HG Has No Computational Advantages over OT

Giorgio Magri
IJN, ENS, CNRS

1. Introduction

The peculiar property of *Optimality Theory* (OT) is that it uses *constraint ranking* and thus enforces *strict domination*, according to which the highest ranked relevant constraint "takes it all"; see Prince & Smolensky (2004). Because of this property, OT looks *prima facie* like an exotic combinatorial framework. Exotic in the sense that it does not seem to have any close correspondent within core Machine Learning. For this reason, the toolkit available nowadays in computational OT for modeling language acquisition, production, and perception consists of autarchic combinatorial algorithms, specifically tailored to the exotic framework of OT, developed from scratch with no connections to methods and results in Machine Learning. Tesar & Smolensky's (1998) powerful ranking algorithms well exemplify this current approach to computational OT.

In order to bridge this gap between computational Phonology and Machine Learning, various scholars have recently started to entertain and explore variants of OT that replace constraint ranking with *constraint weighting* and strict domination with *additive interaction*, and thus fall within the class of *linear models* very well studied in Machine Learning. An important and simple such model is *Harmonic Grammar* (HG); see Legendre et al. (1990b,a). For instance, Pater (2009) writes: "[I will] illustrate and extend existing arguments for the replacement of OT's ranked constraints with [HG's] weighted ones: that the resulting model of grammar [...] is compatible with well-understood algorithms for learning and other computations. [...] The strengths of HG in this area are of considerable importance" (p. 1002): "as these algorithms are broadly applied with connectionist and statistical models of cognition, this forms an important connection between the HG version of Generative Linguistics and other research in cognitive science" (p. 1021). Let me summarize this position as follows.

Conjecture. *HG is computationally superior to OT because it can make use of algorithms from Machine Learning (i.e. algorithms for linear classification), contrary to OT.*

This conjecture of a computational superiority of OT has been endorsed by a number of authors in the recent literature; see Potts et al. (2010), Pater (2009), Hayes & Wilson (2008), Coetzee & Pater (2008), Boersma & Pater (2007), Jesney & Tessier (2009), among others.

In section 2, I briefly review the two frameworks of OT and HG. And in section 3, I then *prove* that this conjecture is false. In fact, I present a simple trick that allows algorithms for HG to be extended to OT. Thus, HG has no computational advantages over OT and the departure from OT to HG is not warranted on the basis of computational considerations. Of course, this result does not in any way provide an argument in favor of OT or against alternative frameworks such as HG. It only shows that the specific argument against OT and in favor of HG based on the conjectured computational superiority of the latter does not go through. Yet, this result on the computational equivalence between OT and HG is significant because it leads to a substantial enrichment of the current toolkit of computational OT with a whole new set of algorithmic tools, obtained by adapting to OT well known algorithms from Machine Learning. Section 4 offers a first illustration of the fruitfulness of this approach, by obtaining the first convergence proof for a variant of Boersma's (1998) *Gradual Learning Algorithm* (GLA) that performs both constraint demotion and promotion. All proofs are omitted for reasons of space; see Magri (2011a) for a more extended version of this paper, that contains detailed proofs.

*I wish to thank Adam Albright for lots of help. I also wish to thank Paul Boersma, Alan Prince, Paul Smolensky, and Bruce Tesar for useful conversations on this material. This work was supported in part by a 'Euryi' grant from the European Science Foundation ("Presupposition: A Formal Pragmatic Approach" to P. Schlenker).

 Proceedings of the 29th West Coast Conference on Formal Linguistics, ed. Jaehoon Choi et al., 380-388. Somerville, MA: Cascadilla Proceedings Project.

2. Review of the frameworks of HG and OT

The basic data unit in HG is a *data triplet* consisting of an underlying form, an intended winner candidate and an intended loser candidate. Within HG, the crucial information provided by one such data triplet can be distilled by collecting into a tuple (1) the *violation differences* for each of the n constraints, namely the difference between the number of violations assigned by constraint C_k to the mapping of the underlying form to the intended loser and the number of violations assigned by that same constraint to the mapping of the underlying form to the winner. One such n-tuple is called an *elementary weighting condition* (EWC). I denote EWCs by $\overline{\mathbf{a}}$ and their components by $\overline{a}_1, \ldots, \overline{a}_n$. If we have (finitely) many data triplets, we can pair up each of them with the corresponding EWC and we can organize these EWCs one underneath the other (the order does not matter), into an *HG-comparative tableau*. I denote by $\overline{\mathbf{A}}$ an arbitrary HG-comparative tableau.

(1) $\quad \overline{\mathbf{a}} = \big[\overline{a}_1 \ldots \overline{a}_k \ldots \overline{a}_n\big] \qquad$ where $\overline{a}_k = C_k(\text{underlying}, \text{loser}) - C_k(\text{underlying}, \text{winner})$

An HG grammar is parameterized by a *weight vector*, namely a tuple $\boldsymbol{\theta}$ with n numerical components $\theta_1, \ldots, \theta_n$, one for every constraint; the kth component θ_k is called the *weight* corresponding to constraint C_k. If the weights are allowed to be negative, unwanted typological consequences follow, as a consequence of the assumption that constraints only assign "violations", and never "rewards". Thus, the definition of HG weight vectors is usually restricted by requiring the weights to be nonnegative. A weight vector $\boldsymbol{\theta}$ is called *HG-compatible* with an EWC $\overline{\mathbf{a}}$ provided condition (2a) holds. Condition (2a) can be unpacked as in (2b), in terms of the underlying/winner/loser form triplet that corresponds to that EWC. The latter condition says that the loser violates the constraints "more severely" than the winner. In the sense that the sum of the constraint violations for the loser multiplied by the corresponding weights is larger than the sum of the constraint violations for the winner multiplied by the corresponding weights. An HG-comparative tableau is called *HG-compatible* provided there exists at least a weight vector that is HG-compatible with it (namely with each of its EWCs).

(2) $\quad$ a. $\displaystyle\sum_{k=1}^{n} \theta_k \overline{a}_k > 0.$ $\qquad$ b. $\displaystyle\sum_{k=1}^{n} \theta_k \cdot C_k(\text{under}, \text{loser}) > \sum_{k=1}^{n} \theta_k \cdot C_k(\text{under}, \text{winner})$

The kernel of any computational problem in HG is the *Weighting problem* (3), that I will thus focus on. The problem is simple because the underlying forms are provided as well as the losers; because the data are guaranteed to be HG-compatible; and because the output weight vector is only required to be HG-compatible with the data, without any further requirement. I will denote by WP($\overline{\mathbf{A}}$) the instance of the Weighting problem (3) corresponding to an HG-comparative tableau $\overline{\mathbf{A}}$.

(3) $\quad$ *given:* an HG-compatible HG-comparative tableau $\overline{\mathbf{A}}$;
find: a weight vector $\boldsymbol{\theta}$ that is HG-compatible with the tableau $\overline{\mathbf{A}}$, according to condition (2).

Let me now turn to OT. Also in the case of OT, data units are triplets of an underlying form together with an intended winner and an intended loser form. Given one such triplet, the constraints can be sorted into *winner-preferring*, *loser-preferring* or *even*, depending on whether the corresponding constraint difference is positive (i.e. the constraint assigns more violations to the loser than to the winner), negative (i.e. the constraint assigns less violations to the loser than to the winner) or null (i.e. the constraint assigns the same number of violations to the loser and to the winner). Following Prince (2002), the information provided by a data triplet that is useful in OT can be distilled into an n-tuple whose kth entry is W, L or e depending on whether the kth constraint C_k is winner-preferrer or loser-preferrer or even. One such n-tuple is called an *elementary ranking condition* (ERC). I denote ERCs by $\mathbf{a}$ and their components by $a_1, \ldots, a_n$. If we have many data triplets, then we can pair up each of them with the corresponding ERC and we can organize these ERCs one underneath the other (the order does not matter), into an *OT-comparative tableau*. I denote by $\mathbf{A}$ an arbitrary OT-comparative tableau.

(4) $\quad \mathbf{a} = \big[a_1 \ldots a_k \ldots a_n\big] \qquad$ where $a_k = \begin{cases} \text{W} & \text{if } C_k \text{ is winner-preferrer} \\ \text{L} & \text{if } C_k \text{ is loser-preferrer} \\ e & \text{if } C_k \text{ is even} \end{cases}$

An OT-grammar is parameterized by a *ranking*, which is a linear order $\gg$ over the constraint set. A ranking is called *OT-compatible* with an ERC provided condition (5a) holds. Condition (5a) can be unpacked as in (5b), in terms of the underlying/loser/winner form triplet that corresponds to the ERC. The latter condition says that the loser violates the constraints "more severely" than the winner. In the sense that, among those constraints that distinguish between the winner and the loser, the top ranked one C_{top} assigns more violations to the loser than to the winner.

(5) a. Once the n entries of the ERC are reordered from left to right in decreasing order according to $\gg$, then the leftmost non-e entry is a W.

b. $C_{\text{top}}(\text{underlying}, \text{loser}) > C_{\text{top}}(\text{underlying}, \text{winner})$

The kernel of any computational problem in OT is the *Ranking problem* (6), that I will thus focus on. This is the equivalent in OT of the Weighting problem (3) in HG. I will denote by RP($\mathbf{A}$) the instance of the Ranking problem (6) corresponding to an OT-comparative tableau $\mathbf{A}$.

(6) *given:* an OT-compatible OT-comparative tableau $\mathbf{A}$;
find: a ranking $\gg$ that is OT-compatible with the tableau $\mathbf{A}$, according to condition (5).

The following claim 1 summarizes what is currently known in the literature concerning the relationship between the HG Weighting problem (3) and the OT Ranking problem (6); see Prince & Smolensky (2004). The idea of the proof is that the highest-takes-all behavior of OT-compatibility (5) can be mimicked by HG-compatibility (2) as long as we use exponentially spaced weights.[1]

Claim 1 *If a finite set $\mathcal{D}$ of underlying/winner/loser form triplets is OT-compatible, then it is also HG-compatible. More precisely, let $\gg$ be a ranking OT-compatible with $\mathcal{D}$. Without loss of generality, assume that it is (7a), with C_n ranked at the top, C_{n-1} ranked below it and so on, until the bottom ranked C_1. Then, the weight vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ in (7b) is HG-compatible with $\mathcal{D}$, where Δ is the largest constraint difference (ignoring sign) over all constraints and all data triplets in the data set $\mathcal{D}$.*

(7) a. $C_n \gg C_{n-1} \gg \cdots \gg C_1$

b. $\theta_n = (\Delta+1)^n$, $\theta_{n-1} = (\Delta+1)^{n-1}$, $\ldots$, $\theta_1 = (\Delta+1)$

To illustrate, consider the two underlying/winner/loser form triplets (/da/, [da], [ta]) and (/rad/, [rat], [rad]) and the three constraints $F_{\text{pos}} = $ IDENT[VOICE]/ONSET, $F_{\text{gen}} = $ IDENT[VOICE], and $M = {}^*$[+VOICE, $-$SONORANT]. The corresponding HG-comparative tableau of constraints differences is $\overline{\mathbf{A}}$ in (8a) and the corresponding OT-comparative tableau is $\mathbf{A}$ in (8b). Of course, $\mathbf{A}$ is OT-compatible with the ranking $F_{\text{pos}} \gg M \gg F_{\text{gen}}$. Since in this case $\Delta = 1$, the corresponding weights according to (7b) are $\theta_{F_{\text{pos}}} = 8$, $\theta_{F_{\text{gen}}} = 2$ and $\theta_M = 4$, that are indeed HG-compatible with $\overline{\mathbf{A}}$.

(8) a. $\overline{\mathbf{A}} = \begin{array}{c} \begin{matrix} F_{\text{pos}} & F_{\text{gen}} & M \end{matrix} \\ \begin{bmatrix} 1 & 0 & -1 \\ 0 & -1 & 1 \end{bmatrix} \end{array}$ b. $\mathbf{A} = \begin{array}{c} \begin{matrix} F_{\text{pos}} & F_{\text{gen}} & M \end{matrix} \\ \begin{bmatrix} \text{W} & \text{W} & \text{L} \\ e & \text{L} & \text{W} \end{bmatrix} \end{array}$

The reverse of claim 1 does not hold, namely there exist data sets $\mathcal{D}$ that are HG-compatible but not OT-compatible. Here is a counterexample. Suppose that the HG-comparative tableau of constraint differences is $\overline{\mathbf{A}}$ in (9a). The corresponding OT-comparative tableau is $\mathbf{A}$ in (9b). The former is HG-compatible (say with the weights $\theta_1 = 3$ and $\theta_2 = \theta_3 = 2$) but the latter is not OT-compatible.

[1] As discussed in detail in Tesar (2007), claim 1 only holds as long as we consider a *finite* set $\mathcal{D}$ of data triplets. Indeed, if the set $\mathcal{D}$ is infinite, then there might not exist any bound Δ on the number of constraint violations.

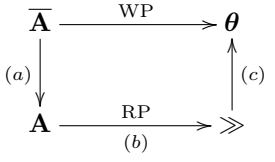
(9) a. $\overline{\mathbf{A}} = \begin{array}{c} \begin{array}{ccc} C_1 & C_2 & C_3 \end{array} \\ \left[\begin{array}{ccc} 1 & -1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 1 \end{array}\right] \end{array}$ b. $\mathbf{A} = \begin{array}{c} \begin{array}{ccc} C_1 & C_2 & C_3 \end{array} \\ \left[\begin{array}{ccc} \text{W} & \text{L} & e \\ \text{W} & e & \text{L} \\ \text{L} & \text{W} & \text{W} \end{array}\right] \end{array}$

The point of example (9) can be made more explicit as follows. In order for a weight vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ to be HG-compatible with the first two EWCs in (9a), the weight of constraint C_1 has got to be larger than both the weights of C_2 and C_3, as in (10a). Analogously, in order for a ranking $\gg$ to be OT-compatible with the first two ERCs of (9b), constraint C_1 has got to be ranked above both constraints C_2 and C_3, as in (10b). No ranking that satisfies the ranking conditions (10b) can ever be OT-compatible with the third ERC in (9b). A weight vector that satisfies the weighting conditions (10a) can instead be HG-compatible with the third EWC in (9a), provided that the two constraints C_2 and C_3, despite their small weight, are allowed to join forces and gang up against C_1, in the sense that the sum $\theta_2 + \theta_3$ of their weights is larger than the weight θ_1 of constraint C_1. The crucial difference between HG and OT is that the former allows for these *gang-up effects*, while the latter doesn't.

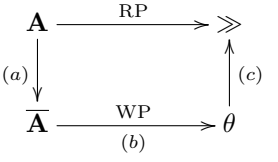
(10) a. $\theta_1 > \theta_2$
 $\theta_1 > \theta_3$ b. $C_1 \gg C_2$
 $C_1 \gg C_3$

The algorithmic implications of claim 1 can be brought out as follows. Suppose we are given a finite set $\mathcal{D}$ of data triplets that happen to be not only HG-compatible but also OT-compatible. And that we are interested in solving the instance WP($\overline{\mathbf{A}}$) of the Weighing problem for the corresponding HG-comparative tableau $\overline{\mathbf{A}}$ of constraint differences. Claim 1 says that, instead of solving the Weighting problem WP($\overline{\mathbf{A}}$) *directly*, we can instead solve it *indirectly*, through the three steps (11a)-(11c). At step (11a), we construct the corresponding OT-comparative tableau $\mathbf{A}$ out of the tableau of constraint differences $\overline{\mathbf{A}}$, as in (4). At step (11b), we solve the corresponding instance RP($\mathbf{A}$) of the Ranking problem. Finally at step (11c), we obtain a weight vector that solves the given Weighting problem WP($\overline{\mathbf{A}}$), through (7).

(11)



(12)



In other words, claim 1 says that the Weighting problem reduces to the Ranking problem, in the sense that we can get a solution to the former by solving the latter instead (provided the data are OT-compatible). Yet, as recalled in section 1, we already know how to solve the Weighting problem, since we can draw on the large literature on linear models; see for instance Potts et al. (2010). What we are really looking for are instead good methods to solve the Ranking problem. The fact that we can reduce the Weighting problem to the Ranking problem is therefore of no algorithmic interest.

3. HG and OT are computationally equivalent

The question addressed in this section can *roughly* be stated as follows: given an arbitrary instance of the Ranking problem (6), is it possible to pair it up with an instance of the Weighting problem (3) such that I get a solution to the former by solving the latter instead? This question can be stated more *precisely* as follows: given an instance RP($\mathbf{A}$) of the Ranking problem, is it possible to find one (or, even better, all) of its solutions without solving the problem *directly* but rather *indirectly*, through the three steps (12a)-(12c)? At step (12a), we pair up the given OT-comparative tableau $\mathbf{A}$ with an HG-comparative tableau $\overline{\mathbf{A}}$. At step (12b), we find a solution $\boldsymbol{\theta}$ of the corresponding Weighting problem WP($\overline{\mathbf{A}}$), by using any algorithm for HG/linear classification. Finally at step (12c), we pair up that solution $\boldsymbol{\theta}$ with a ranking $\gg$, as in (12c). We hope that the latter ranking actually solves the instance RP($\mathbf{A}$) of the Ranking problem that we started with. The scheme in (12) is the inverse of the scheme (11), that summarizes claim 1. Thus, the question considered in this section is whether the algorithmic perspective implicit in claim 1 can be inverted, despite the fact that the inverse of claim 1 does not hold.

In order to implement the scheme in (12), we need to define the two steps (12a) and (12c), namely we need to find proper ways to pair up ERCs with EWCs and weight vectors with rankings. Let me introduce the core, very simple idea with a couple of examples. Consider first the case of the ERC $\mathbf{a}$ in (13). Crucially, it contains a unique entry equal to W. Define the corresponding EWC $\overline{\mathbf{a}}$ as follows: the e of C_1 is replaced with 0; the W of C_2 is replaced with 1; and the L of C_3 is replaced with -1. A weight vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ is HG-compatible with this derived EWC $\overline{\mathbf{a}}$ provided $\theta_2 - \theta_3$ is strictly positive. Equivalently, provided the weight θ_2 corresponding to constraint C_2 is strictly larger than the weight θ_3 corresponding to constraint C_3. Consider a ranking that "respects" the ordering implicit in the relative size of these weights. Any such ranking thus ranks C_2 above C_3. The latter ranking condition ensures OT-compatibility with the ERC $\mathbf{a}$ we started from.

$$\text{(13)} \quad \mathbf{a} = \begin{bmatrix} \overset{C_1}{e} & \overset{C_2}{\text{W}} & \overset{C_3}{\text{L}} \end{bmatrix} \longrightarrow \overline{\mathbf{a}} = \begin{bmatrix} \overset{C_1}{0} & \overset{C_2}{1} & \overset{C_3}{-1} \end{bmatrix}$$

Consider next the case of the ERC $\mathbf{a}$ in (14). Crucially, it contains two entries equal to W. Define the corresponding EWC $\overline{\mathbf{a}}$ as follows: the L of C_3 is replaced again by -1; but the two W's of C_1 and C_2 are now replaced with $1/2$ (rather than with 1), capturing the fact that this ERC has two winner-preferrers. A weight vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ is HG-compatible with this derived EWC $\overline{\mathbf{a}}$ iff $\frac{1}{2}\theta_1 + \frac{1}{2}\theta_2 - \theta_3$ is strictly positive. Equivalently (by multiplying everything by 2), iff $(\theta_1 - \theta_3) + (\theta_2 - \theta_3)$ is strictly positive. This implies in particular that either $(\theta_1 - \theta_3)$ is strictly positive or $(\theta_2 - \theta_3)$ is strictly positive (or both). Again, consider a ranking that "respects" the ordering implicit in the relative size of these weights. Any such ranking either ranks constraint C_1 above C_3 or C_2 above C_3 (or both). The latter ranking conditions ensures OT-compatibility with the ERC $\mathbf{a}$ we started from.

$$\text{(14)} \quad \mathbf{a} = \begin{bmatrix} \overset{C_1}{\text{W}} & \overset{C_2}{\text{W}} & \overset{C_3}{\text{L}} \end{bmatrix} \longrightarrow \overline{\mathbf{a}} = \begin{bmatrix} \overset{C_1}{\frac{1}{2}} & \overset{C_2}{\frac{1}{2}} & \overset{C_3}{-1} \end{bmatrix}$$

The reasoning just illustrated with the two examples (13) and (14) holds in the general case, as follows. Given an ERC $\mathbf{a}$, consider the EWC $\overline{\mathbf{a}}$ *derived* from $\mathbf{a}$ as in (15): every L corresponds to -1; every e corresponds to 0; and every W corresponds to $\frac{1}{w}$, where w is the total number of entries equal to W. Examples (13) and (14) illustrate this mapping from ERCs into EWCs. Let me say that an HG-comparative tableau is *derived* from an OT-comparative tableau iff each row of the former is derived from the corresponding row of the latter according to (15).

$$\text{(15)} \quad \mathbf{a} = \big[a_1, \ldots, a_n\big] \longrightarrow \overline{\mathbf{a}} = \big[\overline{a}_1, \ldots, \overline{a}_n\big] \text{ such that } \overline{a}_k \doteq \begin{cases} -1 & \text{if } a_k = \text{L} \\ 0 & \text{if } a_k = e \\ \frac{1}{w} & \text{if } a_k = \text{W} \end{cases}$$

Following Boersma (1998, 2009), let me say that a ranking $\gg$ is *derived* from a weight vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ iff it is compatible with the order implicitly defined by the relative size of the weights, in the sense that condition (16a) holds for every pair of constraints C_h and C_k.

$$\text{(16)} \quad \text{a.} \quad \theta_h > \theta_k \implies C_h \gg C_k$$

$$\text{b.} \quad \boldsymbol{\theta} = \begin{pmatrix} \overset{C_1}{100} & \overset{C_2}{10} & \overset{C_3}{50} \end{pmatrix} \longrightarrow C_1 \gg C_3 \gg C_2$$

$$\text{c.} \quad \boldsymbol{\theta} = \begin{pmatrix} \overset{C_1}{100} & \overset{C_2}{50} & \overset{C_3}{50} \end{pmatrix} \begin{matrix} \longrightarrow C_1 \gg C_3 \gg C_2 \\ \longrightarrow C_1 \gg C_2 \gg C_3 \end{matrix}$$

Let me unpack this condition (16a), by considering two cases in turn. If all the components of the weight vector $\boldsymbol{\theta}$ are pairwise distinct, then it admits a unique derived ranking, namely the unique ranking that ranks a constraint C_k above a constraint C_h iff the weight θ_k of C_k is larger than the weight θ_h of C_h. This case is illustrated in (16b). If instead the components of the weight vector $\boldsymbol{\theta}$ are *not* all pairwise distinct, then $\boldsymbol{\theta}$ admits multiple derived rankings, because a tie between two weights can be broken differently by different derived rankings. This case is illustrated in (16c).

The main result of this section is the following claim 2. This claim says that the scheme (12) holds provided that the mapping (12a) from OT-comparative tableaux to HG-comparative tableaux is defined as in (15) and the mapping (12c) from weight vectors to rankings is defined as in (16). In other words, claim 2 says that it is possible to get a solution of a given instance of the Ranking problem (6) without solving it directly, but rather by solving the corresponding instance of the Weighting problem (3). For a proof, see the Appendix in Magri (2011a).

Claim 2 *Given an OT-comparative tableau* $\mathbf{A}$*, consider the corresponding HG-comparative tableau* $\overline{\mathbf{A}}$ *derived from* $\mathbf{A}$ *as in (15). If* $\overline{\mathbf{A}}$ *is HG-compatible, then* $\mathbf{A}$ *is OT-compatible. More precisely, if a non-negative weight vector* $\boldsymbol{\theta}$ *solves the instance WP*$(\overline{\mathbf{A}})$ *of the Weighting problem (3), then any ranking derived from* $\boldsymbol{\theta}$ *according to (16) solves the instance RP*$(\mathbf{A})$ *of the Ranking problem (6). If the OT-comparative tableau* $\mathbf{A}$ *has a unique* L *per row,*[2] *then the non-negativity restriction on the weight vector* $\boldsymbol{\theta}$ *can be dropped.*

Claim 2 says that *some* of the solutions of the Ranking problem RP$(\mathbf{A})$ can be obtained by finding some of the solutions of the corresponding Weighting problem WP$(\overline{\mathbf{A}})$ instead, and then constructing the corresponding derived rankings through (16a). Is it possible to obtain *all* of the solutions of the Ranking problem RP$(\mathbf{A})$ this way? That is indeed the case. In fact, consider a ranking $\gg$ that solves the Ranking problem RP$(\mathbf{A})$. Without loss of generality, assume that it is $C_n \gg C_{n-1} \gg \ldots \gg C_1$ (otherwise, just relabel the constraints). Consider the weight vector $\boldsymbol{\theta}$ defined from $\gg$ as in (7b), where Δ is the largest entry (ignoring sign) of the derived HG-comparative tableau $\overline{\mathbf{A}}$. Note that $\gg$ is derived from $\boldsymbol{\theta}$, namely satisfies condition (16a). Furthermore, claim 1 guarantees that $\boldsymbol{\theta}$ solves the Weighting problem WP$(\overline{\mathbf{A}})$. Claims 1 and 2 together thus entail claim 3.

Claim 3 *Given an OT-comparative tableau* $\mathbf{A}$*, consider the corresponding HG-comparative tableau* $\overline{\mathbf{A}}$ *derived from* $\mathbf{A}$ *as in (15). Then,* $\overline{\mathbf{A}}$ *is HG-compatible iff* $\mathbf{A}$ *is OT-compatible. Furthermore, a ranking solves the instance RP*$(\mathbf{A})$ *of the Ranking problem (6) iff it is derived through (16) from a (non-negative) weight vector that solves the corresponding instance WP*$(\overline{\mathbf{A}})$ *of the Weighting problem (3).*

The latter claim says that OT does not raise any new computational challenges beyond HG. Hence, the conjecture of an alleged computational superiority of HG over OT recalled in section 1 is wrong.

4. An application to the GLA's convergence problem

Boersma's (1998) (non-stochastic) *Gradual Learning Algorithm* (GLA) maintains a current weight vector, which represents its current hypothesis on the target OT grammar, through condition (16a). And it keeps updating its current weight vector by looping through three steps. First, the GLA receives an ERC sampled from a fixed, given OT-compatible comparative tableau $\mathbf{A}$, called the *input tableau*. Then, the GLA checks how its current hypothesis fares on the current piece of data, namely whether it is indeed the case that every ranking derived through (16a) from the current weight vector is OT-compatible with the current ERC. If that is not the case, the GLA takes action, updating its current weight vector. The current failure is taken to be evidence that the winner- (loser-) preferrers are currently ranked too low (high), and the GLA thus demotes (promotes) each current loser-preferring (winner-preferring) constraint by 1.

The GLA is said to *converge* provided it can only perform a finite number of updates for any input OT-compatible tableau. If the algorithm converges, then every refinement of its final weight vector solves the instance RP$(\mathbf{A})$ of the Ranking problem (6) corresponding to the input tableau $\mathbf{A}$. Convergence of the GLA has remained an outstanding open issue for a few years. Only recently, Pater (2008) has settled the issue, with a counterexample that shows that the GLA does not convergence; see Magri (2011b) for a detailed explanation of Pater's counterexample. One strategy to cope with Pater's result is to get rid of the promotion component of the GLA's update rule, and only demote (undominated) loser-preferrers by 1. In fact, Boersma (1998:p. 323-327) notes that this variant of the GLA is a slowed-down version of Tesar & Smolensky's (1998) provably converging *Error-driven Constraint Demotion*. Yet, various authors

[2]The assumption of a unique L per ERC is not restrictive. In fact, an ERC with multiple L's can be replaced with multiple ERCs each of which retains only one of the L's while the others are replaced by e's; see Prince (2002).

in the OT acquisition literature have suggested that constraint promotion is needed from a modeling perspective; see for instance Gnanadesikan (2004) and Bernhardt & Stemberger (1998). An explicit computational argument for constraint promotion is provided by Boersma (1997), who argues that it is needed in order for (a stochastic variant of) the GLA to learn certain cases of language variation. In Magri (2011b), I provide another computational argument for constraint promotion (in a non-stochastic setting), namely that it is needed in order to model the acquisition of phonotactics. These considerations lead to the following question: is it possible to derive a variant of the GLA that performs constraint promotion and yet provably converges, so as to retain the good modeling ability of the GLA, without sacrificing computational soundness? This is the question addressed in this section.

For simplicity, suppose that the input tableau has a unique L per row; for the extension to the general case, see Magri (2011a). Suppose the current ERC has two W's corresponding to the two constraints C_h and C_k, as in (17a). This ERC by itself does not say which one of the two winner-preferrers C_h or C_k should in the end be ranked above the loser-preferrer C_ℓ. For instance, the input tableau could contain another ERC like (17b), so that only C_h should be ranked above C_ℓ, while C_k must be ranked below it. Based only on the current ERC (17a), there is no way to choose in a principled way which one between C_h and C_k should be promoted. Nor could we promote both, as shown by Pater's counterexample. I suggest that the way out is to split our confidence between C_h and C_k, promoting each one just by $1/2$.

(17) a.

	C_h	C_k		C_ℓ	
...	W	W	...	L	...

b.

	C_h	C_k		C_ℓ	
...	e	L	...	W	...

In the general case, the total promotion amount of 1 should be split among the many winner-preferrers. I thus suggest the following revised promotion/demotion update rule: the loser-preferrer is demoted by 1 and each winner-preferrer is promoted by $1/w$, where w is the number of winner-preferrers in the current ERC. Let me call the corresponding algorithm the *revised GLA*. To illustrate, consider the beginning of a run of this revised GLA in (18b).

(18) a.

	C_1	C_2	C_3
ERC 1	W	W	L
ERC 2	e	L	W

b.

	$\boldsymbol{\theta}_1$	ERC 1	$\boldsymbol{\theta}_2$	ERC 2	$\boldsymbol{\theta}_3$	ERC 2	$\boldsymbol{\theta}_4$	$\longrightarrow \dots$
C_1	0	$\longrightarrow$	1/2	$\longrightarrow$	1/2	$\longrightarrow$	1/2	
C_2	0		1/2		−1/2		−3/2	
C_3	0		−1		−1/2		1/2	

The algorithm starts from the initial null weight vector $\boldsymbol{\theta}^1$. At time 1, the algorithm is fed an ERC from the input tableau (18a), say ERC 1. As neither of the two winner-preferrers C_1 and C_2 of this ERC are currently ranked above its loser-preferrer C_3 according to $\boldsymbol{\theta}_1$, the algorithm takes action: the loser-preferrer C_3 is demoted by 1 and that same amount is split over the two winner-preferrers C_1 and C_2, that are thus each promoted by $1/2$, obtaining $\boldsymbol{\theta}_2$. At time 2, the algorithm is fed another ERC of the OT-comparative tableau (18a), say ERC 2. Again, the winner-preferrer C_3 of this ERC is not currently ranked above its loser-preferrer C_2 according to $\boldsymbol{\theta}_2$. The algorithm thus takes action: the loser-preferrer C_2 is demoted by 1 and the winner-preferrer C_3 is promoted by 1, obtaining $\boldsymbol{\theta}_3$. And so on and so forth.

Claim 4 guarantees that the revised GLA cannot run into traps like Pater's counterexample: it can only perform a finite number of updates for any input OT-compatible tableau.

Claim 4 *The revised GLA converges for* any *OT-compatible input tableau.*

The proof of the convergence claim 4 rests on OT/HG computational equivalence established in section 3. If OT-comparative rows are paired up with derived HG-comparative rows as in (15) and weight vectors with derived rankings as in (16), then convergent HG error-driven algorithms can be translated into convergent OT error-driven algorithms. In particular, the convergence claim 4 follows as a translation from HG into OT of the convergence claim for the HG *Perceptron* algorithm. In the rest of this section, I sketch this idea; for the details, see Magri (2011a). For a different approach, see Magri (2011b).

The Perceptron algorithm works very similarly to the GLA. It maintains a current weight vector that it keeps updating by looping through three steps. First, the algorithm receives an EWC sampled from a fixed, given HG-compatible comparative tableau $\overline{\mathbf{A}}$, called the *input tableau*. Then, it checks HG-compatibility between the current EWC and the current weight vector. In case HG-compatibility does not hold, the algorithm updates its current weight vector. Update is performed by adding the current EWC

component by component to the current weight vector.[3] It is known that the Perceptron converges, in the sense that it can only perform a finite number of updates when run on an input HG-comparative tableau that is HG-compatible; see for instance Rosenblatt (1958) and Cristianini & Shawe-Taylor (2000:Ch. 2).

To see the connection between the Perceptron and the revised GLA, suppose that the current weight vector in a run of the revised GLA is not OT-compatible with the current ERC, namely it admits derived rankings that are not OT-compatible with that ERC. Consider the corresponding EWC derived through (15): the L is replaced by -1 and each W is replaced by $1/w$, where w is the total number of W's. Claim 2 ensures that, if a derived EWC is HG-compatible with some weight vector, then the original ERC is OT-compatible with each one of its derived rankings. The contrapositive of this claim can thus be stated as follows.

> **Fact I.** If a weight vector is not OT-compatible with an ERC, then that weight vector is also not HG-compatible with the corresponding EWC derived through (15).

Suppose that the current ERC triggers an update of the revised GLA. The winner-preferrers are promoted by $1/w$ and the loser-preferrer is demoted by 1. Equivalently, the current weight vector is updated by adding $1/w$ to the components corresponding to the winner-preferrers and by adding -1 to the component corresponding to the loser-preferrer. In other words again, the current weight vector is updated by adding component by component the corresponding EWC derived through (15), as prescribed by the Perceptron update rule.

> **Fact II.** An update according to the revised GLA triggered by an ERC is equivalent to the update according to the Perceptron triggered by the EWC derived according to (15) from that ERC.

Consider an arbitrary run of the revised GLA. Consider the corresponding derived run of the Perceptron, namely the run that starts with the same initial weight vector and where the Perceptron is fed at every time the EWC derived through (15) from the ERC fed at that time to the revised GLA. The two facts I and II entail that the sequence of weight vectors in the derived Perceptron run is identical to the sequence of weight vectors in the run of the revised GLA. In fact, the two runs start from the same initial weight vector. Furthermore, every time the current weight vector is updated by the revised GLA, it is also updated by the Perceptron, by Fact I. And they are updated in exactly the same way in the two runs, by Fact II. If the revised GLA run could go on for ever, the corresponding Perceptron run would go on for ever too, contradicting the Perceptron convergence claim. In conclusion, the convergence claim 4 for the revised GLA with the new well calibrated promotion/demotion update rule follows straightforwardly from the Perceptron convergence claim through the OT/HG computational equivalence.

Now that we have understood why the revised GLA works, it is instructive to go back to Boersma's original GLA and understand what goes wrong. Every time an ERC triggers an update according to the original GLA update rule, winner-preferrers are promoted by 1 and loser-preferrers are demoted by 1. Equivalently, the current weight vector is updated by adding component by component a corresponding EWC defined as in (19): each W is mapped to a 1 and each L is mapped to a -1. Thus, Fact II used in the convergence proof of the revised GLA also holds in the case of the original GLA: also the updates of the original GLA can be reinterpreted as Perceptron updates, provided that the corresponding EWCs are properly defined as in (19).

$$\text{(19)} \quad \mathbf{a} = \left[a_1, \ldots, a_n\right] \longrightarrow \overline{\mathbf{a}} = \left[\overline{a}_1, \ldots, \overline{a}_n\right] \text{ such that } \overline{a}_k \doteq \begin{cases} -1 & \text{if } a_k = \text{L} \\ 0 & \text{if } a_k = e \\ 1 & \text{if } a_k = \text{W} \end{cases}$$

[3] A remark on the issue of the non-negatvity of the weights is in order here. As reviewed in section 2, HG weights are required to be non-negative, in order to prevent implausible typological predictions. This restriction requires algorithms for HG to enforce the non-negativity of the weights. This is not trivial in the case of the Perceptron, as nothing in the design of the update rule enforces non-negativity of the weights. The recent HG computational literature usually tries to get around this problem by starting out with large positive weights; see for instance Jesney & Tessier (2009). But this trick does not guarantee that the weights will stay non-negative at every iteration until convergence, as the number of updates depends on the size of the initial weights. Furthermore, cutting off the weights at zero jeopardizes the Perceptron convergence claim.

What crucially fails in the case of the original GLA is Fact I used in the convergence proof of the revised GLA. Namely, it is not true that, whenever a weight vector is not OT-compatible with an ERC, that weight vector is also not HG-compatible with the EWC derived through (19) from that ERC. Here is a counterexample: the weight vector (20a) admits derived rankings (such as $C_3 \gg C_1 \gg C_2$) that are not OT-compatible with ERC (20b) and yet this weight vector is HG-compatible with the EWC (20c) derived through (19): despite the fact that the weights of C_1 and C_2 each are smaller than the weight of C_3, they can gang up to overcome the latter.

(20) a. $\boldsymbol{\theta} = \begin{pmatrix} C_1 & C_2 & C_3 \\ 2 & 2 & 3 \end{pmatrix}$ b. $\begin{bmatrix} C_1 & C_2 & C_3 \\ \text{W} & \text{W} & \text{L} \end{bmatrix}$ c. $\begin{bmatrix} C_1 & C_2 & C_3 \\ 1 & 1 & -1 \end{bmatrix}$

Suppose the current weight vector is (20a) and the current ERC is (20b). Thus, the original GLA will perform an update. But this update cannot be mimicked in a corresponding run of the Perceptron: as the current weight vector (20a) is HG-compatible with the EWC (20c), no update will happen. In general, the run of the original GLA can contain many more updates than the corresponding run of the Perceptron. And the Perceptron's convergence thus does not entail convergence of the original GLA, despite the fact that updates by the original GLA can be described as Perceptron updates.

References

Bernhardt, Barbara Handford & Joseph Paul Stemberger (1998). *"Handbook of phonological development from the perspective of constraint-based nonlinear phonology"*. Academic Press.

Boersma, Paul (1997). "How We Learn Variation, Optionality and Probability". *IFA Proceedings 21*, Institute for Phonetic Sciences, University of Amsterdam, 43–58.

Boersma, Paul (1998). *Functional Phonology*. Ph.D. thesis, University of Amsterdam. The Hague: Holland Academic Graphics.

Boersma, Paul (2009). "Some Correct Error-driven Versions of the Constraint Demotion Algorithm". *Linguistic Inquiry* 40, 667–686.

Boersma, Paul & Joe Pater (2007). "Convergence Properties of a Gradual Learner for Harmonic Grammar". *Proceedings of NELS 38*.

Coetzee, Andries W. & Joe Pater (2008). Weighted constraints and gradient restrictions on place co-occurrence in muna and arabic. *Natural Language and Linguistic Theory* 26, 289–337.

Cristianini, Nello & John Shawe-Taylor (2000). *An Introduction to Support Vector Machines and Other Kernel-Based Methods*. Cambridge University Press.

Gnanadesikan, Amalia E. (2004). "Markedness and Faithfulness Constraints in Child Phonology". Kager, René, Joe Pater & Wim Zonneveld (eds.), *Constraints in phonological acquisition*, Cambridge University Press, Cambridge, 73–108. Circulated since 1995.

Hayes, Bruce & Colin Wilson (2008). "A maximum entropy model of phonotactics and phonotactic learning". *Linguistic Inquiry* 39, 379–440.

Jesney, Karen & Anne-Michelle Tessier (2009). "Biases in Harmonic Grammar: the road to restrictive learning". *Natural Language and Linguistic Theory*.

Legendre, Géraldine, Yoshiro Miyata & Paul Smolensky (1990a). "Harmonic Grammar: A formal multi-level connectionist theory of linguistic well-formedness: An application". *Proceedings of the twelfth annual conference of the Cognitive Science Society*, Lawrence Erlbaum, Cambridge, MA, 884–891.

Legendre, Géraldine, Yoshiro Miyata & Paul Smolensky (1990b). "Harmonic Grammar: A formal multi-level connectionist theory of linguistic well-formedness: Theoretical foundations". *Proceedings of the twelfth annual conference of the Cognitive Science Society*, Lawrence Erlbaum, Cambridge, MA, 388–395.

Magri, Giorgio (2011a). "HG has no computational advantages over OT: towards a new toolkit for Computational OT". Manuscript, IJN, ENS. Available as ROA-1104. Submitted to *Linguistic Inquiry*.

Magri, Giorgio (2011b). "On the convergence of error-driven ranking algorithms". Manuscript, IJN, ENS. Available as ROA-1105.

Pater, Joe (2008). "Gradual Learning and Convergence". *Linguistic Inquiry* 39.2, 334–345.

Pater, Joe (2009). "Weighted Constraints in Generative Linguistics". *Cognitive Science* 33, 999–1035.

Potts, Christopher, Joe Pater, Karen Jesney, Rajesh Bhatt & Michael Becker (2010). "Harmonic Grammar with Linear Programming: From linear systems to linguistic typology". *Phonology* 27(1), 1–41.

Prince, Alan (2002). "Entailed Ranking Arguments". ROA 500.

Prince, Alan & Paul Smolensky (2004). *Optimality Theory: Constraint Interaction in Generative Grammar*. Blackwell.

Rosenblatt, F. (1958). "The Perceptron: A probabilistic model for information storage and organization in the brain". *Psychological Review* 65, 386–408.

Tesar, Bruce (2007). "A Comparison of Lexicographic and Linear Numeric Optimization Using Violation Difference Ratios". Ms., Rutgers University; ROA-939.

Tesar, Bruce & Paul Smolensky (1998). "Learnability in Optimality Theory". *Linguistic Inquiry* 29, 229–268.

Proceedings of the 29th West Coast Conference on Formal Linguistics

edited by Jaehoon Choi, E. Alan Hogue, Jeffrey Punske, Deniz Tat, Jessamyn Schertz, and Alex Trueman

Cascadilla Proceedings Project Somerville, MA 2012

ISBN 978-1-57473-451-5 library binding

Ordering information

Orders for the library binding edition are handled by Cascadilla Press.
To place an order, go to www.lingref.com or contact:

Cascadilla Press, P.O. Box 440355, Somerville, MA 02144, USA
phone: 1-617-776-2370, fax: 1-617-776-2271, sales@cascadilla.com

Web access and citation information

This entire proceedings can also be viewed on the web at www.lingref.com. Each paper has a unique document # which can be added to citations to facilitate access. The document # should not replace the full citation.

This paper can be cited as:

Magri, Giorgio. 2012. HG Has No Computational Advantages over OT. In *Proceedings of the 29th West Coast Conference on Formal Linguistics*, ed. Jaehoon Choi et al., 380-388. Somerville, MA: Cascadilla Proceedings Project. www.lingref.com, document #2724.