Finding the Right Words
Implementing Optimality Theory with Simulated Annealing

Tamás Bíró

Work presented developed at:
Humanities Computing
CLCG
University of Groningen

Present affiliation:
Theoretical Linguistics
ACLC
University of Amsterdam

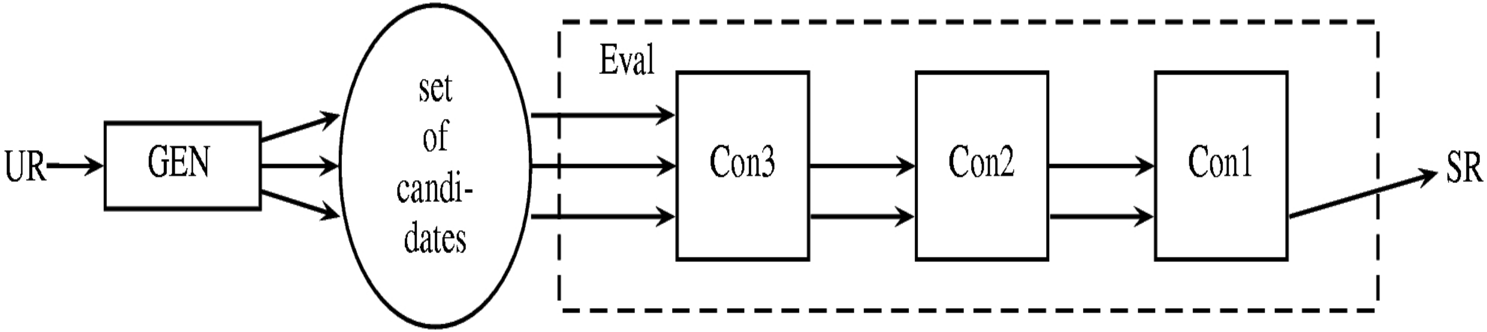
birot@nytud.hu

Amsterdam, ILLC, March 2, 2007

Overview

- Optimality Theory (OT) in a nutshell
- Simulated Annealing for Optimality Theory (SA-OT)
- Examples
- The dis-harmonic mind?
- Conclusion

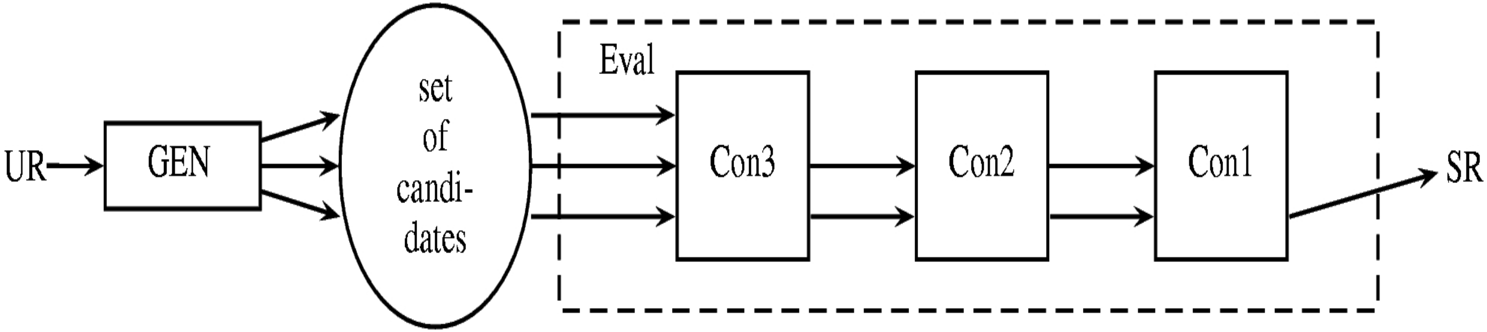
Optimality Theory in a nutshell



OT tableau: search the best candidate w.r.t lexicographic ordering (cf. *abacus*, *abolish*,..., *apple*,..., *zebra*)

	c_n	c_{n-1}	...	c_{k+1}	c_k	c_{k-1}	c_{k-2}	...
w	2	0		1	2	3	0	
w'	2	0		1	3 !	1	2	
w"	3 !	0		1	3	1	2	

Optimality Theory in a nutshell

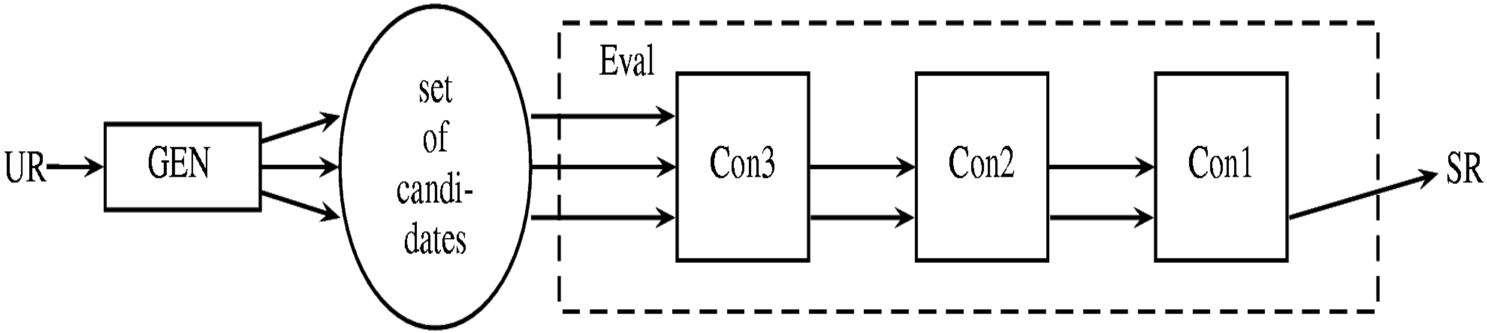


- Pipe-line *vs.* optimize the Eval-function

- Gen: $UR \mapsto \{w | w \text{ } is \text{ } a \text{ } candidate \text{ } corresponding \text{ } to \text{ } UR\}$

  E.g. assigning Dutch metrical foot structure & stress:
  fototoestel $\mapsto$ {fototoe(stél), (fotó)(tòestel), (fó)to(toestèl),...}

Optimality Theory: an optimization problem



$$UR \mapsto \{w | w \ is \ a \ candidate \ corresponding \ to \ UR\}$$
$$E(w) = \Big(C_N(w), C_{N-1}(w), ..., C_0(w)\Big) \in \mathbb{N}_0^{N+1}$$
$$SR(UR) = \mathrm{argopt}_{w \in Gen(UR)} E(w)$$

Optimization: with respect to lexicographic ordering

OT is an optimization problem

The question is:
How can the optimal candidate be found?

- Finite-State OT (Ellison, Eisner, Karttunen, Frank & Satta, Gerdemann & van Noord, Jäger...)

- chart parsing (dynamic programming) (Tesar & Smolensky; Kuhn)

These are perfect for language technology. But we would like a psychologically adequate model of linguistic performance (e.g. errors): Simulated Annealing.

How to find optimum: Gradient Descent 1

```
w := w_init ;
Repeat
        w':= best element of set Neighbours(w);
        Delta :=  E(w') - E(w) ;
        if   Delta < 0   then   w := w' ;
        else
               do nothing

        end-if



Until stopping condition = true

Return w          # w is an approximation to the optimal solution
```

How to find optimum: Gradient Descent 2

```
w := w_init ;
Repeat
        Randomly select w' from the set Neighbours(w);
        Delta :=  E(w') - E(w) ;
        if   Delta < 0   then   w := w' ;
        else
               do nothing

        end-if


Until stopping condition = true

Return w          # w is an approximation to the optimal solution
```

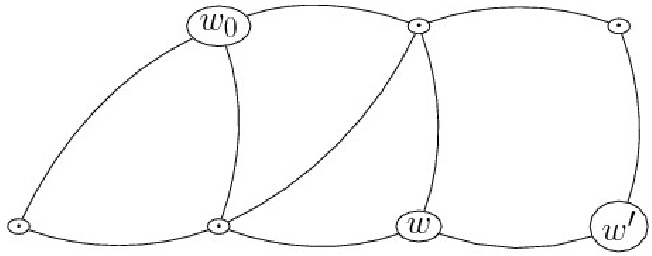
The Simulated Annealing Algorithm

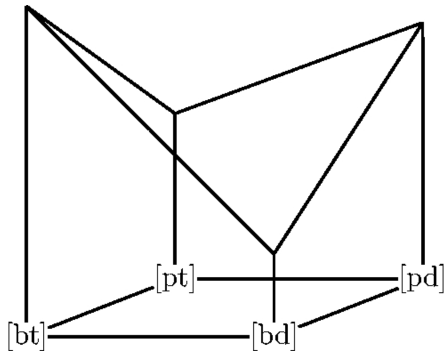
```
w := w_init ;       t := t_max  ;
Repeat
        Randomly select w' from the set Neighbours(w);
        Delta :=  E(w') - E(w) ;
        if   Delta < 0   then   w := w' ;
        else
               generate random r uniformly in range (0,1) ;
               if   r < exp(-Delta / t)   then   w := w' ;
        end-if

   t := alpha(t)                             # decrease t
Until stopping condition = true

Return w         # w is an approximation to the optimal solution
```

Deterministic Gradient Descent for OT

- McCarthy (2006): *persistent OT* (*harmonic serialism*, cf. Black 1993, McCarthy 2000, Norton 2003).
- Based on a remark by Prince and Smolensky (1993/2004) on a "restraint of analysis" as opposed to "freedom of analysis".
- Restricted Gen $\rightarrow$ Eval $\rightarrow$ Gen $\rightarrow$ Eval $\rightarrow$... (n times).
- Gradual progress toward (locally) max. harmony.
- Employed to simulate traditional derivations, opacity.

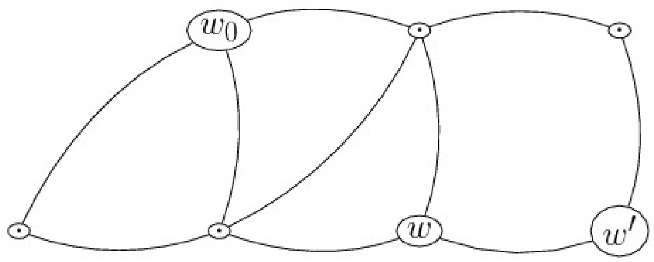
Simulated Annealing for OT – general idea

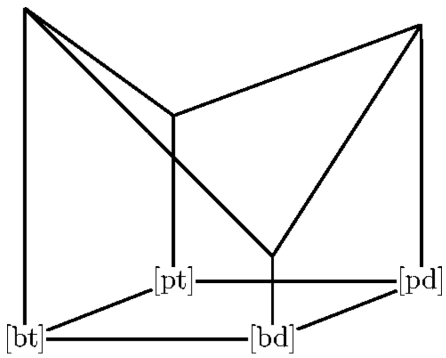




- Neighbourhood structure on the candidate set.
- Landscape's vertical dimension = harmony; random walk.
- If neighbour more optimal: move.
- If less optimal: move in the beginning, don't move later.

Simulated Annealing for OT – general idea





- Neighbourhood structure → local optima.
- System can get stuck in local optima: alternation forms.
- Precision of the algorithm depends on its speed (!!).
- Many different scenarios.

Sim. annealing with non-real valued target function

- Exponential weights if upper bound on $C_i(w)$ violation levels:

$$E(w) = C_N(w) \cdot q^N + C_{N-1}(w) \cdot q^{N-1} + ... + C_1(w) \cdot q + C_0(w)$$

- Polynomials:

$$E(w)[q] = C_N(w) \cdot q^N + C_{N-1}(w) \cdot q^{N-1} + ... + C_1(w) \cdot q + C_0(w)$$

- Ordinal weights:

$$E(w) = \omega^N C_N(w) + ... + \omega C_1(w) + C_0(w)$$

Sim. annealing with non-real valued target function

Transition probability if w' worse than w: what is $e^{\frac{E(w')-E(w)}{t}}$?

- Polynomials:

$$
\begin{aligned}
T[q] &= \langle K_T, t \rangle [q] = t \cdot q^{K_T} \\
P\Big(w \rightarrow w' \mid T[q]\Big) &= \lim_{q \rightarrow +\infty} e^{-\frac{E(w')[q]-E(w)[q]}{T[q]}}
\end{aligned}
$$

- Ordinals: move iff the generated $r \in [0, 1]$ is s.t.

$$
\forall \alpha \in \mathbb{N}^+ : r^{-\alpha} > 2^{q\left(\Delta\left(E(w'), E(w)\right) \cdot \alpha, T\right)}
$$

Domains for temperature and constraints

- Temperature: $T = \langle K_T, t \rangle \in \mathbb{Z} \times \mathbb{R}^+$ (or "$\mathbb{Z}$" $\times \mathbb{R}^+$).

- Constraints associated with domains of K_T:

	$- -$	C_0	C_1	C_2
$\dots$	$K = -1$	$K = 0$	$K = 1$	$K = 2$
$\dots$	... 0.5 1.0 1.5 2.0 2.5	... 0.5 1.0 1.5 2.0 2.5	... 0.5 1.0 1.5 2.0 2.5	... 0.5 1.0 1.5 2.0 2.5

Rules of moving

RULES OF MOVING from w to w'
at temperature $T = \langle K_T, t \rangle$:

If w' is better than w: move! $P(w \rightarrow w'|T) = 1$

If w' loses due to fatal constraint C_k:

If $k > K_T$: don't move! $P(w \rightarrow w'|T) = 0$

If $k < K_T$: move! $P(w \rightarrow w'|T) = 1$

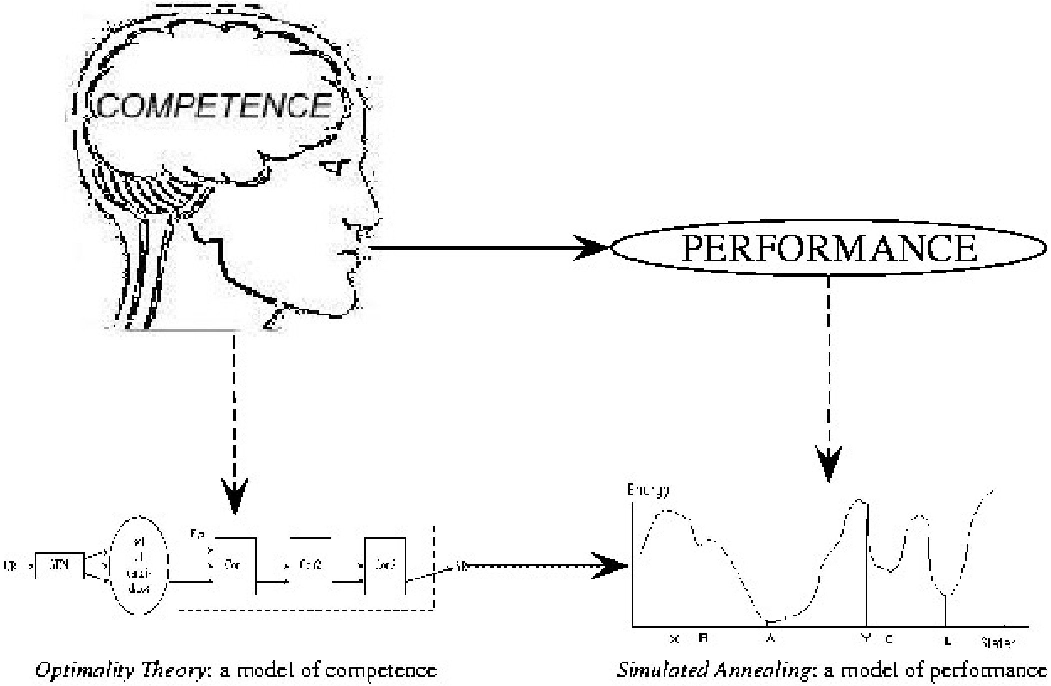
If $k = K_T$: move with probability

$$P = e^{-(C_k(w') - C_k(w))/t}$$

The SA-OT algorithm

```
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) ;
             COMPARE  w' to w:  C := fatal constraint
                                d := C(w') - C(w);
             if d <= 0 then w := w';
             else           w := w' with probability
                        P(C,d;K,t) = 1         , if C < K
                                   = exp(-d/t) , if C = K
                                   = 0         , if C > K
      end-for
end-for
return w
```

SA-OT as a model of linguistic performance



Proposal: three levels

Level	its product	its model	the product in the model
Competence in narrow sense: static knowledge of the language	grammatical form	standard OT grammar	globally optimal candidate
Dynamic language production process	acceptable or attested forms	SA-OT algorithm	local optima
Performance in its outmost sense	acoustic signal, etc.	(phonetics, pragmatics)	??

The art of using Simulated Annealing Optimality Theory

- Take a traditional OT model
- Add *convincing* neighbourhood structure to candidate set
- Local (non-global) optima = alternation forms
- Run simulation (e.g., `http://www.let.rug.nl/~birot/sa-ot`):
  - Slowly: likely to return only the grammatical form
  - Quickly: likely to return local (non-global) optima

Parameters of the algorithm

- t_{step} (and t_{max}, t_{min})
- K_{max} (and K_{min})
- K_{step}
- w_0 (inital candidate)
- Topology (neighbourhood structure)
- Constraint hierarchy

How to make the topology convincing?

A connected (weighted) "graph"; universal;...

- Observation-driven strategies:
  - Many phenomena in many languages
    or even better: cross-linguistic typologies
  - Based on existing theories based on cross-linguistic observations (cf. Hayes's *metrical stress theory*)
- Theory-driven strategies:
  - Principles (e.g. minimal set of basic transformations)
  - Psycholinguistically relevant notions of similarity, etc.

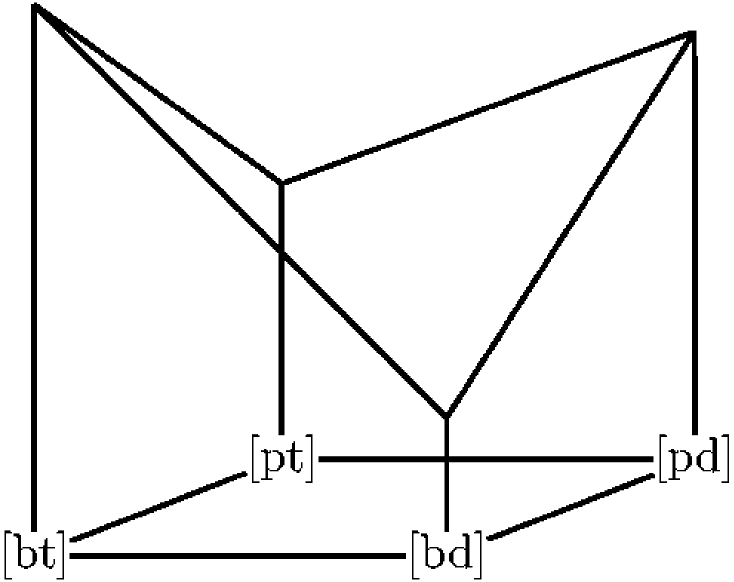
Example: Fast speech: Dutch metrical stress

fo.to.toe.stel 'camera'	*uit.ge.ve.rij* 'publisher'	*stu.die.toe.la.ge* 'study grant'	*per.fec.tio.nist* 'perfectionist'
susu	ssus	susuu	usus
fó.to.tòe.stel fast: *0.82* slow: *1.00*	*ùit.gè.ve.ríj* fast: *0.65* / **0.67** slow: *0.97* / **0.96**	*stú.die.tòe.la.ge* fast: *0.55* / **0.38** slow: *0.96* / **0.81**	*per.fèc.tio.níst* fast: *0.49* / **0.13** slow: *0.91* / **0.20**
fó.to.toe.stèl fast: *0.18* slow: *0.00*	*ùit.ge.ve.ríj* fast: *0.35* / **0.33** slow: *0.03* / **0.04**	*stú.die.toe.là.ge* fast: *0.45* / **0.62** slow: *0.04* / **0.19**	*pèr.fec.tio.níst* fast: *0.39* / **0.87** slow: *0.07* / **0.80**

Simulated / **observed** (Schreuder) frequencies.

In the simulations, $T_{step} = 3$ used for fast speech and $T_{step} = 0.1$ for slow speech.

Example: Irregularities



- Local optimum that is not avoidable.

Example: string-grammar

- *Candidates:* $\{0, 1, ..., P-1\}^L$
  E.g. $(L = P = 4)$: 0000, 0001, 0120, 0123,... 3333.

- *Neighbourhood structure:* w and w' neighbours iff one basic step transforms w to w'.

- Basic step: change exactly one character $\pm 1, \mod P$ (cyclicity).

- Each neighbour with equal probability.

Example: string-grammar

Markedness Constraints ($w = w_0 w_1 ... w_{L-1}$, $0 \leq n < P$):

- No-n: $*n(w) := \sum_{i=0}^{L-1} (w_i = n)$
- No-initial-n: $*\textsc{Initial}n(w) := (w_0 = n)$
- No-final-n: $*\textsc{Final}n(w) := (w_{L-1} = n)$
- Assimilation $\textsc{Assim}(w) := \sum_{i=0}^{L-2} (w_i \neq w_{i+1})$
- Dissimilation $\textsc{Dissim}(w) := \sum_{i=0}^{L-2} (w_i = w_{i+1})$

Example: string-grammar

- Faithfulness to UR σ:

$$\textsc{Faith}_\sigma(w) = \sum_{i=0}^{L-1} d(\sigma_i, w_i)$$

where $d(a, b) = \min(|a - b|, |b - a|)$

(binary square, feature-combination?)

Example: string-grammar

$L = P = 4$, $T_{max} = 3$, $T_{min} = 0$, $K_{step} = 1$.

Each of the 256 candidates used 4 times as w_0.

Grammar:

*0 $\gg$ Assim $\gg$ Faithf$_{\sigma=0000}$ $\gg$ *Init1 $\gg$*Init0 $\gg$ *Init2 $\gg$ *Init3 $\gg$ *Fin0 $\gg$ *Fin1 $\gg$ *Fin2 $\gg$ *Fin3 $\gg$ *3 $\gg$ *2 $\gg$ *1 $\gg$ Dissim

Globally optimal form: 3333

Many other local optima, e.g.: 1111, 2222, 3311, 1333, etc.

Example: string-grammar

Output frequencies for different T_{step} values:

output	0.0003	0.001	0.003	0.01	0.03	0.1
1111	0.40	0.40	0.36	0.35	0.32	0.24
3333	0.39	0.39	0.41	0.36	0.34	0.21
2222	0.14	0.14	0.15	0.18	0.19	0.17
3311	0.04	0.04	0.04	0.05	0.06	0.05
1133	0.03	0.04	0.04	0.04	0.05	0.04
others	–	–	–	–	0.04	0.29

What does SA-OT offer to standard OT?

- A new approach to account for variation:
  - Non-optimal candidates also produced (cf. Coetzee);
  - As opposed to: more candidates with same violation profile; more hierarchies in a grammar.
- A topology (neighbourhood structure) on the candidate set.
- Additional ranking arguments (cf. McCarthy 2006) $\longrightarrow$ learning algorithms (in progress).
- Arguments for including losers (never winning candidates).

The *dis*-harmonic mind?

ICS (Integrated Connectionist/Symbolic Cognitive Architecture):

"[T]here is no symbolic algorithm whose internal structure can predict the time and the accuracy of processing; this can only be done with connectionist algorithms" (Smolensky and Legendre (2006): *The Harmonic Mind*, vol. 1, p. 91).

SA-OT:

- symbolic computation only
- predicts tme and accuracy of processing

Summary of SA-OT

- Implementing OT: lang. technology? cognitively plausible?
- A model of variation / performance phenomena.
- *Errare humanum est*: heuristics in cognitive science.
- Time and accuracy with a symbolic-only architecture.
- Much work needed: learnability, linguistic examples, etc.
- Demo at `http://www.let.rug.nl/~birot/sa-ot`.

Thank you for your attention!

Tamás Bíró

birot@nytud.hu