Giorgio Magri (IJN, ENS)
magrigrg@gmail.com

EALing 2010, ENS
Paris, September 7, 2010

The OT online model of the acquisition of phonotactics
Class 1: constraint demotion and the need for constraint promotion

Summary — This class gives an overview of the goal and the content of these four classes; introduces OT online algorithms in detail; reviews the literature on OT online algorithms that perform demotion only; motivates the need for OT online algorithms that perform promotion too; discusses the computational challenges raised by constraint promotion, offering a detailed analysis of Pater's (2008) counterexample against Boersma's (1997) GLA. Thus, the main point of this class is that constraint promotion is needed from the modeling perspective but hard to get from the computational perspective.

1 Modeling problem: early stage of the acquisition of phonotactics

■ **Preliminaries.** Poggio and Smale (2003) in a classical review paper write:

(1) "The problem of understanding *intelligence* is [...] the greatest problem in science today and *the* problem for this century—as deciphering the genetic code was for the second half of the last one."

And furthermore that:

(2) "Arguably, the problem of *learning* represents a gateway to understanding intelligence in brains and machines, to discovering how the human brain works, and to making intelligent machines that learn from experience and improve their competences as children do."

From this perspective, I am particularly interested in:

(3) The problem of *language learning*, namely the problem of devising algorithms that model how children happen to learn their target language fast, efficiently, and without effort.

One crucial crucial component of the problem of language learning consists of:

(4) The problem of the *learning phonotactics*, namely of the knowledge of licit vs. illicit structures in a language (e.g. English speakers know that [blɪk] is licit while [bnɪk] illicit in English, despite the fact that both are unattested).

In these four classes, we focus on the problem of the acquisition of phonotactics, as a gateway to the problem of understanding intelligence.

■ **The modeling problem.** Hayes (2004) reviews the relevant psycholinguistic literature and concludes with the following three observations:

(5) a. *On the one hand*, "At more or less [eight to ten months], infants start to acquire knowledge of the legal [...] sequences of their language. [...] In carefully monitored experimental situations, [they] come to react differently to legal phoneme sequences in their native language than to illegal [...] ones."

b. *On the other hand*, "Certainly we can say that there are at least some morphological processes which are acquired long after the system of contrasts and phonotactics is firmly in place."

c. *Thus*, "it seems a reasonable guess that in general, the learning of patterns of alternation [that only comes with knowledge of morphology] lags the learning of the constrast and phonotactic systems."

In other words, there is a developmental stage, called the *early stage* of the acquisition of phontactics, characterized by the two (somewhat idealized) properties (6).

(6) a. *Properties of the input.* Throughout this stage, the child does not yet have knowledge of morphology and is thus blind to alternations.

b. *Properties of the output.* By the end of this stage, the child is able to distinguish legal from illegal sequences w.r.t. his target language.

Of course, (6) looks close to a *computational problem*, namely a mapping from an input (6a) to an output (6b). This is the modeling problem tackled in these classes.

■ **A textbook example.** Adult Greek phonotactics displays the restriction in (7), both in non-derived and derived environments.

(7) a. Only palatals occur before front vowels:
/ce/ → [ce], /ec+ete/ → [ecete]
/xe/ → [ce], /ex+ete/ → [ecete]

b. Only velars occur before non-front vowels (V):
/cV/ → [xV], /ec+Vte/ → [ex+Vte]
/xV/ → [xV], /ex+Vte/ → [ex+Vte]

Kazazis (1969) documents the U-shaped learning path (8) at 4.1-5.

(8) Stage 1: the child is correct w.r.t. the phonotactics in (7), both in derived and non-derived forms.
Stage 2: the child is correct in non-derived forms, but now makes mistakes in derived forms (e.g.: /ex+ete/ → [exete]).
Stateg 3: the child is again correct, both in derived and non-derived forms.

This learning path (8) can be accounted for as in (9), under the assumption that morphological awareness lags behind the acquisition of phonotactics.

(9) Stage 1: (IDENTOO) ≫ *XE ≫ *C ≫ IDENTIO

Morphological awarness has not started yet; thus, there is no difference between derived and non-derived forms; and IDENTOO does not play any role.

Stage 2: IDENTOO ≫ *XE ≫ *C ≫ IDENTIO

Morphological awarness kicks in; thus IDENTOO becomes active; assume that it starts out top ranked.

Stage 3: *XE ≫ IDENTOO ≫ *C ≫ IDENTIO

Since the top ranking of IDENTOO is incorrect, learning is triggered; it leads to the correct ranking of IDENTOO below *XE.

This example thus shows that knowledge of morphology lags behind knowledge of phonotactics, as stated in (6).

■ **Motivation.** The early stage of the acquisition of phonotactics is an ideal topic for research on language acquisition, both from a modeling and a computational perspective.

(10) The early developmental onset suggests that the acquisition of phonotactics is somewhat independent from the acquisition of other sub-modules of the linguistic faculty (e.g. of alternations), and can therefore be studied in isolation.

(11) As the problem of the acquisition of phonotactics closely resembles the problem of pattern classification in Machine Learning, it might be the first piece of the puzzle of language acquisition to be broken with the tools of Machine Learning.

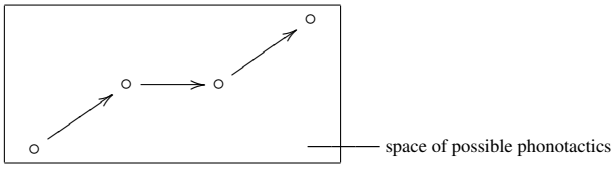
2 Modeling tool: OT online algorithms

■ **Preliminaries.** As stated in (12), the acquisition of phonotactics is gradual; see Levelt et al. (2000), Gnanadesikan (2004) and Pater and Barlow (2003) for classical examples; and Zamuner et al. (2005) or McLeod et al. (2001) for a review.

(12) The target grammar is approached slowly, through a series of more conservative intermediate stages, a progression that starts from the most unmarked grammar and converges toward the target grammar.

I take (at least part of) this gradualness to reflect grammatical development, rather than (only) the slow development of performance factors orthogonal to linguistic competence; see e.g. Smolensky (1996). Thus, (12) can be represented as in (13).

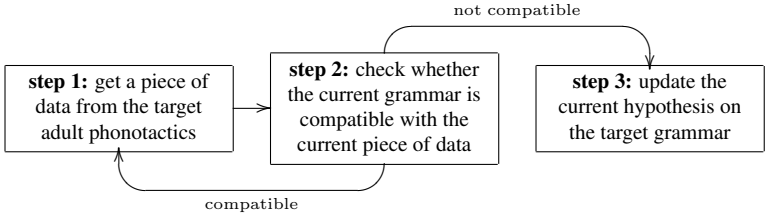
(13)



The learner entertains at each time a current hypothesis of the target phonotactics, that gets updated over time, based on data that consists of phonotactically licit adult forms. The result is a path within the space of possible phonotactics, as in (13).

■ **The modeling tool.** From this perspective, modeling the acquisition of phonotactics consists of developing the following algorithmic scheme: the algorithm maintains a current hypothesis of the target grammar; this hypothesis is initialized to some initial hypothesis; and it is updated by repeating the three steps (14).

(14)



These are called *online learning algorithms*; see Cesa-Bianchi and Lugosi (2006) for a review. The modeling tool of these classes are online algorithms within the framework of *Optimality Theory* (OT); see Prince and Smolensky (2004).

■ **Motivation.** Online algorithms are an interesting modeling tool for various reasons, both from the modeling and the computational perspective.

(15) They define a sequence of grammatical hypotheses that can be matched with data from the huge literature on acquisition paths.

(16) They are memoryless (i.e. do not keep track of previously seen forms), and thus do not impose unrealistic memory requirements.

(17) They raise interesting computational challenges, as they need to achieve correctness through small instantaneous choices based on a single data point.

(18) They are usually assumed in the non-computational OT literature, and thus deserve close investigation; see for instance Gnanadesikan (2004), Levelt et al. (2000) and Bernhardt and Stemberger (1998).

3 Modeling strategy: cognitive computational phonology

■ **Preliminaries.** The theory of vision is the place where phonology should look to foresee its future development. This suggests a strategy reminiscent of Marr's (1982) levels.

■ **The modeling strategy.** The initial step of the strategy is (19). The hope here is that various sub-problems can be tackled somewhat independently. This is particularly true for problems that seem to be solved by the learner very early on.

(19) *Step 1.* The learning task is broken down into its basic building blocks, namely explicit computational problems that single out specific components of the overall learning task. And the computational complexity of these formal problems is assessed with the tools of Complexity Theory.

The next step is (20). The idea here is that the *actual* learning model is likely to be optimal w.r.t. to the specific sub-problem that it is supposed to solve.

(20) *Step 2.* Algorithmic schemes are devised that provably perform optimally with respect to the core computational problems singled out in step 1, relative to their complexity class.

The last step is (21). The idea here is that the data are always too sparse to directly discriminate among competetive algorithmic schemes, and it is thus wiser to go through the intermediate computational step (20).

(21) *Step 3.* Predictions of various implementations of the algorithmic schemes obtained in step 2 are compared with actual acquisition data, in order to fine tune the parameters and select the algorithmic implementation that offers a tighter model of the data.

The strategy (19)-(21) characterizes *Computational Phonology*, construed as a branch of *Cognitive Science*.

- **What is going to come.** In these four classes, we will concentrate on *step 1* and *step 2*. What I have in mind for *step 3* is (22); but this is "under construction", as I have started to work on it only during the summer.

(22) Use the huge literature on the acquisition of consonant clusters to compare the different modeling implications of various algorithmic schemes developed in steps 1 and 2.

Thus, these four classes really address the following two questions:

(23) a. Which are the logically necessary conditions in order for an OT online model of the early stage to work?
b. Can these conditions be met? in the general case? or can we characterize special cases for which they can be met?

4 Plan for the four classes

- **Class 1.** This class shows that OT online algorithms that perform constraint promotion too are needed from the modeling perspective, but hard to get from the computational perspective. More in detail, this class:

(24) a. introduces *OT online algorithms* in full detail from the computational perspective, focusing on the *comparative notation*;
b. reviews existing results on demotion-only OT online algorithms (*Constraint Demotion*, *GLA*, etceteras), within the framework of *ranking vectors*;
c. explains why *constraint promotion* is needed in order to develop an online model the early stage of the acquisition of phonotactics;
d. discusses the computational challenges raised by constraint promotion, offering an explanation of *Pater's (2008) counterexample* against the GLA.

- **Class 2.** This class shows that it is possible to achieve convergence also with OT online algorithms that perform both promotion and demotion, as long as the *promotion amount* is chosen properly. More in detail, this class:

(25) a. introduces a principled family of OT online algorithms that perform *cautious* constraint promotion too, besides demotion;
b. shows that in general OT-compatibility of the data entails *conic independence* of the *update vectors* used by OT online algorithms;
c. uses this new fact (25b) to prove convergence for the family of cautious promotion/demotion OT online algorithms (25a);
d. starts the investigation of *invariants* to characterize the behavior of promotion/demotion OT online algorithms.

- **Class 3.** This class shows that methods and results from the theory of linear classification can be adapted to computational OT, and uses this approach to look at convergence of OT online algorithms from a very different perspective. More in detail, this class:

(26) a. introduces the alternative framework of *Harmonic Grammar*; see Legendre et al. (1990b,a);
b. shows that the latter has no computational advantages over OT, contrary to what has been argued in the recent literature, such as Pater (2009);
c. illustrates the implications of this result by developing an alternative proof of convergence of the cautious OT online algorithms introduced in class 2;
d. presents further developments in the theory of OT online algorithms, in terms of *worst-case number of updates* and alternative *update rules*.

- **Class 4.** This class turns from convergence to correctness, and evaluates correctness of the OT online algorithm as a model of the early stage of the acquisition of phonotactics, both through simulations and theoretical results. More in detail, this class:

(27) a. shows that correctness (contrary to convergence) cannot be achieved in the general case, as the corresponding problem is NP-complete;
b. exhibits a class of simple but non trivial OT typologies such that the OT online algorithm is correct, for any language and any sequence of data;
c. outlines prospects for future research.

5 OT online algorithms: basic description

- **Universal specifications.** A 4-tuple $\tau = (\mathcal{X}, \mathcal{Y}, Gen, \mathcal{C})$ as in (28) is called the *universal specifications* of a typology.

(28) $\mathcal{X}$: (finite) set of *underlying forms*;
$\mathcal{Y}$: (finite) set of *surface forms*;
$Gen(x) \subseteq \mathcal{Y}$: set of *candidate* surface forms for the underlying form x;
$\mathcal{C} = \{C_1, \ldots, C_n\}$: set of n *constraints*.
Constraint C_k takes a pair (x, y) of an underlying form $x \in \mathcal{X}$ and a corresponding candidate $y \in Gen(x)$ and returns a nonnegative number $C_k(x, y)$, called the *number of violations*.

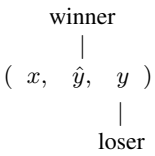
An example of universal specifications is provided in (29).

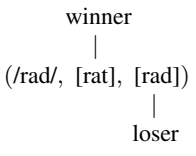
(29) a. $\mathcal{X} = \{$/ta/, /da/, /rat/, /rad/$\}$
b. $\mathcal{Y} = \{$[ta], [da], [rat], [rad]$\}$
c. $Gen($/ta/$) = Gen($/da/$) = \{$[ta], [da]$\}$
$Gen($/rat/$) = Gen($/rad/$) = \{$[rat], [rad]$\}$
d. $\mathcal{C} = \left\{ \begin{array}{rcl} F_{\text{pos}} & = & \text{IDENT[VOICE]/ONSET}, \\ F_{\text{gen}} & = & \text{IDENT[VOICE]}, \\ M & = & {}^*[+\text{VOICE}, -\text{SONORANT}] \end{array} \right\}$

■ **Data triplets.** Consider:

(30) a. an underlying form $x \in \mathcal{X}$;
b. a candidate $\hat{y} \in Gen(x)$ for x, called the *winner* candidate;
c. another candidate $y \in Gen(x)$ for x, called a *loser* candidate.

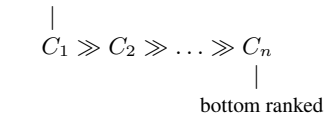
The basic data units in OT are *underlying/winner/loser form triplets* $(x, \hat{y}, y)$ as in (31): the first item of the triplet is the underlying form x, the second item is the intended winner candidate $\hat{y}$, and the third item is a loser candidate y.

(31)


(32)


An example of data triplet is provided in (32) for the typology in (29).

■ **Rankings.** An OT-grammar is parameterized by a *ranking*, i.e. a linear order $\gg$ over the constraint set $\mathcal{C}$, as in (33). A constraint C_h is *$\gg$-ranked above* C_k iff $C_h \gg C_k$.

(33)


(34) $F_{\text{pos}} \gg M \gg F_{\text{gen}}$

An example of ranking over the constraint set in (29d) is provided in (34).

■ **OT-compatibility.** A ranking $\gg$ is *OT-compatible* with an underlying/winner/loser form data triplet $(x,\ \hat{y},\ y)$ iff the intended loser y violates the constraints "more severely" than the intended winner $\hat{y}$, in the sense of condition (35).

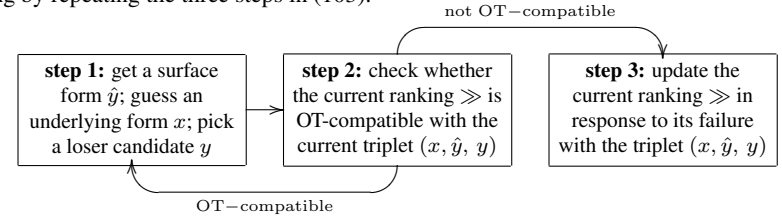
(35) Consider those constraints that differentiate between the winner $\hat{y}$ and the loser y, i.e. $C_h(x, y) \neq C_h(x, \hat{y})$. The top $\gg$-ranked among them (say C_k) assigns more violations to the loser y than to the winner $\hat{y}$, i.e. $C_k(x, y) > C_k(x, \hat{y})$.

Furthermore:

(36) a. a ranking is *OT-compatible* with a set of triplets iff it is OT-compatible with each;
b. A set of data triplets is *OT-compatible* iff it is compatible with some ranking.

To illustrate, the ranking in (34) is OT-compatible with the data triplet in (32).

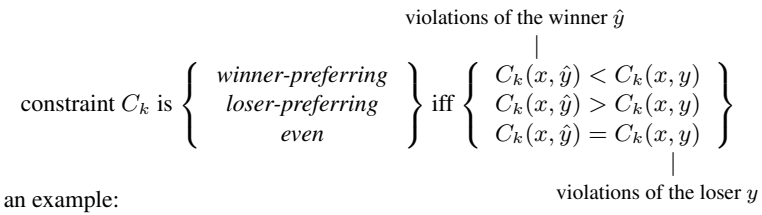
■ **OT online algorithms.** An *OT online algorithm* maintains a *current ranking* $\gg$. It initializes this current ranking to a given *initial ranking* $\gg^{\text{init}}$. And it updates this current ranking by repeating the three steps in (103).

(37)


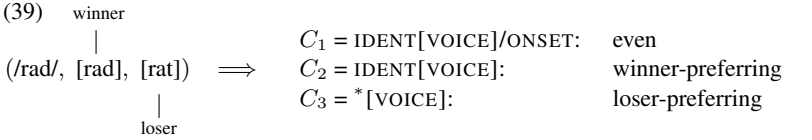
Step 1 of the algorithm asks for an underlying form x and a loser y corresponding to the given winner $\hat{y}$. Thus, we need subroutines that take care of this. For the time being, I will put this issue aside. And assume that underlying and loser forms are provided too.

6 OT online algorithms: restatement with comparative notation

■ **Loser/winner-preferring constraints.** Given an underlying/winner/loser form triplet $(x, \hat{y}, y)$, a constraint C_k is classified as preferring the winner $\hat{y}$ or the loser y as in (38).

(38)


Here is an example:

(39)


Winner- and loser-preferring constraints are *active*, to distinguish them from even ones.

■ **The intuition.** We usually represent a triplet like (32) with the OT-tableau (40).

(40)
$$\overset{\text{winner}}{\underset{\text{loser}}{(\text{/rad/},\ [\text{rat}],\ [\text{rad}])}} \implies$$

/rad/	F_{pos}	F_{gen}	M
[rat]		⋆	
[rad]			⋆

Tableau (40) encodes the numbers of constraint violations (as the number of stars in a cell). Yet, the notion (35) of OT-compatibility has the following property:

(41) OT-compatibility does not care about the actual numbers of constraint violations; it only cares about whether a constraint is winner- or loser-preferring or even.

Thus, the representation (40) can be replaced with the sharper representation (42).

(42)
$$\overset{\text{winner}}{\underset{\text{loser}}{(\text{/rad/},\ [\text{rat}],\ [\text{rad}])}} \implies \begin{bmatrix} F_{\text{pos}} & F_{\text{gen}} & M \\ \text{EVEN} & \text{LOSER-PREF.} & \text{WINNER-PREF.} \end{bmatrix}$$

Let me abbreviate (42) as in (43), with the three symbols W, L and E that stand for "winner-preferring", "loser-preferring" and "even", respectively.

(43)
$$\overset{\text{winner}}{\underset{\text{loser}}{(\text{/rad/},\ [\text{rat}],\ [\text{rad}])}} \implies \begin{bmatrix} F_{\text{pos}} & F_{\text{gen}} & M \\ \text{E} & \text{L} & \text{W} \end{bmatrix}$$

All the information we really need concerning the underlying/winner/loser form triplet (32) is this row (43) with entries equal to either L, or E, or W, one for every constraint.

■ **Comparative rows.** Following Tesar (1995) and Prince (2002), the information provided by a triplet that is useful for the sake of OT-compatibility is distilled as in (44).

(44) The triplet is paired up with a tuple with n entries (one for every constraint), equal to W, L or E depending on whether the corresponding constraint is winner-preferring or loser-preferring or even.

$$\overset{\text{winner}}{\underset{\text{loser}}{(x,\ \hat{y},\ y)}} \implies \mathbf{a} = \begin{bmatrix} a_1 & \dots & a_k & \dots & a_n \end{bmatrix}$$

$$\text{where } a_k = \begin{cases} \text{W} & \text{if } C_k \text{ is winner-preferring} \\ \text{L} & \text{if } C_k \text{ is loser-preferring} \\ \text{E} & \text{if } C_k \text{ is even} \end{cases}$$

One such n-tuple is called an *OT-comparative row*, since it contains all the information needed to compare the winner and the loser within OT.[1]

■ **Comparative tableaux.** Given m data triplets, pair up each of them with the corresponding comparative row (44) and organize these one underneath the other (the order does not matter) into an *OT-comparative tableau* with n columns and m rows, as in (45).

(45)
$$\mathbf{A} = \underbrace{\left.\begin{bmatrix} C_1 & \dots & C_k & \dots & C_n \\ \text{W} & \text{L} & \text{W} & \text{L} & \text{E} \\ \text{L} & \text{W} & \text{W} & \text{E} & \text{E} \\ \text{E} & \text{W} & \text{W} & \text{L} & \text{L} \end{bmatrix}\right\} m \text{ rows}}_{n \text{ columns}}$$

I denote by $\mathbf{A}(D)$ the tableau corresponding to a set of data triplets D; and by $\mathbf{A}$ an arbitrary comparative tableau; I often omit E's. An example is provided in (46).

(46)
$$\overset{\text{winner}}{\underset{\text{loser}}{\begin{matrix} (\text{/da/},\ [\text{da}],\ [\text{ta}]) \\ (\text{/rad/},\ [\text{rat}],\ [\text{rad}]) \end{matrix}}} \begin{bmatrix} F_{\text{pos}} & F_{\text{gen}} & M \\ \text{W} & \text{W} & \text{L} \\ \text{E} & \text{L} & \text{W} \end{bmatrix}$$

The comparative tableau (46) has three columns, because the constraint set (29d) has three constraints; it has two rows, because it corresponds to two triplets; its entries are W's, L's and E's, according to (44).

■ **OT-compatibility.** A ranking $\gg$ is OT-compatible with a set D of data triplets according to (35) iff condition (47) holds for the corresponding comparative tableau $\mathbf{A}(D)$.

(47) Once the n columns of $\mathbf{A}(D)$ are reordered from left to right in decreasing order according to $\gg$, then the leftmost non-E entry is a W in every row.

To illustrate, the OT-comparative tableau in (46) is OT-compatible with the ranking $F_{\text{pos}} \gg M \gg F_{\text{gen}}$, as its columns can be reordered as in (48).

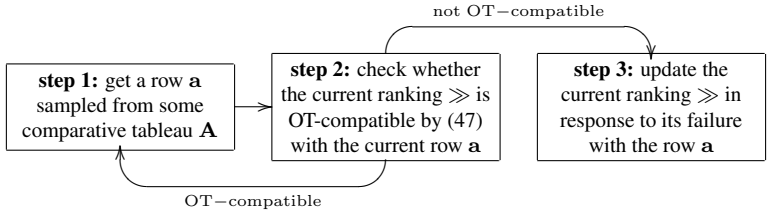
(48)
$$\overset{\curvearrowright}{\begin{bmatrix} F_{\text{pos}} & M & F_{\text{gen}} \\ \text{W} & \text{L} & \text{W} \\ \text{E} & \text{W} & \text{L} \end{bmatrix}}$$

An arbitrary tableau $\mathbf{A}$ is *OT-compatible* with a ranking $\gg$ iff condition (47) holds.

■ **Restatement of OT online algorithms with comparative notation.** As noted at the end of section 5, I assume that the OT online algorithm (103) is provided at step 1 with an underlying/winner/loser form triplet. Or, equivalently, with the corresponding comparative row. It is useful to restate the algorithm in comparative notation, as in (49).

[1] Various alternative names for OT-comparative rows have been used in the literature: Prince (2002) calls them *elementary ranking conditions*; Tesar and Smolensky (1998) call them *mark data pairs*.

(49)



Rows fed to the algorithm (49) in step 1 are sampled from a fixed, given OT-compatible comparative tableau $\mathbf{A}$, called the *input tableau*.

7 OT online algorithms: restatement in terms of ranking vectors

■ **The intuition.** So far, I have represented rankings as total orders on the constraint set.

(50) $C_1 \gg C_3 \gg C_2$

Pair up C_1, C_2, C_3 with arbitrary numbers, say with the three numbers $\theta_1, \theta_2, \theta_3$ in (51).

(51)
$$\begin{array}{ccc} C_1 & C_2 & C_3 \\ | & | & | \\ \theta_1 = 100 & \theta_2 = 2 & \theta_3 = 50 \end{array}$$

As θ_1 is larger than θ_3 which is larger than θ_2, then the triplet $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ in (51) intuitively *represents* the ranking in (50), as stated in (52).

(52)
$$\boldsymbol{\theta} = \begin{array}{c} \begin{array}{ccc} C_1 & C_2 & C_3 \end{array} \\ \begin{pmatrix} 100 & 2 & 50 \end{pmatrix} \end{array} \implies C_1 \gg C_3 \gg C_2$$

It is crucial in (52) that the numbers $\theta_1, \theta_2, \theta_3$ in (51) are distinct. What if, say, two were identical? Well, the triplet $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ can be taken to *represent* at the same time the two rankings that resolve the tie in two different ways, as illustrated in (53).

(53)
$$\boldsymbol{\theta} = \begin{array}{c} \begin{array}{ccc} C_1 & C_2 & C_3 \end{array} \\ \begin{pmatrix} 100 & 50 & 50 \end{pmatrix} \end{array} \begin{array}{l} \nearrow\!\!\nearrow \; C_1 \gg C_2 \gg C_3 \\ \searrow\!\!\searrow \; C_1 \gg C_3 \gg C_2 \end{array}$$

OT-compatibility can thus be extended to $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ via the rankings it represents.

■ **Ranking vectors.** After Boersma (1997, 2008), a *ranking vector* is an n-tuple $\boldsymbol{\theta}$ of numbers $\theta_1, \ldots, \theta_n$ (one for every constraint), as in (54). The kth component θ_k of $\boldsymbol{\theta}$ is called the *ranking value* of the corresponding constraint C_k.

(54)
$$\boldsymbol{\theta} = \begin{array}{c} \begin{array}{ccccc} C_1 & \ldots & C_k & \ldots & C_n \end{array} \\ \begin{pmatrix} \theta_1, & \ldots & \theta_k, & \ldots & \theta_n \end{pmatrix} \end{array}.$$

A ranking vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ *represents* a ranking $\gg$ iff (55) holds for any C_h, C_k, namely $\gg$ respects the ordering implicit in the relative size of the ranking values.

(55) $\theta_h > \theta_k \implies C_h \gg C_k$.

■ **OT compatibility.** A ranking vector $\boldsymbol{\theta}$ is called *OT-compatible* with a comparative tableau $\mathbf{A}$ iff condition (56) holds.[2]

(56) Every ranking represented by $\boldsymbol{\theta}$ is OT-compatible with $\mathbf{A}$.

A ranking vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ is OT-compatible with a comparative row $\mathbf{a} = (a_1, \ldots, a_n)$ according to (56) iff there exists a winner-preferring constraint whose ranking value is larger than the ranking value of every loser-preferring constraint, namely:

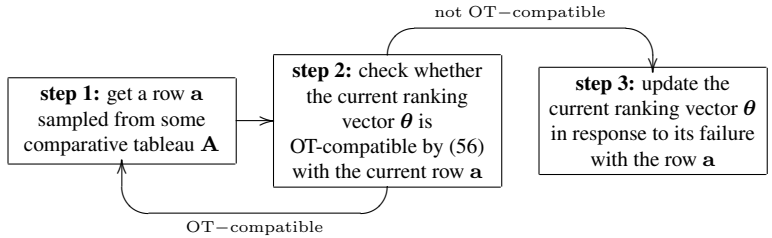
(57) $$\max_{k \in W(\mathbf{a})} \theta_k > \max_{h \in L(\mathbf{a})} \theta_h$$

where $W(\mathbf{a})$ and $L(\mathbf{a})$ are the sets of winner- and loser-preferring constraints:

(58)
$$W(\mathbf{a}) = \left\{ k \in \{1, \ldots, n\} \;\middle|\; a_k = \text{W} \right\} = \text{set of winner-preferring constraints}$$
$$L(\mathbf{a}) = \left\{ k \in \{1, \ldots, n\} \;\middle|\; a_k = \text{L} \right\} = \text{set of loser-preferring constraints}$$

■ **Restatement of OT online algorithms with ranking vectors.** I can of course restate the OT online algorithm (49) as in (59), by assuming that the current hypothesis on the target grammar is represented as a ranking vector rather than as a ranking.

(59)



This final restatement will turn out particularly useful to define update rules for step 3.

8 OT update rules

■ **General shape.** To complete the description of the OT online algorithm (59), we need update rules to be used in step 3. *OT update rules* are functions of the following form:

(60)
$$\begin{array}{l} \text{current comparative row} \\ \quad | \\ (\boldsymbol{\theta}^{\text{old}}, \mathbf{a}) \longrightarrow \boldsymbol{\theta}^{\text{new}} \\ \quad | \qquad\quad | \\ \quad | \qquad \text{updated ranking vector} \\ \quad | \\ \text{current ranking vector} \end{array}$$

[2] The notion of OT-compatibility (56) for ranking vectors with two or more identical components has nothing to do with the notion of OT-compatibility introduced by Tesar and Smolensky (2000), that allows for multiple constraints to be assigned to the same stratum with the corresponding tie resolved additively.

OT update rules are usually constructed out of the following intuition:

(61) a. currently winner-preferring constraints are virtuous, and should thus be *promoted* up the hierarchy by a certain promotion amount;
b. currently loser-preferring constraints are dangerous, and should thus be *demoted* down the hierarchy by a certain demotion amount.

Thus, the updated ranking vector $\boldsymbol{\theta}^{\text{new}} = (\theta_1^{\text{new}}, \dots, \theta_n^{\text{new}})$ is usually constructed out of the current ranking vector $\boldsymbol{\theta}^{\text{old}} = (\theta_1^{\text{old}}, \dots, \theta_n^{\text{old}})$ and the comparative row $\mathbf{a}$ as follows:

(62) $$\theta_k^{\text{new}} = \begin{cases} \theta_k^{\text{old}} + \textit{promotion amount} & \text{if } C_k \text{ is winner-preferring for } \mathbf{a} \text{ (and possibly satisfies other properties)} \\ \theta_k^{\text{old}} - \textit{demotion amount} & \text{if } C_k \text{ is loser-preferring for } \mathbf{a} \text{ (and possibly satisfies other properties)} \\ \theta_k^{\text{old}} & \text{if } C_k \text{ is even for } \mathbf{a} \end{cases}$$

■ **Synopsis of the state of the art on update rules.** Various update rules considered so far in the literature differ along three dimensions, that can be roughly described as follows:

(63) a. *demotion-only* vs *promotion-demotion*:
depending on whether the promotion amounts are always null or not;
b. *minimal* vs *maximal*:
depending on whether only loser-preferring constraints that "need' to be demoted are demoted or else all of them;
c. *gradual* vs *non-gradual*:
depending on whether a given comparative row can trigger two consecutive updates or not.

The main update rules are classified in (64) according to the three dimensions (63), together with the name and main properties of the corresponding OT online algorithm.

(64) *OT update rules*
- *demotion-only*
  - *gradual*
    - *minimal* $\Longrightarrow$ $\text{GLA}_{\text{min}}^{\text{dem}}$: works, and is fast
    - *maximal* $\Longrightarrow$ $\text{GLA}_{\text{max}}^{\text{dem}}$: works, but is slow
  - *non-gradual* $\Longrightarrow$ CD: works, and is suepr-fast
- *demotion-promotion*
  - *gradual*
    - *minimal* $\Longrightarrow$ GLA_{min}: does not work
    - *maximal* $\Longrightarrow$ GLA: does not work
  - *non-gradual* $\Longrightarrow$??

In the rest of this class, we concentrate on the most beautiful update rule: demotion-only, gradual and minimal; and we will compare it to other variants.

9 The most beautiful OT update rule

■ **Currently undominated constraints.** Consider the current comparative row $\mathbf{a}$ and the current ranking vector $\boldsymbol{\theta}^{\text{old}}$ in (65).

(65) a. $\mathbf{a} =$

C_1	C_2	C_3	C_4	C_5	C_6
W	W	E	(L)	E	(L)

b. $\boldsymbol{\theta}^{\text{old}} =$

C_1	C_2	C_3	C_4	C_5	C_6
10	5	20	(15)	100	(5)

Focus on constraints C_4 and C_6 and note that:

(66) a. both are currently loser-preferrers;
b. C_6 is currently ranked below a current winner-preferrer (i.e. C_1);
c. C_4 instead is not currently ranked below any current winner-preferrer.

This difference between C_4 and C_6 is important enough to deserve a name. A constraint C_ℓ is called a *currently undominated loser-preferrer* iff:

(67) a. C_ℓ is currently loser-preferring, namely $a_\ell = \text{L}$;
b. no winner-preferrer is currently ranked above C_ℓ, i.e. $\theta_\ell^{\text{old}} \geq \max_{k \in W(\mathbf{a})} \theta_k^{\text{old}}$.

■ **Definition of the update rule.** The current ranking vector $\boldsymbol{\theta}^{\text{old}} = (\theta_1^{\text{old}}, \dots, \theta_n^{\text{old}})$ is updated to the new ranking vector $\boldsymbol{\theta}^{\text{new}} = (\theta_1^{\text{new}}, \dots, \theta_n^{\text{new}})$ in response to a comparative row $\mathbf{a}$ as in (68): currently undominated loser-preferring constraints are demoted by 1.

(68) $$\theta_k^{\text{new}} = \begin{cases} \theta_k^{\text{old}} - 1 & \text{if } C_k \text{ is a currently undominated loser-preferring constraint} \\ \theta_k^{\text{old}} & \text{otherwise} \end{cases}$$

The OT online algorithm (49) with this demotion-only, gradual and minimal update rule is Boersma's (1997) *minimal demotion-only gradual learning algorithm* ($\text{GLA}_{\text{min}}^{\text{dem}}$).

■ **Example.** The behavior of the algorithm on a small input comparative tableau (starting from the null initial ranking vector) is illustrated by the diagram (69).

(69)

C_1	C_2	C_3
W	W	L
E	L	W

$\Longrightarrow$ $\boldsymbol{\theta}^{\text{init}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

row 1: $\boldsymbol{\theta}^1 = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$, row 2: $\boldsymbol{\theta}^2 = \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$, row 2: $\boldsymbol{\theta}^3 = \begin{bmatrix} 0 \\ -2 \\ -1 \end{bmatrix}$

row 2: $\begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$, row 1: $\begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}$

No matter the order that the rows are fed to the algorithm, after three updates it entertains the ranking vector $\boldsymbol{\theta}^3 = (0, -2, -1)$, that represents the ranking $C_1 \gg C_3 \gg C_2$. Since this ranking is OT-compatible with the input tableau, no further update is triggered.

■ **Tesar/Smolensky's theory.** An elegant analysis of the OT online algorithm (59) with this update rule (68) was developed in Tesar (1995, 1998) and Tesar and Smolensky (1996, 1998, 2000) (T&S).[3] This theory answers the following three questions:

(70) a. Does the OT online algorithm (59) with the update rule (68) converge?
b. If it does, how many updates does it take?
c. And furthermore, what does it converge to?

In the next few sections, I sketch T&S's theory. For simplicity, I assume the initial ranking vector has identical components (say, all 0's). My only contribution is that I restate T&S's analysis in terms of ranking vectors.[4] I think this is the most natural framework to develop T&S's analysis, as most of their reasoning can be straightforwardly algebraized.

10 A useful characterization of OT-compatibility

■ **Why we need characterizations of OT-compatibility.**

(71) a. All analyses of the OT online algorithm presented in these classes assume that the input comparative tableau is OT-compatible.
b. In order to distill the computational consequences of this assumption, we thus need explicit characterizations of OT-compatibility.

This will be a recurrent theme of these classes. T&S offer the following characterization.

■ **Claim 1** A comparative tableau is OT-compatible iff it has the shape in (72) for some $d \leq n$, modulo reordering of its rows and columns and relabeling of the constraints.

(72)

	C_1	C_2	...	C_{d-1}	C_d	C_{d+1}	...	C_n
1st block	W	⋮	...	...	...	...	...	...
	W							
2nd block	E	W						
	⋮	⋮	...	...	...	...	...	...
	E	W						
				⋱		...	...	...
dth block	E	E		E	W			
	⋮	⋮	...	⋮	⋮	...	...	...
	E	E		E	W			

[3] To be precise: T&S consider the update rule (99), which is the non-gradual counterpart of (68). Boersma (1998, p. 323-327) notes that T&S's analysis straightforwardly extends to the gradual (68).

[4] The idea of representing rankings in terms of numerical ranking vectors is actually implicitly already present in T&S's notion of the *offset* of a constraint w.r.t. a ranking, defined as the number of strata above that constraint in that ranking.

Namely, it has a top block of rows whose first entry is W; followed by a second block of rows whose first entry is E and whose second entry is W; and so on, until a final dth block of rows whose first $d-1$ entries are E's and whose dth entry is W.

Proof. To illustrate, consider the comparative tableau in the top left corner in (73). It is OT-comaptible with $C_1 \gg C_2 \gg C_3 \gg C_4 \gg C_5$. Use this ranking as follows:

(73)

	C_2	C_5	C_4	C_1	C_3
$\mathbf{a}_1$	W	L	W	W	L
$\mathbf{a}_2$		L			W
$\mathbf{a}_3$	W	L	W		
$\mathbf{a}_4$	L	L		W	
$\mathbf{a}_5$		W	L		W

given tableau

$\xrightarrow{(a)}$

	C_1	C_2	C_3	C_4	C_5
$\mathbf{a}_1$	W	W	L	W	L
$\mathbf{a}_2$			W		L
$\mathbf{a}_3$		W		W	L
$\mathbf{a}_4$	W	L			L
$\mathbf{a}_5$			W	L	W

reorder the columns according to $C_1 \gg C_2 \gg C_3 \gg C_4 \gg C_5$

$\xrightarrow{(b)}$

	C_1	C_2	C_3	C_4	C_5
$\mathbf{a}_1$	W	W	L	W	L
$\mathbf{a}_4$	W	L			L
$\mathbf{a}_2$			W		L
$\mathbf{a}_3$		W		W	L
$\mathbf{a}_5$			W	L	W

place at the top the rows $\mathbf{a}_1$ and $\mathbf{a}_4$ that have a W corresponding to C_1

$\xrightarrow{(c)}$

	C_1	C_2	C_3	C_4	C_5
$\mathbf{a}_1$	W	W	L	W	L
$\mathbf{a}_4$	W	L			L
$\mathbf{a}_3$		W		W	L
$\mathbf{a}_2$			W		L
$\mathbf{a}_5$			W	L	W

place next the row $\mathbf{a}_3$ that has a W corresponding to C_2

The tableau thus obtained has the shape in (72), with $d = 3$. □

11 A basic invariant

■ **An example.** The input comparative tableau in (69) is only OT-compatible with the ranking (74a). The learning path in (69) displays the property (74b) that the ranking value of the constraint assigned to the kth stratum never goes below $-(k+1)$.

(74) a.
C_1 — stratum 1
|
C_3 — stratum 2
|
C_2 — stratum 3

b. θ_1 never goes below **0**
θ_3 never goes below **-1**
θ_2 never goes below **-2**

T&S note that this property holds in general: current ranking values never get too small.

■ **Claim 2** Assume that the input comparative tableau $\mathbf{A}$ is OT-compatible with a ranking $\gg$. Without loss of generality, assume that this ranking is (75)- otherwise, relabel the constraints.

(75) $C_1 \gg C_2 \gg \ldots \gg C_n$

Then, the ranking vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ entertained at a generic time by the OT online algorithm (59) run on the input tableau $\mathbf{A}$ with the update rule (68) starting from the null initial vector satisfies (76) for every $k = 1, \ldots, n$.

(76) $\theta_k \geq -(k-1)$

Namely, the ranking value of the constraint assigned to the kth stratum (with the 1st stratum being the top stratum) never goes below $-(k-1)$.

Proof. By claim 1, the input tableau is (72) wlg. Here is why (76) holds for $k=1$.

(77) a. The column of (72) corresponding to C_1 does not contain a single L, namely C_1 is never loser-preferring.

b. Thus, the demotion-only update rule (68) never modifies the initial ranking $\theta_1^{\text{init}} = 0$ of constraint C_1.

Here is why (76) holds for $k = 2$:

(78) a. In order for C_2 to be *loser-preferrer*, the current comparative row belongs to the 1st block of (72), which is the only block of (72) where C_2 can have L's.

b. Thus, in order for C_2 to be *undominated*, it must be currently ranked above C_1, as the rows of the 1st block of (72) have a W corresponding to C_1.

c. Thus, in order for C_2 to be undominated loser-preferrer, its current ranking value must be $\theta_2 = 0$, as the ranking value of C_1 is always $\theta_1 = 0$.

d. The constraint C_2 is only demoted (by 1) when it is an undominated loser-preferrer and thus the ranking value of C_2 never gets smaller than $\theta_2 = -1$.

The cases $k > 2$ are dealt with analogously, by induction on k. □

12 Convergence and maximum number of updates

■ **Claim 3** The OT online algorithm (59) with the update rule (68) run on an arbitrary OT compatible input comparative tableau $\mathbf{A}$ with n columns starting from the null initial vector can perform at most $\frac{1}{2}n(n-1)$ updates in step 3.

Proof. Convergence is straightforward, by contradiction:

(79) If the algorithm did not stop after a finite number of updates, some constraint would need to be demoted an infinite number of times. Its ranking value would thus become arbitrarily small, violating the invariant (76).

Thus, the algorithm stops after a total number T of updates. At each update, one or more demotions are performed. Thus, the total number of updates T must be smaller than the total number of demotions, as in (80):

(80) $T \leq$ # of times C_1 is demoted $+$ # of times C_2 is demoted $+ \ldots +$ # of times C_n is demoted

Wlg, assume that the input tableau $\mathbf{A}$ is OT-compatible with the ranking $C_1 \gg C_2 \gg \ldots \gg C_n$. Claim 2 says that C_1 is never demoted; C_2 is demoted at most once; C_3 is demoted at most twice; and so on. Thus, (80) can be made more explicit as in (81).

(81) $T \leq 0 + 1 + \ldots + (n-1)$

Claim 3 follows from the well known identity $0 + 1 + 2 + \ldots (n-1) = \frac{1}{2}n(n-1)$. □

■ **First remark.** The bound provided by claim 3 depends on the number of columns n of the input comparative tableau, but not on the number of rows. This is a general property of (OT and non-OT) online algorithms.

■ **Second remark.** The bound provided by claim 3 is tight. In fact, consider comparative tableaux of the form (82), that Riggle (2007) calls *diagonal*. No matter the order in which the rows are fed to the algorithm, exactly $\frac{1}{2}n(n-1)$ updates are required.

(82)

	C_1	C_2	C_3	C_4	…	C_{n-1}	C_n
$\mathbf{a}_1$	W	L					
$\mathbf{a}_2$		W	L				
			W	L			
⋮					⋱		
$\mathbf{a}_{n-2}$					⋱	L	
$\mathbf{a}_{n-1}$						W	L

To see this, consider for instance the case $n = 4$, illustrated in (83a). As shown by the learning path (83b), it takes $6 = \frac{1}{2}4(4-1)$ updates to reach a ranking vector OT-compatible with the input comparative tableau.

(83) a.

$\mathbf{a}_1$	W	L		
$\mathbf{a}_2$		W	L	
$\mathbf{a}_3$			W	L

b. $\begin{bmatrix}0\\0\\0\\0\end{bmatrix} \xrightarrow{\mathbf{a}_1} \begin{bmatrix}0\\-1\\0\\0\end{bmatrix} \xrightarrow{\mathbf{a}_2} \begin{bmatrix}0\\-1\\-1\\0\end{bmatrix} \xrightarrow{\mathbf{a}_3} \begin{bmatrix}0\\-1\\-1\\-1\end{bmatrix} \xrightarrow{\mathbf{a}_2} \begin{bmatrix}0\\-1\\-2\\-1\end{bmatrix} \xrightarrow{\mathbf{a}_3} \begin{bmatrix}0\\-1\\-2\\-2\end{bmatrix} \xrightarrow{\mathbf{a}_3} \begin{bmatrix}0\\-1\\-2\\-3\end{bmatrix}$

■ **Third remark.** Claim 3 provides a worst-case number of updates over:

(84) a. all possible OT-compatible comparative tableaux $\mathbf{A}$ with n columns;

b. all possible ways of feeding rows of $\mathbf{A}$ to the OT online algorithm (49).

It would be interesting to have worst-case bounds over (84b) only, for a given fixed OT-compatible tableau $\mathbf{A}$. In other words, to have measures of the "online learning complexity" of a given tableau.

■ **Fourth remark.** This initial result shows the benefit of restating the OT online algorithm in terms of comparative notation and ranking vectors:

(85) a. *comparative notation* allows us to depict the data as in (72);

b. *ranking vectors* allow simple algebraic manipulations such as (80).

We will have many many more occasions to appreciate the benefits of such a formulation of the OT online algorithm. It took a bit of time, but it was time well spent.

13 Operations that preserve OT-compatibility

■ **Operations on ranking vectors.** Consider two ranking vectors:

$$\boldsymbol{\theta}' = (\theta'_1, \ldots, \theta'_n), \qquad \boldsymbol{\theta}'' = (\theta''_1, \ldots, \theta''_n) \tag{86}$$

Their *component-wise sum* $\boldsymbol{\theta}' + \boldsymbol{\theta}''$ is the vector whose components are the sum of the corresponding components of $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$:

$$\boldsymbol{\theta}' + \boldsymbol{\theta}'' = \begin{bmatrix} \theta'_1 \\ \vdots \\ \theta'_n \end{bmatrix} + \begin{bmatrix} \theta''_1 \\ \vdots \\ \theta''_n \end{bmatrix} = \begin{bmatrix} \theta'_1 + \theta''_1 \\ \vdots \\ \theta'_n + \theta''_n \end{bmatrix} \tag{87}$$

Their *component-wise maximum* $\max\{\boldsymbol{\theta}', \boldsymbol{\theta}''\}$ is the vector whose components are the maxima of the corresponding components of $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$:

$$\max\{\boldsymbol{\theta}', \boldsymbol{\theta}''\} = \max\left\{ \begin{bmatrix} \theta'_1 \\ \vdots \\ \theta'_n \end{bmatrix}, \begin{bmatrix} \theta''_1 \\ \vdots \\ \theta''_n \end{bmatrix} \right\} = \begin{bmatrix} \max\{\theta'_1, \theta''_1\} \\ \vdots \\ \max\{\theta'_n, \theta''_n\} \end{bmatrix} \tag{88}$$

■ **Claim 4** If two ranking vectors $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$ are OT-compatible with a comparative tableau $\mathbf{A}$, then their component-wise maximum $\boldsymbol{\theta} = \max\{\boldsymbol{\theta}', \boldsymbol{\theta}''\}$ is OT-compatible with $\mathbf{A}$.

Proof. It is enough to assume that $\mathbf{A}$ consists of a single row $\mathbf{a}$. I reason as follows:

$$\begin{aligned}
\max_{h \in W(\mathbf{a})} \theta_h &\overset{(a)}{=} \max_{h \in W(\mathbf{a})} \underbrace{\max\left\{\theta'_h, \theta''_h\right\}}_{\theta_h} \\
&\qquad \text{by the definition of } \boldsymbol{\theta} \text{ as component-wise maximum of } \boldsymbol{\theta}' \text{ and } \boldsymbol{\theta}'' \\
&\overset{(b)}{=} \max\left\{ \max_{h \in W(\mathbf{a})} \theta'_h, \max_{h \in W(\mathbf{a})} \theta''_h \right\} \\
&\qquad \text{by commuting the two maximum operators} \\
&\overset{(c)}{>} \max\left\{ \max_{k \in L(\mathbf{a})} \theta'_k, \max_{k \in L(\mathbf{a})} \theta''_k \right\} \\
&\qquad \text{since both } \boldsymbol{\theta}' \text{ and } \boldsymbol{\theta}'' \text{ are OT-compatible with } \mathbf{a} \\
&\overset{(d)}{=} \max_{k \in L(\mathbf{a})} \underbrace{\max\left\{\theta'_k, \theta''_k\right\}}_{\theta_k} \\
&\qquad \text{by commuting again the two maximum operators} \\
&\overset{(e)}{=} \max_{k \in L(\mathbf{a})} \theta_k \\
&\qquad \text{by the definition of } \boldsymbol{\theta} \text{ as component-wise maximum of } \boldsymbol{\theta}' \text{ and } \boldsymbol{\theta}''.
\end{aligned} \tag{89}$$

The chain of (strict) inequalities (89) shows that $\boldsymbol{\theta}$ is OT-compatible with $\mathbf{a}$. □

■ **Preservation of OT-compatibility.** Consider an operation over ranking vectors (90), such as component-wise sum or maximum:

$$(\boldsymbol{\theta}', \boldsymbol{\theta}'') \mapsto \boldsymbol{\theta} \tag{90}$$

The operation *preserves OT-compatibility* iff (91) holds for any comparative tableau $\mathbf{A}$:

$$\left.\begin{array}{c} \boldsymbol{\theta}' \text{ is OT-compatible with } \mathbf{A} \\ \boldsymbol{\theta}'' \text{ is OT-compatible with } \mathbf{A} \end{array}\right\} \implies \boldsymbol{\theta} \text{ is OT-compatible with } \mathbf{A} \tag{91}$$

The operation of point-wise sum (87) does not preserve OT-compatibility (show it!). By claim 4, the operation (88) of point-wise maximum does.

14 A complete characterization of the final ranking vector

■ **Claim 5** The final ranking vector $\boldsymbol{\theta}^{\text{fin}}$ returned by the OT online algorithm (59) run with the update rule (68) on an input OT-compatible comparative tableau $\mathbf{A}$ starting from the null initial vector is uniquely characterized by (92)

$$\boldsymbol{\theta}^{\text{fin}} = \max \underbrace{\left\{ \mathbf{v} \in \{0, -1, -2, \ldots\}^n \,\middle|\, \mathbf{v} \text{ is OT-compatible with } \mathbf{A} \right\}}_{(*)} \tag{92}$$

namely as the maximum of non-positive integer ranking vectors OT-compatible with $\mathbf{A}$.

Proof. The final vector $\boldsymbol{\theta}^{\text{fin}}$ returned by the algorithm is OT-compatible with the input comparative tableau and has integer non-positive components. Thus, $\boldsymbol{\theta}^{\text{fin}}$ belongs to the set (*) in (92). Thus, to prove (92) it is sufficient to prove that $\boldsymbol{\theta}^{\text{fin}} \geq \mathbf{v}$ holds for every ranking vector $\mathbf{v}$ in the set (*) in (92). This fact is guaranteed by the invariant (76). □

■ **It makes good sense.** The characterization (92) makes sense by claim 4, that ensures that the component-wise maximum of OT-compatible vectors is OT-compatible too.[5]

[5] Strictly speaking, claim 4 does not apply to the case in (92), as the set (*) in (92) is not finite. Yet, consider an arbitrary ranking vector $\overline{\mathbf{v}}$ in the set (*) in (92). Identity (i) trivially holds.

$$\begin{aligned}
\text{(i)} \quad \max \underbrace{\left\{ \mathbf{v} \in \{0, -1, \ldots\}^n \,\middle|\, \mathbf{v} \text{ is OT-compatible with } \mathbf{A} \right\}}_{(*)} = \\
= \max \underbrace{\left\{ \mathbf{v} \in \{0, -1, \ldots\}^n \,\middle|\, \mathbf{v} \text{ is OT-compatible with } \mathbf{A} \text{ and } \mathbf{v} \geq \overline{\mathbf{v}} \right\}}_{(**)}
\end{aligned}$$

Since (**) is finite, claim 4 applies, ensuring that its maximum is OT-compatible with $\mathbf{A}$.

15 A non-minimal variant

■ **Minimal and non-minimal update rules.** *Undominated* loser-preferrer constraints are those that really need to be taken care of. Thus:

(93) An update rule is *minimal* if it only updates those loser-preferrers that are currently undominated, but not the other loser-preferrers.

Update rule (68) is *minimal*, as it only demotes currently *undominated* loser-preferrers.

■ **A non-minimal variant.** What happens if we demote *all* loser-preferers instead? Thus, consider the *non-minimal* variant of the update rule (68) in (94).

(94) $$\theta_k^{\text{new}} = \begin{cases} \theta_k^{\text{old}} - 1 & \text{if } C_k \text{ is a (}\textit{dominated or undominated}\text{) loser-preferrer} \\ \theta_k^{\text{old}} & \text{otherwise} \end{cases}$$

The OT online algorithm (59) with the update rule (94) is Boersma's (1997) *demotion-only Gradual Learning Algorithm* (GLA^{dem}).

■ **Claim 6** Minimal and non-minimal updates compare as in (95); See Magri (2009).

(95) a. *Finite time convergence:*
holds for both the minimal update (68) and the non-minimal update (94).
b. *The invariant (76)*:
holds for the minimal update (68) but not for the non-minimal update (94).
c. *Fast convergence (quadratic in n):*
holds for the minimal update (68) but not for the non-minimal update (94).

Proof. The proof of (95a) is simple but omitted. Counterexample (96) shows (95b): θ_2 drops to -3; yet, the input tableau is OT-compatible with the ranking $C_1 \gg C_2 \gg C_3 \gg C_4$, that assigns C_2 to the second stratum; hence, the invariant (76) fails.

(96)
$$\begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \\ \text{W} & \text{L} & & \\ \text{W} & \text{L} & \text{L} & \\ \text{W} & \text{L} & \text{L} & \text{L} \end{array} \Longrightarrow \begin{array}{c} \boldsymbol{\theta}^{\text{init}} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array} \xrightarrow{\text{row 1}} \begin{array}{c} \boldsymbol{\theta}^{1} \\ \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0 \end{bmatrix} \end{array} \xrightarrow{\text{row 2}} \begin{array}{c} \boldsymbol{\theta}^{2} \\ \begin{bmatrix} 0 \\ -2 \\ -1 \\ 0 \end{bmatrix} \end{array} \xrightarrow{\text{row 3}} \begin{array}{c} \boldsymbol{\theta}^{3} \\ \begin{bmatrix} 0 \\ \mathbf{-3} \\ -2 \\ -1 \end{bmatrix} \end{array}$$

The problem is that an L can be demoted for "unsubstantial" reasons. These extra demotions take extra time and thus slow down convergence, as stated in (95c). □

16 A non-gradual variant

■ **Gradual and non-gradual update rules.** Consider again the behavior of the OT online algorithm (59) with the beautiful update rule (68) as illustrated in (69), repeated in (97).

(97)
$$\begin{array}{ccc} C_1 & C_2 & C_3 \\ \text{W} & \text{W} & \text{L} \\ \text{E} & \text{L} & \text{W} \end{array} \Longrightarrow \begin{array}{c} \boldsymbol{\theta}^{\text{init}} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \end{array} \begin{array}{l} \xrightarrow{\text{row 1}} \\ \xrightarrow{\text{row 2}} \end{array} \begin{array}{c} \boldsymbol{\theta}^{1} \\ \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \\ \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \end{array} \begin{array}{l} \xrightarrow{\text{row 2}} \\ \xrightarrow{\text{row 1}} \end{array} \begin{array}{c} \boldsymbol{\theta}^{2} \\ \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix} \end{array} \xrightarrow{\text{row 2}} \begin{array}{c} \boldsymbol{\theta}^{3} \\ \begin{bmatrix} 0 \\ -2 \\ -1 \end{bmatrix} \end{array}$$

In the top path, row 2 triggers two *consecutive* updates. We might have wanted to save time, replacing those two updates with a single jump. To get the jump, we should have demoted more the first time we encountered row 2. Thus:

(98) An update rule is called *non-gradual* iff update is large enough that no row can trigger two consecutive updates.

■ **A non-gradual variant.** How much more should we have demoted to get the jump? T&S's answer: demote just below the currently top ranked winner-preferrer, as in (99).

(99)
$$\theta_k^{\text{new}} = \begin{cases} \max\limits_{h \in W(\mathbf{a})} \theta_h^{\text{old}} - 1 & \text{if } C_k \text{ is an undominated loser-preferrer} \\ \theta_k^{\text{old}} & \text{otherwise} \end{cases}$$

(ranking value of the currently top-ranked winner-preferrer)

The gradual update rules (68) and the non-gradual variant (99) compare as follows:

(100) a. gradual update rule (68):
demote C_k just below θ_k^{old} = current ranking value of C_k
b. non-gradual update rule (99):
demote C_k just below $\max\limits_{h \in W(\mathbf{a})} \theta_h^{\text{old}}$ = ranking value of the currently top ranked winner-preferrer.

The OT online algorithm (59) with update (99) is T&S's *Constraint Demotion* (CD).

■ **An example.** The behavior of CD on the same input comparative tableau considered in (97) starting from the null initial vector is described in (101).

(101)
$$\begin{array}{ccc} F_{\text{pos}} & F_{\text{gen}} & M \\ \text{W} & \text{W} & \text{L} \\ \text{E} & \text{L} & \text{W} \end{array} \Longrightarrow \begin{array}{c} \boldsymbol{\theta}^{\text{init}} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \end{array} \begin{array}{l} \xrightarrow{\text{row 1}} \\ \xrightarrow{\text{row 2}} \end{array} \begin{array}{c} \boldsymbol{\theta}^{1} \\ \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \\ \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \end{array} \begin{array}{l} \xrightarrow{\text{row 2}} \\ \xrightarrow{\text{row 1}} \end{array} \begin{array}{c} \boldsymbol{\theta}^{2} \\ \begin{bmatrix} 0 \\ -2 \\ -1 \end{bmatrix} \\ \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix} \end{array} \xrightarrow{\text{row 2}} \begin{array}{c} \boldsymbol{\theta}^{3} \\ \begin{bmatrix} 0 \\ -2 \\ -1 \end{bmatrix} \end{array}$$

The two consecutive updates by row 2 in (97) have been replaced by a jump in (101). Update rule (99) can indeed be shown to be non-gradual, in the sense of (98).

■ **Claim 7** Claims 2 and 3 concerning fast convergence and invariants, extend from the beautiful gradual update rule (68) to the nnon-gradual variant (99).

Proof. The update rule (99) is a "sped up" version of the update rule (68). □

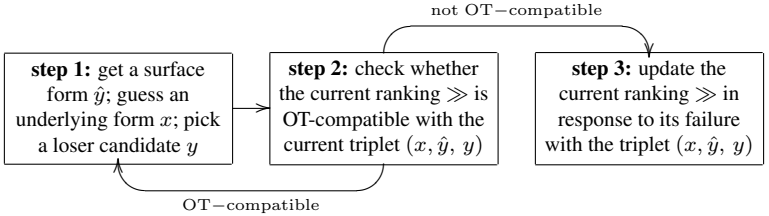
17 From the modeling perspective, promotion is needed

■ **Faithful underlying forms.** Recall the *input property* (6a) of the *early stage*, namely:

(102) *Lack of knowledge of alternations.* Throughout this stage, the child does not yet have knowledge of morphology and is thus blind to alternations.

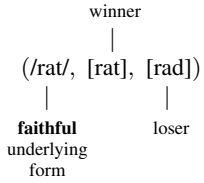
How can this property be modeled? Obviously, as an assumption on the input to the model. Let's go back to the initial description of the OT online algorithm:

(103)

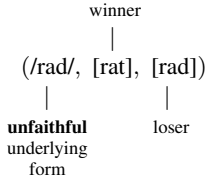


Consider the universal specifications (29); suppose the learner is provided with the surface form $\hat{y}$ = [rat] at step 1. The learner needs to guess the corresponding underlying form x. There are two possibilities:

(104) a. the learner can posit the faithful underlying form x = /rat/:



b. the learner can posit an unfaithful underlying form x = /rad/:



By(104b), the target phonotactics enforces *final devoicing*. But the learner cannot see that, as he is blind to alternations by (102). Following Prince and Tesar (2004) and Hayes (2004) a.o., I thus assume he posits (104a). In general, I model (102) as follows:

(105) The model (103) always assumes in step 1 that the underlying form x is identical to the given surface form $\hat{y}$.

In comparative notation, this means that the input tableau has the shape in (106): the faithfulness constraints cannot contain any L.

(106)

faithfulness constraints	markedness constraints
E, W	L, E, W

■ **Remarks.** Assumption (105) only makes sense provided that (107) is the case. Thus, (105) does not apply to cases such as syllabification, stress assignment, etceteras.

(107) the set of underlying forms $\mathcal{X}$ = the set of surface forms $\mathcal{Y}$

Tesar (2008) proves assumption (105) is sound, as the corresponding comparative tableau (106) is OT-compatible (under mild assumptions on the constraint set).

■ **The problem.** Suppose we try to model the early stage of the acquisition of phonotactics with an OT online algorithm that performs demotion only. Then:

(108) a. by (105), we posit fully faithful underlying forms;
b. thus, the faithfulness constraints are never LPCs;
c. thus, the faithfulness constraints are never re-ranked by demotion-only;
d. thus, their intermediate and final rankings are identical to their initial ranking;
e. namely, their ranking is always the same.

And this cannot be right, for at least three reasons.

■ **First reason.**

(109) a. The typology might contain two languages that require the opposite relative ranking of two faithfulness constraints.
b. If the algorithm starts from the same initial ranking for every language, then it fails to model the early stage for at least one of the two languages.

Examples of (109a): there are lots of such examples; we will consider one in class 4.

■ **Second reason.**

(110) a. Two children exposed to the same target language might entertain a different relative ranking of two faithfulness constraints.
b. If all children start from the same initial ranking (i.e. UG provides the initial ranking), then the algorithm fails to model at least one of the two children.

Examples of (110a): Pater and Barlow (2003) describe three children acquiring English onset clusters that exhibit intermediate stages that differ (also) because of the relative ranking of some faithfulness constraints.

■ **Third reason.**

(111) a. A child might go through two acquisitional stages that differ in particular because of the relative ranking of two faithfulness constraints.

b. The algorithm fails to model at least one of the two stages, no matter how we pick the initial ranking.

Examples of (111a): I haven't found any in the literature I have read so far.

■ **Conclusion.** We thus have the following necessary condition:

(112) The OT online model of the early stage cannot work with demotion only. Rather, we need constraint promotion in order to re-rank the faithfulness constraints too, despite the fact that they are always winner-preferrers.

Let's thus review what is currently known in the literature about constraint promotion.

18 From the computational perspective, promotion is hard to get

■ **T&S's skepticism: the credit problem.** Tesar and Smolensky (1998, pp. 244-245) voice skepticism against constraint promotion in the celebrated passage in (113).

(113) "[The update rules considered in the preceding subsections are] defined entirely in terms of demotion [...]. One could reasonably ask if this is an arbitrary choice; couldn't the learner just as easily promote constraints toward the correct hierarchy? The answer is no, and understanding why reveals the logic behind [demotion-only update rules]. [...] A hypothetical promotion operation would move the [winner-preferring constraints] up in the hierarchy. But [...] it isn't clear which of the [winner-preferring constraints] should be promoted — perhaps all of them, or perhaps just one. Other data might require one of the [winner-preferring constraints] to be dominated by one of the [loser-preferring constraints]. [The current comparative row] gives no basis for choosing."

This is the *credit problem*, after Dresher (1999): if a comparative row has multiple winner-preferrers, which one should be credited the merit of ensuring OT-compatibility with that row, and thus promoted?

■ **Boersma's conjecture.** Boersma (1997, 1998) conjectures that gradualness might solve the credit problem: if promotions are small, no big harm comes by promoting a winner-preferrer that dosen't deserve it. Boersma and Hayes (2001, p. 52) put it as follows:

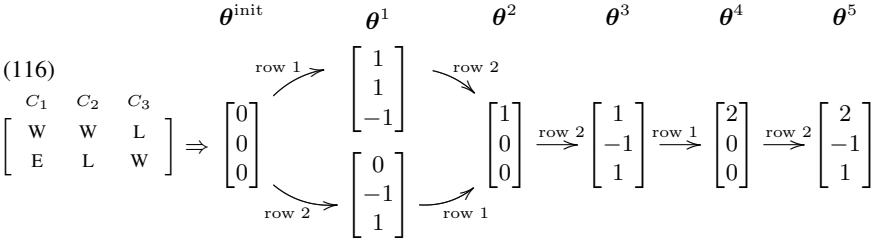
(114) "[A comparative row not OT-compatible with the current ranking] constitutes evidence for two things. First, it is likely that [the loser-preferring] constraints [...] are ranked too high. Second, it is likely that [the winner-preferring] constraints [...] are ranked too low. Neither of these conclusions can be taken as a certainty. However, this uncertainty is not crucial, since the ultimate shape of the grammar will be determined by the ranking values that the constraints will take on in the long term, with exposure to a full range of representative forms. The hypothesis [...] is that moderate adjustments of ranking values will ultimately achieve the right grammar. Therefore, the algorithm is set up so as to make a small adjustment to all [active] constraints [...]."

Thus, Boersma considers the update rule (115). This is the only example of promotion-demotion update rule in the literature. The OT online algorithm (59) with this update rule is Boersma's (1997) (deterministic) *Gradual Learning Algorithm* (GLA).

(115) $$\theta_k^{\text{new}} = \begin{cases} \theta_k^{\text{old}} - 1 & \text{if } C_k \text{ is loser-preferrer} \\ \theta_k^{\text{old}} + 1 & \text{if } C_k \text{ is winner-preferrer} \\ \theta_k^{\text{old}} & \text{if } C_k \text{ is even} \end{cases}$$

The behavior of the GLA on the same input comparative tableau considered in (97) starting from the null ranking vector is described in the diagram (116).

(116)



Note the non-monotonic dynamics of the components of the current ranking vector: for instance, the ranking value θ_2 first grows from 0 to 1, then it drops from 1 to -1, then it grows again to 0 only to finally drop again down to -1.

■ **Pater's counterexample.** It is easy to prove that:

(117) The GLA converges in the case of input comparative tableaux that have a unique winner-preferring constraint per row, and thus do not raise any credit problem.[6]

But what about the case of input tableaus with multiple winner-preferrers per row? Does Boersma' conjecture (114) prove correct? As stressed in Keller and Asudeh (2002), this question has regrettably remained open for various years. Until recently Pater (2008) has provided a negative answer:

(118) The GLA does *not* converge for input tableaux with multiple winner-preferrers per row. Thus, gradualness does not cure the credit problem, contrary to (114).

In fact, Pater considers the comparative tableau in (119), that exacerbates the credit problem by stacking multiple rows with two winner-preferring constraints.

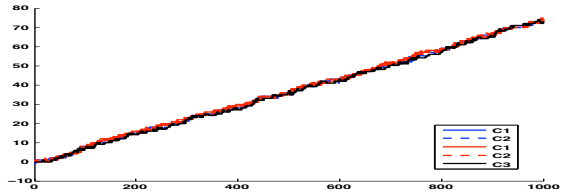
[6] As far as I know, no proof of this fact actually exists so far in the literature. Yet, claim (118) follows straightforwardly from the developments of class 2.

(119)

C_1	C_2	C_3	C_4	C_5
W	L	W		
	W	L	W	
		W	L	W
			W	L

He notes that the GLA run on the input comparative tableau (119) starting from the null initial vector and the rows sampled uniformly keeps increasing the current ranking values as in (120), without ever converging to an OT-compatible vector.

(120)



■ **Conclusion.** Fikkert and De Hoop (2009, pp. 314-315) note that

(121) "The field of the acquisition of phonology in OT is in fact split into two subdivisions. In one division, research is based on empirical data: child language acquisition data are studied and developmental patterns are unraveled. The other division investigates learnability issues. [...] Ideally, the model mimics real language acquisition, but this is [not] of much concern in actual learnability studies. Often the term 'learning' is used in the latter context, while 'acquisition' refers to child language development [...]. So far, learnability studies have not taken actual acquisition patterns, or real learners, into account."

The main conclusion of this class is a striking example of this divide between computational simplicity and modeling complexity, as we have seen that:

(122) a. constraint promotion is needed from the *modeling perspective*, as in (112);

b. constraint promotion is hard to get from the *computational perspective*.

This *impasse* will keep us busy in the next three classes.

19 A detailed explanation of Pater's counterexample (119)

■ **First step.** The beginning of a possible run of the GLA on Pater's counterexample (119) starting from the null initial ranking vector is (123).

(123)

$$\begin{bmatrix}0\\0\\0\\0\\0\end{bmatrix} \xrightarrow{\text{row 1}} \begin{bmatrix}+1\\-1\\+1\\0\\0\end{bmatrix} \xrightarrow{\text{row 2}} \begin{bmatrix}+1\\-1\\+1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\+1\\-1\\+1\\0\end{bmatrix} \xrightarrow{\text{row 3}} \begin{bmatrix}+1\\-1\\+1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\+1\\-1\\+1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\+1\\-1\\+1\end{bmatrix} \xrightarrow{\text{row 4}} \begin{bmatrix}+1\\-1\\+1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\+1\\-1\\+1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\+1\\-1\\+1\end{bmatrix} + \begin{bmatrix}0\\0\\0\\+1\\-1\end{bmatrix} \xrightarrow{\cdots}$$

Thus, the current ranking vector $\boldsymbol{\theta}$ entertained by the GLA has the shape in (124).

(124)

$$\boldsymbol{\theta} = \begin{bmatrix}\theta_1\\\theta_2\\\theta_3\\\theta_4\\\theta_5\end{bmatrix} = \underbrace{\alpha_1 \begin{bmatrix}1\\-1\\1\\0\\0\end{bmatrix}}_{\substack{\text{\# of updates triggered}\\\text{by the 1st row of (119)}\\\text{corresponds to}\\\text{the 1st row of (119)}}} + \alpha_2 \begin{bmatrix}0\\1\\-1\\1\\0\end{bmatrix} + \cdots + \underbrace{\alpha_4 \begin{bmatrix}0\\0\\0\\1\\-1\end{bmatrix}}_{\substack{\text{\# of updates triggered}\\\text{by the 4th row of (119)}\\\text{corresponds to}\\\text{the 4th row of (119)}}}$$

Thus, the search space of the GLA run on Pater's counterexample (119) starting from the null initial vector is a subset of the set of ranking vectors of the form (125).

(125) $$\boldsymbol{\theta} = \begin{bmatrix}\theta_1\\\theta_2\\\theta_3\\\theta_4\\\theta_5\end{bmatrix} = \begin{bmatrix}\alpha_1\\\alpha_2-\alpha_1\\\alpha_1+\alpha_3-\alpha_2\\\alpha_2+\alpha_4-\alpha_3\\\alpha_3-\alpha_4\end{bmatrix}, \quad \alpha_1, \alpha_2, \alpha_3, \alpha_4 \geq 0$$

■ **Second step.** Pater's tableau (119) is only OT-compatible with the ranking (126).

(126) $C_1 \gg C_2 \gg C_3 \gg C_4 \gg C_5$

Thus, a ranking vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$ is OT-compatible with Pater's comparative tableau (119) iff it satisfies the four strict inequalities (127).

(127) $\theta_1 > \theta_2 > \theta_3 > \theta_4 > \theta_5$.

By virtue of the characterization in (125), the four inequalities (127) can be rewritten as the four strict inequalities (128), in terms of the four coefficients $\alpha_1, \alpha_2, \alpha_3, \alpha_4$.

(128)
a. $\theta_1 > \theta_2 \quad \Rightarrow \quad \alpha_1 > \alpha_2 - \alpha_1$
b. $\theta_2 > \theta_3 \quad \Rightarrow \quad \alpha_2 - \alpha_1 > \alpha_1 + \alpha_3 - \alpha_2$
c. $\theta_3 > \theta_4 \quad \Rightarrow \quad \alpha_1 + \alpha_3 - \alpha_2 > \alpha_2 + \alpha_4 - \alpha_3$
d. $\theta_4 > \theta_5 \quad \Rightarrow \quad \alpha_2 + \alpha_4 - \alpha_3 > \alpha_3 - \alpha_4$

■ **Third step.** It turns out that:

(129) There exist no coefficients $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ that satisfy all four inequalities (128).

Here is a way to see (129). If (128d) is subtracted from (128b), we get (130a); if (128c) and (128d) are summed together, we get (130b); if (128a) and (128d) are summed together and then the result divided by 2, we get (130c); if (130b) and (130c) are summed together, we get (130d). Since (130d) contradicts (130a), then the four strict inequalities (128) cannot all be satisfied at the same time.

(130)
a. $-2\alpha_1 + \alpha_2 + \alpha_3 - 2\alpha_4 > 0$
b. $\alpha_1 - \alpha_2 + \alpha_4 > 0$
c. $\alpha_1 - \alpha_3 + \alpha_4 > 0$
d. $-2\alpha_1 + \alpha_2 + \alpha_3 - 2\alpha_4 < 0$

Thus, the GLA fails on Pater's comparative tableau (119) because its search space (125) contains no ranking vector (128) OT-compatible with the tableau. In other words, the algorithm struggles to reach something which is outside of its reach.

References

Bernhardt, Barbara Handford, and 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". In *IFA Proceedings 21*, 43–58. University of Amsterdam: Institute for Phonetic Sciences.

Boersma, Paul. 1998. Functional Phonology. Doctoral Dissertation, University of Amsterdam. The Hague: Holland Academic Graphics.

Boersma, Paul. 2008. "Some Correct Error-driven Versions of the Constraint Demotion Algorithm". To appear in *Linguistic Inquiry*.

Boersma, Paul, and Bruce Hayes. 2001. "Empirical Tests for the Gradual Learning Algorithm". *Linguistic Inquiry* 32:45–86.

Cesa-Bianchi, Nicolò, and Gábor Lugosi. 2006. *Prediction, Learning, and Games*. Cambridge University Press.

Dresher, E. 1999. "Charting the Learning Path: Cues to Parameter Setting". *Linguistic Inquiry* 30:27–67.

Fikkert, Paula, and Helen De Hoop. 2009. "Language acquisition in optimality theory". *Linguistics* 47.2:311–357.

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

Hayes, Bruce. 2004. "Phonological Acquisition in Optimality Theory: The Early Stages". In *Constraints in Phonological Acquisition*, ed. R. Kager, J. Pater, and W. Zonneveld, 158–203. Cambridge University Press.

Kazazis, Kostas. 1969. "Possible evidence for (near-)underlying forms in the speech of a child". In *Papers from the Fifth Regional Meeting of the Chicago Linguistic Society (CLS5)*, ed. Robert I. Binnick, Alice Davison, Georgia McGreen, and Jerry L. Morgan, 382–388. Chicago, IL: Chicago Linguistic Society.

Keller, Frank, and Ash Asudeh. 2002. "Probabilistic Learning Algorithms and Optimality Theory". *Linguistic Inquiry* 33.2:225–244.

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

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

Levelt, Clara C., Niels O. Schiller, and Willem J. Levelt. 2000. "The Acquisition of Syllable Types". *Language Acquisition* 8(3):237–264.

Magri, Giorgio. 2009. A Theory of Individual Level Predicates Based on Blind Mandatory Implicatures. Constraint Promotion for Optimality Theory. Doctoral Dissertation, MIT.

Marr, David. 1982. *Vision. A computational Investigation into the human representation and processing of visual information*. Freeman and Company.

McLeod, Sharynne, Jan van Doorn, and Vicki A. Reed. 2001. "Normal Acquisition of Consonant Clusters". *American Journal of Speech-Language Pathology* 10:99–110.

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.

Pater, Joe, and Jessica A. Barlow. 2003. "Constraint conflict in cluster reduction". *Journal of Child Language* 30:487–526.

Poggio, Tomaso, and Steve Smale. 2003. "The Mathematics of Learning: Dealing with Data". *Notices of the American Mathematical Society* 50.5:537–544.

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

Prince, Alan, and Paul Smolensky. 2004. *Optimality Theory: Constraint Interaction in Generative Grammar*. Blackwell. As Technical Report CU-CS-696-93, Department of Computer Science, University of Colorado at Boulder, and Technical Report TR-2, Rutgers Center for Cognitive Science, Rutgers University, New Brunswick, NJ, April 1993. Rutgers Optimality Archive 537 version, 2002.

Prince, Alan, and Bruce Tesar. 2004. "Learning Phonotactic Distributions". In *Constraints in Phonological Acquisition*, ed. R. Kager, J. Pater, and W. Zonneveld, 245–291. Cambridge University Press.

Riggle, Jason. 2007. "The Complexity of Ranking Hypotheses in Optimality Theory". *Computational Linguistics* 1:–.

Smolensky, Paul. 1996. "On the Comprehension/Production Dilemma in Child Language". *Linguistic Inquiry* 27.4:720–731.

Tesar, Bruce. 1995. "Computational Optimality Theory". Doctoral Dissertation, University of Colorado, Boulder. ROA 90.

Tesar, Bruce. 1998. "Error-Driven Learning in Optimality Theory via the Efficient Computation of Optimal Forms". In *Is the Best Good Enough? Optimality and Competition in Syntax*, ed. Pilar Barbosa, Danny Fox, Paul Hagstrom, Martha McGinnis, and David Pesetsky, 421–435. Cambridge, MA: MIT Press.

Tesar, Bruce. 2008. "Output-Driven Maps". Ms., Rutgers University; ROA-956.

Tesar, Bruce, and Paul Smolensky. 1996. "Learnability in Optimality Theory (long version)". Technical Report 96-3, Department of Cognitive Science, Johns Hopkins University, Baltimore. Available as Rutgers Optimality Archive 156, http://ruccs.rutgers.edu/roa.html.

Tesar, Bruce, and Paul Smolensky. 1998. "Learnability in Optimality Theory". *Linguistic Inquiry* 29:229–268.

Tesar, Bruce, and Paul Smolensky. 2000. *Learnability in Optimality Theory*. Cambridge, MA: The MIT Press.

Zamuner, Tania S., LouAnn Gerken, and Michael Hammond. 2005. "The acquisition of phonology based on input: a closer look at the relation of cross-linguistic and child language data". *Lingua* 115(10):1329–1474.