Learning an Optimality Theoretical grammar with a structured candidate set

Tamás Bíró

Work presented developed at:
Humanities Computing
CLCG
University of Groningen

Present affiliation:
Theoretical Linguistics
ACLC
University of Amsterdam

birot@nytud.hu

Bielefeld, January 10, 2007
Learning
(an Optimality Theoretical grammar with a structured candidate set)

Tamás Bíró

Work presented developed at:
Humanities Computing
CLCG
University of Groningen

Present affiliation:
Theoretical Linguistics
ACLC
University of Amsterdam

birot@nytud.hu

Bielefeld, January 10, 2007
*Learning
(an Optimality Theoretical grammar)
(with a structured candidate set)

Tamás Bíró

Work presented developed at:
Humanities Computing
CLCG
University of Groningen

Present affiliation:
Theoretical Linguistics
ACLC
University of Amsterdam

birot@nytud.hu

Bielefeld, January 10, 2007
Overview

• Optimality Theory (OT) in a nutshell

• Simulated Annealing for Optimality Theory (SA-OT)

• Examples

• Learnability?

• Conclusion
Optimality Theory in a nutshell

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

<table>
<thead>
<tr>
<th></th>
<th>$c_n$</th>
<th>$c_{n-1}$</th>
<th>...</th>
<th>$c_{k+1}$</th>
<th>$c_k$</th>
<th>$c_{k-1}$</th>
<th>$c_{k-2}$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w$</td>
<td>2</td>
<td>0</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$w'$</td>
<td>2</td>
<td>0</td>
<td></td>
<td>1</td>
<td>3!</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$w''$</td>
<td>3!</td>
<td>0</td>
<td></td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Optimality Theory in a nutshell

- Pipe-line vs. optimize the Eval-function

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

  E.g. assigning Dutch metrical foot structure & stress:
  fototoestel $\mapsto \{\text{fototoe(stél)}, (\text{fotó})(\text{tòestel}), (\text{fó})\text{to}(\text{toestèl}), \ldots\}$
Optimality Theory: an optimisation problem

\[ UR \mapsto \{ w \mid w \text{ is a candidate corresponding to } UR \} \]

\[ E(w) = \left( C_N(w), C_{N-1}(w), \ldots, C_0(w) \right) \in \mathbb{N}_0^{N+1} \]

\[ SR(UR) = \text{argopt}_{w \in Gen(UR)} E(w) \]

Optimisation: 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 programing)** (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

\[
\begin{align*}
w &:= w_{\text{init}}; \\
\text{Repeat} & \\
\quad &\text{Randomly select } w' \text{ from the set } \text{Neighbours}(w); \\
\quad &\text{Delta} := E(w') - E(w); \\
\quad &\text{if } \Delta < 0 \text{ then } w := w'; \\
\quad &\text{else} \\
\quad &\quad \text{do nothing} \\
\text{end-if} & \\
\text{Until stopping condition} &= \text{true} \\
\text{Return } w & \quad \# w \text{ is an approximation to the optimal solution}
\end{align*}
\]
The Simulated Annealing Algorithm

\[ w := w_{\text{init}} \; \quad t := t_{\text{max}} \; \]
Repeat

Randomly select \( w' \) from the set Neighbours\((w)\);
\[ \text{Delta} := E(w') - E(w) ; \]
if \( \text{Delta} < 0 \) then \( w := w' \) ;
else

  generate random \( r \) uniformly in range \((0,1)\);
  if \( r < \exp(-\text{Delta} / t) \) then \( w := w' \) ;
end-if

\[ t := \alpha(t) \quad \# \text{decrease } t \]
Until stopping condition = true

Return \( w \) \quad \# w \text{ is an approximation to the optimal solution}
Gradient descent for OT?


- Based on a remark by Prince and Smolensky (1993/2004) on a “restraint of analysis” as opposed to “freedom of analysis”.

- Restricted Gen → Eval → Gen → Eval → … (n times).

- Gradual progress toward (locally) max. harmony.

- Employed to simulate traditional derivations, opacity.
Simulated Annealing for OT

- 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

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

<table>
<thead>
<tr>
<th></th>
<th>$C_0$</th>
<th>$C_1$</th>
<th>$C_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K$</td>
<td>$K = 0$</td>
<td>$K = 1$</td>
<td>$K = 2$</td>
</tr>
<tr>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
</tr>
<tr>
<td>$K = -1$</td>
<td>$K = 0$</td>
<td>$K = 1$</td>
<td>$K = 2$</td>
</tr>
<tr>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
</tr>
<tr>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
<td>. . .</td>
</tr>
</tbody>
</table>
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

\[ w := w_{\text{init}} ; \]

for \( K = K_{\text{max}} \) to \( K_{\text{min}} \) step \( K_{\text{step}} \)

\[
\quad \text{for } t = t_{\text{max}} \text{ to } t_{\text{min}} \text{ step } t_{\text{step}} \\
\qquad \text{CHOOSE random } w' \text{ in neighbourhood}(w) ; \text{COMPARE } w' \text{ to } w: \ C := \text{fatal constraint} \\
\qquad d := C(w') - C(w); \]

if \( d \leq 0 \) then \( w := w' \);
else \( w := w' \) with probability

\[
P(C,d;K,t) = \begin{cases} 
1, & \text{if } C < K \\
\exp(-d/t), & \text{if } C = K \\
0, & \text{if } C > K 
\end{cases}
\]

end-for

end-for

return \( w \)
SA-OT as a model of linguistic performance

Optimality Theory: a model of competence

Simulated Annealing: a model of performance
**Proposal: three levels**

<table>
<thead>
<tr>
<th>Level</th>
<th>its product</th>
<th>its model</th>
<th>the product in the model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competence in narrow sense: static knowledge of the language</td>
<td>grammatical form</td>
<td>standard OT grammar</td>
<td>globally optimal candidate</td>
</tr>
<tr>
<td>Dynamic language production process</td>
<td>acceptable or attested forms</td>
<td>SA-OT algorithm</td>
<td>local optima</td>
</tr>
<tr>
<td>Performance in its outmost sense</td>
<td>acoustic signal, etc.</td>
<td>(phonetics, pragmatics)</td>
<td>??</td>
</tr>
</tbody>
</table>
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](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 $K_{min}$)
- $K_{step}$
- $w_0$ (initial 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**

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

*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 1

- **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 2

Markedness Constraints \((w = w_0w_1...w_{L-1}, 0 \leq n < P)\):

- **No-\(n\):** \(*n(w) := \sum_{i=0}^{L-1} (w_i = n)\)
- **No-initial-\(n\):** \(*\text{INITIAL}n(w) := (w_0 = n)\)
- **No-final-\(n\):** \(*\text{FINAL}n(w) := (w_{L-1} = n)\)
- **Assimilation** \(\text{ASSIM}(w) := \sum_{i=0}^{L-2} (w_i \neq w_{i+1})\)
- **Dissimilation** \(\text{DISSIM}(w) := \sum_{i=0}^{L-2} (w_i = w_{i+1})\)
Example: string-grammar 3

- Faithfulness to UR $\sigma$:

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

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

(binary square, feature-combination?)
Example: string-grammar 4

\[ L = P = 4, \ T_{\text{max}} = 3, \ T_{\text{min}} = 0, \ K_{\text{step}} = 1. \]

Each of the 256 candidates used 4 times as \( w_0 \).

Grammar:

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

Globally optimal form: 3333

Many other local optima, e.g.: 1111, 2222, 3311, 1333, etc.
Example: string-grammar 5

Output frequencies for different $T_{step}$ values:

<table>
<thead>
<tr>
<th>output</th>
<th>0.0003</th>
<th>0.001</th>
<th>0.003</th>
<th>0.01</th>
<th>0.03</th>
<th>0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1111</td>
<td>0.40</td>
<td>0.40</td>
<td>0.36</td>
<td>0.35</td>
<td>0.32</td>
<td>0.24</td>
</tr>
<tr>
<td>3333</td>
<td>0.39</td>
<td>0.39</td>
<td>0.41</td>
<td>0.36</td>
<td>0.34</td>
<td>0.21</td>
</tr>
<tr>
<td>2222</td>
<td>0.14</td>
<td>0.14</td>
<td>0.15</td>
<td>0.18</td>
<td>0.19</td>
<td>0.17</td>
</tr>
<tr>
<td>3311</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04</td>
<td>0.05</td>
<td>0.06</td>
<td>0.05</td>
</tr>
<tr>
<td>1133</td>
<td>0.03</td>
<td>0.04</td>
<td>0.04</td>
<td>0.04</td>
<td>0.05</td>
<td>0.04</td>
</tr>
<tr>
<td>others</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>0.04</td>
</tr>
</tbody>
</table>
Learnability?

Why? What to learn?? Learning: for whom?

• For a linguist: find parameters exactly matching the observations to make the publication nice.

• For language technology (create a complex, but nicely working system; but OT not very used in NLP).

• For a cognitive scientist: a language acquisition model.
Parameters of the algorithm (reminder)

- $t_{step}$ (and $t_{max}$, $t_{min}$)
- $K_{max}$
- $K_{step}$
- Topology (neighbourhood structure)
- $w_0$ (initial candidate)
- Constraint hierarchy
Learnability?

What to learn??

- Find values for the parameters of the algorithm (e.g. $T_{step}$) that return the same frequencies:
  - Nice for the linguist;
  - But this is learning performance.

- Learn the underlying OT grammar, “despite” perf. errors:
  - Exact quantitative match is not the goal;
  - This is learning competence, also a relevant issue.
A first trial for learning

- Observed forms must be local optima.

- Guess: more harmonic form has higher frequency.
  - This is what SA-OT would like to achieve but it doesn’t.
    What are the consequences?
  - Cf. Coetzee’s OT proposal.

I employed *Recursive Constraint Demotion* (RCD), an off-line standard learning algorithm.

Result: a grammar that produces a superset of forms.
Preliminary remarks on learning / acquisition

- No negative evidence = hard to make something \textit{not} a local optimum.

- Already infants “measure” frequencies (e.g., Gervain). Even at pre-production age: off-line algorithm makes sense.

- Children’s forms: superset of adults’ forms.

Future work: gradually refine the grammar learned by RCD to reach adult grammar (existing on-line algorithms: EDCD, GLA).
What does SA-OT offers 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.


- Arguments for including losers (never winning candidates).
Summary of SA-OT

- Implementation: can OT be useful to language technology? is OT cognitively plausible?

- A model of variation / performance phenomena.

- *Errare humanum est* – a general cognitive principle: the role of heuristics.

- Learning is being worked on.

Thank you for your attention!

Tamás Bíró

birot@nytud.hu