}, c_i, c_{i+1})$, the indexes are interpreted cyclically (i.e. c_1 and c_m are neighbours). Thus we have a circle-topology.

Φ is arbitrary.

We need the following general definition:

Definition. In a CA (C, A, N, Φ) the cell c_i *depends* on its neighbour c_j, if there are two neighbourhoods of c_i such, that they differ only in state of c_j, and the corresponding new states of c_i are different.

Using this notion we take a remark to the definition of (C, A_0, N_0, Φ): if Φ is such, that c_1 and c_m are independent each of other, then the circle-topology we may replace with a section-topology. So our definition contains the section-topology too.

Two lemmas will be proved in further. In proofs we shall use often the fact, that for reversibility is necessary condition that all local maps are balanced. (It results from the second corollary in point 2.) Moreover we shall use the notation $\bar{a}$, which denotes the opposite of cell-state a.

Lemma 1. Suppose that Φ is reversible, and its reverse is Φ'. In this case if c_{i-1} depends on c_{i-2} by the function-system Φ, then c_i is independent of c_{i-1} by Φ'.

Proof. Suppose, that c_{i-1} depends on c_{i-2}, i.e. there are a, b such, that $f_{i-1}(0, a, b)=x$, and $f_{i-1}(1, a, b)=\bar{x}$.

Now let's consider the function f_{i+1}! We have two different cases:

(i) $\exists y\colon \forall c, d\colon f_{i+1}(b, c, d) = y$.

The function f_{i+1} is balanced, therefore $\forall c, d$: $f_{i+1}(\bar{b}, c, d)=\bar{y}$, that is to say, c_{i+1} depends only on c_i. Thus by the reverse c_i depends only on c_{i+1}.

(ii) $\exists c, d$ and $\exists c', d'\colon f_{i+1}(b, c, d) = y$ and $f_{i+1}(b, c', d') = \bar{y}$.

Let $f_i(a, b, c)=p$, $f_i(a, b, c')=q$. So the ICT of cell c_i contains the following part:

$$\begin{array}{ccccc|ccc} & & t & & & & t+1 & \\ 0 & a & b & c & d & x & p & y \\ 1 & a & b & c & d & \bar{x} & p & y \\ 0 & a & b & c' & d' & x & q & \bar{y} \\ 1 & a & b & c' & d' & \bar{x} & q & \bar{y} \end{array}$$

The four binary triples in column $t+1$ are different, and the reverse function f_i' constructed by the table assigns to each triple the same state b. But f_i' is balanced, so it assigns to the other four triple the state $\bar{b}$. By this the table of f_i' is known. We can see from it, that c_i is independent of c_{i-1}. □

The second lemma needs the following definition:

Definition. Let $c_i, \ldots, c_j$ be a section of cells. We say, that it is *isolated*, if c_i is independent of c_{i-1}, and c_j of c_{j+1}.

Lemma 2. Suppose that Φ is reversible, and its reverse is Φ'. In this case if the section $c_i, \ldots, c_j$ is isolated by Φ, then it is isolated by Φ' too.

Proof. Two configurations will be called equivalent (with respect to the section $c_i, \ldots, c_j$), if their sections corresponding to the $c_i, \ldots, c_j$ are equal. So a classification is obtained on the set $\mathscr{A}$.

It is easy to prove the following chain: $c_i, \ldots, c_j$ is isolated by $\Phi \Rightarrow$ the previous classification is F-compatible (i.e. $\forall \alpha, \beta$: $\alpha \sim \beta \Rightarrow F(\alpha) \sim F(\beta)$) $\Rightarrow$ it is F^{-1}-compatible too (because F is one-to-one) $\Rightarrow$ $c_i, \ldots, c_j$ is isolated by Φ. □

Definition. A function-system we call a *shift function-system*, if each cell depends only on its left (or only on its right) neighbour.

Theorem. If the CA (C, A_0, N_0, Φ) is reversible, then there exists one of the following two cases:

(i) Each cell stands in an isolated section containing maximum three cells.

(ii) Φ is a shift function-system.

Proof. (i) Suppose that there are c_i and c_j such, that c_i is independent of its left neighbour, and c_j is independent of its right neighbour. The cellular automaton has circle-topology, consequently the section $c_i, \ldots, c_j$ always exists. Furthermore this section is isolated, and — having applied the lemma 2. — it is isolated by Φ' too.

Now let's consider an arbitrary cell c_k. According to the lemma 1. either c_{k-1} is independent of c_{k-2}, or by the reverse c_k is independent of c_{k-1}. In the first case the section $c_{k-1}, \ldots, c_j$, in the second case the section $c_k, \ldots, c_j$ is isolated. Applying the geometrical inverse of lemma 1. we get: either $c_i, \ldots, c_{k+1}$ or $c_i, \ldots, c_k$ is isolated. The common part of two isolated sections is isolated too, so we have: c_k stands in an isolated section containing maximum three cells.

(ii) Suppose the negation of the previous case, that is each cell depends (for example) on the left neighbour. We shall prove, that in this case each cell is independent of the right neighbour: suppose, that for any k c_{k+1} depends on c_{k+2}. At the same time c_{k-1} depends on c_{k-2}, and from the lemma 1. we get, that c_k is an isolated cell. This fact contradicts to the original assumption.

So each cell has only two real neighbours: the left cell and itself. We may classify the balanced local maps for two neighbours in three types:

$$\text{I.}\ \begin{array}{cc|c} 0 & 0 & a \\ 0 & 1 & a \\ 1 & 0 & b \\ 1 & 1 & b \end{array} \qquad \text{II.}\ \begin{array}{cc|c} 0 & 0 & a \\ 0 & 1 & b \\ 1 & 0 & a \\ 1 & 1 & b \end{array} \qquad \text{III.}\ \begin{array}{cc|c} 0 & 0 & a \\ 0 & 1 & b \\ 1 & 0 & b \\ 1 & 1 & a \end{array}$$

In our case each cell depends on the left neighbour, so the type II. is out of the question. If all functions have the type III., then $\forall\alpha$: $F(\alpha)=F(\bar{\alpha})$, thus the global map is not one-to-one.

If there are functions type I. and type III. at the same time, then there exists a cell c_i such, that f_i has the type I., and f_{i+1} has the type III. Therefore the ICT of c_i contains the following part:

$$\begin{array}{ccccc|ccc} & & t & & & & t+1 & \\ a & b & c & d & e & x & y & z \\ a & b & \bar{c} & \bar{d} & e & x & y & z \end{array}$$

These two lines exclude the reversibility.

So we get: all local maps have the type I., i.e. Φ is a shift function-system.

Corollaries. 1. If (C, A_0, N_0, Φ) has section topology, then each reversible Φ has the type (i).

2. If (C, A_0, N_0, Φ) is homogeneous, then we have only the six trivial reversible function-systems: the identical one, and its contrary (where each cell alters its state independently of neighbours), the left and right shift function-systems, and their contrary.

THE BUILDING COMPANY OF
SOUTHERN HUNGARY (DÉLÉP)
H-6721 SZEGED, HUNGARY
BOCSKAI U. 10—12.

References

[1] CODD, E. F., *Cellular automata*, Academic Press, Inc., New York, 1968.
[2] FÁY, GY., Introduction to cellular automata, *Lecture notes on Univ. Eötvös L.*, Budapest, 1974, (in Hungarian).
[3] LEGENDI, T., Cellprocessors in computer architecture, *Computational Linguistics and Computer Languages XI*, 1977, pp. 147—167.
[4] LEGENDI, T., Programming of cellular processors, *Proceedings of the Braunschweig Cellular Meeting*, June 2—3, 1977.
[5] RICHARDSON, D., Tessellations with local transformations, *J. Comput. System Sci.*, v. 6, 1972, pp. 373—388.
[6] MARUOKA, A. & M. KIMURA, Condition for injectivity of global maps for tessellation automata, *Information and Control*, v. 32, 1976, pp. 158—162.
[7] HARAO, M. & SH. NOGUCHI, Finite cellular automata as models of self-organizing systems, *Conf. on Formal Language, Automata and Development*, 1975.
[8] TOFFOLI, T., On the computability of backward steps in cellular spaces, *CNR, Inst. per le Appliziori del Calcolo*, Roma, 1972.
[9] TOFFOLI, T., Computation- and construction-universality of reversible cellular automata, *Like the previous publ.*, 1976.
[10] AMOROSO, S., G. COOPER & Y. PATT, Some clarifications of the concept of a garden-of-eden configuration, *U. S. Army Electronics Command*, 1971.
[11] KATONA, E., The realisation of addition, multiplication and associative storage with cellular automaton, *Manuscript* (in Hungarian).

(Received Oct. 31, 1977)

On two modified problems of synchronization in cellular automata*

By R. Vollmar

1. A modified firing squad synchronization problem

As Moore (1964) states, the problem to synchronize a finite (but arbitrary long) chain of finite automata was devised about 1957 by Myhill. In the meantime this problem has become well-known as the firing squad synchronization problem (fssp).

Among other people Waksman (1966) and Balzer (1967) have given minimal-time solutions. Moreover there exist some modifications, especially the synchronization of two- and three-dimensional arrays (Shinahr (1974), Nguyen and Hamacher (1974), Grasselli (1975)) and of growing arrays (Herman et al. (1974)) have been investigated. In the "classical" fssp the synchronization process is started by one automaton, the so-called general, at the border. Moore and Langdon (1968) and Varshavsky et al. (1970) have renounced this assumption and they have stated minimal-time solutions for this modification.

We consider a further modification. Starting point is a chain of n automata where each automaton is directly connected with its two neighbours. In the "classical" fssp at time $t=0$ all automata except one of the border automaton are in the quiescent state. This quiescent state is distinguished by the property that an automaton will retain it at time $t+1$ if itself and its two neighbours have been in the quiescent state at time t. Here we will assume that initially k automata, where $1 \leqq k \leqq n$, are allowed to be set to the "general state" — all the other automata assume the quiescent state — and after that it is also possible that automata become generals at later moments.

The problem is to specify the structure of the automata such that independently of the number of automata and generals all automata enter a special state, called "fire" state at exactly the same time and this state may not be assumed at any earlier moment by any automaton.

* Presented at the Conference on Automata and Formal Languages, Szeged (August 30— September 2, 1977).

This generalization is motivated by the consideration of models of neural layers and their interpretation as cellular automata (an example is given in Vollmar/Spreng (1976)): One of the layers has to detect some changes within an other layer and afterwards it must give simultaneously signals to the following layer. A change is obtained if at least one of the automata in the layer receives a certain number of special signals in a certain interval. It may happen that several automata identify changes and they start independently synchronization processes.

The basis of our solution are time-optimal algorithms of the problem to synchronize chains which contain one general but at an arbitrary place (see e.g. Moore/Langdon (1968) or Varshavsky/Marakhovsky/Peschansky (1970)): The general sends out signals (waves) in the two directions which halve the chain, then halve the two new chains etc. Our concept of the age of signals is applicable to any algorithm of this kind.

Our solution is composed of two independently working procedures; they have been combined in such a manner that the procedure which is the first to end, will cause the synchronization. This is done because the two procedures have incomparable synchronization times.

The synchronization time by one of the procedures, which has been described in Vollmar (1976), is achieved in $<2.5n$.

To find a "good" solution of our problem it is necessary to decide quickly which of two waves coming from distinct generals will survive. We have chosen a strategy such that whenever two signals collide, the signal coming from the "elder" general will survive. This is motivated by the fact that with respect to the synchronization process in some but not in all cases the elder general has "done more" than a younger one. However there are space-time configurations for which this is not valid (see fig. 1). If at the border (or nearby) a general g_1 originates at time t_0 and at time t_0+1 a general g_2 will originate nearby the center of the chain, at time t_c the signals transmitted from g_2 will have passed almost the double number of automata than those ones transmitted from g_1. This disadvantage of our procedure could only be repaired if it were possible to determine the age of the generals *and* their positions relative to the center. Up to now we did not succeed in doing this fast enough.

First we will describe the part of the procedure which has to detect the elder signal. Afterwards it will be shown how the synchronization process can be delayed in dependence on the time needed for this detection.

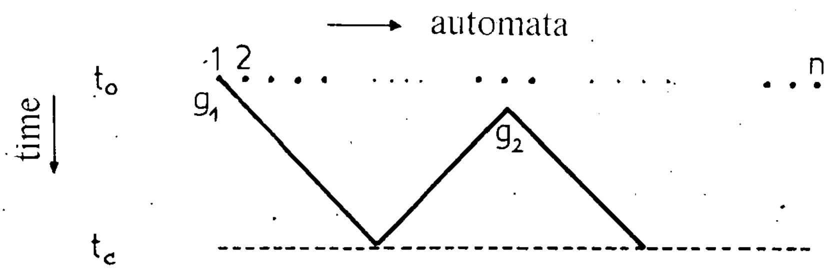

Fig. 1
A "bad" space-time configuration

To be able to classify the signals according to their ages it is necessary that the signals immediately coming from generals "drag along" its ages: For this reason the state set of the automata is increased in such a way that among others the digits of a corresponding number system and some marks can be stored. The age or more precisely the number of automata which have been passed through, is represented in the top automaton in which the signal is arrived and possibly in some automata which have been reached earlier (see fig. 2; only the information relevant for the age is displayed).

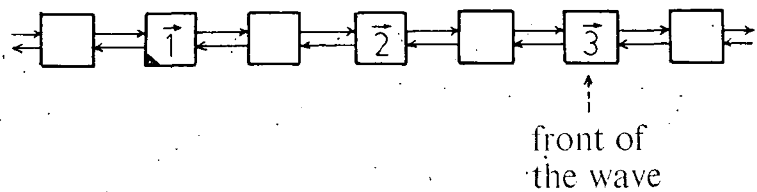

Fig. 2
Configuration with the age of the wave

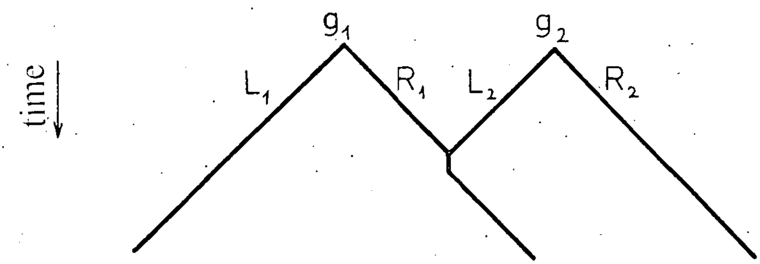

Fig. 3
The problem of overtaking waves

Whenever two signals collide, the distances to the corresponding transmitting generals have to be compared. To do this the propagation of these signals stops and the numbers are subtracted. For this the digits of one of the numbers travel successively to the corresponding place of the other number, i.e. it is stored there in a reversal order. Simultaneously to this shifting process the two numbers are subtracted digit by digit, whenever this is possible. When the first digit of the number — especially marked — has reached the "valid position", the subtraction is finished, and the result is sent out in the corresponding direction to restart the transmission of the "elder" signal. If two signals have the same age by definition the left one will survive. The time needed to make such a comparison is given by $c \log n$ where $c \in \mathbf{N}$.

But still another problem arises (see fig. 3). If the two generals g_1 and g_2 have been originated at the same time, according to our agreement after the collision of R_1 and L_2 R_1 survives. After the comparison it propagates to the right following R_2 and writes its information over that one of R_2. Dependent on the distance between g_1 and g_2 and the running time it is possible that the number representing the age of R_2 is overtaken by R_1. From this time on the age of R_2 is incorrectly represented but as the R_1-signal following from the left (resp. the last of the following signals) is the valid one, there will be no confusion. It is impossible to "inform" the R_2-signal about these occurences because it propagates with unit speed. On the other hand during the comparison of the ages this overtaking will be detected and will cause an interrupt of the subtraction, and the comparison will be done with the following R_1-signal, etc.

If two or more generals exist, it is possible that one of the signals transmitted from a general stops for a certain interval and the other signal moves on (in the opposite direction). To prevent any disturbance of the synchronization process at each step a signal does not propagate a delay signal is transmitted. This signal moves into the opposite direction of the (original) movement of the stopped signal and the transition of each automaton is delayed for one time unit. In fig. 4 the movement and the effect of a delay signal is displayed.

It is clear that the time of the sketched algorithm depends on the number of generals: Since each general causes a delay of about $c \log n$ of the synchronization

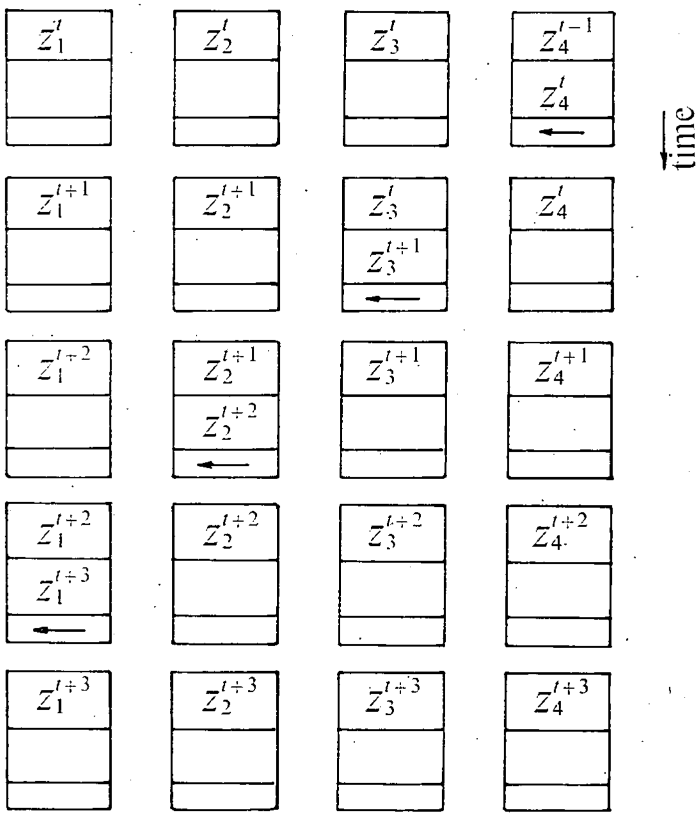

Fig. 4
The movement of a delay signal

time, an upper bound for the total synchronization time is given by

$$2n + cn \log n.$$

The quality of this bound is illustrated by fig. 5.

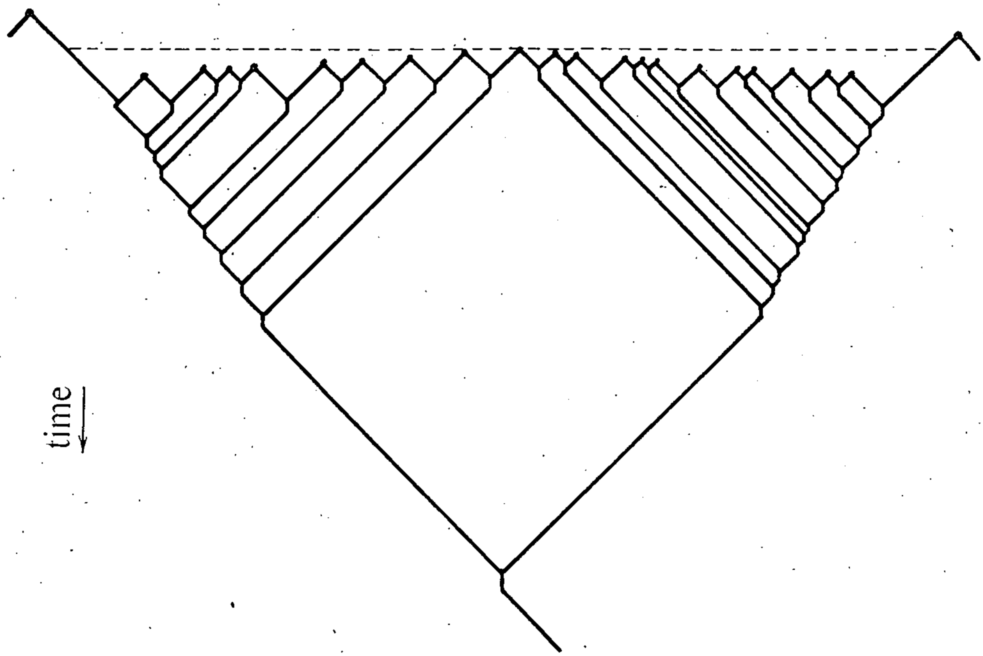

Fig. 5
Space-time diagram of a strongly delayed synchronization
(Between the generals (●) there are other automata.)

On the other hand it should be mentioned that the minimal time is obtained if only one general exists. Moreover there exist configurations for which the sketched algorithm needs a shorter synchronization time as the other procedure mentioned above and vice versa. Therefore we combine the two procedures such that the synchronization time will be

$$< 2.5n.$$

It should be remarked that the method sketched above is also applicable for several generals at arbitrary positions in a rectangular array.

2. A modification of the early bird problem

The method described above is also applicable to a modification of the early bird problem. Rosenstiehl et al. (1972) have described the following problem: To each of the n vertices of an elementary cyclic graph there is assigned an automaton.

These automata may be "excited" at different moments (from the outside); for simplicity we will also say that they assume a "general state". The automata must be designed in such a way that the (single) automaton, which has been excited first, eventually will assume a distinguished state E and all the other automata will assume states I. Rosenstiehl et al. give a solution of this problem, which needs $2n$ steps. They emphasize that this solution does not work if two or more excitations occur at the same moment.

We will discuss a solution of the following problem: At each time an arbitrary number of automata in a chain of n automata may be excited with the only restriction that at time $t=0$ at least one automaton has to be excited and it is not allowed to excite automata which have leaved the quiescent state. After a certain period automata which have been initially excited must assume the state E, and all the other automata assume the state I.

The solution is obtained by the following procedure: Each of the originating generals sends out age signals, as described above. If they collide with other signals a comparison is made. Irrespective of the states in any case the elder signal is transmitted. If two signals of the same age collide, both signals are reflected — with special marks —.

These reflected signals are transmitted backwards, subtracting 1 at each step, until they are decremented to the value 0. In this case, the corresponding automata are marked. An automaton is an early bird (EB) if it is marked by signals from the right and from the left and if the chain of automata has reached a certain age. The last condition is necessary to exclude "local" EBs.

Each automaton contains information about the age of the signal and about the distance to the sending general. In contrast to the procedure in the foregoing paragraph the age is also increased at each step the signal transmission is stopped (because the signals have collided and the comparison takes place).

The number representing the age has to be stored in the automata located between the sending automaton and the automaton where the collision occurs. In general this number will be greater than the number representing the distance, but there are no storage problems if the numbering system is appropriately chosen.

After a collision the numbers representing the *age* are compared: If these numbers are equal, i.e. they come from generals of the same age, or if a signal reaches a border automaton, then the numbers representing the *distance* are reflected. These numbers are transmitted and at each step the value is decremented by 1 until the value 0 is assumed. The corresponding automaton has sent the original signal. It is marked with a label indicating that a reflected signal has arrived. Another label is set if two reflected signals have arrived; in such cases it may be that the marked automaton is *not* an EB (see fig. 6).

Decrementing the numbers a special consideration is necessary if the lowest digit of the number equals 0; but we will not discuss this here. These delays are not illustrated in fig. 6. The total time for the return of these signals depends linearly on the number of the automata.

To compute the total time we have to take into account the following: the time is greater than that one in the paragraph above because the comparisons have to be made with the age numbers (and not with the distances); and those numbers are given as the sum of the distance numbers and the sum of the times for comparisons.

A rough estimation of the total time is given by

$$n+c'\sum_{i=1}^{\lceil n/2\rceil}\log(i\cdot n)$$

if $n>1$, where c' is a constant depending on the algorithm which performs the comparisons. An estimation of the term is given by

$$n\cdot(1+\bar{c}\log n).$$

The age of the signals and the distances to the generals are represented in a polyadic numbering system; therefore the maximum of the values is estimable by $k\cdot\log_B n$. It is possible to give a basis B — which is independent of n — such that the numbers can be stored in the automata between the generals and the collision automata.

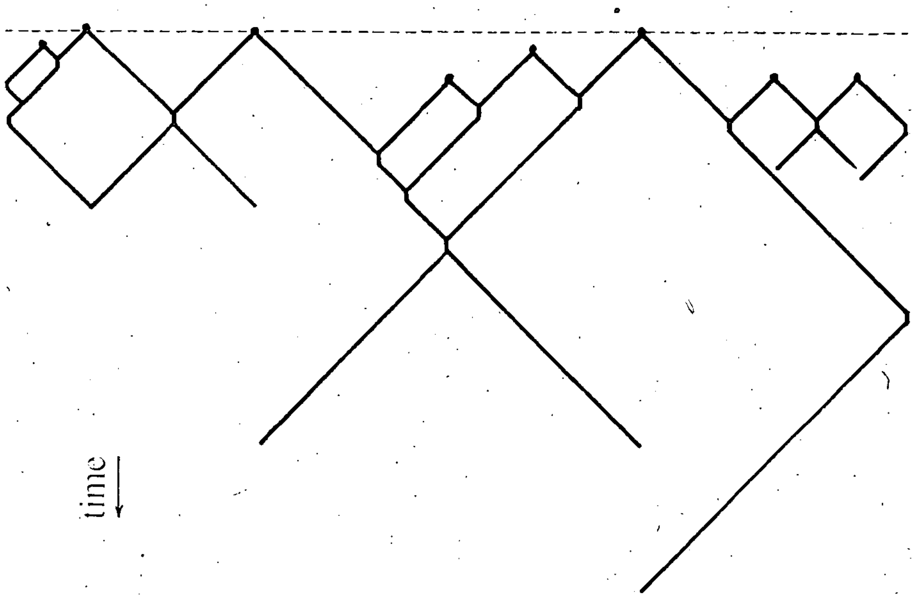

Fig. 6
Space-time diagram of an Early-Bird solution
(without the synchronization of the E- and I-signals)
(Between the generals (●) there are other automata.)

At the time all comparisons have terminated the reflected signals must go back to the corresponding automata. As mentioned above, this time depends linearly on n, and therefore we can find a constant c such that the total amount of time is given by $n\cdot(1+c\log n)$.

To guarantee the synchronized transition to the states E and I, we start at time $t=0$ — independently of the processes described above — a counting procedure which counts (using all the automata for storage) until $\lceil n\cdot(1+c\log n)\rceil$.

If this value is reached, a border automaton starts a synchronization process (following an usual fssp algorithm) such that the states E and I are assumed synchronously.

The solution to the modified EBP needs about $n\cdot(1+\log n)$ time steps. The solution time does not depend on the number of excitations.

It should be noted that this procedure does not solve the modified version of the original problem. As mentioned above in our procedure it is necessary to determine one of the automata as general to start the synchronization process; by reason of the homogeneity of the connections and the determinism of the automata this determination cannot be done in an elementary cyclic graph. On the other side our procedure does not produce either a correct non-synchronous solution because we must wait a certain period — and it is not possible to represent this time in the graph — to make the decision whether doubly marked automata are "real" EBs (see fig. 6).

Abstract

We will introduce the concept of the age of signals which is well-suited for the solution of modifications of the "firing squad synchronization problem" (fssp) and of the "early-bird problem" (ebp).

TECHN. UNIV. BRAUNSCHWEIG
D-3300 BRAUNSCHWEIG, FRG

References

BALZER, R., An 8-state minimal time solution to the firing squad synchronization problem, *Information and Control*, v. 10, 1967, pp. 22—42.

GRASSELLI, A., Synchronization of cellular arrays: The firing squad problem in two dimensions, *Information and Control*, v. 28, 1975, pp. 113—124.

HERMAN, G. T., W. H. LIU, S. ROWLAND and A. WALKER, Synchronization of growing cellular arrays, *Information and Control*, v. 25, 1974, pp. 103—121.

MOORE, E. F. (ed.), *Sequential machines, Selected papers*, Addison-Wesley, Reading, Massachusetts, 1964.

MOORE, F. R. and G. G. LANGDON, A generalized firing squad problem, *Information and Control*, v. 12, 1968, pp. 212—220.

NGUYEN, H. B. and V. C. HAMACHER, Pattern synchronization in two-dimensional cellular spaces, *Information and Control*, v. 26, 1974, pp. 12—23.

ROSENSTIEHL, P., J. R. FIKSEL and A. HOLLIGER, Intelligent graphs: Networks of finite automata capable of solving graph problems, in: Read, R. C. (ed.), *Graph theory and computing*, Academic Press, New York, 1972, pp. 219—265.

SHINAHR, I., Two- and three-dimensional firing squad synchronization problems, *Information and Control*, v. 24, 1974, pp. 163—180.

VARSHAVSKY, V. I., V. B. MARAKHOVSKY and V. A. PESCHANSKY, Synchronization of interacting automata, *Math. Systems Theory*, v. 4, 1970, pp. 212—230.

VOLLMAR, R., Yet another generalization of the firing squad problem, *Informatik-Berichte der Technischen Universität Braunschweig*, Nr. 7601, Braunschweig, 1976.

VOLLMAR, R. and M. SPRENG, Ein zellulares Netz als Modell eines Teilsystems der akustischen Informationsverarbeitung, *Arbeitsberichte des Instituts für Mathematische Maschinen und Datenverarbeitung der Universität Erlangen-Nürnberg*, Bd. 9, Nr. 8, Erlangen, 1976, pp. 193—211.

WAKSMAN, A., An optimum solution to the firing squad synchronization problem, *Information and Control*, v. 9, 1966, pp. 66—78.

(Received Nov. 1, 1977.)

On α_i-products of automata

By B. IMREH

The purpose of this paper is to study the α_i-products (see [1]) from the point of view of isomorphic completeness. Namely, we give necessary and sufficient conditions for a system of automata to be isomorphically complete with respect to the α_i-product. It will turn out that there exists no minimal isomorphically complete system of automata with respect to α_i-product and if $i \geqq 1$ then isomorphically complete systems coincide with each other with respect to different α_i-products. Moreover, we prove that if $i<j$ then the α_j-product is isomorphically more general than the α_i-product.

By an automaton we mean a finite automaton without output. Let $\mathbf{A}_t = (x_t, A_t, \delta_t)$ $(t=1, \ldots, n)$ be a system of automata. Moreover, let X be a finite nonvoid set and φ a mapping of $A_1 \times \ldots \times A_n \times X$ into $X_1 \times \ldots \times X_n$ such that $\varphi(a_1, \ldots, a_n, x) = (\varphi_1(a_1, \ldots, a_n, x), \ldots, \varphi_n(a_1, \ldots, a_n, x))$, and each φ_j $(1 \leqq j \leqq n)$ is independent of states having indices greater than or equal to $j+i$, where i is a fixed nonnegative integer. We say that the automaton $\mathbf{A} = (X, A, \delta)$ with $A = A_1 \times \ldots \times A_n$ and

$$\delta((a_1, \ldots, a_n), x) = (\delta_1(a_1, \varphi_1(a_1, \ldots, a_n, x)), \ldots, \delta_n(a_n, \varphi_n(a_1, \ldots, a_n, x)))$$

is the *α_i-product* of $\mathbf{A}_t$ $(t=1, \ldots, n)$ with respect to X and φ. For this product we use the shorter notation $\mathbf{A} = \prod_{t=1}^{n} \mathbf{A}_t(X, \varphi)$.

Let Σ be a system of automata. Σ is called *isomorphically complete* with respect to the α_i-product if any automaton can be embedded isomorphically into an α_i-product of automata from Σ. Furthermore, Σ is called *minimal* isomorphically complete system if Σ is isomorphically complete and for arbitrary $\mathbf{A} \in \Sigma$ the system $\Sigma \setminus \{\mathbf{A}\}$ is not isomorphically complete.

Take a set M of automata, and let i be an arbitrary nonnegative integer. Let $\alpha_i(M)$ denote the class of all automata which can be embedded isomorphically into an α_i-product of automata from M. It is said that the α_i-product is *isomorphically more general* than the α_j-product if for any set M of automata $\alpha_j(M) \subseteq \alpha_i(M)$ and there exists at least one set $\bar{M}$ such that $\alpha_j(\bar{M})$ is a proper subclass of $\alpha_i(\bar{M})$.

The following statement is obvious for arbitrary natural number $i \geqq 0$.

Lemma. If $\mathbf{A}$ can be embedded isomorphically into an α_i-product $\mathbf{B}$ with a single factor and $\mathbf{B}$ can be embedded isomorphically into an α_i-product $\mathbf{C}$ with a single factor, then $\mathbf{A}$ can be embedded isomorphically into an α_i-product $\mathbf{C}$ with a single factor.

For any natural number $n \geqq 1$ denote by $\mathbf{T}_n = (T_n, N, \delta_N)$ the automaton for which $N = \{1, \ldots, n\}$, T_n is the set of all transformations t of N, and $\delta_N(j, t) = t(j)$ for all $j \in N$ and $t \in T_n$.

The next Theorem gives necessary and sufficient conditions for a system of automata to be isomorphically complete with respect to α_0-product.

Theorem 1. A system Σ of automata is isomorphically complete with respect to α_0-product if and only if for any natural number $n \geqq 1$, there exists an automaton $\mathbf{A} \in \Sigma$ such that $\mathbf{T}_n$ can be embedded isomorphically into an α_0-product of $\mathbf{A}$ with a single factor.

Proof. The necessity and sufficiency of these conditions will be proved in a similar way as that of the corresponding statement for generalized α_0-product in [2].

In order to prove the necessity assume that Σ is isomorphically complete with respect to the α_0-product. Let $n > 1$ be a natural number and take $\mathbf{T}_n$. By our assumption, $\mathbf{T}_n$ can be embedded isomorphically into an α_0-product $\mathbf{B} = (T_n, B, \delta_{\mathbf{B}}) = \prod_{t=1}^{m} \mathbf{A}_t(T_n, \varphi)$ of automata from Σ. Assume that $m > 1$, and let μ denote a suitable isomorphism. Define partitions π'_j $(j = 1, \ldots, m)$ on B in the following way: $(a_1, \ldots, a_m) \equiv (a'_1, \ldots, a'_m)(\pi'_j)$ $(a_1, \ldots, a_m), (a'_1, \ldots, a'_m) \in B$ if and only if $a_1 = a'_1, \ldots, a_j = a'_j$. Now let π_j $(j = 1, \ldots, m)$ be partitions on N given as follows: for any $(a_1, \ldots, a_m), (a'_1, \ldots, a'_m) \in B$ we have $\mu^{-1}(a_1, \ldots, a_m) \equiv \mu^{-1}(a'_1, \ldots, a'_m)(\pi_j)$ if and only if $(a_1, \ldots, a_m) \equiv (a'_1, \ldots, a'_m)$ (π'_j). It is easy to prove that π_j $(j = 1, \ldots, m)$ have the Substitution Property (SP). On the other hand, for $\mathbf{T}_n$ only the two trivial partitions have SP. Thus, we get that each π_j has one-element blocks only, or it has one block only. Among these partitions there should be at least one which has more than one block, since $n > 1$. Let l be the least index for which π_l has at least two bloks. Then the blocks of π_l consist of single elements. Therefore, the number of all blocks of π_l is n. We show that $\mathbf{T}_n$ can be embedded isomorphically into an α_0-product $\mathbf{A}_l$ with a single factor. Let $(a_{i1}, \ldots, a_{im})$ denote the image of i $(i = 1, \ldots, n)$ under μ. From our assumption and the definition of π_j it follows that $a_{ks} = a_{1s}$ if $1 \leqq k \leqq n$ and $1 \leqq s \leqq l-1$. Take the α_0-product $\mathbf{C} = (T_n, A_l, \delta_{\mathbf{C}}) = \Pi \mathbf{A}_l(T_n, \Psi)$ where $\Psi(t) = \varphi_l(a_{11}, \ldots, a_{1l-1}, t)$ for all $t \in T_n$. It is easy to prove that mapping $v: i \to a_{il}$ $(i = 1, \ldots, n)$ is an isomorphism of $\mathbf{T}_n$ into $\mathbf{C} = \Pi \mathbf{A}_l(T_n, \Psi)$.

The case $n = 1$ is obvious.

To prove the sufficiency take an automaton $\mathbf{A} = (X, A, \delta_{\mathbf{A}})$ with n states. Let μ be an arbitrary $1-1$ mapping of A onto N. Take the α_0-product $\mathbf{C} = \Pi \mathbf{T}_n(X, \varphi)$ with a single factor, where $\varphi(x) = t$ if and only if $\mu(\delta_{\mathbf{A}}(a, x)) = t(\mu(a))$ for any $a \in A$. Then μ is an isomorphism of $\mathbf{A}$ into $\mathbf{C}$. On the other hand, by our assumption, there exists an automaton $\mathbf{B}$ in Σ such, that $\mathbf{T}_n$ can be embedded isomorphically into an α_0-product of $\mathbf{B}$ with a single factor. Therefore, by our Lemma, $\mathbf{A}$ can be embedded isomorphically into an α_0-product of $\mathbf{B}$, which completes the proof of Theorem 1.

Corollary. There exists no system of automata which is isomorphically complete with respect to α_0-product and minimal.

Proof. Take a system Σ of automata which is isomorphically complete with respect to α_0-product, and let $\mathbf{A}\in\Sigma$ be an automaton with n states. It is obvious that $\mathbf{A}$ can be embedded isomorphically into an α_0-product of $\mathbf{T}_m$ with a single factor if $m\geqq n$. Take a natural number $m>n$. By Theorem 1, there exists a $\mathbf{B}\in\Sigma$ such that $\mathbf{T}_m$ can be embedded isomorphically into an α_0-product of $\mathbf{B}$ with a single factor. Therefore, by our Lemma, $\mathbf{A}$ can be embedded isomorphically into an α_0-product of $\mathbf{B}$ with a single factor. Thus, $\Sigma\setminus\{\mathbf{A}\}$ is isomorphically complete with respect to α_0-product, showing that Σ is not minimal.

For any natural number $n\geqq 1$ denote by $\mathbf{D}_n=(\{x_{pq}\}_{\substack{1\leqq p\leqq n\\ 1\leqq q\leqq n}}, \{1, \dots, n\}, \delta_n)$ the automaton for which for any $l\in\{1, \dots, n\}$ and $x_{sk}\in\{x_{pq}\}$

$$\delta_n(l, x_{sk}) = \begin{cases} k & \text{if } \ l=s \\ l & \text{otherwise.}\end{cases}$$

The following Theorem holds for α_i-products with $i\geqq 1$.

Theorem 2. A system Σ of automata is isomorphically complete with respect to α_i-product ($i\geqq 1$) if and only if for any natural number $n\geqq 1$, there exists an automaton $\mathbf{A}\in\Sigma$ such that $\mathbf{D}_n$ can be embedded isomorphically into an α_i-product of $\mathbf{A}$ with a single factor.

Proof. First we prove that $\mathbf{D}_n$ $(n>1)$ can be embedded isomorphically into an α_i-product of automata from Σ with at most i factors if $\mathbf{D}_n$ can be embedded isomorphically into an α_i-product of automata from Σ. Indeed, assume that $\mathbf{D}_n$ can be embedded isomorphically into the α_i-product $\mathbf{B}=\prod_{t=1}^{k}\mathbf{A}_t(\{x_{pq}\}, \varphi)$ of automata from Σ with $k>i$, and let μ denote the isomorphism. For any $l\in\{1, \dots, n\}$ denote by $(a_{l1}, \dots, a_{lk})$ the image of l under μ. We may suppose that there exist natural numbers $r\neq s$ $(1\leqq r, s\leqq n)$ such that $a_{r1}\neq a_{s1}$ since otherwise $\mathbf{D}_n$ can be embedded isomorphically into an α_i-product of automata from Σ with $k-1$ factors. Now assume that there exist natural numbers $u\neq v$ $(1\leqq u, v\leqq n)$ such that $a_{ut}=a_{vt}$ $(t=1, \dots, i)$. Then $\varphi_1(a_{u1}, \dots, a_{ui}, x_{lr})=\varphi_1(a_{v1}, \dots, a_{vi}, x_{lr})$ for any $x_{lr}\in\{x_{pq}\}$. Thus in the α_i-product $\mathbf{B}$ the automaton $\mathbf{A}_1$ obtains the same input signal in the states a_{u1} and a_{v1} for any $x_{lr}\in\{x_{pq}\}$. On the other hand since μ is an isomorphism and $u\neq v$, thus the automaton $\mathbf{A}_1$ from the state a_{u1} goes into the state a_{r1} and from the state a_{v1} it goes into the state a_{v1} for any input signal x_{ur} $(1\leqq r\leqq n)$. This implies $a_{v1}=a_{r1}$ $(1\leqq r\leqq n)$, which contradicts our assumption. Thus we get that the elements $(a_{t1}, \dots, a_{ti})$ $(1\leqq t\leqq n)$ are pairwise different. Take the following α_i-product $\mathbf{C}=(\{x_{pq}\}, C, \delta_c)=\prod_{t=1}^{i}\mathbf{A}_t(\{x_{pq}\}, \psi)$ where for any $j=1, \dots, i$, $(a_1, \dots, a_i)\in A_1\times\dots\times A_i$ and $x\in\{x_{pq}\}$

$$\psi_j(a_1, \dots, a_i, x) = \begin{cases} \varphi_j(a_{t1}, \dots, a_{tj+i-1}, x) & \text{if } j+i-1\leqq k \text{ and there exists} \\ & 1\leqq t\leqq n \text{ such that } a_s=a_{ts} \ (s=1, \dots, i,) \\ \varphi_j(a_{t1}, \dots, a_{tk}, x) & \text{if } j+i-1>k \text{ and there exists} \\ & 1\leqq t\leqq n \text{ such that } a_s=a_{ts} \ (s=1, \dots, i), \\ \text{arbitrary input signal from } X_j & \text{otherwise.}\end{cases}$$

It is clear that the correspondence $\nu: l \to (a_{l1}, \ldots, a_{li})$ is an isomorphism of $\mathbf{D}_n$ into $\mathbf{C}$.

Now we show that if $\mathbf{D}_n$ $(n>1)$ can be embedded isomorphically into an α_i-product of automata from Σ with at most i factors then there exists an automaton $\mathbf{A} \in \Sigma$ such that $\mathbf{D}_{[\sqrt[i]{n}]}$ can be embedded isomorphically into an α_i-product of $\mathbf{A}$ with a single factor, where $[\sqrt[i]{n}]$ denotes the largest integer less than or equal to $\sqrt[i]{n}$. Indeed, assume that $\mathbf{D}_n$ can be embedded isomorphically into the α_i-product $\mathbf{B} = \prod_{t=1}^{k} \mathbf{A}_t(\{x_{pq}\}, \varphi)$ of automata from Σ with $k \leqq i$ factors. Let μ denote a suitable isomorphism, and for any $l \in \{1, \ldots, n\}$ let $(a_{l1}, \ldots, a_{lk})$ be the image of l under μ. Since μ is a $1-1$ mapping, thus the elements $(a_{t1}, \ldots, a_{tk})$ $(t=1, \ldots, n)$ are pairwise different. Therefore, there exists an s $(1 \leqq s \leqq k)$ such that the number of pairwise different elements among $a_{1s}, a_{2s}, \ldots, a_{ns}$ is greater than or equal to $[\sqrt[i]{n}]$. Let $a_{j_1 s}, \ldots, a_{j_r s}$ denote pairwise different elements, where $r \geqq [\sqrt[i]{n}]$, and denote by $\bar{X}$ the set of input signals x_{pq} $(1 \leqq p, q \leqq [\sqrt[i]{n}])$. Take the following α_i-product $\mathbf{C} = \prod \mathbf{A}_s(\bar{X}, \Psi)$ with single factor, where for any $a_{j_t s} \in \mathbf{A}_s$ and $x_{uv} \in \bar{X}$

$$\psi(a_{j_t s}, x_{uv}) = \begin{cases} \varphi_s(a_{j_t 1}, \ldots, a_{j_t k}, x_{j_t j_v}) & \text{if} \quad u = t \\ \varphi_s(a_{j_t 1}, \ldots, a_{j_t k}, x_{j_t j_t}) & \text{otherwise.} \end{cases}$$

It can be proved easily that the correspondence $\nu: t \to a_{j_t s}$ $(t=1, \ldots, [\sqrt[i]{n}])$ is an isomorphism of $\mathbf{D}_{[\sqrt[i]{n}]}$ into $\mathbf{C}$.

The case $n=1$ is again obvious. To prove the sufficiency by our Lemma, it is enough to show that arbitrary automaton with n states can be embedded isomorphically into an α_i-product of $\mathbf{D}_n$ with a single factor. This is trivial.

Corollary. There exists no system of automata which is isomorphically complete with respect to α_i-product $(i \geqq 1)$ and minimal.

In the sequel we shall study general properties of α_i-products $(i=0, 1, \ldots)$. For this we need some preparation.

Take a set A and a system $\pi_0, \ldots, \pi_n$ of partitions on A. We say that this system of partitions is *regular* if the following conditions are satisfied:

(1) π_0 has one block only,

(2) π_n has one-element blocks only,

(3) $\pi_0 \geqq \pi_1 \geqq \ldots \geqq \pi_n$.

Let π be a partition of A. For any $a \in A$, denote by $\pi(a)$ the block of π containing a. Moreover, set $M_{j,a} = \{\pi_{j+1}(b) : b \in A \text{ and } b \equiv a(\pi_j)\}$, where $a \in A$ and $j=0, \ldots, n-1$. Finally, let $\pi_j/\pi_{j+1} = \max \{|M_{j,a}| : a \in A\}$.

It holds the following.

Theorem 3. Let $l>2$ be a natural number and $i \geqq 1$. An automaton $\mathbf{A} = (X, A, \delta_{\mathbf{A}})$ can be embedded isomorphically into an α_i-product of automata having fewer states than l, if and only if there exists a regular system $\pi_0, \ldots, \pi_n$ of partitions of A such that

(I) $\pi_j/\pi_{j+1} < l$ for all $j=0, \ldots, n-1$,

(II) $a \equiv b(\pi_j)$ implies $\delta_{\mathbf{A}}(a, x) = \delta_{\mathbf{A}}(b, x)$ (π_{j-i+1}) for all $i-1 \leqq j \leqq n$, $x \in X$ and $a, b \in A$.

Proof. Theorem 3 will be proved in a similar way as the corresponding statement for generalized α_i-products in [2].

In order to prove necessity assume that the automaton $\mathbf{A}$ can be embedded isomorphically into an α_i-product $\prod_{t=1}^{n} \mathbf{A}_t(X, \varphi)$ of automata with $|A_t| < l$ $(t=1, \dots, n)$ and $l > 2$. Let μ denote a suitable isomorphism. Define partitions π_j $(j=0, 1, \dots, n)$ on A in the following way: π_0 has one block only, and $a \equiv a'(\pi_j)$ $(1 \leqq j \leqq n)$ if and only if $\mu(a) = (a_1, \dots, a_n)$, $\mu(a') = (a'_1, \dots, a'_n)$ and $a_1 = a'_1, \dots, a_j = a'_j$. It is obvious that $\pi_0, \pi_1, \dots, \pi_n$ is a regular system of partitions and conditions (I) and (II) are satisfied by this system.

Conversely, assume that for an $\mathbf{A} = (X, A, \delta)$ there exists a regular system $\pi_0, \dots, \pi_n$ of partitions satisfying conditions (I) and (II). We construct automata $\mathbf{A}_j = (X_j, A_j, \delta_j)$ $(j=1, \dots, n)$ with $|A_j| = \pi_{j-1}/\pi_j (< l)$ such that the automaton $\mathbf{A}$ can be embedded isomorphically into an α_i-product of automata $\mathbf{A}_j$ $(j=1, \dots, n)$.

Let A_j be arbitrary abstract sets with $|A_j| = \pi_{j-1}/\pi_j$ and $X_j = A_1 \times \dots \times A_{j+i-1} \times X$ if $j+i-1 \leqq n$ and $X_j = A_1 \times \dots \times A_n \times X$ otherwise. Now let μ_j be a mapping of $M_j = \{\pi_j(a) : a \in A\}$ onto A_j such that the restriction of μ_j to any $M_{j-1,a}$ is $1-1$. Define the transition function δ_j in the following way:

(1) if $j+i-1 \leqq n$ then for any $a_j \in A_j$ and $(b_1, \dots, b_{j+i-1}, x) \in X_j$

$$\delta_j(a_j, (b_1, \dots, b_{j+i-1}, x)) = \begin{cases} \mu_j(\pi_j(\delta(a, x))) \text{ if } a_j = b_j \text{ and there exists an } a \in A \\ \qquad \text{such that } \mu_t(\pi_t(a)) = b_t \text{ for all } t = 1, \dots, i+j-1, \\ \text{arbitrary element from } A_j \text{ otherwise,} \end{cases}$$

(2) if $j+i-1 > n$ then for any $a_j \in A_j$ and $(b_1, \dots, b_n, x) \in X_j$

$$\delta_j(a_j, (b_1, \dots, b_n, x)) = \begin{cases} \mu_j(\pi_j(\delta(a, x))) \text{ if } a_j = b_j \text{ and there exists an } a \in A \\ \qquad \text{such that } \mu_t(\pi_t(a)) = b_t \text{ for all } t = 1, \dots, n, \\ \text{arbitrary element from } A_j \text{ otherwise.} \end{cases}$$

First we prove that δ_j is well defined. Assume that in case (1) there exists a $b \in A$ such that $\mu_t(\pi_t(b)) = b_t$ $(t=1, \dots, j+i-1)$. It is enough to show that $b \equiv a(\pi_{j+i-1})$ since this by (II), implies that $\delta(b, x) \equiv \delta(a, x)$ for any $x \in X$. We proceed by induction on t. $b \equiv a(\pi_1)$ obviously holds since μ_1 is a $1-1$ mapping of M_1 onto A_1. Assume that our statement has been proved for $t-1$ $(1 \leqq t-1 < j+i-1)$ that is $b \equiv a(\pi_{t-1})$. Therefore, since μ_t is $1-1$ on $M_{t-1,a}$ and $\mu_t(\pi_t(a)) = \mu_t(\pi_t(b))$ thus $\pi_t(b) = \pi_t(a)$. Case (2) can be proved by a similar argument.

Take the α_i-product $\mathbf{B} = \prod_{t=1}^{n} \mathbf{A}_t(X, \varphi)$ where the mapping φ_j is defined in the following way:

(1) if $j+i-1 \leqq n$ then for any $(a_1, \dots, a_{j+i-1}) \in A_1 \times \dots \times A_{j+i-1}$ and $x \in X$

$$\varphi_j(a_1, \dots, a_{j+i-1}, x) = (a_1, \dots, a_{j+i-1}, x),$$

(2) if $j+i-1 > n$ then for any $(a_1, \dots, a_n) \in A_1 \times \dots \times A_n$ and $x \in X$

$$\varphi_j(a_1, \dots, a_n, x) = (a_1, \dots, a_n, x).$$

It is easy to prove that the mapping $\nu: a \to (\mu_1(\pi_1(a)), \ldots, \mu_n(\pi_n(a)))$ is an isomorphism of **A** into **B**, which completes the proof of Theorem 3.

Let us denote by $\mathbf{A}_2 = (\{x, y\}, \{0, 1\}, \delta_2)$ the automaton for which $\delta_2(0, x) = \delta_2(1, y) = 1$ and $\delta_2(1, x) = \delta_2(0, y) = 0$.

Now we prove

Theorem 4. Automaton $\mathbf{D}_n$ can be embedded isomorphically into an α_i-product of $\mathbf{A}_2$ $(i \geqq 1)$ if and only if $1 \leqq n \leqq 2^i$.

Proof. The necessity follows from Theorem 3. Indeed, if $\mathbf{D}_n$ can be embedded isomorphically into an α_i-product of $\mathbf{A}_2$, then by Theorem 3, there exists a regular system $\pi_0, \pi_1, \ldots, \pi_k$ of partitions of the set $\{1, \ldots, n\}$ such that (I) and (II) are satisfied. If $n > 2^i$ then there exists a subsystem $\pi_{t_1} > \pi_{t_2} > \ldots > \pi_{t_i}$ of $\pi_0, \ldots, \pi_k$ such that $\pi_0 > \pi_{t_1}$ and $\pi_{t_i} > \pi_k$. Since $\pi_{t_i} > \pi_k$ thus there exists at least one block of π_{t_i} which has more than one element, that is there exist l and r $(1 \leqq l, r \leqq n)$ with $l \neq r$ and $l \equiv r(\pi_{t_i})$. From this, by condition (II), we get that for all $x_{sv} \in \{x_{pq}\}_{\substack{1 \leqq p \leqq n \\ 1 \leqq q \leqq n}}$ $\delta_n(l, x_{sv}) \equiv \delta_n(r, x_{sv})(\pi_{t_1})$. This implies $\pi_0 = \pi_{t_1}$, which contradicts the assumption that $\pi_0 > \pi_{t_1}$.

To prove the sufficiency let n be an arbitrary natural number with $1 \leqq n \leqq 2^i$. We take the α_i-product $\mathbf{B} = \prod_{t=1}^{i} \mathbf{A}_2(\{x_{pq}\}, \varphi)$ of $\mathbf{A}_2$, where the mapping φ_j is defined in the following way: for any

$$(a_1, \ldots, a_i, x_{sr}) \in \{0, 1\} \times \{0, 1\} \times \ldots \times \{0, 1\} \times \{x_{pq}\}$$

$$\varphi_j(a_1, \ldots, a_i, x_{sr}) = \begin{cases} x \text{ if } \sum_{t=1}^{i} a_t 2^{i-t} + 1 = s \text{ and } r = \sum_{t=1}^{i} b_t 2^{i-t} + 1 \text{ and } a_j \neq b_j, \\ y \text{ otherwise.} \end{cases}$$

It is not difficult to prove that $\mathbf{D}_n$ can be embedded isomorphically into the automaton **B** under the isomorphism μ defined as follows: if $k = \sum_{t=1}^{i} a_t 2^{i-t} + 1$ then $\mu(k) = (a_1, \ldots, a_i)$ for all $k = 1, \ldots, n$. This ends the proof of Theorem 4.

Let $\mathbf{C}_n$ denote the automaton $(\{x\}, \{1, \ldots, n\}, \delta_n)$ where for all $1 \leqq k < n$ $\delta_n(k, x) = k + 1$ and $\delta_n(n, x) = n$.

It can easily be seen that for any natural number $n \geqq 1$ $\mathbf{C}_n$ can be embedded isomorphically into an α_1-product of $\mathbf{A}_2$. On the other hand it is not difficult to prove that if $n > 1$ then $\mathbf{C}_n$ cannot be embedded isomorphically into an α_0-product of $\mathbf{A}_2$. From this we obtain that the α_1-product is isomorphically more general than the α_0-product.

In [3] V. M. Gluskov introduced the concept of the general product and proved that system $\{\mathbf{A}_2\}$ is isomorphically complete with respect to the general product. This, by Theorem 4, implies that for any natural number i the general product is isomorphically more general than the α_i-product.

Our results can be summarized by

Theorem 5. The general product is isomorphically more general than any α_j-product $(j = 0, 1, 2, \ldots)$ and any i, j $(i, j \in \{0, 1, 2, \ldots\})$ if $i < j$ then the α_j-product is isomorphically more general than the α_i-product.

Finally we consider that what kind automata can be embedded isomorphically into an α_i-product $(i=0, 1, 2, \ldots)$ of automata from the given finite set of automata. For this the following is valid.

Theorem 6. For any natural number $i(\geqq 0)$, automaton $\mathbf{A}$ and finite set M of automata it can be decided whether or not $\mathbf{A}\in\alpha_i(M)$.

Proof. Assume that automaton $\mathbf{A}=(X, A, \delta_{\mathbf{A}})$ with m states can be embedded isomorphically into an α_i-product $\mathbf{B}=\prod_{t=1}^{s} \mathbf{A}_t(X, \varphi)$ of automata from M under the isomorphism μ. Let $V=\max\{|A_t|:\mathbf{A}_t\in M\}$, and for all $a_i\in A$ $(i=1, \ldots, m)$ denote by $(a_{i1}, \ldots, a_{is})$ the image of a_i under μ. We define partition π on the set of indices of the α_i-product $\mathbf{B}$. Any k, l $(1\leqq k, l\leqq s)$ $k\equiv l(\pi)$ if and only if $\mathbf{A}_k=\mathbf{A}_l$ and $a_{tk}=a_{tl}$ for all $t=1, \ldots, m$. It can easily be seen that the partition π has at most $|M|\cdot V^m$ blocks. Since μ is an isomorphism, thus if $a_{tk}=a_{tl}$ $(t=1, \ldots, m)$ then the k-th component of $\mu(\delta_{\mathbf{A}}(a_t, x))$ is equal to the l-th component of $\mu(\delta_{\mathbf{A}}(a_t, x))$ for all $t=1, \ldots, m$ and $x\in X$. By this it is not difficult to prove, that the automaton $\mathbf{A}$ can be embedded isomorphically into an α_i-product of automata from the set M with at most $|M|\cdot V^m$ factors.

DEPT. OF COMPUTER SCIENCE
A. JÓZSEF UNIVERSITY
H-6720 SZEGED, HUNGARY
SOMOGYI U. 7.

References

[1] GÉCSEG, F., Composition of automata, *Proceedings of the 2nd Colloquium on Automata, Languages and Programming*, Saarbrücken, 1974, Springer Lecture Notes in Computer Science, v. 14, pp. 351—363.

[2] GÉCSEG, F., On products of abstract automata, *Acta Sci. Math.*, v. 38, 1976, pp. 21—43.

[3] GLUSKOV, V. M., Абстрактная теория автоматов, *Успехи матем. наук*, 16:5 (101), 1961, pp. 3—62.

(Received Nov. 22, 1977)

Rational representation of forests by tree automata

By G. Maróti

1. Introduction

In this paper we give a new representation of forests which is more powerful than the usual one in the following sense: for this representation there exists a proper variety which is complete, i.e., every regular forest can be represented (in this new sense) by a tree automaton built on a finite algebra belonging to this variety (Theorem 5). This representation is a generalization of the rational one developed by F. Gécseg in [1]. Moreover our Theorem 5 yields immediately the result of F. Gécseg and G. Horváth [2]: there exists a proper variety over the type $G=\{g, h\}$, where the arities of g and h are 2 and zero, respectively, such that every context-free language can be recognized by a finite tree-automaton belonging to this variety.

2. *Fr*-homomorphism and *Fr*-embedding

Let F be a nonvoid set and r a mapping of F into the set N of all nonnegative integers. We call the ordered pair $\langle F, r\rangle$ a *type*. The elements of F are the *operational symbols*. If $f\in F$ and $r(f)=n$ $(n\in N)$ then we say that the *arity* of f is n (or f is an *n-ary* operational symbol). We will refer to the type $\langle F, r\rangle$ simply by F. The subset of all 0-ary operational symbols will be denoted by F^0.

Take the set $X=\{x_0, x_1, \dots\}$ and a type F. The set $T_{F,n}$ of the *n-ary polynomial symbols over* F is defined by

1) $x_0, \dots, x_{n-1}\in T_{F,n}$,

2) if $p_0, \dots, p_{m-1}\in T_{F,n}$ and $f\in F$ is an m-ary operational symbol $(m\geqq 0)$ then $f(p_0, \dots, p_{m-1})\in T_{F,n}$,

3) $T_{F,n}$ is the smallest set satisfying 1) and 2).

The set T_F of all polynomial symbols over F is defined as the union of all $T_{F,n}$

$$T_F = \bigcup_{n=0}^{\infty} T_{F,n}.$$

Every polynomial symbol $p\in T_F$ can be represented by a tree P (by a loop-free connected graph) whose nodes are labelled by the elements of the set $F\cup X$ in such a way that if a node has the label $f \in F$ then there are exactly $r(f)$ edges leaving it. We use the terminology that P is the tree belonging to the polynomial symbol p.

Consider the polynomial symbols $p \in T_{F,m}$ and $p_0, \dots, p_{m-1} \in T_{F,n}$. Then $p(p_0, \dots, p_{m-1})$ denotes the following n-ary polynomial symbol over F:

1) if $p = x_i$ $(0 \leqq i \leqq m-1)$ then $p(p_0, \dots, p_{m-1}) = p_i$,

2) if $p = f(q_0, \dots, q_{k-1})$, where $f \in F$ and $r(f) = k$, then

$$p(p_0, \dots, p_{m-1}) = f\big(q_0(p_0, \dots, p_{m-1}), \dots, q_{k-1}(p_0, \dots, p_{m-1})\big).$$

Next we define the *mapping fr (frontier): fr* is a mapping of T_F into the free monoid generated by the set X satisfying the following conditions:

1) $fr(x_i) = x_i$ $(i = 0, 1, \dots)$,

2) if $h \in F^0$, then $fr(h) = \varepsilon$ (ε denotes the empty word),

3) if $p = f(p_0, \dots, p_{m-1})$, where $f \in F$ and $r(f) = m$, then

$$fr(p) = fr(p_0) \dots fr(p_{m-1}).$$

Let us consider now two types F and G. The mapping $\alpha: T_F \to T_G$ is called an *fr-homomorphism (frontier-homomorphism)* if it satisfies the following conditions:

(i) $\alpha(x_i) = x_i$ $(i = 0, 1, \dots)$

(ii) $fr\big(\alpha(f(x_0, \dots, x_{n-1}))\big) = fr\big(f(x_0, \dots, x_{n-1})\big)$, where $f \in F$ and $r(f) = n$ $(n \geqq 0)$,

(iii) $\alpha\big(f(p_0, \dots, p_{n-1})\big) = \alpha(f)\big(\alpha(p_0), \dots, \alpha(p_{n-1})\big)$.

Corollary 1. For every polynomial symbol $p \in T_F$ and for every fr-homomorphism $\alpha: T_F \to T_G$ we have

$$fr\big(\alpha(p)\big) = fr(p).$$

Let $d(p)$ denote the *depht* of the polynomial symbol p, i.e., if p is equal to x_i or a 0-ary operational symbol then $d(p) = 0$, and if p is of the form $p = f(p_0, \dots, p_{m-1})$ then $d(p) = \max_{i=0, \dots, m-1} \{d(p_i)\} + 1$.

Corollary 2. Let $\alpha: T_F \to T_G$ be an fr-homomorphism, and assume that for every $f \in F$, $d\big(\alpha(f(x_0, \dots, x_{r(f)-1}))\big) \geqq 1$. Then for each $p \in T_F$

$$d\big(\alpha(p)\big) \geqq d(p)$$

holds.

Proof. Let $p \in T_F$. If $d(p) = 0$ then the assertion is trivial. Assume that Corollary 2 is true for all polynomial symbols whose depht is less than that of $p = f(p_0, \dots, p_{m-1})$. Then

$$d\big(\alpha(p)\big) = d\big(\alpha(f(p_0, \dots, p_{m-1}))\big) = d\big(\alpha(f)(\alpha(p_0), \dots, \alpha(p_{m-1})\big) \geqq$$
$$\geqq 1 + \max_{i=0, \dots, m-1} \big\{d\big(\alpha(p_i)\big)\big\} \geqq 1 + \max_{i=0, \dots, m-1} \{d(p_i)\} = d(p).$$

If the fr-homomorphism $\alpha: T_F \to T_G$ is one-to-one then it is called *fr-embedding*.

Let us denote by $T_F[1]$ the set of all polynomial symbols from F with depth less than or equal to 1:

$$T_F[1] = \{p \mid p \in T_F \text{ and } d(p) \leqq 1\}.$$

For every mapping $\varphi: T_F[1] \to T_G$ satisfying condition (ii) there exists exactly one fr-homomorphism $\alpha: T_F \to T_G$ such that $\alpha \upharpoonright T_F[1] = \varphi$, where $\alpha \upharpoonright T_F[1]$ denotes the restriction of α to $T_F[1]$.

If we take two types F and G then an fr-homomorphism not necessarily exists between T_F and T_G. For example if G consists of unary operational symbols only and in F there exists an operational symbol with arity greater than or equal to 2 then, obviously, there is no fr-homomorphism between T_F and T_G.

Consider the type F and denote by $S(F)$ the following set of nonnegative integers

$$S(F) = \{n \mid \exists f \in F \text{ with } r(f) = n\}.$$

The set $\{0, \dots, m-1\}$ will be denoted by $\tilde{m}$ for all natural number m.

Theorem 1. Let F and G be types, $S(F) = \{n_0, \dots, n_{r-1}\}$ and $S(G) = \{m_0, \dots, m_{s-1}\}$. If there exists an fr-homomorphism α: $T_F \to T_G$, then for a suitable mapping φ: $\tilde{r} \to \tilde{s}$ we have

$$(m_0-1, \dots, m_{s-1}-1) \mid (n_0 - m_{0\varphi}, \dots, n_{r-1} - m_{(r-1)\varphi}). \tag{1}$$

Proof. Let α: $T_F \to T_G$ be an fr-homomorphism. If G has an operational symbol with arity zero, then (1) holds for every mapping φ: $\tilde{r} \to \tilde{s}$ because of

$$(m_0-1, \dots, -1, \dots, m_{s-1}-1) = 1.$$

In the opposite case take a $p \in T_F$ of depth 1, and let $q = \alpha(p)$. Then

$$|fr(q)| = |fr(p)| = n_k \tag{2}$$

for some $k \in \tilde{r}$. Now consider the tree Q belonging to q. In consequence of (2) Q has n_k leaves. Delete in Q all leaves belonging to a given subtree of Q with depth 1. We get a tree with $n_k - (m_{i_1} - 1)$ leaves, where $i_1 \in \tilde{s}$. Continue the deletion of the leaves of the subtrees from Q with depth 1 as long as we get a tree of depth 1. At each step the number of leaves of the current tree was reduced by $(m_{i_v} - 1)$ for some $i_v \in \tilde{s}$. At the end of the process, the tree of depth 1 must have m_j leaves, where $j \in \tilde{s}$. In this way for suitable nonnegative integers $l_0, \dots, l_{s-1}$ we have

$$n_k - l_0(m_0-1) - \dots - l_{s-1}(m_{s-1}-1) = m_j. \tag{3}$$

Hence

$$n_k - m_j = l_0(m_0-1) + \dots + l_{s-1}(m_{s-1}-1). \tag{4}$$

Let d be the greatest common divisor of $m_0-1, \dots, m_{s-1}-1$. Then d divides the right side of (4). Therefore d divides $n_k - m_j$ as well.

Take the correspondence $k \to j$, and denote it by ψ

$$k\psi = j. \tag{5}$$

Since $p \in T_F[1]$ was arbitrary, while it runs over the set $T_F[1]$ in (5), thus k must run (not necessarily once) over the numbers $0, \dots, r-1$ and, meanwhile, for every $k \in \tilde{r}$, $k\psi$ assigns a subset of $\tilde{s}$. Let φ be a choice function of the system of sets $\{k\psi \mid k \in \tilde{r}\}$. Because of (4), $n_k - m_{k\varphi}$ can be divided by d for every $k \in \tilde{r}$. Therefore, d divides their greatest common divisor, as we stated. □

Unfortunately, condition (1) is not sufficient. Indeed, let F consist of a single unary operational symbol and let $G=\{g\}$ with $r(g)=2$. It is clear that condition (1) holds, but that in T_G there is no tree with a single leaf.

Theorem 2. Using the notations of the previous theorem, the necessary and sufficient condition of the existence of an fr-homomorphism between T_F and T_G is the validity of the following equalities

$$n_k = m_{k\varphi}+l_0(m_0-1)+\ldots+l_{s-1}(m_{s-1}-1) \quad (k=0, \ldots, r-1), \tag{6}$$

where φ is a mapping of $\tilde{r}$ into $\tilde{s}$ and l_i are nonnegative integers for $i=0, \ldots, s-1$.

Proof. The necessity of conditions is trivial by the proof of the previous theorem.

Before we are going to prove the sufficiency let us note, that if a natural number n is of the form

$$n = m_i+y_0(m_0-1)+\ldots+y_{s-1}(m_{s-1}-1),$$

where $i\in\tilde{s}$ and $y_0, \ldots, y_{s-1}$ are nonnegative integers, then there exists a q in T_G such that

$$fr(q) = x_0\ldots x_{n-1}.$$

We proof this statement by induction on s. For $s=1$,

$$n = m_0+y_0(m_0-1).$$

If $g\in G$ with $r(g)=m_0$, then the polynomial symbol

$$g(\ldots g(g(x_0, \ldots, x_{m_0-1}), x_{m_0}, \ldots, x_{2m_0-1}), \ldots, x_{n-1})$$

is appropriate. Remark, that this choice is possible since $n>0$ implies $m_0>0$.

Now assume that our statement has been shown for $s=v$, i.e. for each natural number n' of the form

$$n' = m_i+y_0(m_0-1)+\ldots+y_{v-1}(m_{v-1}-1)$$

there exists the desired q' in $T_{G'}$ and let

$$n = m_i+y_0(m_0-1)+\ldots+y_v(m_v-1) = n'+y_v(m_v-1).$$

We distinguish three cases. If $m_v>1$, then we can choose for q the polynomial symbol

$$g(\ldots g(q', x_{n'}, \ldots, x_{n'+m_v-1})\ldots x_{n-1}),$$

where $g\in G$ and $r(g)=m_v$. If $m_v=1$ then $n=n'$ and, therefore, q' itself is suitable. Finally, if $m_v=0$ and h is 0-ary operational symbol in G, then let q be the polynomial symbol which can be obtained from q' by replacing the variables $x_n, x_{n+1}, \ldots, x_{n'-1}$ by h.

Now assume that conditions (6) hold for the types F and G. In order to show the sufficiency of our conditions it is enough to define a mapping α: $T_F[1]\to T_G$ with $fr(\alpha(p))=fr(p)$ for every $p\in T_F[1]$. If in F there is no 0-ary operational symbol then for $f(x_{j_0}, \ldots, x_{j_{n_k-1}})$ let

$$\alpha(f(x_{j_0}, \ldots, x_{j_{n_k-1}})) = q(x_{j_0}, \ldots, x_{j_{n_k-1}}),$$

where $q \in T_G$ the polynomial symbol with $fr(q)=x_0 \dots x_{n_k-1}$, whose existence was shown above. In the opposite case in G there must be a 0-ary operational symbol as well, say h. For every $f \in F^0$ let $\alpha(f)=h$. Furthermore if $f \in F \setminus F^0$ is of arity n_k $(k \in \tilde{r})$ and $q \in T_G$ is the polynomial symbol with $fr(q)=x_0 \dots x_{n_k-1}$, then for $f(y_{j_0}, \dots, y_{j_{n_k-1}}) \in T_F[1]$ let

$$\alpha(f(y_{j_0}, \dots, y_{j_{n_k-1}})) = q(z_{j_0}, \dots, z_{j_{n_k-1}}),$$

where

$$z_{j_i} = \begin{cases} y_{j_i} & \text{if} \quad y_{j_i} \in X, \\ h & \text{if} \quad y_{j_i} \in F^0 \quad (i = 0, \dots, n_k-1). \end{cases}$$

Theorem 2 provides two necessary and sufficient conditions for the existence of fr-embedding $\alpha: T_F \to T_G$ for every type F.

Theorem 3. The following three conditions are equivalent:
1) for every type F there exists an fr-homomorphism of T_F into T_G,
2) in G there exist a 0-ary and an at least binary operational symbols,
3) for every type F there exists an fr-embedding T_F into T_G.

Proof. Because of the previous theorem, 1) is equivalent to 2), and it is clear that 3) implies 2). Therefore, it is enough to prove the implication 2)⇒3).

For this let $g, h \in G$ with $r(h)=0$ and $r(g) \geqq 2$. Consider an arbitrary type F and take a one-to-one mapping γ of F into T_G, for which

$$|fr(\gamma(f))| = r(f)$$

holds for every $f \in F$. Now we define the mapping β: $T_F[1] \to T_G$ in the following manner:
1) $\beta(x_i)=x_i$
2) $\beta(f)=g(h, \dots, h, \gamma(f))$ if $f \in F^0$,
3) $\beta(f(y_{i_0}, \dots, y_{i_{n-1}})) = g(h, \dots, h, \gamma(f)(\beta(y_{i_0}), \dots, \beta(y_{i_{n-1}})))$, where $y_{i_j} \in X \cup F^0$ $(j=0, \dots, n-1)$ and $f \in F \setminus F^0$.

Obviously β is one-to-one. Moreover, for every $p \in T_F[1]$ we have

$$fr(\beta(p)) = fr(p).$$

Assume that $F=\{f_0, \dots, f_{k-1}\}$ and take the following unary polynomial symbols from T_G

$$q_0 = g(x_0, h, \dots, h)$$
$$q_j = g(q_{j-1}, h, \dots, h) \quad (j = 0, \dots, k-1).$$

Finally, let us denote by α' the mapping of $T_F[1]$ into T_G for which

$$\alpha'(p) = q_j(\beta(p)),$$

where $p=f_j(p_0, \dots, p_{n-1}) \in T_F[1]$. Obviously, α' can be extended to an fr-homomorphism $\alpha: T_F \to T_G$. We claim that α is an fr-embedding. Indeed, assume that for the polynomial symbols p and q in T_F, $\alpha(p)=\alpha(q)$. We proceed by induction on the depth of p.

If $p=x_i$ then $\alpha(p)=x_i$. Moreover,

$$0 = d(x_i) = d(\alpha(p)) = d(\alpha(q)) \geqq d(q) \geqq 0$$

implies $d(q) = 0$, which yields $q = x_i$. If $p = f_j \in F^0$ then $d(p) = \alpha(f_j) = q_j(g(h, \ldots, h, \gamma(f_j)))$. Assume that q has the form $f_k(t_0, \ldots, t_{m-1})$. Then

$$\alpha(q) = \alpha(f_k(t_0, \ldots, t_{m-1})) = q_k(g(h, \ldots, h, \gamma(f_k)(\alpha(t_0), \ldots, \alpha(t_{m-1})))).$$

This and the assumption $\alpha(p)=\alpha(q)$ jointly imply

$$q_j(g(h, \ldots, h, \gamma(f_j))) = q_k(g(h, \ldots, h, \gamma(f_k)(\alpha(t_0), \ldots, \alpha(t_{m-1})))).$$

But this yields $j=k$.

Finally, assume that $p=f_j(p_0, \ldots, p_{n-1})$ and that the statement has been shown for every p' with $d(p')<d(p)$. Let $q=f_k(q_0, \ldots, q_{m-1})$. Then $\alpha(p)=\alpha(q)$ implies

$$q_j(g(h, \ldots, h, \gamma(f_j)(\alpha(p_0), \ldots, \alpha(p_{n-1})))) = \\ = q_k(g(h, \ldots, h, \gamma(f_k)(\alpha(q_0), \ldots, \alpha(q_{m-1})))). \tag{7}$$

But this holds only if $q_j=q_k$, which is equivalent to $j=k$. Thus (7) yields that $\alpha(p_i)=\alpha(q_i)$ $(i=0, \ldots, k-1)$, which makes the proof complete.

3. *Fr*-representation

Let F be a finite type and $\mathfrak{A}=\langle A, F\rangle$ a finite F-algebra (for terminology, see [3] and/or [1]). The triple $\overline{\mathfrak{A}}=(\mathfrak{A}, \underline{a}, A')$ is called an *n-ary tree automaton over F*, or shortly *n-ary F-automaton*, where $A' \subseteq A$ is the *set of final states* and $\underline{a} \in A^n$ is the *initial vector*.

According to the terminology used in the theory of tree automata the polynomial symbols over F and the subsets of T_F will be called *F-trees* and *F-forests*, respectively.

Consider the n-ary F-automaton $\overline{\mathfrak{A}}=(\mathfrak{A}, \underline{a}, A')$ and let us denote by $T(\overline{\mathfrak{A}})$ the following subset of $T_{F,n}$

$$T(\overline{\mathfrak{A}}) = \{p \mid p \in T_{F,n} \text{ and } p_{\mathfrak{A}}(\underline{a}) \in A'\}.$$

We say that the forest $T \subseteq T_{F,n}$ *can be recognized* by $\overline{\mathfrak{A}}$ (or $\overline{\mathfrak{A}}$ *represents the forest T*) if $T=T(\overline{\mathfrak{A}})$.

Let $T_1, T_2 \subseteq T_{F,n}$ and $0 \leqq i \leqq n-1$. The *x_i-product* of T_1 and T_2 is the forest which can be obtained by replacing every occurence of x_i of some tree from T_2 by a tree in T_1. We denote the x_i-product of T_1 and T_2 by $T_1 x_i T_2$. Let $T^{0,i}=\{x_i\}$ and $T^{k,i}=T^{k-1,i} \cup T^{k-1,i} x_i T$ $(k=1, 2, \ldots)$. Finally, let us denote by $T^{*,i}$ the union of all forests $T^{k,i}$:

$$T^{*,i} = \bigcup_{k=0}^{\infty} T^{k,i}.$$

$T^{*,i}$ is called the *x_i-iteration* of the forest T.

We say that the forest $T \subseteq T_{F,n}$ is *m-regular* if it can be obtained from finitely many trees of $T_{F,m}$ by finitely many applications of union, x_i-product and x_i-iteration. A forest T is called regular if it is m-regular for some m.

It is well known that a forest is regular if and only if it can be recognized by a tree automaton [1].

Take a forest $T \subseteq T_{F,n}$ and an n-ary G-automaton $\overline{\mathfrak{A}}=(\mathfrak{A}, \underline{a}, A')$. We say that $\overline{\mathfrak{A}}$ *fr-represents* the forest T (or T *can be fr-recognized* by $\overline{\mathfrak{A}}$) if there exists an *fr*-embedding $\alpha\colon T_F \to T_G$ such that $\alpha(T)=T(\overline{\mathfrak{A}})$.

Theorem 4. A forest is regular if and only if it can be *fr*-recognized by a tree automaton.

Proof. We shall show that the image and the complete inverse image of a regular forest under an *fr*-homomorphism are regular as well. This yields for us the sufficience of our conditions. The necessity is trivial.

Let $\alpha\colon T_F \to T_G$ be *fr*-homomorphism. From the definition of union and x_i-product of forests immediately follows that for each $T_1, T_2 \subseteq T_{F,n}$ we have

$$\alpha(T_1 \cup T_2) = \alpha(T_1) \cup \alpha(T_2), \tag{8}$$

$$\alpha(T_1 x_i T_2) = \alpha(T_1) x_i \alpha(T_2). \tag{9}$$

After this by induction on k it is easy to show that

$$\alpha(T_1^{k,i}) = \alpha(T_1)^{k,i} \quad (k = 0, 1, \ldots).$$

From this we get

$$\alpha(T_1^{*,i}) = \alpha\left(\bigcup_{k=0}^{\infty} T_1^{k,i}\right) = \bigcup_{k=0}^{\infty} \alpha(T_1^{k,i}) = \bigcup_{k=0}^{\infty} \alpha(T_1)^{k,i} = \alpha(T_1)^{*,i}. \tag{10}$$

Consider now the regular forest $T \subseteq T_F$, and assume that it can be obtained from the trees $p_0, \ldots, p_{k-1} \in T_F$ by finitely many application of regular operations (union, x_i-product, x_i-iteration). Because of (8)—(10), $\alpha(T)$ must be obtained from $\alpha(p_0), \ldots, \alpha(p_{k-1})$ by finitely many applications of the regular operations, namely in exactly such a manner as T is built up from $p_0, \ldots, p_{k-1}$. Therefore, $\alpha(T)$ is regular as well.

Now take two forests $T \subseteq T_{G,n}$ and $T' \subseteq T_{F,n}$, and assume that $T'=\alpha^{-1}(T)$ and that T is regular. Then for some n-ary G-automaton $\overline{\mathfrak{A}}$, $T=T(\overline{\mathfrak{A}})$. Take the F-algebra $\mathfrak{B}=\langle B, F\rangle$ such that $B=A$ and for every $f\in F$, $f_{\mathfrak{B}}=\alpha(f)_{\mathfrak{A}}$. Moreover consider the n-ary F-automaton $\overline{\mathfrak{B}}=(\mathfrak{B}, \underline{a}, A')$. We claim that $T(\overline{\mathfrak{B}})=T'$. Indeed for every $p\in T_{F,m}$, $p\in T(\overline{\mathfrak{B}})$ if and only if $p_{\mathfrak{B}}(\underline{a})\in A'$. But $p_{\mathfrak{B}}(\underline{a})=\alpha(p)_{\mathfrak{A}}\ (\underline{a})\in A'$ is equivalent to $\alpha(p)\in T(=T(\overline{\mathfrak{A}}))$. Finally, $\alpha(p)\in T$ if and only if $p\in\alpha^{-1}(T)\ (=T')$. The proof is complete.

Let K be a class of G-algebras. We say that K is *fr-complete*, if for every regular forest T (not necessarily over the type G) there exists a finite algebra $\mathfrak{A}=\langle A, F\rangle$ in K, an $\underline{a}\in A^n$ and $A'\subseteq A$ such that the tree automaton $\overline{\mathfrak{A}}=(\mathfrak{A}, \underline{a}, A')$ *fr*-represents the forest T.

Our aim is to prove the existence of a nontrivial *fr*-complete variety. In order to show this, take the type G in which there exist two operational symbols g and h with $r(g)\geqq 2$ and $r(h)=0$. Furthermore consider the equation

$$g(h, \ldots, h, g(h, \ldots, h)) = g(h, \ldots, h, g(h, \ldots, h), g(h, \ldots, h)). \tag{11}$$

Theorem 5. The variety defined by the equation (11) is *fr*-complete.

Proof. Let $\alpha\colon T_G \to T_G$ be an *fr*-homomorphism such that:

1) $\alpha(h)=g(g(h, \dots, h), h, \dots, h)$,

2) $\alpha(g(x_{i_0}, \dots, x_{i_{m-1}})) = g(g(x_{i_0}, \dots, x_{i_{m-1}}), h, \dots, h)$,

3) on the set of all other polynomial symbols of T_G with depth less than or equal to 1 α is the identity mapping.

We claim that α is *fr*-embedding. Indeed, let $\alpha(p)=\alpha(q)$. If $p=x_i$ then obviously q must be equal to x_i. If $p=h$ then because of $\alpha(h)=g(g(h, \dots, h), h, \dots, h)$, q does not contain any operational symbols different from g and h. Therefore, if $d(q)\geqq 1$, then q must have the form $g(p_0, \dots, p_{m-1})$. In this way from

$$g(g(h, \dots, h), h, \dots, h) = g(g(\alpha(p_0), \dots, \alpha(p_{m-1})), h, \dots, h)$$

it follows that $h=\alpha(p_0)$ which is a contradiction. Therefore, $d(q)=0$ and thus q must be equal to h. Finally, if p is 0-ary operational symbol different from h then $p=q$ obviously holds.

Now assume that $d(p)\geqq 1$ and that our statement has been shown for every polynomial symbol with depth less than that of p. Moreover, let $p=g_1(p_0, \dots, p_{k-1})$ and $q=g_2(q_0, \dots, q_{l-1})$. Then

$$\alpha(g_1)(\alpha(p_0), \dots, \alpha(p_{k-1})) = \alpha(g_2)(\alpha(q_0), \dots, \alpha(q_{l-1})) \tag{12}$$

yields that $\alpha(g_1)$ and $\alpha(g_2)$ must begin with the same operational symbol, but this is possible only if $g_1=g_2$. Therefore, from (12) we get that $k=l$ and $\alpha(p_i)=\alpha(q_i)$ $(i=0, \dots, k-1)$. According to our induction hypothesis, this yields that $p=q$.

Now take an arbitrary type F and an *fr*-embedding $\beta\colon T_F \to T_G$. Then $\gamma=\alpha\beta$ is an *fr*-embedding of T_F into T_G as well. For the sake of simplicity introduce the notations

$$t_1 = g(h, \dots, h, g(h, \dots, h))$$

and

$$t_2 = g(h, \dots, h, g(h, \dots, h), g(h, \dots, h)).$$

Then

$$\operatorname{sub}(t_i)\cap\gamma(T_F) = \emptyset \quad (i = 1, 2). \tag{13}$$

Moreover, for every $p\in\gamma(T_F)$

$$t_i \notin \operatorname{sub}(p) \quad (i = 1, 2). \tag{14}$$

Let $T\subseteqq T_{F,n}$ be a regular forest which can be obtained from the trees $p_0, \dots, p_{k-1}\in T_{F,m}$ by finitely many applications of regular operations. According to (14), $\gamma(p_0), \dots, \gamma(p_{k-1})$ can be represented by the m-ary G-automata $\mathfrak{A}_0, \dots, \mathfrak{A}_{k-1}$ such that on the algebras $\mathfrak{A}_0, \dots, \mathfrak{A}_{k-1}$ the equation $t_1=t_2$ holds ([1] lemma 2).

Note that the power set of $\gamma(T_F)$ is closed under the regular operations, that is if $T_1, T_2\subseteqq(T_F)$ then $T_1\cup T_2$, $T_1x_iT_2$ and $T_1^{*,i}\subseteqq(T_F)$ as well. Indeed,

$$T_1\cup T_2 = \gamma(\bar\gamma^1(T_1)\cup\bar\gamma^1(T_2)) \subseteqq \gamma(T_F), \tag{15}$$

$$T_1x_iT_2 = \gamma(\bar\gamma^1(T_1)x_i\bar\gamma^1(T_2)) \subseteqq \gamma(T_F), \tag{16}$$

$$T_1^{*,i} = \gamma(\bar\gamma^1(T_1)^{*,i}) \subseteqq \gamma(T_F). \tag{17}$$

Therefore, for every forest $T' \subseteq T_G$ which can be obtained from $\gamma(p_0), \dots, \gamma(p_{k-1})$ by finitely many applications of regular operations we have

$$\operatorname{sub}(t_i) \cap T' = \emptyset \quad (i = 1, 2).$$

By lemmas 3, 4 and 5 of [1] if for the forests T_1 and T_2

1) $\operatorname{sub}(t_i) \cap T_j = \emptyset$ $(i, j = 1, 2)$, and

2) T_1 and T_2 can be recognized by the tree automata $\overline{\mathfrak{A}}_1$ and $\overline{\mathfrak{A}}_2$, respectively, such that on the algebras $\mathfrak{A}_1$ and $\mathfrak{A}_2$ $t_1 = t_2$ holds,

then the forests $T_1 \cup T_2$, $T_1 x_i T_2$ and $T_1^{*,i}$ can be represented by the tree automata $\mathfrak{B}_1$, $\mathfrak{B}_2$ and $\mathfrak{B}_3$, respectively, such that on the algebras $\mathfrak{B}_i$ $(i = 1, 2, 3)$ $t_1 = t_2$ holds as well.

From this and from statements (14)—(17) we get, that every forest which can be obtained from $\gamma(p_0), \dots, \gamma(p_{k-1})$ by finitely many applications of regular operations (among them $\gamma(T)$) can be represented by a G-automaton belonging to the variety defined by the equation (11). This ends the proof of our theorem.

From the above theorem we can see that the existence of a 0-ary and an at least binary operational symbols in the type G is sufficient for the existence of a proper fr-complete variety. But, by Theorem 3 it is necessary as well. Therefore, the simplest types over which there exist fr-complete varieties are those which consist of exatly one 0-ary and one at least binary operational symbols.

By the languages over the alphabet $X = \{x_0, \dots, x_{n-1}\}$ accepted by an n-ary F-automaton $\overline{\mathfrak{A}}$ we mean

$$L(\overline{\mathfrak{A}}) = \{fr(p) \mid p \in T(\overline{\mathfrak{A}})\}.$$

In [2] it was shown by F. Gécseg and G. Horváth that there exists a proper variety over the type $G = \{g, h\}$ with $r(g) = 2$ and $r(h) = 0$ such that every context-free language can be accepted by a finite tree automaton belonging to this variety. This result directly follows from Theorem 5.

4. *Fr*-equivalence of tree automata

In [1] F. Gécseg introduced the concept of *rational equivalence* of tree automata. Namely, two tree automata $\overline{\mathfrak{A}}$ and $\overline{\mathfrak{B}}$ (not necessarily of the same type) are called rationally equivalent if for every forest T, T can be rationally represented by $\overline{\mathfrak{A}}$ if and only if T can be rationally represented by $\overline{\mathfrak{B}}$. Now we define the analogous concept for fr-representation. We call two tree automata $\overline{\mathfrak{A}}$ and $\overline{\mathfrak{B}}$ *fr-equivalent* if the class of forests fr-representable by $\overline{\mathfrak{A}}$ is equal to the class of all those forests, which can be fr-represented by $\overline{\mathfrak{B}}$.

One can naturally raise the following questions:

1) Is the rational equivalence of tree automata decidable? In other words, does there exist an algorithm to decide for arbitrary two tree automata whether they are rationally equivalent or not?

2) Is the fr-equivalence of tree automata decidable?

In this section we give positive answers to each of these questions.

We shall need the following two simple lemmas.

Lemma 1. Let $\alpha\colon T_F \to T_F$ be an *fr*-embedding and assume that there exists a forest $T \subseteq T_F$ such that $\alpha(T)=T$. Then for each $p \in T$ we have

$$d(\alpha(p)) = d(p).$$

Proof. For every natural number n let

$$T_n = \{p \mid p \in T \text{ and } d(p) = n\}.$$

We shaw that for every n, $\alpha(T_n)=T_n$. Indeed, let n_0 be the least natural number with $T_{n_0} \neq \emptyset$. If $q \in T_{n_0}$ then $\alpha^{-1}(q) \in T$ and $d(\alpha^{-1}(q)) \leqq n_0$ which implies that $\alpha^{-1}(q) \in T_{n_0}$. Therefore, $\alpha^{-1}(T_{n_0}) \subseteq T_{n_0}$. But α^{-1} is one-to-one and T_{n_0} is finite. Thus the restriction of α^{-1} to T_{n_0} is onto, i.e., $\alpha^{-1}(T_{n_0})=T_{n_0}$. Hence $\alpha(T_{n_0})=T_{n_0}$. Now take an arbitrary natural number n such that $T_n \neq \emptyset$ and assume that for every $m<n$, $\alpha(T_m)=T_m$. For each $q \in T_n$ we have $d(\alpha^{-1}(q)) \leqq n$. If $d(\alpha^{-1}(q))<n$ then $\alpha^{-1}(q) \in T_m$ for some $m<n$ implying $q \in T_m$, which is impossible. Therefore, $d(\alpha^{-1}(q))=n$, or equivalently $\alpha^{-1}(q) \in T_n$. Finally, again from the finiteness of T_n we get that $\alpha(T_n)=T_n$. □

Consider the types F and G. We call the mapping γ of F onto G a *projection* if γ preserves arity. If we have an *fr*-homomorphism $\alpha\colon T_F \to T_G$ such that

1) for every $f \in F$, $d(\alpha(f))=1$,

2) for every $f \in F$, $\alpha(f)$ has exatly $r(f)$ leaves,

3) for every $g \in G$, $g(x_0, \ldots, x_{r(g)-1}) \in \alpha(T_F)$, then we can take the projection $\gamma\colon F \to G$ for which $\gamma(f)=g$ if and only if $\alpha(f(x_0, \ldots, x_{r(f)-1}))=g(x_0, \ldots, x_{r(f)-1})$. For this we use the notation $\gamma=\alpha{\restriction}F$.

The next result is obvious.

Lemma 2. Take three *fr*-embeddings $\alpha\colon T_F \to T_G$, $\beta\colon T_G \to T_H$ and $\gamma\colon T_F \to T_H$ such that $\gamma=\beta\alpha$. Then $\gamma{\restriction}F$ is a projection if and only if $\alpha{\restriction}F$ and $\beta{\restriction}G$ are projections as well.

Consider an F-automaton $\mathfrak{A}$ and a G-automaton $\mathfrak{B}$. We say that $\mathfrak{A}$ and $\mathfrak{B}$ are *equivalent up to the notation of their operational symbols* if there exists a one-to-one projection γ of F onto G such that $\gamma(T(\mathfrak{A}))=T(\mathfrak{B})$. Moreover, we use the terminology that *F is reduced for $\mathfrak{A}$* if for every $f \in F$ there is a tree p in $T(\mathfrak{A})$ such that f occurs in p.

Theorem 6. Take an F-automaton $\mathfrak{A}$ and a G-automaton $\mathfrak{B}$ such that F and G are reduced for $\mathfrak{A}$ and $\mathfrak{B}$, respectively. Then the following three conditions are equivalent:

1) $\mathfrak{A}$ and $\mathfrak{B}$ are rationally equivalent,

2) $\mathfrak{A}$ and $\mathfrak{B}$ are *fr*-equivalent,

3) $\mathfrak{A}$ and $\mathfrak{B}$ are equivalent up to the notation of their operational symbols.

Proof. The equivalence of 1) and 3) was proved in [1]. Furthermore, it is obvious that 3) implies 2). Thus it is enough to show that 3) follows form 2).

First we prove, that if for an *fr*-embedding $\alpha\colon T_F \to T_F$ there exists a $q \in T_F$ such that $\alpha(q)=q$, than for every operational symbol f occuring in q we have $\alpha(f)=f$. Indeed, if $d(q) \leqq 1$ then this statement is trivial. Now let $q=$

$=f(q_0, \ldots, q_{k-1})$ and assume that for every tree q' with $d(q')<d(q)$ our statement is true. From $\alpha(q)=q$ we get

$$\alpha(f)(\alpha(q_0), \ldots, \alpha(q_{k-1})) = f(q_0, \ldots, q_{k-1}).$$

But this yields that $\alpha(f)=f$ and that $\alpha(q_i)=q_i$ $(i=0, \ldots, k-1)$.

Now take an F-automaton $\mathfrak{A}$ and a G-automaton $\mathfrak{B}$ such that F and G are reduced for $\mathfrak{A}$ and $\mathfrak{B}$, respectively. Assume that $\mathfrak{A}$ and $\mathfrak{B}$ are *fr*-equivalent. Then there exist two *fr*-embeddings $\alpha\colon T_F \to T_G$ and $\beta\colon T_G \to T_F$ such that $\alpha(T(\mathfrak{A}))= =T(\mathfrak{B})$ and $\beta(T(\mathfrak{B}))=T(\mathfrak{A})$. Therefore, for the *fr*-embedding $\gamma=\beta\alpha$ we have $\gamma(T(\mathfrak{A}))=T(\mathfrak{A})$. Thus, by Lemma 1, γ preserves the depth of trees in $T(\mathfrak{A})$. For the sake of simplicity let us denote $T(\mathfrak{A})$ by T.

Consider the trees $p_0, \ldots, p_{m-1} \in T$ such that for every $f \in F$ there exists a $j \in \tilde{m}$ for which f occurs in p_j. Let $d(p_0)=n_0, \ldots, d(p_{m-1})=n_{m-1}$. Therefore, $p_j \in T_{n_j}$ $(j=0, \ldots, m-1)$. (We recall that T_{n_j} is the set of all trees from T whose depth is n_j.) Let

$$\gamma_j = \gamma \restriction T_{n_j} \quad (j = 0, \ldots, m-1).$$

Since T_{n_j} is finite and γ_j is one-to-one thus there exist natural numbers $k_0, \ldots, k_{m-1}$ such that

$$\gamma_j^{k_j} = \mathrm{id}_{T_{n_j}} \quad (j = 0, \ldots, m-1). \tag{18}$$

Take $d=k_0 \cdot \ldots \cdot k_{m-1}$. From (18) it follows that

$$\gamma^d \restriction (T_{n_0} \cup \ldots \cup T_{n_{m-1}}) = \mathrm{id}_{T_{n_0} \cup \ldots \cup T_{n_{m-1}}}.$$

Therefore, for the *fr*-embedding $\gamma^d\colon T_F \to T_F$ we have

$$\gamma^d(p_j) = p_j \quad (j = 0, \ldots, m-1).$$

Because of the choice of the trees $p_0, \ldots, p_{m-1}$ the first assertion of this proof yields that $\gamma^d \restriction F = \mathrm{id}_F$. Thus $\gamma^d \restriction F$ is a one-to-one projection of F onto F, but by Lemma 2 this is true if and only if $\gamma \restriction F$ is a projection of F onto F as well. Then Lemma 2, $\gamma=\beta\alpha$ and the fact that $\gamma \restriction F$ is a projection jointly imply that $\alpha \restriction F$ is a projection of F onto G. The proof is complete.

According to the above theorem in order to decide the rational equivalence (*fr*-equivalence) of arbitrary two tree automata $\mathfrak{A}$ and $\mathfrak{B}$ it is enough to check whether there exists a one-to-one projection γ between the types of $\mathfrak{A}$ and $\mathfrak{B}$ such that $\gamma(T(\mathfrak{A}))=T(\mathfrak{B})$. But the set of all one-to-one projections between two finite types is finite, and for a given one-to-one projection γ the equality $\gamma(T(\mathfrak{A}))=T(\mathfrak{B})$ is decidable by taking the minimal tree automata recognizing $\gamma(T(\mathfrak{A}))$ and $T(\mathfrak{B})$. Thus we have

Theorem 7. The rational equivalence and the *fr*-equivalence of tree automata are decidable.

DEPT. OF COMPUTER SCIENCE
A. JÓZSEF UNIVERSITY
H-6720 SZEGED, HUNGARY
SOMOGYI U. 7.

References

[1] GÉCSEG, F., Universal algebras and tree automata, *Proceedings of the Conference on Fundamentals of Computation Theory*, Poznan-Kórnik v. 56, 1977, p. 89.
[2] GÉCSEG, F. and GY. HORVÁTH, On representation of trees of context-free languages by tree automata, *Foundations of Control Engineering*, v. 1, 1976, pp. 161—168.
[3] GRÄTZER, G., *Universal Algebras*, D. Van Nostrand Company, Princeton, New Yersey, 1968.

(Received Nov. 22, 1977)

A note on symmetric Boolean functions

By P. ECSEDI—TÓTH,* F. MÓRICZ** and A. VARGA**

To the memory of Professor GÉZA FODOR

Introduction

The notion of a symmetric function can be found in any textbook on switching theory or logical design. It is well-known (SHANNON [1]) that the truth-value of a symmetric function depends only on the number of literals for which the truth-value TRUE is substituted. More precisely, the following theorem holds.

Theorem (Shannon). *Let φ be a Boolean function of n variables. φ is a symmetric function if and only if there exists a set of integers* $\{n_1, n_2, \ldots, n_k\}$ (called the Shannon set of φ) ($k \leqq n$, $0 \leqq n_i \leqq n$ *for* $i \leqq k$) *such that the truth-value of* φ *is* TRUE *iff for exactly* n_i *of the literals* TRUE *is substituted.*

The proof of this theorem gives no idea how to determine the set $\{n_1, n_2, \ldots, n_k\}$. Since symmetric functions have nice properties, it is important to decide whether a given function φ is symmetric or not. As far as we know, there are only trivial methods (i.e., to test all possible cases) for the solution of this problem.

In this paper we present an effective algorithm to determine the Shannon set of a Boolean function if it exists. The method is based on the tree-representation of Boolean functions used by the present authors [2] to get irredundant normal-forms as representation of them. In particular, we associate a number — the number of negative literals — to each path of this tree. Then by a simple comparison of the endnodes of the paths and of the associated numbers, we can collect the Shannon set provided it exists.

1. The tree-representation of Boolean functions

To make the paper self-contained, we present here the tree-construction algorithm, too. A more detailed explanation and the basic results can be found in [2].

Let a Boolean function φ be given in which at least one variable occurs. Choose a variable of φ according to some rule (a so-called selection function), fixed previously. First, substitute the truth-values TRUE and FALSE, respectively, for the chosen variable. Then eliminate the truth-values from both expressions obtained by using the following transformation rules:

$$\varphi\wedge 1 \leftrightarrow \varphi; \qquad 1\wedge\varphi \leftrightarrow \varphi$$
$$\varphi\wedge 0 \leftrightarrow 0; \qquad 0\wedge\varphi \leftrightarrow 0$$
$$\varphi\vee 1 \leftrightarrow 1; \qquad 1\vee\varphi \leftrightarrow 1$$
$$\varphi\vee 0 \leftrightarrow \varphi; \qquad 0\vee\varphi \leftrightarrow \varphi$$
$$\varphi \rightarrow 1 \leftrightarrow 1$$
$$\varphi \rightarrow 0 \leftrightarrow \bar{\varphi}$$
$$1 \rightarrow \varphi \leftrightarrow \varphi$$
$$0 \rightarrow \varphi \leftrightarrow 1$$
$$(\varphi \leftrightarrow 1) \leftrightarrow \varphi; \qquad (1 \leftrightarrow \varphi) \leftrightarrow \varphi$$
$$(\varphi \leftrightarrow 0) \leftrightarrow \bar{\varphi}; \qquad (0 \leftrightarrow \varphi) \leftrightarrow \bar{\varphi}$$
$$\varphi\wedge\varphi \leftrightarrow \varphi$$
$$\varphi\wedge\bar{\varphi} \leftrightarrow 0; \qquad \bar{\varphi}\wedge\varphi \leftrightarrow 0$$
$$\varphi\vee\varphi \leftrightarrow \varphi$$
$$\varphi\vee\bar{\varphi} \leftrightarrow 1; \qquad \bar{\varphi}\vee\varphi \leftrightarrow 1$$
$$\varphi \rightarrow \varphi \leftrightarrow 1$$
$$(\varphi \leftrightarrow \varphi) \leftrightarrow 1$$
$$\bar{\bar{\varphi}} \leftrightarrow \varphi$$
$$\bar{1} \leftrightarrow 0$$
$$\bar{0} \leftrightarrow 1$$

As a result of the elimination process we come to one of the following two cases:

(i) The expression obtained contains at least one variable. Then let us choose a variable in it according to our rule, and repeat the substitution and the elimination described above.

(ii) The expression obtained is a single truth-value. Then the algorithm stops.

We note that the function φ, together with a selection function, determines its tree uniquely up to isomorphism, and conversely, every binary tree determines a Boolean function uniquely up to logical equivalence.

The following example illustrates the method. We use the usual logical connectives ($\wedge$ for conjunction, $\vee$ for disjunction, $\rightarrow$ for implication, $\leftrightarrow$ for equivalence, and $^{-}$ (bar) for negation). 1 and 0 will denote the truth-values TRUE and FALSE, respectively.

Example. Let the Boolean function φ be as follows:

$$\varphi\colon\ ((\overline{A \to B}) \wedge ((\overline{C} \vee B) \to \overline{A})) \leftrightarrow A.$$

Let us choose the variables alphabetically. The substituted and "simplified" expressions can be arranged in a tree as indicated in Fig. 1.

As it was proved in [2, Corollary 6], the function φ and also its sub-functions can be omitted since they are obtainable from the shape of the tree, so it is enough to draw the simpler form as indicated in Fig. 2.

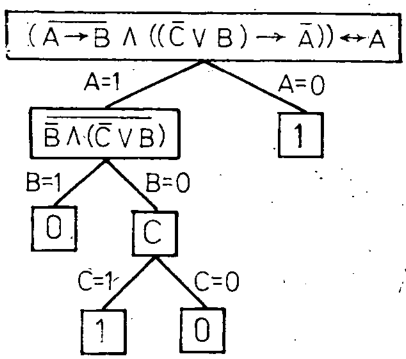

Fig. 1

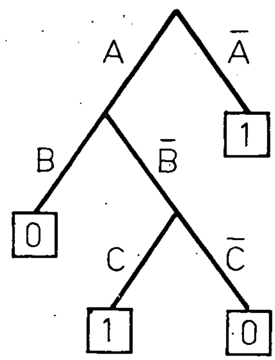

Fig. 2

The concept of a complete tree was introduced also in [2]. A tree of a Boolean function is *complete* iff all paths from the root to an endnode of the tree have the same length, which is equal to the number of the variables of φ. The reader can easily verify the following two assertions.

Lemma 1. *Let φ be a Boolean function of n variables. Then one can find a Boolean function φ' with the same variables, the tree of which is complete and φ' is logically equivalent to φ. φ' and its complete tree are uniquely determined.*

In practice, it is very easy to get a complete tree from any incomplete one as Fig. 3 shows.

Lemma 2. *Let φ be a Boolean function of n variables and suppose that the tree of φ is complete. Then there exist exactly 2^n paths in the tree of φ.*

Convention. In the rest of this paper we shall assume that every tree is drawn in such a way that the positive sub-expressions (those which can be obtained by substituting TRUE for a variable) are drawn on the left-hand side, while the negative sub-expressions are drawn on the right-hand side of the tree. Observe that trees in Fig. 1—3 correspond to this convention.

Definition. Let φ be a Boolean function of n variables. Then its *complete tree* is the tree of φ' determined by Lemma 1.

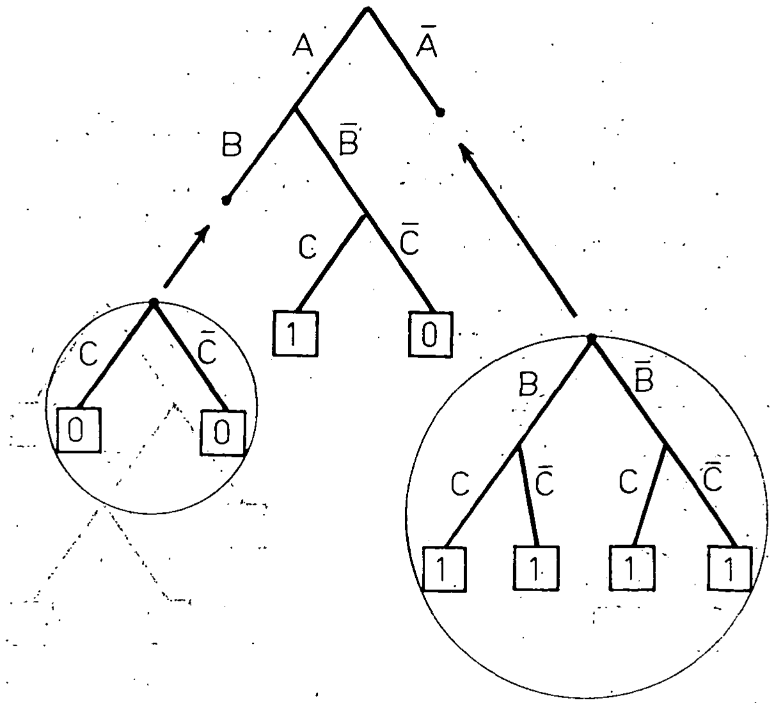

Fig. 3

2. The sequence ξ

Definition. Let us define the sequence of non-negative integers $\langle \xi_k \rangle$ by

ξ_k=the number of 1 in the binary expansion of $k-1$ $(k=1, 2, \ldots)$.

Definition. Let $\langle \zeta_k \rangle$ be defined by the following recurrence:

$$\zeta_1 = 0,$$
$$\zeta_{2k-1} = \zeta_k,$$
$$\zeta_{2k} = \zeta_k + 1.$$

Lemma 3. *We have* $\xi_k = \zeta_k$ *for every* $k=1, 2, \ldots$.

Proof. If $k=1$, then the lemma holds by definition. For every $k \geqq 2$ there exists exactly one non-negative integer n such that $2^n < k \leqq 2^{n+1}$. We proceed by induction on n.

Let n be fixed. Assume $2^n < k \leqq 2^{n+1}$ and that $l \leqq 2^n$ implies $\xi_l = \zeta_l$.

Let $k=2l-1$. Obviously, $k-1=2l-2$ is even and $l \leqq 2^n$, so

$$\zeta_k = \zeta_l = \xi_l = \xi_k.$$

Note that the last equation holds, since multiplication by 2 simple means a shifting in the binary expansion of $k-1$.

Let $k=2l$. Obviously $k-1=2l-1$ is odd and $l\leqq 2^n$, so

$$\zeta_k=\zeta_{2l}=\zeta_l+1=\xi_l+1=\xi_{2l-1}+1,$$

where again the last equation holds by the shifting property mentioned above. We have to prove that

$$\xi_{2l-1}+1=\xi_{2l}.$$

However, this readily follows by definition from the fact that $2l-1$ is odd and $2l-1=2l-2+1$.

The proof of Lemma 3 is complete.

Definition. For each non-negative integer n we define $\xi_{2^n}(k)$ by the following recurrence:

(i) $\xi_{2^0}(1)=0$,

(ii) $\xi_{2^{n+1}}(k)=\begin{cases}\xi_{2^n}(k) & \text{if } 0<k\leqq 2^n,\\ \xi_{2^n}(l) & \text{if } 2^n<k\leqq 2^{n+1} \text{ and } k=2^n+l.\end{cases}$

Lemma 4. *We have* $\zeta_k=\xi_{2^n}(k)$ *provided* $0<k\leqq 2^n$.

Proof. It is enough to prove that

$$\xi_{2^{n+1}}(2k-1)=\xi_{2^n}(k)$$

and

$$\xi_{2^{n+1}}(2k)=\xi_{2^n}(k)+1, \tag{1}$$

since if we assume that $0<k\leqq 2^n$ entails

$$\zeta_k=\xi_{2^n}(k), \tag{2}$$

then if $l=2k-1$ $(2^n<l\leqq 2^{n+1})$, then

$$\zeta_l=\zeta_{2k-1}=\zeta_k=\xi_{2^n}(k)$$

by (2); and if $l=2k$ $(2^n<l\leqq 2^{n+1})$, then

$$\zeta_l=\zeta_{2k}=\zeta_k+1=\xi_{2^n}(k)+1.$$

We prove (1) by induction on n. If $n=0$, then (1) trivially holds. If $n\neq 0$, then we prove that

$$\xi_{2^{n+2}}(2k-1)=\xi_{2^{n+1}}(k)$$

and

$$\xi_{2^{n+2}}(2k)=\xi_{2^{n+1}}(k)+1 \quad (k=1,2,\ldots,2^{n+1}).$$

In each case two subcases will be distinguished.

1) If k is odd and $2k-1\leqq 2^n$, then

$$\xi_{2^{n+2}}(2k-1)=\xi_{2^{n+1}}(2k-1)=\xi_{2^n}(k)=\xi_{2^{n+1}}(k).$$

2) If k is odd and $2k-1>2^n$, then

$$2k-1=2^n+l,$$

where $l\leqq 2^n$ and l is odd, thus

$$2k-1=2^n+2m-1.$$

We have also for $k=2^n+m$,

$$\xi_{2^{n+2}}(2k-1) = \xi_{2^{n+2}}(2^n+l) = \xi_{2^{n+1}}(l)+1 =$$
$$= \xi_{2^n}(m)+1 = \xi_{2^{n+1}}(2^{n-1}+m) = \xi_{2^{n+1}}(k).$$

3) If k is even and $2k \leqq 2^n$, then

$$\xi_{2^{n+2}}(2k) = \xi_{2^{n+1}}(2k) = \xi_{2^n}(k)+1 = \xi_{2^{n+1}}(k)+1.$$

4) If k is even and $2k > 2^n$, then

$$\xi_{2^{n+2}}(2k) = \xi_{2^{n+2}}(2^n+l) = \xi_{2^{n+1}}(l)+1 = \xi_{2^{n+1}}(2m)+1 =$$
$$= \xi_{2^n}(m)+2 = \xi_{2^{n+1}}(2^{n-1}+m)+1 = \xi_{2^{n+1}}(k)+1.$$

The proof of Lemma 4 is complete.

The sequence $\xi_{2^n}(k)$ can be easily generated, so by Lemma 3 and Lemma 4 we have a "fast" algorithm to obtain the sequence $\langle \xi_k \rangle$. The use of this sequence is shown by the following

Theorem 5. *Let φ be a Boolean function of n variables. Let us number the endnodes in its complete tree by $k=1, 2, \ldots, 2^n$ from the left to the right. Then ξ_k means the number of the negative literals in the path, the endnode of which is numbered by k.*

Proof. Denote by $n(k)$ the number of the negative literals in the path labelled by k. Actually, one can prove by induction on the number of the variables in φ that

$$n(k) = \zeta_k.$$

3. Symmetric functions

Definition. Let T be the set of indices of those paths, whose endnodes are TRUE.

Corollary 6. *Let φ be an n-ary symmetric function and let m be an arbitrary non-negative integer such that $m \leqq n$. Then m is an element of the Shannon set of φ if and only if*

$$\{k \mid n-\xi_k = m\} \subseteqq T.$$

Proof. It is quite easy by Theorem 5.

* RESEARCH GROUP ON MATHEMATICAL LOGIC AND THEORY OF AUTOMATA OF THE HUNGARIAN ACADEMY OF SCIENCES
H-6720 SZEGED, HUNGARY
SOMOGYI U. 7.

** BOLYAI INSTITUTE OF THE JÓZSEF ATTILA UNIVERSITY
H-6720 SZEGED, HUNGARY
ARADI VÉRTANÚK TERE 1.

References

[1] SHANNON, C. E., A symbolic analysis of relay and switching circuits, *Trans. A.I.E.E.*, v. 57, 1938, pp. 713—723.
[2] VARGA, A., P. ECSEDI-TÓTH and F. MÓRICZ, An effective method for minimizing Boolean polynomials, *Research Report* (Research Group on Mathematical Logic and Theory of Automata of the Hungarian Academy of Sciences), 1977, p. 40.

(Received Oct. 13, 1977)

Truth functions and problems in graph colouring

By G. LÜKŐ

To the memory of Professor LÁSZLÓ KALMÁR

Introduction

The aim of this paper is to introduce some truth functions, which seem to be useful in the theory of graph colouring, and to study their basic properties and their interrelations.

It can be hoped that a future article will contain a some more detailed analysis of these functions and some applications of the results presented now.

Theorem 3 includes (somewhat implicitly) a purely graph-theoretical assertion. In fact, a simple representation of the maximal v-critical graphs may be given: these can be produced as the intersection of N graphs each of which is the complement of a partition graph[1].

§ 1. Concepts and notations for graphs

1.1. By a graph, always a non-directed finite graph is meant without loops and parallel edges. Later the vertex set of any graph will be viewed to be labelled, a vertex will be identified with the corresponding number (except when it is emphasized explicitly that a graph is considered abstractly, i.e., apart from isomorphy).

If a natural number is denoted by a letter N, then denote by $\mathcal{N}$ (the script form of the *same* latter) the set $\{1, 2, \dots, N\}$; furthermore, we define $\mathcal{N}_i$ by

$$\mathcal{N}_i = \{1, 2, \dots, i-1, i+1, \dots, N\}$$

for an arbitrary i ($1 \leqq i \leqq N$). The letter $\mathcal{H}$ denotes an arbitrary set of natural numbers (not necessarily of form $\{1, 2, \dots, H\}$). The cardinality of a set $\mathcal{H}$ is denoted by $|\mathcal{H}|$. $\mathfrak{C}_{\mathcal{H}}$ is the complete graph with the vertex set $\mathcal{H}$. If $\mathcal{H}$ is a subset of $\mathcal{N}$, then we put $\overline{\mathcal{H}} = \mathcal{N} - \mathcal{H}$.

[1] The notions occuring here will be defined later.

For a graph $\mathfrak{G}$, $\mathscr{V}(\mathfrak{G})$ is the set of vertices of $\mathfrak{G}$ and $\Gamma(\mathfrak{G})$ is the set of edges of $\mathfrak{G}$. If the number N is fixed and $\mathscr{V}(\mathfrak{G})\subseteq\mathscr{N}$ for a graph $\mathfrak{G}$, then we denote the complement of $\mathfrak{G}$ (with respect to $C_{\mathscr{N}}$) by $\overline{\mathfrak{G}}$.

The isomorphy of graphs is denoted by $\approx$. The sign $\subseteq$ can express both subset and subgraph; we write $\subset$ if the inclusion is proper. $\varkappa(\mathfrak{G})$ is the chromatic number of $\mathfrak{G}$.

A graph $\mathfrak{G}$ is called *partition graph* if each connected component of $\mathfrak{G}$ is complete.

Let us fix the set $\mathscr{H}$. By $\mathscr{P}^c_{\mathscr{H}}$ the set of all partition graphs $\mathfrak{G}$ is meant such that $\mathscr{V}(\mathfrak{G})=\mathscr{H}$ and the number of connected components of $\mathfrak{G}$ is c.

1.2. Let $\Lambda=\|\lambda_{ij}\|$ be a symmetric matrix of size $N\times N$ such that the entries of Λ are truth values and $\lambda_{11}=\lambda_{22}=\ldots=\lambda_{NN}=\downarrow$. Let the function Φ assign to Λ the graph $\mathfrak{G}=\Phi(\Lambda)$ with $V(\mathfrak{G})=\mathscr{N}$ such that the edge $\overline{ij}$ exists in $\mathfrak{G}$ if and only if $\lambda_{ij}=\uparrow$. Φ is obviously a one-to-one mapping and the range of Φ exhausts the set of all graphs on the vertex set V.[2]

1.3. An abstract graph $\mathfrak{G}$ is called *edge-critical* (or *e-critical)* if $\varkappa(\mathfrak{G}')<\varkappa(\mathfrak{G})$ for every $\mathfrak{G}'$ such that $\mathfrak{G}'$ results from $\mathfrak{G}$ by deleting one edge.

Analogously, $\mathfrak{G}$ is called *vertex-critical* (or *v-critical*) if $\varkappa(\mathfrak{G}')<\varkappa(\mathfrak{G})$ holds for any $\mathfrak{G}'$ such that may be obtained from $\mathfrak{G}$ by deleting one vertex (and the edges incident to it). Any e-critical graph is evidently v-critical.

A v-critical graph $\mathfrak{G}$ is called *maximal v-critical* if $\varkappa(\mathfrak{G}^*)>\varkappa(\mathfrak{G})$ holds for every choice of $\mathfrak{G}^*$ such that $\mathfrak{G}^*$ is v-critical and $\mathfrak{G}$ is a subgraph of $\mathfrak{G}^*$.

If $\mathfrak{G}$ is e-critical and $\varkappa(\mathfrak{G})=c$, then $\mathfrak{G}$ is called c-edge-critical.

Let the natural numbers c, N be fixed $(c<N)$. Denote by $\mathscr{K}^c_N$ the set of all c-edge-critical abstract graphs such that $\mathscr{V}(\mathfrak{G})\leqq\mathscr{N}$.

We get the graph class $\mathscr{V}^c_N$ or $\mathscr{M}^c_N$ in a similar manner if "edge-critical" is replaced by "vertex-critical" or "maximal vertex-critical" (respectively) in the above definition. And, moreover, if $|\mathscr{V}(\mathfrak{G})|\leqq N$ is replaced by $|\mathscr{V}(\mathfrak{G})|=N$, then the resulting graph classes are denoted by $\hat{\mathscr{K}}^c_N$, $\hat{\mathscr{V}}^c_N$ and $\hat{\mathscr{M}}^c_N$ (respectively, in analogy to how $\mathscr{K}^c_N$, $\mathscr{V}^c_N$, $\mathscr{M}^c_N$ have been defined).

§ 2. Introduction of truth functions defined on graphs

2.1. Consider a number N and the vertex set $\mathscr{N}$, let a graph $\mathfrak{G}_0$ be fixed with $V(\mathfrak{G}_0)=\mathscr{N}$. Define a truth function $\chi_{\mathfrak{G}_0}[\Lambda]$ by

$$\chi_{\mathfrak{G}_0}[\Lambda]=\bigwedge_{\lambda^0_{ij}=\uparrow}\lambda_{ij} \tag{2.1}$$

where

Λ is a symmetric matrix of size $N\times N$ (as in Section 1.2.),[3]
the variables of Λ are the entries λ_{ij} of Λ fulfilling $i<j$,

[2] Cf. the first sentence of 1.1. $\Phi(\Lambda)$ can be viewed as a non-directed graph because of the symmetry of Λ.

[3] Hence $\Phi(\Lambda)$ is a graph whose vertex set is $\mathscr{N}$.

on the right-hand side of (2.1) the conjunction is taken for all pairs (i, j) such that $\lambda_{ij}^{0} = \uparrow$ where λ_{ij}^{0} is the entry of $\Phi^{-1}(\mathfrak{G}_0)$ being in crossing of the i-th row and j-th column.

An obvious consequence of the above definition is:

Proposition 1. *The value $\chi_{\mathfrak{G}_0}[\Lambda]$ is $\uparrow$ if and only if any edge of $\mathfrak{G}_0$ is an edge of $\Phi(\Lambda)$, too.*

2.2. Let c be a natural number $(c<N)$. In analogy to the above definition of $\chi_{\mathfrak{G}_0}$, we define the truth function D^c by

$$D^c[\Lambda] = \bigvee_{\mathfrak{G}^* \in \mathscr{P}_{\mathscr{N}}^{c}} \chi_{\mathfrak{G}^*}[\Lambda]$$

where the disjunction is taken for all elements $\mathfrak{G}^*$ of the set $\mathscr{P}_{\mathscr{N}}^{c}$. The meaning of D^c is expressed in the following evident assertion:

Proposition 2. *The following three statements are equivalent for any matrix Λ:*
(i) $D^c[\Lambda] = \uparrow$,
(ii) $\mathfrak{G} = \Phi(\Lambda)$ *contains a partition graph consisting of c connected components,*
(iii) *the complement of $\Phi(\Lambda)$ is c-colourable (i.e., $\varkappa(\overline{\mathfrak{G}}) \leqq c$).*

2.3. In the particular case when $\mathfrak{G}_0$ has only one edge e, the function $\chi_{G_0}[\Lambda]$ expresses whether this edge e is present in $\Phi(\Lambda)$ or not. In this special case we write also $\chi_e[\Lambda]$.

Let an abstract graph $\mathfrak{K}$ with at most N vertices be chosen. Define the function $L_{\mathfrak{K}}$ by

$$L_{\mathfrak{K}}[\Lambda] = \bigwedge_{\mathfrak{K}'} \bigvee_{e} \chi_e[\Lambda]$$

where $\mathfrak{K}'$ runs through all graphs such that
$\mathscr{V}(\mathfrak{K}') \subseteqq \mathscr{N}$ and
$\mathfrak{K}'$ is isomorphic to $\mathfrak{K}$;
for any choice of $\mathfrak{K}'$, e runs through the edges of $\mathfrak{K}'$.

The next result follows easily from this definition:

Proposition 3. $L_{\mathfrak{K}}[\Lambda] = \uparrow$ *if and only if no subgraph of the complement of $\Phi(\Lambda)$ is isomorphic to $\mathfrak{K}$.*

2.4. Let the functions E^c and F^c be defined by

$$E^c[\Lambda] = \bigwedge_{\mathfrak{K} \in \mathscr{K}_{N-1}^{c+1}} L_{\mathfrak{K}}[\Lambda]$$

and

$$F^c[\Lambda] = \bigwedge_{\mathfrak{K} \in \hat{\mathscr{K}}_{N}^{c+1}} L_{\mathfrak{K}}[\Lambda]$$

The following two assertions follow easily from these definitions and from Proposition 3.

Proposition 4. $E^c[\Lambda] = \uparrow$ *if and only if the complement of $\Phi(\Lambda)$ has no $(c+1)$-edge-critical subgraph with at most $N-1$ vertices.*

Proposition 5. *$F^c[A]=\uparrow$ if and only if the complement of $\Phi(A)$ has no $(c+1)$ edge-critical subgraph containing each of the N vertices.*

Proposition 6. *The equality*

$$D^c[A] = E^c[A] \wedge F^c[A]$$

holds for any matrix A.

Proof. Let us consider four assertions:

(i) $E^c[A] \wedge F^c[A]=\uparrow$,

(ii) the complement of $\Phi(A)$ has no $(c+1)$-edge-critical subgraph,

(iii) the complement of $\Phi(A)$ is c-colourable,

(iv) $D^c[A]=\uparrow$.

Propositions 4, 5 imply the equivalence of (i) and (ii). Proposition 2 has stated that (iii), (iv) are equivalent. If (ii) is false then $\varkappa(\overline{\Phi(A)})>c$, this means the falsity of (iii). As it was shown in [2], the falsity of (iii) implies the falsity of (ii).

2.5. We mention some obvious consequences of the definitions occuring in this §. $\chi_{\mathfrak{G}_0}$ is an elementary conjunction. D^c was defined in a disjunctive normal form. Each of $L_{\mathfrak{R}}$, E^c, F^c was introduced as the conjunction of functions expressed in disjunctive normal form. All these functions are isotonic.

In what follows we shall write e.g. $D^c_{\mathcal{N}}$ instead of D^c if we want to emphasize that graphs with the vertex set $\mathcal{N}$ are considered.

§ 3. Results

The most important interrelation concerning the defined truth functions is expressed by

Theorem 1. *For any matrix A we have*

$$E^c_{\mathcal{N}}[A] = \bigwedge_{i=1}^{N} D^c_{\mathcal{N}_i}[A].$$

From Theorem 1 we shall infer to

Theorem 2. *There is exactly one truth function $A^c_{\mathcal{N}}[A]$ such that*

(i) $A^c_{\mathcal{N}}[A]$ is isotonic

(ii) any matrix A fulfils the equality $E^c_{\mathcal{N}}[A]=D^c_{\mathcal{N}}[A] \vee A^c_{\mathcal{N}}[A]$,

and

(iii) $A^c_{\mathcal{N}}[A]$ and $D^c_{\mathcal{N}}[A]$ have no prime implicant in common.

Remark. $A^c_{\mathcal{N}}$ is identically true if and only if

$$\mathfrak{G} \in \mathcal{K}^c_N \Rightarrow |\mathcal{V}(\mathfrak{G})| \leqq N-1.$$

In the next assertion $A^c_{\mathcal{N}}$ is characterized by means of vertex-critical graphs.

Theorem 3. *Suppose that the numbers N, c are such that there is a $(c+1)$-v-critical graph with N vertices. $\chi_0=\chi_{\mathfrak{G}_0}[A]$ is a prime implicant of $A^c_{\mathcal{N}}[A]$ if and only if*

(a) $\overline{\mathfrak{G}}_0 \in \mathcal{M}^{\hat{c}+1}_N$ and

(b) $\mathcal{V}(\overline{\mathfrak{G}}_0) = \mathcal{N}$.

We are able to give for a disjunctive normal form of F^c a characterization which is somewhat less explicit in comparison to how E^c has been characterized in Theorem 2.

Theorem 4. *Let N, c be numbers as in Theorem 3. $\chi_0=\chi_{\mathfrak{G}_0}[\Lambda]$ is a prime implicant of $F^c_{\mathcal{N}}[\Lambda]$ if and only if*

(a) *$\overline{\mathfrak{G}}_0$ has no subgraph $\mathfrak{G}_1$ such that all the vertices of $\overline{\mathfrak{G}}_0$ are contained in $\mathfrak{G}_1$ and $\mathfrak{G}_1\in\hat{\mathcal{K}}^{c+1}_N$ and*

(b) *whenever $\mathfrak{G}_2$ is a subgraph of $\mathfrak{G}_0$ then there is a subgraph $\mathfrak{G}_3$ of $\overline{\mathfrak{G}}_2$ such that $\mathfrak{G}_3\in\hat{\mathcal{K}}^{c+1}_N$.*

Moreover, if χ_0' is a prime implicant of $F^c_{\mathcal{N}}[\Lambda]$, then either $\mathfrak{G}_0\in\mathcal{P}^c_{\mathcal{N}}$ or $\mathfrak{G}_0$ contains a $(c+1)$-critical graph $\mathfrak{G}_4$ such that $\mathfrak{G}_4$ has at most $N-1$ vertices.

§ 4. Proofs

We shall use the following well-known fact (see [1], p. 40):

Lemma 1. *An isotonic truth function has a single irredundant disjunctive normal form, this form consists of all its prime implicants.*

Proof of Theorem 1. Since $D^c_{\mathcal{N}}[\Lambda]$ is isotonic, we can use Lemma 1. By Proposition 6 and the definitions of E^c, F^c, we have

$$D^c_{\mathcal{N}_i}[\Lambda]=\bigwedge_{\mathfrak{K}\in\mathcal{K}^{c+1}_{\mathcal{N}_i}}L_{\mathfrak{K}}[\Lambda]$$

for any i $(1\leqq i\leqq N)$. If we form the conjunction of these N equalities (in such a manner that the conjunction of the left-hand sides and the conjunction of the right-hand sides is taken, with an equality sign between them), then the right-hand side can be simplified to $E^c_N[\Lambda]$, thus we get the assertion of Theorem 1.

Proof of Theorem 2. Let us distinguish three cases. If $N<c+1$, $\mathcal{P}^c_{\mathcal{N}}=\emptyset$ and so D^c is undefined. If $N=c+1$, then, by Proposition 4, $E^c_N\equiv\uparrow$, as there exists no $(c+1)$-edge-critical graph with at most c vertices. So $D^c[\Lambda]\equiv F^c[\Lambda]$ whence follows the existency and unicity (in the sense of the assertion) of $A^c_N[\Lambda]$, namely $A^c_N[\Lambda]\equiv\uparrow$. If $N>c+1$, the proof runs as follows.

Our first aim is to verify that each prime implicant χ_0 of $D^c_N[\Lambda]$ is a prime implicant of $E^c_N[\Lambda]$. By Proposition 6, any implicant χ_0 of $D^c_N[\Lambda]$ is an implicant of $E^c_N[\Lambda]$. Let χ_0' be a prime implicant of $E^c_N[\Lambda]$ such that χ_0' is a subconjunction of χ_0. By the definition of D^c, there is a graph $\mathfrak{G}_0(\in\mathcal{P}^c_{\mathcal{N}})$ such that $\chi_0=\chi_{\mathfrak{G}_0}[\Lambda]$. Let $\mathfrak{T}_1,\mathfrak{T}_2,\ldots,\mathfrak{T}_c$ be the connected components of $\mathfrak{G}_0$ (any of them is a complete graph). As $N>c+1$, $|\mathcal{V}(\mathfrak{T}_k)|>1$ for at least one k $(1\leqq k\leqq c)$. Fixing such a k, let r be an arbitrary element of $\mathcal{V}(\mathfrak{T}_k)$. Let an edge e be chosen in $\mathfrak{G}_0$ such that r, e are not incident. We have $\chi_0'=\chi_{G_0'}[\Lambda]$ for a suitable subgraph $\mathfrak{G}_0'$ of $\mathfrak{G}_0$. Let $\mathfrak{G}'_{or}$ be defined by $\mathfrak{G}'_{or}=\mathfrak{G}_0'\cap\mathfrak{C}_{N_r}$. By Theorem 1, there is a partition graph $\mathfrak{G}_P'(\in\mathcal{P}^c_{\mathcal{N}_r})$ such that $\mathfrak{G}_P'\subseteq\mathfrak{G}'_{or}$. If $\mathfrak{G}_P(\in\mathcal{P}^c_{\mathcal{N}_r})$ is defined by $\mathfrak{G}_P=\mathfrak{G}_0\cap\mathfrak{C}_{N_r}$, then we have $\mathfrak{G}_P'\subseteq\mathfrak{G}'_{or}\subseteq\mathfrak{G}_P$. Since $\mathfrak{G}_P'$, $\mathfrak{G}_P$ are partition graphs on the same vertex set and the number of their connected components coincide, $\mathfrak{G}_P'\subset\mathfrak{G}_P$ is impossible, hence $\mathfrak{G}'_P=\mathfrak{G}'_{or}=\mathfrak{G}_P$. The (arbitrarily chosen) edge e of $\mathfrak{G}_0$ belongs to $\mathfrak{G}'_P(\subseteqq\mathfrak{G}_0)$, thus $\chi'_0=\chi_0$.

$A^c_N[\Lambda]$ is defined as the disjunction of the prime implicants φ of $E^c_N[\Lambda]$ such that φ is not a prime implicant of $D^c_N[\Lambda]$.

Lemma 2. *Let $\mathfrak{G}$ be a graph such that $\mathscr{V}(\mathfrak{G})=\{1, 2, \ldots, N\}$. The following three assertions are equivalent for $\mathfrak{G}$:*

(i) $\mathfrak{G}$ *is $(c+1)$-vertex-critical,*

(ii) $\mathfrak{G}\cap\mathfrak{C}_{\mathscr{N}_i}$ *is c-chromatic for any i $(1\leqq i\leqq N)$,*

(iii) $\overline{\mathfrak{G}}\cap\mathfrak{C}_{\mathscr{N}_i}$ *includes a partition graph with connected components.*

Remarks. $\mathfrak{G}\cap\mathfrak{C}_{\mathscr{N}_i}$ results from $\mathfrak{G}$ by deleting the vertex i and the edges incident to it. $\overline{\mathfrak{G}}\cap\mathfrak{C}_{\mathscr{N}_i}$ is the complement of $\mathfrak{G}\cap\mathfrak{C}_{\mathscr{N}_i}$ with respect to $\mathfrak{C}_{\mathscr{N}_i}$.

Proof of Lemma 2. (i) and (ii) are equivalent in consequence of the definition of vertex-critical graphs. The equivalence of (ii), (iii) is obvious (cf. the statements (ii), (iii) in Proposition 2).

Proof of Theorem 3. Assume that the first sentence of Theorem 3 holds for N, c.

Necessity. Let $\chi_{\mathfrak{G}_0}[\Lambda]$ be a prime implicant of $A^c_N[\Lambda]$.

First we prove that condition (ii) of Lemma 2 holds for $\overline{\mathfrak{G}}_0$. Let k be an arbitrary element of $\mathscr{N}$. χ_0 is an implicant of $D^c_{\mathscr{N}_k}$ because of Theorem 1. So $\mathfrak{G}_0$ includes an element of $\mathscr{P}^c_{\mathscr{N}_k}$, say $\mathfrak{P}_k$. For this element $\mathfrak{P}_k\subseteqq\mathfrak{G}_0\cap\mathfrak{C}_{\mathscr{N}_k}$, thus $\mathfrak{G}_0$ satisfies condition (iii) of Lemma 2, and so — by the lemma — conditions (i) and (ii) too.

Hence $\overline{\mathfrak{G}}_0$ is $(c+1)$-vertex-critical (by Lemma 2). The necessity will completely be proved if we show the maximality of $\overline{\mathfrak{G}}_0$.

Let $e=\overline{ij}$ be an arbitrary edge of $\mathfrak{G}_0$. Define the graphs $\mathfrak{G}_1$ and $\mathfrak{G}_2$ (again on the vertex set $\{1, 2, \ldots, N\}$ such that

the edges of $\mathfrak{G}_1$ are the edges of $\overline{\mathfrak{G}}_0$ and e,

the edges of $\mathfrak{G}_2$ are the edges of $\mathfrak{G}_0$ except e.

It is clear that $\mathfrak{G}_1, \mathfrak{G}_2$ are complements of each other, and

$$(\chi_0 =)\chi_{\mathfrak{G}_0}[\Lambda] = \chi_{\mathfrak{G}_2}[\Lambda]\wedge\lambda_{ij}.$$

Let the short notation χ_2 be used for $\chi_{\mathfrak{G}_2}[\Lambda]$. χ_2 is not an implicant of $A^c_N[\Lambda]$, consequently there exists a $k(\in\mathscr{N})$ such that χ_2 is not an implicant of $D_{\mathscr{N}_k}[\Lambda]$ (by Theorem 1).

If $\mathfrak{G}_3$ is defined by $\mathfrak{G}_3=\mathfrak{G}_2\cap\mathfrak{C}_{\mathscr{N}_k}$, it is clear that $\chi_{G_3}[\Lambda]$ is not an implicant of $D_{\mathscr{N}_k}[\Lambda]$.

By Proposition 2 this means that $\mathfrak{G}_3$ has no subgraph $\mathfrak{P}$ such that $\mathfrak{P}\in\mathscr{P}^c_{\mathscr{N}_k}$. From Lemma 2 it follows that $\overline{\mathfrak{G}}_2\notin\hat{\mathscr{V}}^c_N$. As $\overline{\mathfrak{G}}_2=\overline{\mathfrak{G}}_0\cup\{e\}$ and e is an arbitrary edge of $\mathfrak{G}_0$, $\overline{\mathfrak{G}}_0$ is maximal v-critical indeed, which completes the necessity proof.

Sufficiency. If conditions (a) and (b) are fulfilled by $\mathfrak{G}_0$, then

(1) $\chi_{\mathfrak{G}_0}[\Lambda]=\chi_0$ is an implicant of A^c_N.

This can be shown in two steps.

(1.1) χ_0 is an implicant of E^c_N. Indeed, $\mathfrak{G}_0$ satisfies condition (i) of Lemma 2, and so also condition (iii) of this lemma. This implies that the graph $\overline{\mathfrak{G}}_0\cap\mathfrak{C}_{N_i}$ includes an element $\mathfrak{P}$ of $\mathscr{P}^c_{\mathscr{N}_i}$ and therefore χ_0 is an implicant of $D^c_{\mathscr{N}_i}$ (for every $i(\in N)$) by Proposition 2.

Now from Theorem 1 it follows that χ_0 is an implicant of E_N^c.

(1.2) χ_0 is not an implicant of D_N^c. By Proposition 2 this is true if and only if $\mathfrak{G}_0$ includes no element of $\mathscr{P}_{\mathscr{N}}^c$, that is $\varkappa(\overline{\mathfrak{G}}_0)>c$. But this is now in consequence of condition (a) of our theorem.

From (1.1) and (1.2) we conclude that (1) is true. It remains to prove that

(2) χ is a prime implicant of A_N^c.

To prove this chose an arbitrary edge e of $\mathfrak{G}_0$. Let us introduce a new graph $\mathfrak{G}_1$ by $\mathfrak{G}_1=\overline{\mathfrak{G}}_0\cup\{e\}$. As $\mathfrak{G}_0$ is maximal $(c+1)$-vertex-critical, $\mathfrak{G}_1$ is not $(c+1)$-vertex-critical. By Lemma 2, there exists an $r(\in\mathscr{N})$ such that the graph $\overline{\mathfrak{G}}_1\cap\mathfrak{C}_{\mathscr{N}_r}$ includes no partition graph $\mathfrak{P}\in P_{\mathscr{N}_r}^c$. By Proposition 2, for this r $\chi_{\mathfrak{G}_1}[A]$ is not an implicant of $D_{\mathscr{N}_r}^c[A]$.

By Theorem 1, $\chi_{\mathfrak{G}_1}$ is not an implicant of $E_N^c[A]$, thus we have proved assertion 2. This completes the sufficiency proof.

Proof of Theorem 4. The first part of the assertions — the sufficient and necessary condition — is equivalent to Proposition 5; so it does not require any proof. To prove the last sentence of the theorem, let us distinguish two cases: (i) $\varkappa(\overline{\mathfrak{G}}_0)\geqq c+1$ and (ii) $\varkappa(\overline{\mathfrak{G}}_0)<c$. In case (i) by the first part of this theorem $|V(\overline{\mathfrak{G}}_0)|\leqq N-1$, which is the second alternative of the assertion to be proved. In case (ii) there exists a graph $\mathfrak{P}\in\mathscr{P}_{\mathscr{N}}^c$ such that $\mathfrak{G}_0\supseteqq\mathfrak{P}$ and so $\chi_{\mathfrak{P}}[A]$ is a subconjunction of $\chi_{\mathfrak{G}_0}$. But $\chi_{\mathfrak{P}}[A]$ is an implicant of $F_N^c[A]$ because it is an implicant of $D_N^c[A]$. As $\chi_{\mathfrak{G}_0}[A]$ is a prime implicant of F_N^c, it cannot include $\chi_{\mathfrak{P}}[A]$ properly, therefore $\chi_{\mathfrak{G}_0}[A]=\chi_{\mathfrak{P}}[A]$, that is $\mathfrak{G}_0=\mathfrak{P}$, proving the second part of the theorem. Thus Theorem 4 is proved.

*

Acknowledgement. The author wishes to thank to A. Ádám for his many valuable comments and criticisms. He has rewritten the original paper to a great extent, introducing a number of notions and propositions to ease the reading.

References

[1] Ádám, A., *Truth functions and the problem of their realization by two-terminal graphs*, Akadémiai Kiadó, Budapest, 1968.

[2] Dirac, G. A., Some theorems on abstract graphs, *Proc. London Math. Soc.*, v. 2, 1952, pp. 69—81.

(Received Sept. 29, 1977)

INDEX—TARTALOM

ISSN 0324—721 X

Felelős szerkesztő és kiadó: Gécseg Ferenc
A kézirat a nyomdába érkezett: 1977. december hó
Megjelenés: 1978. május hó
Példányszám: 1000. Terjedelem: 5,81 (A/5) ív
Készült monószedéssel, íves magasnyomással
az MSZ 5601 és az MSZ 5602—55 szabvány szerint
77-5590—Szegedi Nyomda—F. v.: Dobó József