# A sz.ot.ag
# Optimalitáselmélet szimulált hőkezeléssel

Bíró Tamás

Eötvös Loránd Tudományegyetem, Budapest
University of Groningen, Hollandia
birot@nytud.hu

**Kivonat:** Az SA-OT algoritmus bevezeti a *szomszédsági struktúra* (*neighbourhood structure*; geometria vagy topológia) fogalmát a jelöltek halmazán. A szimulált hőkezelés egy véletlen bolyongást valósít meg azon a halmazon, amelynek a legjobb elemét keresi, és ehhez a szomszédsági struktúra adja meg azt, hogy egy adott állapotból (SA-OT esetén: jelöltből) mely szomszédos állapotokba lehet lépni. Sőt, a topológiának azt is meg kell határoznia, hogy melyik szomszédot milyen *a priori* valószínűséggel választjuk. A jelen cikk alkalmazza az SA-OT algoritmust a szótagolásra, különböző lehetséges topológiákat hasonlít össze, és azt mutatja be, hogy a topológia definíciója – a többi paraméterhez hasonlóan – jelentősen befolyásolja az algoritmus kimenetelét.

## 1 Keresés a jelöltek halmazán

Az Optimalitáselmélet (*Optimality Theory,* OT, Prince és Smolensky, 1993 / 2004; magyarul lásd például: Rebrus, 2001) alapfeltevése szerint a mögöttes reprezentációból generált jelöltek halmaza (*candidate set*) optimális eleme (vagy elemei) felel(nek) meg a leírandó nyelv felszíni alakjainak. Az $E$ Evaluációs (vagy Harmónia-) függvény egy-egy vektort rendel az egyes $w$ jelöltekhez, amelynek $i$-ik komponense az adott jelölt számára az $i$-ik *constraint* ($C_i$) által kiosztott sértések száma, $C_i(w)$:

$$E(w) = \left(C_N(w), C_{N-1}(w), \ldots, C_0(w)\right) \qquad (1)$$

Az OT-paradigmán belül dolgozó hagyományos nyelvész (leggyakrabban fonológus) feladata meghatározni, hogy egy adott jelölt, mint potenciális nyelvi alak, mennyi sértést gyűjt be egy-egy constraint-től (megszorítástól, korláttól) – azaz, milyen constraint-eket definiáljunk, univerzális nyelvi megfigyelések alapján. Mi most ezeket a $C_i$ megszorításokat adottnak vesszük, hiszen a klasszikus OT filozófiája szerint a constraint-ek halmaza univerzális (csak éppen nem jelenleg ismert).

Ha a vektorkomponensek sorrendje megfelel a constraint-ek csökkenő hierarchiájának, akkor az optimalitást a vektorok *lexikografikus rendezése* határozza meg. A szavak ábécésorrendjéhez hasonló módon, két vektor közül azt a vektort tekintjük harmonikusabbnak, amelynek az első nem-azonos komponense kisebb:

*Definíció:* Legyen $w_1$ és $w_2$ a jelöltek halmazának két eleme. A $w_1$ jelölt akkor és csak akkor *harmonikusabb* a $w_2$ jelöltnél, ha létezik olyan $k \in \{N, N\text{-}1,\ldots, 0\}$, hogy $C_k(w_1) < C_k(w_2)$, továbbá minden $i \in \{N, N\text{-}1,\ldots, 0\}$-re: ha $i > k$, akkor $C_i(w_1) = C_i(w_2)$.

A $w_1$ és $w_2$ jelöltek *ekvivalensek*, ha minden $i \in \{N, N\text{-}1,\ldots, 0\}$-re $C_i(w_1) = C_i(w_2)$.

Azt a $C_k$ megszorítást, amely a $w_1$ és $w_2$ jelöltek összehasonlítása során eldönti, hogy melyik jelölt a harmonikusabb, *fatális constraint*-nek fogjuk nevezni. Bebizonyítható formálisan (Bíró, 2006), hogy bizonyos, a nyelvészeti alkalmazásokban általában teljesülő feltételek mellett, a jelöltek halmaza egy *jólrendezett* halmaz, sőt bármely részhalmazának van *optimális részhalmaza*:

*Definíció:* Legyen *S a jelöltek valamely (rész)halmaza.* Ekkor létezik $S_ö \subseteq S$ úgy, hogy (a) ha $w_1$ és $w_2 \in S_ö$, akkor $w_1$ és $w_2$ ekvivalensek; (b) ha $w_1 \in S_ö$ és $w_2 \notin S_ö$, akkor $w_1$ harmonikusabb $w_2$-nél. Ezt az optimális részhalmazt $S_ö$ = opt ($S$)-sel jelöljük.

Ezek alapján formálisan is megfogalmazhatjuk az Optimalitáselmélet alapfeltevését. Mint oly sok nyelvészeti modell az elmúlt ötven évben, egy *UR* mögöttes reprezentáció és egy *SR* felszíni alak közötti leképezést szeretnénk mi is megadni. Ha *GEN*(*UR*) jelöli az *UR*-ből az univerzális *Generátor függvény* által legyártott jelöltek halmazát, *E*(*w*) pedig az univerzális constraint-ek alapján (1) szerint definiált *Evaluációs függvényt*, akkor az UR-nek megfelelő grammatikus alak(ok) azok, amelyek *GEN*(*UR*)-en optimalizálják az *E*(*w*) függvényt:

$$SR(UR) = \underset{w \in GEN(UR)}{\arg\text{opt}} \left(E(w)\right) \qquad (2)$$

Véges, kevés elemből álló jelölthalmaz esetén az optimális elem megtalálása nem jelent nehézséget. Nagyobb, és főleg végtelen jelölthalmaz esetén viszont az Optimalitáselmélet egy ritkán végiggondolt komputációs kihívást jelent – legyen szó a modell számítógépes implementációjáról vagy a kognitív plauzibilitásáról. Eisner (2000) bebizonyítatta, hogy az optimális elem megtalálása NP-teljes probléma a nyelvtan méretében. Bizonyos erősen korlátozott feltételek között használhatóak véges állapotú modellek (pl. Ellison, 1994; Frank és Satta, 1998; Gerdemann és van Noord, 2000; Bíró, 2003, 2005a). Dinamikus programozással (Tesar és Smolensky, 2000; Kuhn, 2000) egy jóval tágabb problémahalmazra implementálható az Optimalitáselmélet, de ez utóbbi technika – a véges állapotú automaták megépítéséhez hasonlóan – nem csekély számítási kapacitást igényel.

Kérdés, persze, hogy van-e egyáltalán szükség olyan algoritmusra, amely habár jelentős számítógépes kapacitások (memória, idő) igénybe vétele árán, de garantáltan megtalálja a jelöltek halmazának optimális elemét. A nyelvtechnológia talán még használhatná ezeket az eljárásokat – feltéve, hogy a nyelvtechnológia bármikor hasznosítani fogja az OT-t –, de nehéz elképzelni, hogy biológiai (pszichológiai, kognitív) szempontból adekvátak lenének. Az emberi beszédprodukció ugyanis nem hibátlan, viszont olyan algoritmusra van szükségünk, amely valós időben működik. Beszéd közben nem forgathatjuk a homokórát a képernyőn („bocs, számolok"), de hajlandóak vagyunk a pontosságból áldozni, különösen akkor, ha ezáltal felgyorsítható a számolás. Az SA-OT algoritmus ezt lehetővé teszi.

## 2 A szimulált hőkezelés alkalmazása az Optimalitáselméletre

Egy házi dolgozatban (Bíró, 1997), a disszertációmban (Bíró, 2006), valamint néhány cikkben (Bíró, 2005b,c) javasoltam a *szimulált hőkezelés* (Kirkpatrick et al., 1983)[2] alkalmazását a legjobb jelölt megkeresésére és a beszédtempó modellezésére. A statisztikus fizikából származó (Metropolis *et al.*, 1953[3]), az elmúlt húsz évben széles körben elterjedt optimalizációs algoritmus előnye ugyanis az egyszerűsége és a kis számítási igénye.

Az SA-OT (*Simulated Annealing Optimality Theory*; 1. ábra) algoritmus ún. heurisztikus technika (például ld. Reeves, 1995), vagyis nem garantálja, hogy megtalálja a keresett legjobb megoldást. Lassú futtatás (sok iterációs lépés) esetén nagy valószínűséggel rátalál a „helyes” megoldásra, míg gyors futtatás (kevés iteráció) mellett könnyebben hibázik. Tehát, akárcsak az emberi beszéd, az algoritmus felgyorsítható, amelynek az árát a pontossággal kell megfizetni. Viszont nem bármilyen hibát követ el az algoritmus: csak bizonyos „helytelen” alakokat talál meg, amelyek egy sikeres modellben épp a gyorsbeszéd jellegzetes alakjainak felelnek meg.

```
algorithm SA-OT
    w := w_init;
    for K = K_max to K_min step K_step
         for t = t_max to t_min step t_step
           choose random w' in Neighbourhood(w);
           calculate <k,d> = | E(w')-E(w) | ;
           if d <= 0 then   w:=w';
           else             w:=w' with probability
               P(C,d;K,t)   = 1          , if k < K
                            = exp(-d/t) , if k = K
                            = 0          , if k > K ;
         end-for;
    end-for;
```

1. ábra: Az SA-OT algoritmus (*Simulated Annealing Optimality Theory*) (Bíró, 2006)

Az SA-OT algoritmus megértése érdekében először tekintsük az egyik legegyszerűbb optimalizációs eljárást. Tegyük fel, hogy egy $E(w)$ függvényt szeretnénk *minimalizálni* egy $S$ alaphalmazon. Definiáljunk egy *topológiát* (egy gráfszerű geometriát, szomszédsági struktúrát) az $S$ alaphalmazon: egy *Neighbour*($w$) függvényt, amely $S$ mindegyik $w$ eleméhez hozzárendeli $S$ egy részhalmazát, $w$ szomszédjait. Ezen szomszédsági struktúra lehetővé teszi azt, hogy egy képzeletbeli bolhát (egy *véletlen bolyongást*) indítsunk el $S$-en, valamely $w_{init}$ elemből indulva.

Egy iteráció két részből áll. Amikor a bolha a $w$ pontban tartózkodik, kiválasztjuk véletlenszerűen $w$ valamelyik $w'$ szomszédját (azaz *Neighbour*($w$) egyik elemét), majd eldöntjük, hogy a bolha vajon $w'$-ba ugrik-e, vagy $w$-ben marad. Első megközelítésben legyen az a szabály, hogy a bolha akkor és csak akkor ugrik $w'$-be, ha $E(w') \leq$

---

[2] Az ELTE fizikus szakán *szimulált hőkezelésnek* neveztük azt a technikát, amelyet a BME-n *szimulált lehűtésként* ismernek. A cikkben az előző elnevezést fogjuk használni.

[3] Érdekesség, hogy az említett cikk utolsó szerzője Teller Ede.

$E(w)$. Az algoritmus addig tart, amíg adott számú iterációt el nem végeztünk, vagy pedig amíg a bolha be nem ragad egy *lokális minimumba*. Ugyanis a lokális minimumnak, definíció szerint, nincs olyan szomszédja, ahova a bolha átugorhatna. Az algoritmus az $S$ alaphalmaznak azon elemét adja vissza, amelyben a bolha végül leledzik. Ha az eljárás elég hosszan fut, az algoritmus kimenetele egy *lokális minimum* lesz – de semmi sem garantálja, hogy megtaláljuk a keresett *globális minimumot*. Sőt, ha rossz „völgyből" indulunk, el sem juthatunk a globális minimumhoz.

A hagyományos *szimulált hőkezelés* (vagy *szimulált lehűtés*, *simulated annealing*), ezzel szemben, lehetővé teszi „hegyek" megmászását is azáltal, hogy a bolha bizonyos valószínűséggel akkor is átugorhat $w$-ből $w'$-be, ha $E(w') > E(w)$. Ezt az *állapotátmeneti valószínűséget* egy $T$ paraméter segítségével számoljuk ki:

$$P(w \to w' \mid T) = \begin{cases} e^{-\left(E(w')-E(w)\right)/T} & \text{ha } E(w') > E(w) \\ 1 & \text{ha } E(w') \le E(w) \end{cases} \quad \textbf{(3)}$$

A $T$ paramétert *hőmérsékletnek* hívjuk, utalva a statisztikus fizikai (termodinamikai) gyökerekre. Jelentése az az $E(w') - E(w)$ érték, amennyit a bolha $1/e = 0{,}37$ valószínűséggel ugrik felfelé. Kisebb ugrás valószínűsége magasabb, nagyobb ugrásé alacsonyabb. A szimuláció kezdetén $T$ értéke magas, és a bolha bármely szomszédos állapotba könnyen átugorhat. Majd a $T$ hőmérsékletet lépésről lépésre csökkentjük, a rendszert „hűtjük": a bolha fokozatosan fárad, és csak egyre kisebb „dombokat" hajlandó megmászni. Végül a rendszer „megfagy", és a bolha leereszkedik a legközelebbi völgy aljára. A szimulált hőkezelés sikerét az adja, hogy minél lassabban csökkentjük a hőmérsékletet (vagyis minél több iterációs lépést engedünk közepes hőmérsékleten), annál nagyobb a valószínűsége annak, hogy a bolha a globális minimumot találja meg az algoritmus végén, mert kikerüli a többi lokális minimum csapdáját.

Végül ezt a technikát alkalmazzuk az Optimalitáselméletre. A nehézséget az adja, hogy – a hagyományos szimulált hőkezeléssel ellentétben – az optimalizálandó $E(w)$ függvény nem valósértékű, általános esetben nem helyettesíthető egy valósértékű függvénnyel. Hogyan értelmezzük akkor az exponenciális kifejezést a (3) egyenletben? A különböző megfontolásokból javasolt megoldás (Bíró 2005c, 2006) szerint legyen két, (1) szerinti $E(w)$ vektor „különbsége" a következő rendezett pár:

$$\left|E(w') - E(w)\right| := < k, C_k(w') - C_k(w) > \quad \textbf{(4)}$$

ahol $C_k$ a fatális constraint $w$ és $w'$ jelöltek összehasonlításakor. Vagyis két OT-jelölt *Evaluációs függvényének* különbségét az a legmagasabb $C_k$ korlát adja meg, amelyben eltérnek. A különbség egy rendezett pár, amelynek az első eleme $C_k$ indexe, a második eleme pedig az ezen megszorítás által kiosztott sértések számának a különbsége.

Most már érthetővé válik az 1. ábrán bemutatott SA-OT algoritmus magja. A kettős ciklus belsejében előbb a jelenlegi $w$ állapot egy szomszédos $w'$ szomszédját választjuk ki véletlenszerűen, majd (4) segítségével kiszámoljuk a megfelelő *Eval*-függvények értékének a különbségét. Ha a különbség második tagja nem-pozitív, a bolha átugrik a $w$-nél harmonikusabb $w'$ állapotba. Ellenkező esetben az átmenet valószínűségét a különbség két tagján kívül a $T = <K,t>$ hőmérséklet is befolyásolja, és a valószínűséget a (3) egyenlet analógiájára számoljuk ki. Ezt az analógiát alaposabban kidolgozza Bíró (2005c, 2006). A $P(w \to w' \mid T)$ állapotátmeneti valószínűség jelentése az, hogy generálunk egy *r* véletlen számot 0 és 1 között egyenletes eloszlással, és ha $r < P(w \rightarrow w' \mid \mathrm{T})$, akkor a bolha átugrik *w*-ből *w'*-be.

Már megelőlegeztük azt, hogy az SA-OT algoritmusban a *T* hőmérséklet sem egy valós szám, hanem éppúgy egy rendezett pár, mint |*E*(*w'*) – *E*(*w*)| = <k,d>. E két mennyiség azonos szerkezetét megköveteli a (3) egyenlet, illetve annak az interpretációja: *T* jelentése azon ugrás magassága, amelynek a valószínűsége 1/*e* = 0,37. Az SA-OT algoritmusban szintén igaz, hogy ha a |*E*(*w'*) – *E*(*w*)| és *T* egyenlő (ugyanaz a rendezett pár), akkor a bolha 1/*e* valószínűséggel ugrik felfelé.

Végezetül, a hőmérséklet komplex voltából következik az, hogy a hagyományos szimulált hőkezelés egyszeres ciklusával ellentétben, az SA-OT algoritmus egy kettős ciklus segítségével csökkenti a *T* = <*K*,*t*> hőmérsékletet. A két ciklus kétszer három paramétere ($K_{max}$, $K_{min}$, $K_{step}$, $t_{max}$, $t_{min}$, $t_{step}$) határozza meg az iterációk számát, vagyis a globális minimum megtalálásának a valószínűségét (Bíró 2005b, 2006).

# 3 A topológia jelentősége

A jelenlegi cikk célja a topológia jelentőségének a bemutatása. A topológiát eddig mint egy *Neighbour*(*w*) függvényt definiáltuk, amely – az SA-OT esetén – a jelöltek halmazát képezi le ezen halmaz hatványhalmazára: minden jelölthöz hozzárendeli a jelöltek halmazának egy részhalmazát. De valójában ennél többre van szükségünk, méghozzá egy *valószínűségi eloszlásra* minden egyes *Neighbour*(*w*) halmazon.

Ugyanis a szomszédsági struktúrát arra használjuk, hogy a bolha jelenlegi *w* helyének egy *w'* szomszédját véletlenszerűen kiválasszuk. A *P*(*w'*|*w*) valószínűség fogja megadni azt, hogy milyen valószínűséggel választjuk *w'*-t következő potenciális ugrási célpontként, feltéve, hogy *w'* ∈ *Neighbour*(*w*). A *P*(*w'*|*w*) valószínűséget mostantól *a priori valószínűségnek* nevezzük, ellentétben a korábban, például (3)-ban, definiált *P*(*w* □ *w'* | *T*) *állapotátmeneti valószínűséggel.* Annak az esélye, hogy a bolha a következő iteráció során a *w'* pontban lesz, feltéve, hogy most *w*-ben található, e két valószínűség szorzata: előbb a *P*(*w'*|*w*) *a priori* valószínűséggel választjuk ki *w'*-t, majd a *P*(*w* □ *w'* | *T*) állapotátmeneti valószínűség határozza meg, hogy a bolha tényleg átugrik-e oda.

E két valószínűség egymástól független. Az állapotátmeneti valószínűségek a véletlen bolyongás tájképének „függőleges struktúrájától”, az *Eval*-függvény által meghatározott hegyek magasságától, völgyek mélységétől, lejtők meredekségétől függ, valamint a fokozatosan csökkenő hőmérséklettől. Ha a constraint-eket más sorrendbe rakjuk, azaz megváltoztatjuk az E-függvényt, nagyon különböző valószínűségeket kapunk. Ezzel szemben, az *a priori* valószínűségek függetlenek a constraintek-től, azok rendezésétől, és a szimuláció során sem változnak. Az említett tájkép „vízszintes struktúráját” határozzák meg, az egyes jelöltek „szomszédsági fokát”, „közelségét”.

Amint a hagyományos OT a jelöltek halmazát egyetemesnek feltételezi (a *Richness of the Base* alapelve és a GEN leképezés univerzalitása folytán), úgy a jelöltek halmazának a struktúráját sem tekintem – alapvetően – nyelvspecifikusnak. A legegyszerűbb javaslat szerint *w'* ∈ *Neighbour*(*w*) akkor és csak akkor, ha *w*-ből egy elemi transzformáció segítségével konstruálható *w'*. Mivel ezek az elemi transzformációk – például egy szegmentum beszúrása vagy törlése, egy szerkezeti határ eltolása – a nyelvi jel szerkezetéből adódnak, a jelöltek halmazának a topológiája természetes módon kell, hogy kapcsolódjon a jelöltek univerzális ábrázolási formalizmusához.

Jegyezzük itt meg, hogy a topológia definiálását néhány további megszorítás is befolyásolja. Habár nem szükségszerű, de valószínűleg a szomszédsági relációt ésszerű szimmetrikusnak megadni. Fontos viszont a véletlen bolyongás szempontjából, hogy a jelöltek halmaza egy összefüggő struktúrát alkosson, vagyis bármely elemből el lehessen jutni bármely másik elembe véges számú lépéssel. Harmadszor, valószínűleg nem érdemes egy jelölthöz túl sok, egyaránt valószínű szomszédot rendelni, hiszen ekkor a szimulált hőkezelés – amelynek a lényege a szomszédsági struktúra értelmes kihasználása – elveszíti a hatékonyságát, és egy hihetetlenül ügyetlen, véletlenszerű próbálkozássá redukálódik.

Mindazonáltal, a topológiához kapcsolódó valószínűségi eloszlás definiálása már nem ennyire triviális feladat. Amennyiben mindegyik jelöltnek néhány szomszédja van, a legegyszerűbb megközelítés mindegyik szomszédnak egyenlő valószínűséget rendel: $P(w'|w) = P(w'' \mid w)$, ha mind $w'$ és $w'' \in$ *Neighbour*($w$). Ez a javaslat például sikerrel járt a holland gyorsbeszédben megfigyelhető hangsúlyeltolódások magyarázata során (Bíró, 2005b). De más lehetőségek is elképzelhetőek, és az alábbiakban a topológia, amelybe ezentúl beleértjük a $P(w'|w)$ *a priori* valószínűségeket is,[4] szerepét vizsgáljuk az SA-OT-ben.

Ezen a ponton kicsit árnyaljuk a topológia univerzalitásáról tett korábbi állításunkat. A topológiát meghatározó alapelvek, az *a priori* valószínűségek kiszámításának a logikája valóban univerzális, de a pontos részleteket esetleg paraméterek határozhatják meg. Amint rövidesen látni fogjuk, elképzelhető például, hogy egy jelölt szomszédjait úgy kapjuk, hogy bizonyos számú elemi transzformációk közül pontosan egyet hajtunk végre, de ezen elemi transzformációkhoz nem szükséges egyenlő valószínűségeket rendelni. Lehetséges tehát, hogy az egyes elemi transzformációk pontos valószínűsége, mint a topológia paraméterei, nyelvenként, regiszterenként, esetleg beszélőnként kis mértékben változhatnak.

---

[4] Eddig a $P(w'|w)$ *a priori* valószínűséget adott $w$ esetén a *Neighbour*($w$) halmazon vezettük be. De tulajdonképpen a *Neighbour*($w$) függvényre nem is lesz szükségünk mostantól, elegendő lesz a $P(w'|w)$: $S \times S \rightarrow [0,1]$ leképezést megadnunk. A *Neighbour*($w$) pedig azon $w'$-k halmaza lesz, amelyekre $P(w'|w)$ pozitív (vagy nagyobb egy adott $\varepsilon$-nál). Más megközelítésben, *Neighbour*($w$)-t felvehetjük úgy is, mint a jelöltek teljes $S$ halmazát, csak éppen e halmaz jelentős részén nulla (elenyésző) az *a priori* valószínűség. Ne felejtsük el azt sem, hogy bármely $w$-re a

$$\sum_{w'} P(w' \mid w) = 1$$

feltételnek teljesülnie kell, mivel a valószínűségi eloszlást 1-re kell normálni.

# 4 Szó.ta.go.lás

## 4.1 Az SA-OT komponensei a klasszikus szótagolási modellhez

A jelen cikkben az SA-OT algoritmust az egyik klasszikus optimalitáselméleti példán, a szótagoláson mutatom be: hogyan bontsunk szótagokra (szótagkezdetre, szótagmagra és kódára) egy magánhangzó-mássalhangzó sorozatot? (Magyarul lásd például: Rebrus, 2001.) A paradigma nem csupán a *faktoriális tipológia* klasszikus példája 1993. óta, de az Optimalitáselmélet különböző implementációit is ezen a feladaton illusztrálták (véges állapotú automatákkal: Gerdemann és van Noord, 2000; dinamikus programozással: Tesar és Smolensky, 2000). Az egyszerű constraint-ek ellenére a legjobb jelölt megtalálása nem triviális feladat, mivel a jelöltek halmaza végtelen a rekurzív epentézis (*over-parsing*) lehetőségének a következtében.

A feladat ismert: egy bemeneti sztringhez, mint mögöttes reprezentációhoz, rendelünk egy kimeneti sztringet (fűzért), mint felszíni alakot, amely alapvetően a bemeneti sztring másolata, de

(a) a bemeneti sztring néhány karaktere törölhető (*underparsing*, alulelemzés),
(b) epentetikus szegmensek szúrhatók be (*overparsing*, túlelemzés),
(c) a szótaghatárok jelölve vannak (például egy ponttal).

A szótagok pontosan egy *nukleuszt* (*szótagmagot*) tartalmazhatnak, és nyelvenként változik, hogy mi lehet szótagmag. Az egyszerűség kedvéért feltételezzük most, hogy a nyelv hangkészletének egy része (magánhangzók, diftongusok, szillabikus szegmentumok) szerepelhetnek csak nukleuszként a szótagban, és ezek a szegmentumok szerepet nem is tölthetnek be.

A szótagnak a nukleuszt megelőző részét *szótagkezdetnek* (*onset*-nek), a nukleuszt követő darabját pedig *kódának* nevezzük. Ismét az egyszerűség kedvéért azt is feltételezzük, hogy a szótagkezdet és a kóda nem lehetnek komplexek (elágazóak), csupán egyetlen fonémát tartalmazhatnak. A nyelvek egy részében ez valóban létező megszorítás, és most elsősorban az algoritmus tulajdonságai érdekelnek bennünket.

Adott mögöttes reprezentációhoz tartozó jelölthalmaz a fenti feltételeknek megfelelő valamennyi sztring. Gerdemann és van Noord (2000) bemutatja, hogy habár végtelen, de a jelölthalmaz egy reguláris nyelvet alkot. Ezen halmaz optimális elemét ezek után az alábbi constraint-ek valamilyen rendezése mellett kell megtalálnunk:[5]

ONSET: a szótagkezdettel *nem* rendelkező szótagok száma
NOCODA: a kódával rendelkező szótagok száma
PARSE: az *alulelemzett* (a mögöttes formából törölt) szegmentumok száma.
FILLNUCLEUS: az *túlelemzett* (epentetizált), szótagmagi pozícióban található szegmentumok száma.
FILLONSET: az *túlelemzett* (epentetizált), szótagkezdeti pozícióban található szegmentumok száma.

---

[5] A constraint-eket, a szokásos fonológiai irodalommal ellentétben, nem azon feltételek megadásával határozom meg, amelyeket az alakoknak lehetőleg teljesíteniük kell. Hanem, mivel a constraint-ek egészértékű függvények, a jelölteknek kiosztott sértések számát definiálom.

Hogyan lássunk most neki e modell SA-OT-vel történő implementációjához? Egy SA-OT szimuláció előkészítése a következő négy lépésből áll:

1. A jelöltek halmazának (azaz a *Generátor- függvénynek*) a megadása.
2. A topológia és az *a priori* valószínűségek definiálása a jelöltek halmazán.
3. A constraint-ek és azok rendezésének (hierarchiájának) a meghatározása.
4. Az SA-OT paramétereinek, elsősorban a „hűtés menetének” (*cooling schedule*, a hőmérséklet csökkentési ritmusának) a megadása.

Ezen teendők közül az első és a harmadik pont a hagyományos Optimalitáselméletből ismert, és a nem-számítógépes nyelvészet területe. A szótagolás kapcsán fentebb már definiáltuk a jelöltek halmazát és a constraint-eket, és rövidesen kipróbáljuk a modellt több lehetséges hierarchiára (v.ö. *factorial typology*). Az utolsó pontra kitérünk rövidesen, így most összpontosítsunk a topológiára, pontosabban szólva az *a priori* valószínűségekre.

Két jelöltsztringet akkor tekintünk szomszédnak, ha pontosan egy *elemi lépés* (*elemi transzformáció*) segítségével eljuthatunk egyikből a másikba. Az elemi lépések – első megközelítésben – a következők:

1. A szóalak hosszának növelése epentetikus fonéma beszúrásával (magánhangzó nukleuszi pozícióba, mássalhangzó szótagkezdeti vagy kódai pozícióba).
2. A szóalak hosszának a csökkentése egy epentetikus szegmentum törlésével, vagy egy eredeti (nem-epentetikus) szegmentum alulelemzésével. Egy alulelemzett eredeti szegmentum visszaelemzése.
3. Egy szótaghatár eltolása, vagyis egy szótagkezdet átértelmezése kódává, vagy egy kóda átértelmezése szótagkezdetté.

Valójában, az SA-OT implementációnk nem *választ* egyet az előre kiszámolt szomszédok közül, hanem *legyártunk* egyet közülük. Az *a priori* valószínűségeket az határozza meg, ahogyan pontosan eljárunk. Először is, eldöntjük, hogy újraszótagolunk-e (a 3. lehetőség), vagy megváltoztatjuk-e a szó hosszát. Az újraszótagolás esélye $p_{reparse}$, míg az első két lehetőségé $1 – p_{reparse}$. A tapasztalat szerint a 60% körüli $p_{reparse}$ érték volt a legcélravezetőbb. Amennyiben az újraszótagolás mellett döntünk, véletlenszerűen választjuk ki, hogy melyik intervokális mássalhangzónak változtassuk meg a státuszát (ha *onset* volt, kódává tesszük, és fordítva). Ellenkező esetben, 50% valószínűséggel beszúrunk egy epentetikus szegmentumot (40%, 40% és 20% rendre az esélye annak, hogy szótagkezdetet, nukleuszt vagy kódát szúrunk be). És szintén 50% eséllyel törlünk egy szegmentumot[6]: ha epentetikus volt, töröljük, ha parszolt eredeti szegmentum volt, alulelemezzük, de ha eddig alulelemezve volt, akkor újraparszoljuk. Végül, amint eldöntöttük, hogy melyik műveletet választjuk, a jelöltsztringen belüli pozíciók közül egyenlő eséllyel kiválasztjuk, hogy hol hajtsuk azt végre – amennyiben ezáltal megengedett alakot kapunk.

[6] Az epentézis és a törlés egyenlő valószínűségének a célja az, hogy egyensúlyban legyenek a stringet növelő és az azt csökkentő operációk. A szimuláció korai fázisában, amikor még a magas hőmérséklet megenged bármely állapotátmenetet, fontos, hogy a véletlen bolyongó bejárhassa a jelöltek halmazának egy jelentős részét – az epentetikus szegmentumokat törlő operációk túlsúlya ezt megakadályozná. Ugyanakkor, a túl sok epentézis szükségtelenül megnöveli, „felfújja” a sztringet.

### 4.2 Az SA-OT futtatása

Ha ezt a szimulációt lefuttatjuk, csak a legritkább esetben kapjuk vissza a várt alakot, mivel egy sor *lokális minimum* állít csapdát a modell számára. Korábbi modellekkel ellentétben (például Bíró 2005b), e lokális optimumok nem felelnek meg performanciahibáknak. Így ki kell egészíteni a topológiát olyan további, *ad hoc* transzformációknak megfelelő szomszédsági relációkkal, amelyek megszüntetik ezeket a csapdákat. E *posztprocesszálás* során $p_{postproc}$ valószínűséggel két transzformációt hajtunk végre az előbbiekben leírt algoritmus eredményeként kapott sztringen. Egyrészt törölhetünk egy szótagot, ha pontosan egy epentetikus szótagkezdetből és egy epentetikus szótagmagból áll: az ilyen szótag törlése ugyanis két lépésből áll, de az első lépés az eredetinél rosszabb jelöltet hozna létre. Hasonló okokból nekünk kell közbelépnünk, ha egy epentetikus nukleusz mellett egy alulelemzett magánhangzó, vagy egy epentetikus onset mellett egy alulelemzett eredeti mássalhangzó áll.

1. Táblázat: A $p_{reparse}$ és a $p_{postproc}$ paraméterek hatása az SA-OT algoritmus pontosságára

| $p_{reparse}$ | % | $p_{reparse}$ | % |
|---|---|---|---|
| 0.00 | 15 | 0.60 | 20 |
| 0.10 | 15 | 0.70 | 15 |
| 0.20 | 15 | 0.80 | 14 |
| 0.30 | 16 | 0.90 | 9 |
| 0.40 | 14 | 1.00 | 3 |
| 0.50 | 17 | | |

| $p_{postproc}$ | % | $p_{postproc}$ | % | $p_{postproc}$ | % |
|---|---|---|---|---|---|
| 0.00 | 19 | 0.35 | 19 | 0.70 | 14 |
| 0.05 | 11 | 0.40 | 15 | 0.75 | 15 |
| 0.10 | 8 | 0.45 | 12 | 0.80 | 16 |
| 0.15 | 10 | 0.50 | 13 | 0.85 | 14 |
| 0.20 | 14 | 0.55 | 11 | 0.90 | 16 |
| 0.25 | 18 | 0.60 | 11 | 0.95 | 21 |
| 0.30 | 14 | 0.65 | 14 | 1.00 | 25 |

Az 1. Táblázat azt mutatja be, hogy a két tárgyalt paraméter – $p_{reparse}$ és $p_{postproc}$ – miként befolyásolja az SA-OT algoritmus pontosságát, a topológia *a priori* valószínűségeinek modulálása révén. Az *an.ta* bemenettel indítottuk el az algoritmust, mindegyik ($p_{reparse}$ , $p_{postproc}$) paraméterkombináció esetén tízszer-tízszer; a használt hierarchia ONSET ≫ FILLNUCLEUS ≫ PARSE ≫ FILLONSET ≫ NOCODA volt ebben az előzetes kísérletben. Mindkét táblázat csak az egyik paraméter szerepét mutatja be, miközben a másik paraméterre átlagoltunk. Látható, hogy bizonyos paraméterkombinációk jelentősen eltérő valószínűséggel találják meg a helyes alakot – esetünkben *Tan.ta*-t, egy szókezdő epentetikus mássalhangzóval. Az SA-OT algoritmus paraméterei a következők voltak: $K_{max}$ = 5, $K_{min}$ = -2, $K_{step}$ = 1, $t_{max}$ = 4, $t_{min}$ = 0, $t_{step}$ = 0,1. Az öt constraintet a 0, 1,…, 4 indexekkel azonosítottuk.

Hasonló feltételek mellett futottak azok a szimulációk, amelyekről a 2. Táblázat számol be. Ezúttal a topológia paraméterei rögzítettek: $p_{reparse} = 0{,}60$, $p_{postproc} = 0{,}95$.

Figyeljük meg, hogy különböző hierarchiák nagyon eltérő pontossággal produkálhatók. Ezt a tényt interpretálhatjuk úgy, hogy az SA-OT lehetőséget biztosít arra, hogy habár egyes nyelvtípusokat elvileg megengedne a faktoriális tipológia (azaz az emberi agy), de megmagyarázzuk, miért nem fordulnak elő (egyáltalán nem, vagy csupán ritkán) a világ nyelvei között.

Ugyanis ezeket az elvileg lehetséges nyelveket nehéz produkálni, valamint a sok hiba miatt a következő generáció is nehezen sajátítja azt el. Vagyis az adott nyelvtípus evolúciósan nem stabil.[7] Ha az SA-OT algoritmus, legalább elvileg, képes ilyen típusú magyarázatra, az jó hír a számunkra: ugyanis reményt ad arra, hogy az Univerzális Grammatika modelljeit nem kell túlkomplikálni, ha a faktoriális tipológia történetesen nem képes megmagyarázni egy kis űrt a megfigyelhető nyelvtipológiában. Lehetséges ugyanis, hogy ezt az űrt nem az UG szintjén kell megmagyarázni: az UG megengedné, de azért nem fordul elő a típus, mert nehéz lenne helyesen produkálni.

2. Táblázat: Az SA-OT algoritmus pontossága különböző hierarchiák esetén

| *Hierarchia* | *%* |
|---|---|
| prs fio fin noc ons | 31 |
| prs fio fin ons noc | 26 |
| prs fio noc fin ons | 78 |
| prs fio noc ons fin | 84 |
| prs fio ons fin noc | 14 |
| prs fio ons noc fin | 72 |
| prs fin fio noc ons | 38 |
| prs fin fio ons noc | 25 |
| prs fin noc fio ons | 30 |

# 5 Összefoglalás

A jelen cikk célja egyrészt az volt, hogy a magyar nyelvű szakmai közönségnek bemutassam a szimulált hőkezelés társítását az Optimalitáselmélettel, azaz az SA-OT algoritmust. Az eljárás részletei mellett több korábbi cikkben, valamint a készülő disszertációmban részletesen érvelek, különböző típusú – matematikai, „filozófiai”, empirikus – érveket hozva. Az érdeklődő kipróbálhatja az algoritmus demonstrálására elkészített weblapot is a *http://www.let.rug.nl/~birot/sa-ot/* oldalon.

A cikk fő célja viszont annak a bemutatása volt, hogy milyen szerepet játszik az algoritmusban a jelöltek halmazán bevezetett topológia (szomszédsági struktúra), különösen pedig az *a priori* valószínűségek. Ezen fogalmak újak a hagyományos

[7] Ez a gondolatmenet beleilleszkedik abba a kutatási programba, amely a Chomskyánus megközelítést – ha egy nyelvtípus nem létezik, akkor azt az Univerzális Grammatika modelljének kell kizárnia – más megközelítésekkel igyekszik kiegészíteni. Jäger (2003) megmutatja, hogy a faktoriális tipológia által megengedett egyes típusok azért nem léteznek a világ nyelvei között, mert ezek a típusok evolucionárisan nem stabilak. Boersma (2004) pedig olyan magyarázatot javasol, amely szerint egyes típusokat nehezebb elsajátítani egy tanulóalgoritmussal.

Optimaitáselméletben, habár *Paul Smolensky* konnekcionista munkái is használnak egy szomszédsági struktúrát. Ebből a célból megvizsgáltuk, hogy miként alkalmazható az SA-OT az Optimalitáselmélet talán leggyakrabban használt példájára, a bemenet szótagstruktúrájának a kiszámítására. Láttuk, hogy hogyan vezethetők be nem-triviális *a priori* valószínűségek néhány paraméter segítségével – szemben például a Bíró (2005b) által használt modellel, amely a topológia szempontjából rendkívül egyszerű. Valamint azt is láttuk, hogy e paraméterek variálásával az algoritmus kimenete, pontossága erősen változhat. Hipotézisem szerint az *a priori* valószínűségek kiszámításának a módja univerzális, de a paraméterek pontos értékei változhatnak nyelvenként, regiszterenként, vagy akár beszélőnként.

A cikkben bemutatott eredmények egyelőre nem meggyőzőek abból a szempontból, hogy konkrétan a szótagolás jól implementálható dinamikus programozással (Tesar és Smolensky, 2000), és a véges állapotú automatákkal is meglepően jól működik, szemben az elméleti várakozásokkal (Gerdemann és van Noord, 2000), miközben az SA-OT algoritmus egyelőre meglehetősen alacsony pontosságot produkált. De a bemutatott eredmények egy folyamatban levő munka részeredményei, amelyek továbbgondolásra szorulnak. Továbbá, a pontosság radikálisan növelhető, ha néhány szimulációt párhuzamosan futtatunk. Még ha egy szimuláció csupán 20% valószínűséggel találja is meg a helyes alakot, ha tíz szimulációt futtatunk párhuzamosan, 90%-ot meghaladja annak az esélye, hogy a tíz közül legalább az egyik a keresett alak lesz. Márpedig tíz output alak közül megkeresni a legjobbat egy könnyű feladat, összehasonlíthatatlanul könnyebb, mint egy végtelen halmaz legjobb elemének a megtalálása. Emlékezzünk, épp ebből az utóbbi célból alkalmaztuk a szimulált hőkezelést.

## Köszönetnyilvánítás

A köszönetemet fejezem ki a hollandiai Groningeni Egyetem (Rijksuniveriteit Groningen) *High Performance Computing* programjának, valamint témavezetőimnek és konzulenseimnek, *John Nerbonne*-nak, *Gosse Bouma*-nak és *Gertjan van Noord*-nak a segítségükért a cikk alapjául szolgáló kutatásban.

## Bibliográfia

1. Bíró, T.: Az Optimalitáselmélet megvalósítása szimulált hőkezeléssel, szemináriumi dolgozat Rebrus Péter órájára, ELTE Elméleti Nyelvészet, 1997.
2. Bíró, T.: Quadratic Alignment Constraints and Finite State Optimality Theory, In: Proc. FSMNLP, pp. 119-126, Budapest, (valamint: *ROA-600,* http://roa.rutgers.edu), 2003.
3. Bíró, T.: Squeezing the Infinite into the Finite: Handling the OT Candidate Set with Finite State Technology, Proc. FSMNLP, Helsinki, 2005a.
4. Bíró, T.: When the Hothead Speaks: Simulated Annealing Optimality Theory for Dutch Fast Speech, In: Cremers, C., Reckman, H., Poss, M., van der Wouden, T. (eds.): Selected Papers of the 15th Meeting of Computational Linguistics in the Netherlands, Leiden, 2005b.
5. Bíró, T.: How to Define Simulated Annealing for Optimality Theory? In: Proc. 10th Conf. on Formal Grammar and 9th Meeting on Mathematics of Language, Edinburgh, 2005c

6. Bíró, T.: Finding the Right Words: Implementing Optimality Theory with Simulated Annealing, PhD disszertáció, University of Groningen, Hollandia, 2006.
7. Boersma, P.: Review of *B. Tesar* & *P. Smolensky* (2000): Learnability in Optimality Theory, *ROA-63*8 (Rutgers, Optimality Archives, http://roa.rutgers.edu), 2004.
8. Eisner, J.: Easy and Hard Constraint Ranking in Optimality Theory: algorithms and complexity, In: Eisner, J., Karttunen, L., Thériault, A. (eds.): Finites-State Phonology, Proceedings of SIGPHON 5, pp. 57-67, Luxemburg, 2000.
9. Ellison, T.M.: Phonological Derivation in Optimality Theory, In: Proc. COLING-94, pp. 1007-1013, Kyoto, 1994.
10. Frank, R., Satta, G.: Optimality Theory and the Generative Complexity of Constraint Violability, In: Computational Linguistics, 24(2):307-315, 1998
11. Gerdemann, D., van Noord, G.: Approximation and Exactness in Finite State Optimality Theory, In: Eisner, J., Karttunen, L., Thériault, A. (eds.): Proc. SIGPHON, 2000.
12. Jäger, G.: Simulating language change with functional OT,. In: Kirby, S. (ed.): *Language Evolution and Computation, Proc. of the Workshop at ESSLLI, Vienn*a, pp. 52–61, 2003.
13. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science no. 4598, vol. 20, pp. 671-680, 1983
14. Kuhn, J.: Processing Optimality-theoretic Syntax by Interleaved Chart Parsing and Generation, In: Proc. ACL-2000, pp. 360-367, Hongkong, 2000.
15. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of State Calculation by Fast Computing Machines. J. Chemical Physics, vol. 21, no. 6, pp. 1087-1092, 1953.
16. Prince, A., Smolensky, P.: Optimality Theory: Constraint Interaction in Generative Grammar, RuCCS-TR-2, 1993 / Blackwell, Malden, MA, 2004.
17. Rebrus, P.: Optimalitáselmélet, In: Siptár, P. (szerk.): Szabálytalan fonológia, Tinta, 2001.
18. Reeves, C.R.: Modern Heuristic Techniques for Combinatorial Problems, McGraw-Hill, London, 1995.
19. Tesar, B., Smolensky, P.: Learnability in Optimality Theory, MIT Press, Cambridge, 2000.