# From Performance Errors to Optimal Competence
## Learnability of OT and HG with Simulated Annealing


Tamás Biró

University of Amsterdam, Netherlands

t.s.biro@uva.nl



Amsterdam Center for Language and Communication ACLC

Tools for OT

NWO Netherlands Organisation for Scientific Research


### COMPETENCE as linguistic optimization

Generative linguistics as an optimization problem: how to map underlying form $U$ onto surface form SF($U$)?

$$\mathrm{SF}(U) = \underset{w \in \mathrm{Gen}(U)}{\arg\,\mathrm{opt}}\; H(w)$$

*Target function* ("Harmony") $H(w)$ derived from *elementary functions* $C_i(w)$ ("constraints" – a misnomer):

1. Hard constraints: $H(w) = C_1(w)\ \&\ C_2(w)\ \&\ ...\ \&\ C_N(w)$ → P&P
2. Weighted sum: $H(w) = g_N \cdot C_N(w) + g_{N-1} \cdot C_{N-1}(w) + ... + g_1 \cdot C_1(w)$. → HG
3. Exponential weights: $H(w) = -C_N(w) \cdot q^N - C_{N-1}(w) \cdot q^{N-1} - ... - C_1(w) \cdot q$ → $q$-HG
4. OT tableau row: $H(w) = \boxed{C_N(w)}\boxed{C_{N-1}(w)}\boxed{...}\boxed{C_1(w)}$ → OT

**Rank** $r_i$ for each constraint $C_i$. We focus on OT and $q$-HG.

**Grammar = set of ranks.**

Sort constraints by rank:

- OT hierarchy: $C_i \gg C_j$ iff $r_i > r_j$.
- HG weights: $g_i = q^{K_i}$ such that $K_i > K_j$ iff $r_i > r_j$, and $K_i \in \{1, ..., N\}$.

### PERFORMANCE or production as implementation

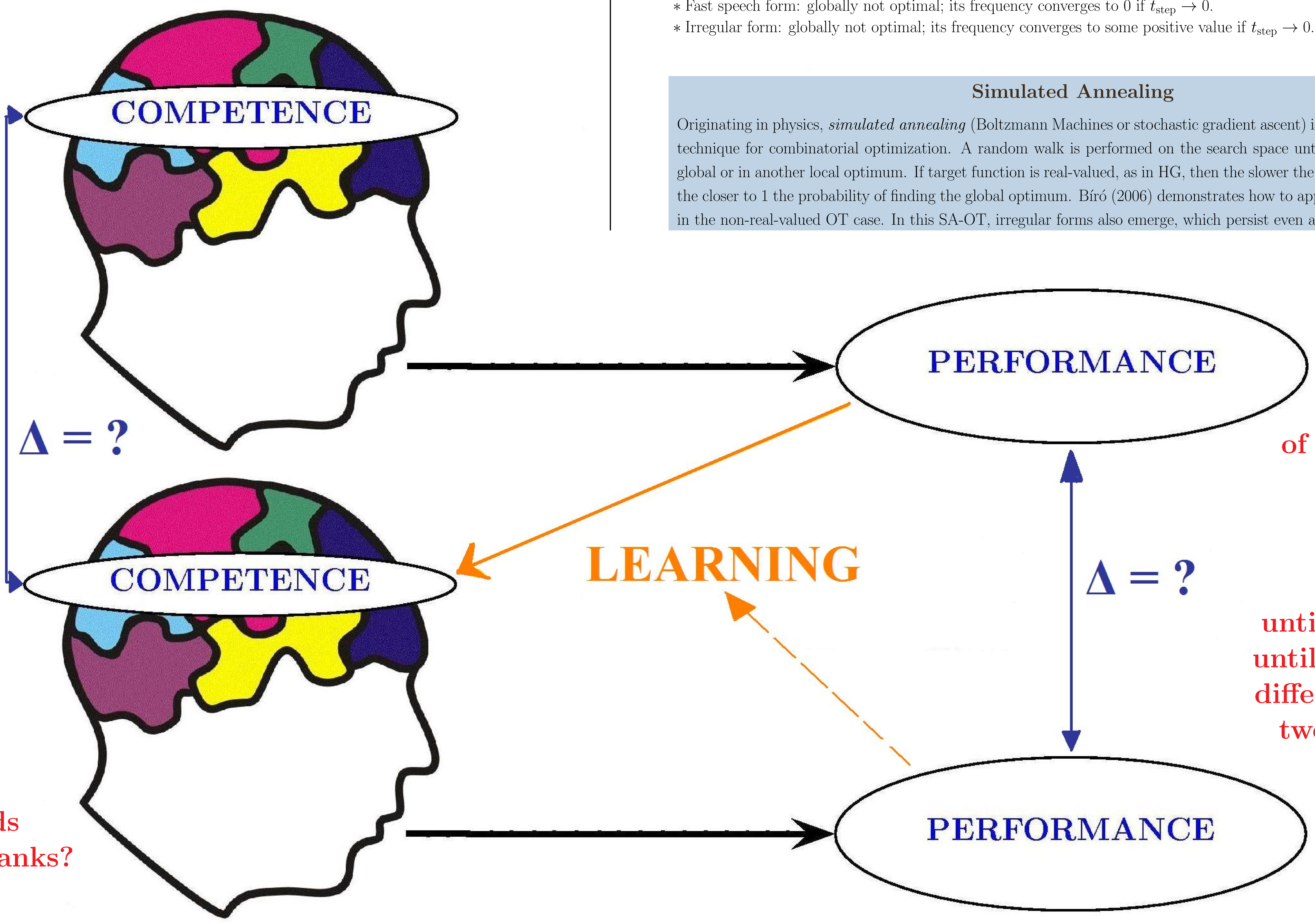
| | | |
|---|---|---|
| 1. *Competence*: the static knowledge | grammatical forms | (explained by) grammar |
| 2. *Mental computation* in the brain | produced forms | implementation of grammar |
| 3. *Performance* in its outmost sense | produced forms | phonetics, pragmatics, etc. |

Cf. Bíró (2006:43); Smolensky and Legendre (2006:vol. 1. p. 228). Ways to implement HG and OT:

- **Grammatical:** return the most harmonic candidate (exhaustive search; FS-OT, dynamic programming).
- **Simulated annealing:** return local optima, depending on *cooling schedule* ($t_{\text{step}}$: step by which temperature is decreased in each iteration, "speed").
  - HG: sa converges to gr (frequency of global optimum converges to 1) if $t_{\text{step}} \to 0$ (more iterations).
  - OT: grammatical forms, irregular forms and fast speech forms are returned (Biró 2007):
    - Grammatical form: globally optimal.
    - Fast speech form: globally not optimal; its frequency converges to 0 if $t_{\text{step}} \to 0$.
    - Irregular form: globally not optimal; its frequency converges to some positive value if $t_{\text{step}} \to 0$.

#### Simulated Annealing

Originating in physics, *simulated annealing* (Boltzmann Machines or stochastic gradient ascent) is a widespread heuristic technique for combinatorial optimization. A random walk is performed on the search space until being trapped in the global or in another local optimum. If target function is real-valued, as in HG, then the slower the speed of the algorithm, the closer to 1 the probability of finding the global optimum. Bíró (2006) demonstrates how to apply simulated annealing in the non-real-valued OT case. In this SA-OT, irregular forms also emerge, which persist even at slow speed.




Can you look into the brain of the teacher?

Online learning from teacher's performance.

Any need to learn until convergence on the grammar? Until the learner finds the teacher's set of ranks?

Distance of performances: *JSD.*

Learn until convergence: until no significant difference between two performance samples.

### Gradual online LEARNING needs production (that is, performance)

Repeated error-driven updates of the constraint ranks $r_i$, until convergence:

- Initially: fix random target grammar, fix underlying form, initial random grammar for learner.
- **Error-driven:** "winner" produced by target grammar vs. "loser" produced by learner's current grammar.
- **Update rule:** update the rank $r_i$ of every constraint $C_i$, depending on whether $C_i$ prefers the winner or the loser. Two approaches ($\epsilon = 0.1$, while ranks are initially random numbers between 0 and $N = 15$):
  - Boersma (1997): increase rank by $\epsilon$ if winner-preferring; decrease rank by $\epsilon$ if loser-preferring constraint.
  - Magri (2009): increase rank of all winner-preferring constraints by $\epsilon$; decrease rank of highest ranked loser-preferring constraint by $W \cdot \epsilon$, where $W$ is the number of winner-preferring constraints.

### Converging on performance: Δ is Jensen-Shannon divergence

- **Convergence criterion:** *JSD* between sample produced by target grammar and sample produced by learner's current grammar ≤ average *JSD* of two samples produced by target grammar. (Sample size = 100).

A measure of the "distance" of two distributions: $JSD(P\|Q) = \frac{D(P\|M)+D(Q\|M)}{2}$
where $D(P\|Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$ (relative entropy, Kullback-Leibler divergence), and $M(x) = \frac{P(x)+Q(x)}{2}$.

- Symmetric: $JSD(P\|Q) = JSD(Q\|P)$. Finite and non-negative: $0 \leq JSD(P\|Q) \leq 1$.
- $JSD(P\|Q) = 0$ if and only if $P(x) = Q(x),\ \forall x$. $JSD(P\|Q) = 1$ if and only if $P(x) \cdot Q(x) = 0,\ \forall x$.
- Same language: $JSD(L_t\|L_l) = 0$. Not a single overlap: $JSD(L_t\|L_l) = 1$.

#### String Grammar: "toy grammar" imitating typical OT phonology:

- *Candidates:* $\mathrm{Gen}(U) = \{0, 1, ..., P-1\}^L$. We have used $L = P = 4$: 0000, 0001, 0120, 0123,... 3333.
- *Neighborhood structure* on this candidate set: $w$ and $w'$ neighbors iff one basic step transforms $w$ to $w'$.
  *Basic step:* change exactly one character $\pm 1 \pmod P$ (cyclicity). Each neighbor with equal probability.
  Example: neighbors of 0123 are exactly 1123, 3123, 0023, 0223, 0113, 0133, 0122 and 0120.
- *Constraints* (for all $n \in \{0, 1, ..., P-1\}$):
  - No-$n$ (number of character $n$ in string): $*n(w) := \sum_{i=0}^{L-1}(w_i = n)$.
  - No-initial-$n$: $*\textsc{Initial}n(w) := (w_0 = n)$. No-final-$n$: $*\textsc{Final}n(w) := (w_{L-1} = n)$.
  - Assimilation (number of different adjacent character pairs): $\textsc{Assim}(w) := \sum_{i=0}^{L-2}(w_i \neq w_{i+1})$.
  - Dissimilation (number of identical adjacent character pairs): $\textsc{Dissim}(w) := \sum_{i=0}^{L-2}(w_i = w_{i+1})$.
  - Faithfulness to underlying form $U$ (using pointwise distance modulo $P$):
    $\textsc{Faith}(w) = \sum_{i=0}^{L-1} d(U_i, w_i)$ where $d(a,b) = \min(|(a-b) \bmod P|, |(b-a) \bmod P|)$.

#### Experiment: Measuring number of learning steps

2000 times learning (rnd target, rnd underlying form) per grammar type, production method and learning method.
Distribution of the number of learning steps until convergence: *1st quartile* ; **median** ; *3rd quartile* ; 90th percentile

| | | OT | 10-HG | 4-HG | 1.5-HG |
|---|---|---|---|---|---|
| gramm. | M | *13* ; **27** ; *45* ; 67 | *13* ; **28** ; *46* ; 70 | *12* ; **27** ; *48* ; 69 | *15* ; **30** ; *47* ; 67 |
| | B | *23* ; **43** ; *65* ; 102 | *22* ; **41** ; *64* ; 107 | *22* ; **42** ; *64* ; 107 | *23* ; **40** ; *60* ; 90 |
| sa, | M | *53* ; **109** ; *233* ; 497 | *63* ; **140** ; *328* ; 1681 | *60* ; **148** ; *366* ; 1517 | *83* ; **199** ; *508* ; 1702 |
| $t_{\text{step}} = 0.1$ | B | *80* ; **171** ; *462* ; 1543 | *92* ; **240** ; *772* ; 7512 | *92* ; **239** ; *785* ; 8633 | *117* ; **290** ; *694* ; 1956 |
| sa, | M | *64* ; **131** ; *305* ; 1022 | *62* ; **134** ; *304* ; 1127 | *63* ; **137** ; *329* ; 1278 | *72* ; **163** ; *437* ; 2229 |
| $t_{\text{step}} = 1$ | B | *90* ; **212** ; *560* ; 1966 | *92* ; **233** ; *572* ; 3116 | *84* ; **212** ; *646* ; 3005 | *101* ; **242** ; *616* ; 2091 |

#### OBSERVATIONS: very long tail (significance based on Wilcoxon rank-sum test)

- Performance errors make grammars more difficult to learn: *gramm < 0.1-sa < 1-sa.*
- But reversed for HG and SA (either significant, or not significant tendency). Why?
- Magri's update rule (M) quicker than Boersma's (B) (extremely significant). Due to larger update steps?
- Grammar type (OT vs. $q$-HG): only minor influence ("hardly any" and "small, but very significant"). OT much easier to learn than 1.5-HG (significant difference for sa cases). NB: also quicker to produce.

Does learning speed depend on initial grammar? On order of learning data?
**New experiment:** Run two learners learning the same target grammar:

- With same initial grammar: strong correlation in their nr. of learning steps. Learning data not the same: slightly decreased correlation.
- With different initial grammars: correlation (almost) lost, large differences in learning time.

Cf. long tail: children must start with same initial grammar, but need not receive same (correct or erroneous) data (if algorithm is correct). SLI is being born with initial grammar causing 10-20 times higher learning steps?
**Stability of the algorithms**: if learner reached target, will she stay there? Much improvement needed.

### References


T. Bíró (2006). *Finding the Right Words.* PhD thesis, University of Groningen. ROA-896.
T. Bíró (2007). 'The benefits of errors'. In *Proc. Workshop Cognitive Aspects of Comp. Lang. Acquisition*, ACL. ROA-929.
P. Boersma (1997). 'How we learn variation, optionality, and probability'. IFA Proceedings 21: 43-58.
G. Magri (2009). 'New update rules for on-line algorithms for the Ranking problem in OT'. Handout. LMA workshop, DGfS 31.
P. Smolensky and G. Legendre (eds.) (2006). *The Harmonic Mind.* MIT Press, Cambridge.