TOOLS FOR THE ROBUST ANALYSIS OF ERROR-DRIVEN RANKING ALGORITHMS AND THEIR IMPLICATIONS FOR MODELING THE CHILD ACQUISITION OF PHONOTACTICS

GIORGIO MAGRI

[Accepted subject to revisions at the *Journal of Logic and Computation*;
last revised on December 2012]

Abstract — *Error-driven ranking algorithms* (EDRAs) perform a sequence of slight re-rankings of the constraint set triggered by mistakes on the incoming stream of data. In general, the sequence of rankings entertained by the algorithm, and in particular the final ranking entertained at convergence, depend not only on the grammar the algorithm is trained on, but also on the specific way data are sampled from that grammar and fed to the algorithm. The *robust analysis* of EDRAs pinpoints at properties of the predicted sequence of rankings that are robust, namely only depend on the target grammar, not on the way the data are sampled from it. Tesar and Smolensky (1998) develop a tool for the robust analysis of EDRAs that perform constraint demotion only, that is reviewed in detail. The paper then develops a new tool for the robust analysis of EDRAs that perform both constraint demotion and promotion. The latter tool is applied to the robust analysis of the EDRA model of the child early acquisition of phonotactics, through a detailed discussion of *restrictiveness* on three case studies from Prince and Tesar (2004), that crucially require EDRAs that perform both demotion and promotion.

1. Introduction

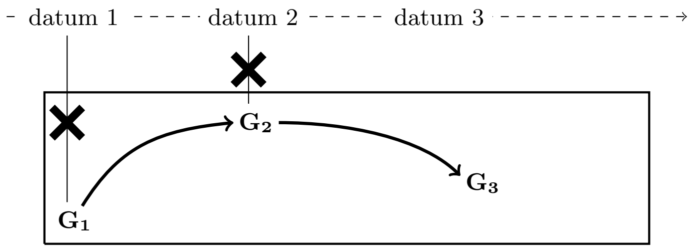
This Section provides and informal overview of the problem of the robust analysis of error-driven ranking algorithms, of the analytical tools that will be presented in the paper and of their application to the problem of modeling the child acquisition of phonotactics.

1.1. **Error-driven ranking algorithms (EDRAs).**

Assume that the learner is provided with the space of possible grammars G_1, G_2, etcetera. Data come in a stream, one piece of data at the time. And the learner maintains a *current grammar*, which represents its current hypothesis on the *target grammar* the learner is being trained on. Suppose that at a certain time the current grammar, say G_1, is inconsistent with the current piece of data, say datum 1. Prompted by this error, the learner updates the current grammar G_1 to a slightly different grammar, say G_2, that sits nearby in the space of grammars, as depicted in (1). This process is repeated over and over again. Until the learner eventually stops making errors, namely it converges to a *final grammar* consistent with the stream of data, and learning ceases.

I wish to thank Adam Albright for lots of help and discussion. I also wish to thank Paul Boersma, Alan Prince, Paul Smolensky, Donca Steriade, and Bruce Tesar, as well as a JLC anonymous reviewer. Part of this paper has been presented at the 47th annual meeting of the Chicago Linguistic Society (Chicago, April 2011; see Magri 2011). This work was supported in part by a grant from the Fyssen Research Foundation ("Child acquisition of sound patterns: a computational modeling approach").

(1)



This learning scheme is called *error-driven*, as the learning dynamics is driven by the errors performed on the incoming stream of data. This scheme has been thoroughly investigated in the Machine Learning literature (under the heading of *online learning*; for a review, see Kivinen 2003 and Cesa-Bianchi and Lugosi 2006, chapters 11, 12). Within the linguistic literature, error-driven learning dates back to at least Wexler and Culicover (1980).

In this paper, I focus on error-driven learning within the mainstream phonological framework of *Optimality Theory* (OT; Prince and Smolensky 2004, Kager 1999). According to OT, the underlying typology of grammars is parameterized by the collection of rankings of a given constraint set. The error-driven learner thus maintains a current ranking of the constraints and performs a slight re-ranking whenever the current ranking turns out to be inconsistent with the current piece of data. Re-ranking can take different forms. For instance, the algorithm could *demote* those constraints that incorrectly punish the current piece of data. Or it could *promote* those constraints that instead favor the current piece of data. Or it could adopt a mixed re-ranking strategy that combines both constraint demotion and promotion. An error-driven learner within OT is called an *error-driven ranking algorithm* (henceforth: EDRA; Tesar and Smolensky 1998, Boersma 1998). Section 2 introduces EDRAs in detail.

1.2. **Tools for the robust analysis of EDRAs.** The sequence of data fed to the error-driven learner (1) is called the *training sequence*. This training sequence is sampled from a *target* (or, equivalently, *training*) *grammar*, that I will denote by G_{target}. The algorithm then converts the training sequence into a path $G_1, G_2, \ldots$ in the space of grammars, called the *predicted learning sequence*. These basic ingredients are flashed out in (2).

(2)
a. *Target* (or *training*) *grammar*: G_{target}.
b. *Training sequence:* datum 1, datum 2, datum 3, ... sampled from G_{target}.
c. *Predicted learning sequence:* $G_1, G_2, G_3, \ldots$ within the typology.

In general, the predicted learning sequence depends not only on the target grammar but also on the training sequence, namely on the specific way in which the data are sampled from the target grammar and organized into a sequence: the frequency which which the various pieces of data are sampled, the order with which they are presented to the error-driven algorithm, and so on. A certain property of the predicted learning sequence is called *robust* if it only depends on the target grammar but not on the way the data from that target grammar are organized into a training sequence. The goal of this paper is to develop tools for the robust analysis of error-driven learning within OT: are there properties of the sequence of rankings entertained by an EDRA that only depend on the target grammar it is trained on, and not on the the specific training data sequences?

Tesar and Smolensky (1998) focus on EDRAs that only perform constraint demotion, but no constraint promotion. For this class of EDRAs, they show that constraints can never be demoted lower than a certain position, which is completely determined by the training grammar, independently of the training sequence. Using this robust property of the current ranking, they are able to show that all possible learning sequences on a given target grammar must converge to the same final grammar, which can therefore be completely characterized independently of the actual training sequences. Subsection 3.1

reviews in full detail this remarkable result, that provides the first tool for the robust analysis of EDRAs.

Lack of constraint promotion allows Tesar and Smolensky to pinpoint at a remarkably robust property. Yet, Bernhardt and Stemberger (1998), Stemberger and Bernhardt (1999, 2001), and Stemberger et al. (1999) discuss child acquisition data that seem to require some degree of constraint promotion. Boersma (1997, 1998) provides a computationally explicit argument in favor of constraint promotion, arguing that it is needed to model certain cases of language variation within a stochastic variant of the classical OT framework. Finally in Magri (2012b), I provide an argument for constraint promotion within the classical OT framework based on the EDRA model of the child early acquisition of phonotactics, as reviewed below in 4.1.5. In conclusion, various modeling and computational considerations suggest that some constraint promotion is needed. New tools for the robust analysis of EDRAs that perform both constraint demotion and promotion are thus called for.

Subsection 3.2 fills this theoretical gap. My starting point is the intuition that the number of updates triggered by a certain piece of data depends not only on the frequency with which that piece of data is sampled and fed to the EDRA. But also on the frequency with which the current ranking entertained by the EDRA happens not to be consistent with that piece of data. Suppose that a piece of data is "easy" to account for, so that the current ranking is consistent with it most of the time. That piece of data will thus trigger few updates, even if it is fed with high frequency. I show that this intuition can be formalized into a bound on the number of updates that each piece of data can trigger. This bound will only depend on the target grammar, not on the frequencies with which various pieces of data are fed to the algorithm. As the current ranking entertained by the EDRA at a certain time is determined by the number of updates triggered by the various pieces of data up to that time, my distribution-independent bound on the number of updates yields a tool for the robust analysis of the current ranking.

1.3. Implications for the EDRA model of the child acquisition of phonotactics.

An error-driven learner (1) is trained on a single piece of data at the time and does not keep track of previously seen data, so that is does not impose unrealistic memory requirements. Furthermore, a run of the algorithm describes a learning sequence that can be matched with child acquisition paths, thus providing a straightforward tool for modeling the observed child acquisition gradualness. For these reasons, error-driven learning has been endorsed in much of the linguistic acquisition literature, at least since Wexler and Culicover (1980). The robust analysis of error-driven learning develops the formal tools to provide solid ground to this modeling enterprise.

In a classical modeling application, EDRAs are used to model the child early acquisition of phonotactics. In this specific application, the data fed to the algorithm consist of phonotactically licit forms according to a certain target OT grammar G_{target}. The EDRA performs a series of re-rankings until it converges to a final grammar G_{final}. In order for an EDRA to qualify as a computationally sound model of the child acquisition of phonotactics, the final grammar G_{final} must indeed capture the phonotactics induced by the target grammar G_{target} the algorithm has been trained on. This is the core issue of the computational theory of the EDRA model of the child acquisition of phonotactics.

The robust analysis of EDRAs provides the tools to address this core question. In Section 4, I focus on three abstract case studies for phonotactics learning, considered in Prince and Tesar (2004). I point out that EDRAs that perform only constraint demotion but no constraint promotion are unable to learn the target phonotactics in these three case studies. This observation provides further motivation for the new tool for the robust analysis of promotion/demotion EDRAs developed in Subsection 3.2. Furthermore, I use this new tool to obtain robust guarantees of success of the EDRA model on these three case studies of phonotactics learning.

2. Error-driven ranking algorithms (EDRAs)

Subsection 2.1 reviews the OT framework and introduces the basic formulation of EDRAs. Subsections 2.2 and 2.4 offer reformulations of the basic EDRA scheme, in terms of *ranking vectors* and *elementary ranking conditions.* Subsection 2.3 reviews various EDRAs' re-ranking rules from the perspective of convergence.

2.1. **Combinatorial formulation of EDRAs.**

2.1.1. *Typological specifications.* An OT typology is defined through a 4-tuple of *typological specifications* ($\mathcal{X}$, $\mathcal{Y}$, *Gen*, $\mathcal{C}$) as in (3a). The first two ingredients are the set of *underlying forms* $\mathcal{X}$ and the set of *surface forms* $\mathcal{Y}$. The third ingredient is a *generating function Gen* that maps an underlying form into a set of surface forms, called its *candidates.* The fourth ingredient is a *constraint set* $\mathcal{C}$ consisting of n functions $C_1, \dots, C_n$ called *constraints.* Each constraint C_k maps a pair of an underlying form and a corresponding candidate form into a (nonnegative) integer called the *number of violations* assigned by constraint C_k to the mapping of that underlying form to that candidate.

(3) a.
$$\begin{aligned}\mathcal{X} &= \text{set of } \textit{underlying forms};\\ \mathcal{Y} &= \text{set of } \textit{surface forms};\\ \textit{Gen} &= \textit{generating function}\\ \mathcal{C} &= \textit{constraint set}\end{aligned}$$

b.
$$\begin{aligned}\mathcal{X} &= \big\{\text{/ta/, /da/, /rat/, /rad/}\big\}\\ \mathcal{Y} &= \big\{\text{[ta], [da], [rat], [rad]}\big\}\\ \textit{Gen} &= \begin{bmatrix} \text{/ta/, /da/} \rightarrow \big\{\text{[ta], [da]}\big\} \\ \text{/rad/, /rat/} \rightarrow \big\{\text{[rat], [rad]}\big\}\end{bmatrix}\\ \mathcal{C} &= \left\{\begin{array}{l} F_{\text{pos}} = \text{I}\textsc{dent}[\textsc{voice}]/\textsc{onset} \\ F_{\text{gen}} = \text{I}\textsc{dent}[\textsc{voice}] \\ M = {}^*[+\textsc{voice},\ -\textsc{sonorant}] \end{array}\right\}\end{aligned}$$

An example of typological specifications is provided in (3b). The set of underlying forms $\mathcal{X}$ and the set of surface forms $\mathcal{Y}$ coincide, and basically consist of voiced and voiceless stops in onset and coda position. The generating function *Gen* is only allowed to modify voicing. The constraint set $\mathcal{C}$ contains a markedness constraint $M = {}^*[+\textsc{voice}, -\textsc{sonorant}]$ against voiced obstruents together with general and positional faithfulness constraints for voicing $F_{\text{gen}} = \text{I}\textsc{dent}[\textsc{voice}]$ and $F_{\text{pos}} = \text{I}\textsc{dent}[\textsc{voice}]/\textsc{onset}$.

2.1.2. *Data triplets.* The basic data unit in OT is a *data triplet* (4a), consisting of an underlying form $/x/$ and two surface forms $[y]$ and $[z]$, both drawn from the set $\textit{Gen}(/x/)$ of candidates for $/x/$. The first of the two candidates $[y]$ is called the intended *winner*, while the other candidate $[z]$ is called a *loser.* As a mnemonic, I adopt the convention of striking out the loser candidate in a triplet.

(4) a.
$$\Big(\ /x/,\ \overset{\substack{\text{winner}\\ |}}{[y]},\ \underset{\substack{|\\ \text{loser}}}{\cancel{[z]}}\ \Big)$$

b.
$$\Big(\text{/rad/},\ \overset{\substack{\text{winner}\\ |}}{\text{[rat]}},\ \underset{\substack{|\\ \text{loser}}}{\cancel{\text{[rad]}}}\Big)$$

An example of underlying/winner/loser form triplet for the OT typology (3b) is provided in (4b). The underlying form /rad/ is paired up with the two candidate surface forms [rat] and [rad], together with the information that the former is the intended winner while the latter is a loser. This triplet thus provides evidence for a grammar that enforces devoicing, at least in coda position.

2.1.3. *Winner- and loser-preferring constraints.* An underlying/winner/loser form triplet $(/x/, [y], \cancel{[z]})$ sorts the constraints into those that prefer the winner $[y]$ and those that prefer the loser $[z]$ (or those that are even between the two), as stated in (5a).

(5) a. Constraint C_k is *winner-preferring* (respectively, *loser-preferring* and *even*) relative to an underlying/winner/loser form triplet ($/x/, [y],$ ~~$[z]$~~) provided it assigns less (respectively, more and equal) violations to the mapping of the underlying form $/x/$ to the winner $[y]$ than to the loser $[z]$.

b. $$(/\text{rad}/,\ [\text{rat}],\ \cancel{[\text{rad}]}) \Longrightarrow \begin{array}{ll} F_{\text{pos}} = \textsc{ident}[\textsc{voice}]/\textsc{onset}\text{:} & \text{even} \\ F_{\text{gen}} = \textsc{ident}[\textsc{voice}]\text{:} & \text{loser-preferrer} \\ M = {}^*[\textsc{voice}]\text{:} & \text{winner-preferrer} \end{array}$$

To illustrate, in (5b) I classify accordingly the three constraints in (3b) relative to the underlying/winner/loser form triplet (/rad/, [rat], ~~[rad]~~) in (4b). The positional faithfulness constraints F_{pos} for voicing is even, because it is not violated by neither candidates [rat] and [rad]. The general faithfulness constraint F_{gen} is loser-preferrer, as it is violated by the intended winner [rat] but not by the intended loser [rad]. And the markedness constraint M against voicing is instead winner-preferrer, as it is violated by the intended loser [rad] but not by the intended winner [rat].

2.1.4. *Rankings.* Constraints can conflict. For instance, the constraint IDENT[VOICE] prefers the candidate [da] over [ta] for the underlying form /da/, while *[+VOICE] exerts the opposite preference. Grammars differ in how they prioritize or *rank* these constraints. In other words, OT-grammars are parameterized by *rankings*, which are linear orders over the constraint set $\mathcal{C}$, represented as in (6a) or equivalently in (6a′). We say that a constraint C_h is *$\gg$-ranked above* another constraint C_k provided that $C_h \gg C_k$. Given an arbitrary ranking $\gg$, I can assume without loss of generality that it is $C_1 \gg C_2 \gg \ldots \gg C_n$, as the numerical labels assigned to the constraints are arbitrary to start with.

(6) a. $$\begin{tikzcd} \text{top ranked} \arrow[d, dash] & & & \\ C_1 \gg C_2 \gg \ldots \gg C_n & & & \\ & & \text{bottom ranked} \arrow[u, dash] & \end{tikzcd}$$

a′. $$\begin{tikzcd} C_1 \arrow[d, dash] \\ C_2 \arrow[d, dash] \\ \vdots \arrow[d, dash] \\ C_n \end{tikzcd}$$

b. $$\begin{tikzcd} F_{\text{pos}} \arrow[d, dash] \\ M \arrow[d, dash] \\ F_{\text{gen}} \end{tikzcd}$$

An example of ranking over the constraint set in (3b) is provided in (6b). It ranks the positional faithfulness constraint F_{pos} at the top and the general faithfulness constraint F_{gen} at the bottom, with the markedness constraint M ranked in between.

2.1.5. *Consistency.* A ranking $\gg$ is called *OT-consistent* with an underlying/winner/loser form triplet provided condition (7) holds. Crucially, this consistency condition (7) is only sensitive to winner- and loser-preferrers, while even constraints play no role.

(7) A winner-preferring constraint is $\gg$-ranked above every loser-preferring constraint.

To illustrate, the ranking (6b) is consistent with the underlying/winner/loser form triplet (/rad/, [rat], ~~[rad]~~) in (4b) because the markedness constraint M is winner-preferring according to the computation in (5b), and ranked above the loser-preferring constraint F_{gen}, while constraint F_{pos} is even and thus plays no role in establishing consistency.

2.1.6. *Grammars.* A candidate $[y]$ for an underlying form $/x/$ is called a *winner* relative to a ranking $\gg$ provided $\gg$ is consistent with the underlying/winner/loser form triplet ($/x/, [y],$ ~~$[z]$~~) corresponding to any candidate ~~$[z]$~~ for $/x/$ different from $[y]$. Assume that the constraint set is rich enough to distinguish between any two candidates for any underlying form. Then, the winner exists and is unique for any underlying form, relative to any ranking. In this case, a ranking $\gg$ thus defines a corresponding *grammar* $\mathsf{OT}_{\gg}$, namely a function that maps an underlying form $/x/$ in $\mathcal{X}$ into the winner $\mathsf{OT}_{\gg}(/x/)$ for $/x/$ relative to $\gg$.

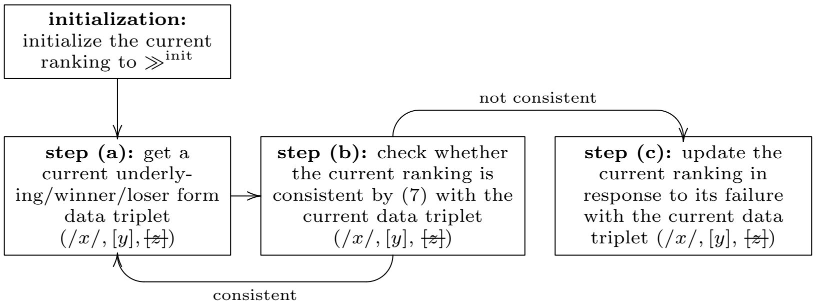
(8) a. $\left[\begin{array}{ccc} /\text{ta}/ & \rightarrow & [\text{ta}] \\ /\text{da}/ & \rightarrow & [\text{da}] \\ /\text{rat}/ & \rightarrow & [\text{rat}] \\ /\text{rad}/ & \rightarrow & [\text{rat}] \end{array}\right]$

b. triplet 1: (/da/, [da], ~~[ta]~~)
triplet 2: (/rad/, [rat], ~~[rad]~~)
triplet 3: (/ta/, [ta], ~~[da]~~)
triplet 4: (/rat/, [rat], ~~[rad]~~)

To illustrate, the grammar corresponding to the universal specifications (3b) and the ranking (6b) is provided in (8a). It neutralizes voicing contrast in final position but not in onset position. The corresponding complete set of underlying/winner/loser form triplets is provided in (8b).

2.1.7. *Basic EDRAs.* The error-driven learning scheme sketched in (1) can be formalized within the framework of OT as the EDRA (9). The algorithm maintains a *current ranking*, which represents its current hypothesis on the target adult grammar it is being trained on. This current ranking is initialized to an *initial ranking* $\gg^{\text{init}}$. And it is updated over time by looping through the three steps (9a)-(9c).

(9)


initialization: initialize the current ranking to $\gg^{\text{init}}$
not consistent
step (a): get a current underlying/winner/loser form data triplet (/x/, [y], ~~[z]~~)
step (b): check whether the current ranking is consistent by (7) with the current data triplet (/x/, [y], ~~[z]~~)
step (c): update the current ranking in response to its failure with the current data triplet (/x/, [y], ~~[z]~~)
consistent

At step (9a), the algorithm is provided with a piece of data consisting of an underlying form /x/, the corresponding winner [y] according to the target grammar the algorithm is being trained on, together with a loser candidate [z]. In certain modeling applications, this step (9a) is broken down further, as discussed in more detail below in Subsection 4.1. At step (9b), the algorithm checks whether the current underlying/winner/loser form data triplet (/x/, [y], ~~[z]~~) is consistent with the current ranking, according to condition (7). If consistency holds, then the algorithm has nothing to learn from this piece of data, it loops back to step (9a) and waits for more data. If instead consistency fails, then the algorithm takes action at step (9c) by modifying its current ranking, and then loops back to step (9a) and starts all over again.

2.1.8. *Robust analysis.* The EDRA (9) is trained on a sequence of data triplets drawn from the grammar corresponding to a certain target ranking $\gg_{\text{target}}$. The algorithm then predicts a sequence of rankings $\gg^1, \gg^2, \gg^3, \ldots$, starting with the assigned initial ranking $\gg^{\text{init}}$. The robust analysis of the EDRA (9) addresses the following question: are there properties of the predicted sequence of rankings $\gg^1, \gg^2, \gg^3, \ldots$ that only depend on the target ranking $\gg_{\text{target}}$, and thus hold for whatever training sequence of triplets drawn from the corresponding grammar?

2.2. **Numerical reformulation of EDRAs.**

2.2.1. *Ranking vectors and their refinements.* Boersma (1997, 1998, 2009) notes that rankings can be represented numerically, exploiting the natural ordering among numbers. A *ranking vector* is an n-tuple $\boldsymbol{\theta}$ of numbers $\theta_1, \ldots, \theta_n$ as in (10a), one for each of the n constraints. The kth component θ_k is called the *ranking value* of constraint C_k.

(10) a. $\begin{array}{c} C_1 \\ \vdots \\ C_k \\ \vdots \\ C_n \end{array}\begin{bmatrix} \theta_1 \\ \vdots \\ \theta_k \\ \vdots \\ \theta_n \end{bmatrix}$ b. $\begin{array}{c} F_{\text{pos}} \\ F_{\text{pos}} \\ M \end{array}\begin{bmatrix} 100 \\ 50 \\ 70 \end{bmatrix}$ c. $\begin{array}{c} F_{\text{pos}} \\ F_{\text{pos}} \\ M \end{array}\begin{bmatrix} 100 \\ 50 \\ 50 \end{bmatrix}$

A ranking $\gg$ is a *refinement* of a ranking vector $\boldsymbol{\theta} = (\theta_1, \dots, \theta_n)$ provided condition (11) holds for any pair of constraints C_h, C_k. If the two ranking values θ_h and θ_k tie, the antecedent of (11) fails, and different refinements can break the tie in either way ($C_h \gg C_k$ or $C_k \gg C_h$). Otherwise, any refinement satisfies the ordering implicitly defined by the relative size of the two ranking values θ_h and θ_k.

(11) If $\theta_h > \theta_k$, then $C_h \gg C_k$.

To illustrate, consider the two ranking vectors (10b) and (10c) for the constraint set in (3b). The former assigns the largest ranking value to the positional faithfulness constraint F_{pos} and the smallest one to the general faithfulness constraint F_{gen}, while the markedness constraint M receives a ranking value in between. This ranking vector thus admits a unique refinement, namely the ranking (6b) that ranks F_{pos} at the top and F_{gen} at the bottom, with M in between. The ranking vector (10c) assigns the same ranking value to the general faithfulness constraint F_{gen} and to the markedness constraint M and therefore admits two refinements $F_{\text{pos}} \gg M \gg F_{\text{gen}}$ and $F_{\text{pos}} \gg F_{\text{gen}} \gg M$, that split the tie between F_{gen} and M in the two possible ways.

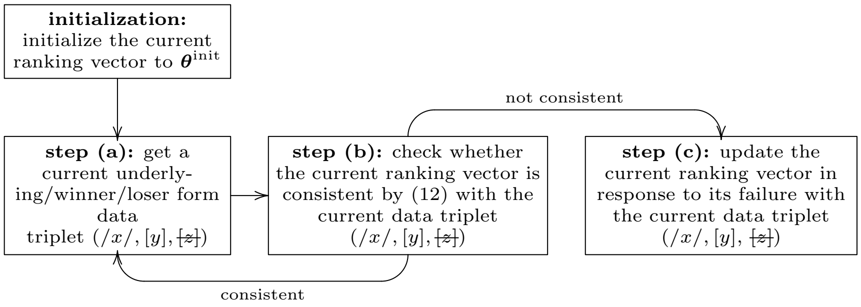
2.2.2. *Consistency for ranking vectors.* Once ranking vectors are paired up with rankings through (11), notions that pertain to rankings can be extended to ranking vectors. In particular, the notion of consistency stated in (7) for rankings can be extended to ranking vectors. A ranking vector is called *consistent* with an underlying/winner/loser data triplet provided that *each* of its refinements is consistent with that triplet (Boersma 2009). This consistency condition for ranking vectors can be equivalently restated as in (12).

(12) There is a winner-preferring constraint whose ranking value is strictly larger than the ranking value of any loser-preferring constraint.

For instance, the ranking vector in (10b) is consistent with the underlying/winner/loser form triplet (/rad/, [rat], ~~[rad]~~) in (4b), as its only refinement is the ranking (6b) that is indeed consistent with that data triplet according to the original condition (7). The ranking vector in (10c) is instead inconsistent with that data triplet, as it admits the refinement $F_{\text{pos}} \gg F_{\text{gen}} \gg M$, that is inconsistent with that data triplet according to (7).

2.2.3. *Numerical EDRAs.* Boersma (1997, 1998, 2009) suggests to restate the EDRA (9) in terms of ranking vectors as in (13). The current hypothesis on the target grammar is represented by the algorithm as a numerical ranking vector, rather than as a combinatoric ranking. The *current* ranking vector is initialized to an *initial ranking vector* $\boldsymbol{\theta}^{\text{init}}$ and slightly updated whenever it is found to be inconsistent with the current data triplet.

(13)


initialization: initialize the current ranking vector to $\boldsymbol{\theta}^{\text{init}}$
not consistent
step (a): get a current underlying/winner/loser form data triplet (/x/, [y], ~~[z]~~)
step (b): check whether the current ranking vector is consistent by (12) with the current data triplet (/x/, [y], ~~[z]~~)
step (c): update the current ranking vector in response to its failure with the current data triplet (/x/, [y], ~~[z]~~)
consistent

This restatement of EDRAs in terms of ranking vectors rather than rankings has proven crucial in the development of the theory of EDRAs, from various perspectives. Boersma motivates the switch from rankings to ranking vectors from the perspective of modeling adult and child variation. In Magri (in press), I point out another advantage of a numerical re-parameterization of OT grammars in terms of ranking vectors, namely that it allows methods and results from the theory of *online linear classification* to be ported into OT. Finally, in Magri (2012b) I show that the reformulation of EDRAs in (13) in terms of ranking vectors provides the proper framework for the development of the theory of EDRA's convergence, summarized below in Subsection 2.3.

2.2.4. *Robust analysis.* The EDRA (13) is trained on a sequence of data triplets drawn from the grammar corresponding to a certain target ranking $\gg_{\text{target}}$. The algorithm then predicts a sequence of rankings $\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \boldsymbol{\theta}^3, \ldots,$ starting with the assigned initial ranking vector $\boldsymbol{\theta}^{\text{init}}$. The robust analysis of the EDRA (13) addresses the following question: are there properties of the predicted sequence of rankings vectors $\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \boldsymbol{\theta}^3, \ldots$ that only depend on the target ranking $\gg_{\text{target}}$, and thus hold for whatever training sequence of triplets drawn from the corresponding grammar?

2.3. **Re-ranking rules.** The failure of the current ranking vector on the current data triplet detected at step (13b) provides evidence that loser-preferring constraints are currently ranked too high while winner-preferring constraints are ranked too low. The re-ranking rule used at step (13c) tries to remedy to these shortcomings. On the one hand, it demotes the loser-preferrers by a certain amount, say 1 for concreteness, as in (14a). Following Tesar and Smolensky (1998), I assume here that not all loser-preferrers are demoted, but only those that really need to. Namely those that are currently *undominated*, in the sense that their ranking value is not already smaller than the ranking value of some winner-preferrer. On the other hand, the re-ranking rule promotes the winner-preferring constraints, by a certain *promotion amount* $p \geq 0$ that can be null or positive, as in (14b).

(14) a. Demote each undominated loser-preferring constraint by 1;
b. promote each winner-preferring constraint by p.

The re-ranking rules considered in the literature differ (mainly) for the choice of the promotion amount p. Let me quickly review the main options explored in the literature.

2.3.1. *(Gradual) EDCD.* Tesar and Smolensky (1998) study EDRAs that perform constraint demotion but no constraint promotion. In other words, they set the promotion amount p equal to zero, as in the re-ranking rule (15). The EDRA (13) with this re-ranking rule (15) is called *(gradual) Error-Driven Constraint Demotion* (henceforth: EDCD).

(15) a. Demote each undominated loser-preferring constraint by 1;
b. do not promote the winner-preferring constraints.

Tesar and Smolensky show that EDCD *converges*, namely can make only a finite number of mistakes, no matter the training grammar the input triplets are sampled from.[1] Furthermore, they show that the worst-case number of mistakes grows slowly (i.e. only quadratically) with the number of constraints, so that the algorithm remains efficient also for very large constraint sets.

[1] Tesar and Smolensky actually consider the non-gradual counterpart of the re-ranking rule (15). Boersma (1998, p. 323-327) notes that their analysis straightforwardly extends to the gradual counterpart (15). Furthermore, Boersma (2009) fixes a small glitch in their analysis. For a detailed discussion of the relationship between Tesar and Smolensky's original algorithm and the variant considered here, see Magri (2012b, Subsection 3.7).

2.3.2. *GLA.* Boersma (1997) sets the promotion amount p equal to 1, and thus considers the re-ranking rule (16), that performs promotion and demotion by the same amount. The EDRA (13) with this re-ranking rule (16) is called the (non-stochastic) *Gradual Learning Algorithm* (henceforth: GLA).[2]

(16) a. Demote each undominated loser-preferring constraint by 1;
b. promote each winner-preferring constraint by 1.

This algorithm has been reported to have good modeling properties (Boersma and Levelt 2000, Curtin and Zuraw 2002, Boersma and Hayes 2001). Yet, the GLA's convergence has remained an open issue for almost ten years (Keller and Asudeh 2002). Until Pater (2008) has shown that the GLA can make an infinite number of mistakes, even on a small set of consistent data triplets; see Magri (2012b) for an explanation of Pater's counterexample. The promotion component (16b) of the GLA's re-ranking rule thus disrupts the good convergent behavior of the demotion component (16a).

2.3.3. *CEDRAs.* Tesar and Smolensky have shown that constraint demotion converges fast. Intuitively, in order to retain convergence, the promotion component of the re-ranking rule should not overwhelm the demotion component. But that is precisely what happens with the GLA's update rule (16): if only one loser-preferrer is demoted but two winner-preferrers are promoted, the algorithm overall promotes more (namely by 2) than it demotes (namely by 1). In order for the promotion component of the update rule not to overwhelm the demotion component, the promotion amount p needs to be *calibrated* on the number of winner-preferrers: if there are w winner-preferrers, the promotion amount must be strictly smaller than $1/w$ (indeed a total of w promotions by less than $1/w$ gives an overall promotion smaller than 1, namely smaller than the smallest overall demotion). The *calibrated EDRA* (henceforth: CEDRA) thus sets the promotion amount equal to c/w for some $c < 1$, as in (17b).

(17) a. Demote each undominated loser-preferring constraint by 1;
b. promote each of the w winner-preferring constraints by c/w, for some $c < 1$.

In Magri (2012b,a), I show that indeed a CEDRA converges, namely it can only perform a finite number of mistakes. And that the worst-case number of mistakes again grows only quadratically with the number of constraints, just as in the case of EDCD (15).

2.4. **Reformulation of EDRAs in terms of ERCs.**

2.4.1. *Elementary ranking conditions (ERCs).* An underlying/winner/loser form triplet $(/x/, [y], \cancel{[z]})$ sorts the constraints into winner- and loser-preferring (as well as even), according to the classification in (5a). Following Prince (2002), this information is collected into the corresponding *elementary ranking condition* (henceforth: ERC), namely the n-tuple (where n is the number of constraints) whose kth entry is equal to $\textsc{w}$, $\textsc{l}$ or e depending on whether the kth constraint C_k is winner-preferring or loser-preferring or even, as in (18a). I denote an ERC by $\mathbf{a}$ and its entries by $a_1, \ldots, a_n$; I denote a collection of ERCs by $\mathbf{A}$, and I represent it by stacking its ERCs one on top of the other into a matrix. I often omit e's for readability.

(18) a. $$(/x/,\ [y],\ \cancel{[z]}) \implies \mathbf{a} = \begin{bmatrix} a_1 \ldots a_n \end{bmatrix} \text{ where } a_k = \begin{cases} \textsc{w} & \text{if } C_k \text{ is winner-preferrer} \\ \textsc{l} & \text{if } C_k \text{ is loser-preferrer} \\ e & \text{if } C_k \text{ is even} \end{cases}$$

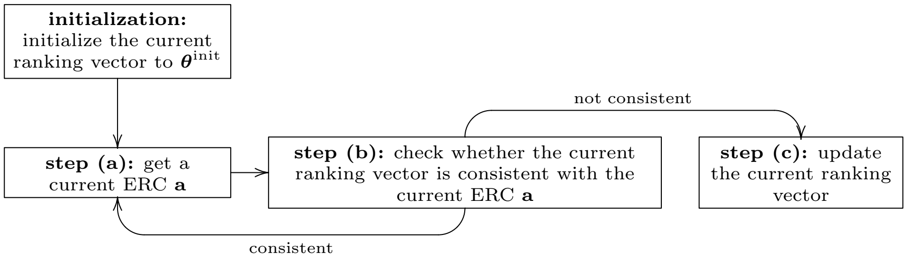
[2] Boersma's actual formulation of the GLA demotes all loser-preferrers, not just the undominated ones.

$$\text{b.}\quad \left\{ \begin{array}{c} (/\text{da}/,\ [\text{da}],\ \cancel{[\text{ta}]}) \\ (/\text{rad}/,\ [\text{rat}],\ \cancel{[\text{rad}]}) \\ (/\text{ta}/,\ [\text{ta}],\ \cancel{[\text{da}]}) \\ (/\text{rat}/,\ [\text{rat}],\ \cancel{[\text{rad}]}) \end{array} \right\} \Longrightarrow \begin{array}{cc} & \begin{array}{ccc} F_{\text{pos}} & M & F_{\text{gen}} \end{array} \\ \begin{array}{c} \text{ERC 1} \\ \text{ERC 1} \\ \text{ERC 3} \\ \text{ERC 4} \end{array} & \left[\begin{array}{ccc} \text{W} & \text{L} & \text{W} \\ e & \text{W} & \text{L} \\ \text{W} & \text{W} & \text{W} \\ e & \text{W} & \text{W} \end{array} \right] \end{array}$$

To illustrate, I provide in (18b) the ERC matrix corresponding to the underlying/winner/loser form triplets listed in (8b), relative to the constraint set in (3b).

2.4.2. *Restatement of EDRAs in terms of ERCs.* At each iteration, the EDRA (13) is fed with a current underlying/winner/loser form triplet $(/x/, [y], \cancel{[z]})$ at step (13a). The notion of consistency (12) used at step (13b) is really only sensitive to the ERC corresponding to the current triplet, namely to the way it sorts the constraints into loser- and winner-preferring. Other properties of the current triplet (such as the actual numbers of constraint violations for the winner or the loser) play no role. Also the re-ranking rules (15), (16), and (17) used by EDRAs at step (13c) only care about the ERC corresponding to the current triplet and not about the current triplet itself. In conclusion, EDRAs can be restated as in (19), whereby the training sequence consists of ERCs, rather than of triplets.

(19)


initialization: initialize the current ranking vector to $\boldsymbol{\theta}^{\text{init}}$
not consistent
step (a): get a current ERC **a**
step (b): check whether the current ranking vector is consistent with the current ERC **a**
step (c): update the current ranking vector
consistent

The abstract formulation of EDRAs in (19) will prove well suited to the computational developments of Session 3.

2.4.3. *Robust analysis.* The EDRA (19) is provided with a *training ERC sequence* drawn from a given *training ERC set* **A**. The latter is usually a subset of the set of ERCs corresponding to (the set of triplets corresponding to) a target grammar. The algorithm then predicts a sequence of ranking vectors $\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \boldsymbol{\theta}^3, \ldots,$ starting with the assigned initial ranking vector $\boldsymbol{\theta}^{\text{init}}$. The robust analysis of the EDRA (19) addresses the following question: are there properties of the predicted sequence of rankings vectors $\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \boldsymbol{\theta}^3, \ldots$ that only depend on the training ERC set **A**, and thus hold no matter how ERCs are sampled from it and arranged into a training ERC sequence?

2.4.4. *An example.* Assume that the EDRA is trained on the ERC set in (18b); that it starts from null initial ranking values; and that it adopts the GLA update rule (16), that demotes (respectively, promotes) each undominated loser-preferrer (respectively, each winner-preferrer) by 1. The sequence of ranking vectors entertained by the algorithm depends on the training sequence of ERCs, as illustrated in (20).

(20)

$$\begin{tikzcd}
\boldsymbol{\theta}^{\text{init}} & \boldsymbol{\theta}^1 & \boldsymbol{\theta}^2 & \boldsymbol{\theta}^3 & \boldsymbol{\theta}^4 & \boldsymbol{\theta}^5 \\
& \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \arrow[dr, "\text{ERC 2}"] & & & & \\
\begin{array}{r} F_{\text{pos}} \\ M \\ F_{\text{gen}} \end{array}\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \arrow[ur, "\text{ERC 1}"] \arrow[dr, "\text{ERC 2}"'] & & \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \arrow[r, "\text{ERC 2}"] & \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} \arrow[r, "\text{ERC 1}"] & \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} \arrow[r, "\text{ERC 2}"] & \begin{bmatrix} 2 \\ 1 \\ -1 \end{bmatrix} \\
& \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \arrow[ur, "\text{ERC 1}"'] & & & &
\end{tikzcd}$$

At the first iteration, the algorithm can receive any of the four input ERCs. Yet, the ERCs 3 and 4 corresponding to the triplets (/ta/, [ta], ~~[da]~~) and (/rat/, [rat], ~~[rad]~~) are consistent with any ranking (vector). Hence, they never trigger an update and can therefore be ignored. Suppose for concreteness that at the first iteration, the EDRA receives ERC 1. Since the current ranking vector $\boldsymbol{\theta}^{\text{init}}$ is not OT-consistent with that ERC, then the algorithm updates $\boldsymbol{\theta}^{\text{init}}$ to $\boldsymbol{\theta}^1$ by decreasing by 1 the ranking value of the currently undominated loser-preferrer M and by increasing by 1 the current ranking values of the two winner-preferrers F_{pos} and F_{gen}. At the next iteration, the algorithm can again receive either ERC 1 or ERC 2. Since the current ranking vector $\boldsymbol{\theta}^1$ is OT-consistent with ERC 1, this ERC triggers no update and nothing happens until the algorithm is fed ERC 2. When that happens, the algorithm detects that the current ranking vector $\boldsymbol{\theta}^1$ is not OT-consistent with ERC 2 and it consequently updates it to $\boldsymbol{\theta}^2$ by decreasing by 1 the ranking value of the currently undominated loser-preferrer F_{gen} and by increasing by 1 the current ranking value of the winner-preferrer M. And so on.

3. Tools for the robust analysis of EDRAs

This Section presents two tools for the robust analysis of EDRAs. In Subsection 3.1, I review Tesar and Smolensky's (1998) robust analysis for the demotion-only update rule (15). In Subsection 3.2, I then develop a new tool for the robust analysis of EDRAs, that also holds for update rules that perform constraint promotion, such as (16) or (17). Section 4 will then illustrate the latter new tool on some case studies singled out in Prince and Tesar (2004), that crucially require update rules that perform constraint promotion.

3.1. Tesar and Smolensky's analysis.

3.1. Tesar and Smolensky's analysis. Tesar and Smolensky (1998; henceforth: TS) focus on the demotion-only re-ranking rule (15). Quite surprising, they show that the final ranking vector can be completely characterized in terms of the training ERC set, independently of the specific way the ERCs are sampled from it and arranged into a training ERC sequence. They thus provide the first result of the robust analysis of EDRAs. Furthermore, some of the intermediate results in their analysis extend beyond the demotion-only case. In order to set the stage for the developments in Subsection 3.2, this Subsection thus offers a detailed introduction to TS's analysis. My reconstruction of their analysis underscores the benefits of framing the theory of EDRAs in terms of ranking vectors rather than in terms of rankings, as they originally did.[3] For concreteness, I will assume throughout that the initial ranking values are all null; the extension to arbitrary initial ranking values is straightforward.

3.1.1. *The consistency lattice.* Let $\mathbf{A}$ be the set of ERCs the EDRA is trained on. To start, we need to pair up $\mathbf{A}$ with some formal object that captures those properties of $\mathbf{A}$ that will be crucial in the robust analysis of the algorithm on $\mathbf{A}$. A natural choice would be the collection of all ranking vectors consistent with $\mathbf{A}$. In the specific case of the demotion-only update rule (15) that TS focus on, the current ranking values can only take on the values $0, -1, -2, \ldots$, as they start at zero and are decreased by 1 every time they are updated. Thus, let the *consistency lattice* corresponding to the the training ERC set $\mathbf{A}$ be the collection $\mathfrak{L}_{\mathbf{A}}$ of all those ranking vectors $\boldsymbol{\theta}$ that are consistent with each ERC in $\mathbf{A}$ and furthermore have non-positive integer components, as in (21a). If the ERCs in $\mathbf{A}$ are consistent (with at least a ranking), their consistency lattice $\mathfrak{L}_{\mathbf{A}}$ is not empty.

(21) a. $$\mathfrak{L}_{\mathbf{A}} = \left\{ \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \;\middle|\; \begin{array}{l} \theta_k \in \{0, -1, -2, \ldots\} \\ \boldsymbol{\theta} \text{ is consistent with } \mathbf{A} \end{array} \right\}$$

[3] Although 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.

$$\text{b. } \mathfrak{L}_{\mathbf{A}} = \left\{ \begin{bmatrix} \theta_{F_{\text{pos}}} \\ \theta_M \\ \theta_{F_{\text{gen}}} \end{bmatrix} \;\middle|\; \begin{array}{l} \theta_{F_{\text{pos}}}, \theta_M, \theta_{F_{\text{gen}}} \in \{0, -1, -2, \ldots\} \\ \theta_{F_{\text{pos}}} > \theta_M > \theta_{F_{\text{gen}}} \end{array} \right\}$$

To illustrate, the consistency lattice $\mathfrak{L}_{\mathbf{A}}$ of the ERC set $\mathbf{A}$ in (18b) is provided in (21b). It consists of all triplets of non-positive integer ranking values $\theta_{F_{\text{pos}}}, \theta_M, \theta_{F_{\text{gen}}}$ such that the ranking value $\theta_{F_{\text{pos}}}$ of the positional faithfulness constraint F_{pos} is larger than the ranking value θ_M of the markedness constraint M and the latter is in turn larger than the ranking value $\theta_{F_{\text{gen}}}$ of the general faithfulness constraint F_{gen}, so that the only refinement of $\boldsymbol{\theta}$ is the ranking $F_{\text{pos}} \gg M \gg F_{\text{gen}}$ consistent with the ERCs in $\mathbf{A}$.

3.1.2. *Consistency is preserved under maxima.* Denote by $\boldsymbol{\theta} = \max\{\boldsymbol{\theta}', \boldsymbol{\theta}''\}$ the *component-wise maximum* of the two ranking vectors $\boldsymbol{\theta}' = (\theta_1', \ldots, \theta_n')$ and $\boldsymbol{\theta}'' = (\theta_1'', \ldots, \theta_n'')$, namely the vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ whose component θ_k is the largest between the two corresponding components θ_k' and θ_k'', as in (22a). A concrete example is provided in (22b).

$$\text{(22)} \quad \text{a. } \theta_k = \max\{\theta_k', \theta_k''\} \qquad \text{b. } \max\left\{ \begin{bmatrix} 5 \\ -4 \\ 2 \end{bmatrix}, \begin{bmatrix} -2 \\ 7 \\ 2 \end{bmatrix} \right\} = \begin{bmatrix} 5 \\ 7 \\ 2 \end{bmatrix}$$

This operation over ranking vectors is natural because it preserves consistency, as stated in the following Lemma. The straightforward proof is provided in Appendix A.

LEMMA 1. *If two ranking vectors $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$ are consistent with an ERC $\mathbf{a}$, then their component-wise maximum $\boldsymbol{\theta} = \max\{\boldsymbol{\theta}', \boldsymbol{\theta}''\}$ is consistent with $\mathbf{a}$ as well.* ■

3.1.3. *The maximal consistent vector.* Given a consistent training ERC set $\mathbf{A}$, let the *maximal consistent ranking vector* $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ be the component-wise maximum of the ranking vectors in the corresponding consistency lattice $\mathfrak{L}_{\mathbf{A}}$, as stated in (23a). This vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ is well defined (as the maxima are computed over ranking values that are at most zero) and unique. Furthermore, Lemma 1 ensures that this vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ is consistent with each ERC in $\mathbf{A}$,[4] as it is the component-wise maximum of ranking vectors consistent with $\mathbf{A}$.[5]

$$\text{(23)} \quad \text{a. } \overline{\boldsymbol{\theta}}_{\mathbf{A}} = \max \mathfrak{L}_{\mathbf{A}} \qquad \text{b. } \overline{\boldsymbol{\theta}}_{\mathbf{A}} = \begin{bmatrix} \overline{\theta}_{F_{\text{pos}}} \\ \overline{\theta}_M \\ \overline{\theta}_{F_{\text{gen}}} \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ -2 \end{bmatrix}$$

To illustrate, the maximal consistent ranking vector for the training ERC set $\mathbf{A}$ in (18b) is computed in (23b). Of course the vector $[0, -1, -2]$ has integer non-positive components and is consistent with $\mathbf{A}$, so that it belongs to the consistency lattice $\mathfrak{L}_{\mathbf{A}}$. Thus, I only need to show that there is no ranking vector in the consistency lattice $\mathfrak{L}_{\mathbf{A}}$ that has larger components. To start, suppose by contradiction that there existed a ranking vector in $\mathfrak{L}_{\mathbf{A}}$ with a ranking value corresponding to F_{pos} larger than 0. That is absurd, because

[4] The maximal consistent vector is a restatement in terms of ranking vectors of TS's notion of the *h-dominant target stratified hierarchy*, introduced in their definition (52). The claim that it is well defined and unique corresponds to TS's Lemma (53). The claim that it is indeed consistent with the training ERC set corresponds to TS's Theorem (55).

[5] Strictly speaking, Lemma 1 does not apply in this case, as the consistency lattice $\mathfrak{L}_{\mathbf{A}}$ is not finite. But this is only a small glitch that can be immediately overcome as follows. Consider the subset (*) of those ranking vectors in the consistency lattice whose ranking values all larger than or equal to Δ. Choose $\Delta < 0$ is such a way that this set (*) is non-empty. The identity (i) then trivially holds: the component-wise maximum over the entire consistency lattice $\mathfrak{L}_{\mathbf{A}}$ is identical to the component-wise maximum over its subset (*).

$$\text{(i)} \quad \max \mathfrak{L}_{\mathbf{A}} = \max \underbrace{\left\{ \boldsymbol{\theta} = (\theta_1, \ldots, \theta_n) \in \mathfrak{L}_{\mathbf{A}} \;\middle|\; \theta_1, \ldots, \theta_n \geq \Delta \right\}}_{(*)}$$

Since the set (*) is finite, Lemma 1 does apply, ensuring that its component-wise maximum is consistent with the training ERC set $\mathbf{A}$.

all ranking vectors in $\mathfrak{L}_{\mathbf{A}}$ have non-positive components. Next, suppose by contradiction that there existed a ranking vector in $\mathfrak{L}_{\mathbf{A}}$ with a ranking value corresponding to M larger than -1, namely equal to 0. That is absurd, because the ranking value corresponding to M needs to be smaller than the ranking value corresponding to F_{pos}, which is in turn 0. Finally, suppose by contradiction that there existed a ranking vector in $\mathfrak{L}_{\mathbf{A}}$ with a ranking value corresponding to F_{gen} larger than -2, namely equal to -1 or 0. That is again absurd, because the ranking value corresponding to F_{gen} needs to be smaller than the ranking value corresponding to M, which is in turn -1.

3.1.4. *Characterization of intermediate ranking vectors.* TS show how to use the maximal consistent ranking vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ to characterize the behavior of an EDRA on the training ERC set $\mathbf{A}$. To illustrate the idea, consider again the diagram (20) of all possible runs over the training ERC set $\mathbf{A}$ in (18b), whose maximal consistency vector has been computed in (23b). The EDRA walks through different sequences of intermediate ranking vectors, depending on the training sequence of ERCs (e.g., depending on whether the first ERC fed to the algorithm is ERC 1 or 2). Yet, these sequences of intermediate ranking vectors share a common property: the ranking value of F_{pos} never gets smaller than 0, the ranking value of M never gets smaller than -1, and the ranking value of F_{gen} never gets smaller than -2, as stated in (24). Crucially, the ranking vector at the right hand side of (24) is the maximal consistent ranking vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ computed in (23b). In other words, the current ranking vector entertained by the EDRA in any run on the training set $\mathbf{A}$ is at least as large (component-wise) as the corresponding maximal consistent vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$.

(24)
$$\overset{\substack{\text{current ranking vector in (20)}\\ |}}{\begin{bmatrix}\theta_{F_{\text{pos}}}\\ \theta_M\\ \theta_{F_{\text{gen}}}\end{bmatrix}} \quad\geq\quad \underset{\substack{|\\ \text{maximal consistent vector } \overline{\boldsymbol{\theta}}_{\mathbf{A}}}}{\begin{bmatrix}0\\ -1\\ -2\end{bmatrix}}$$

Let's look into (24) in some detail. The current ranking value corresponding to the positional faithfulness constraint F_{pos} in (20) is always at least 0. In fact, F_{pos} starts at 0 and is never demoted, as it has no L's in the training ERC set. The ranking value corresponding to the markedness constraint M in (20) is always at least -1. In fact, M can only be demoted through update by ERC 1. In order for that ERC to trigger an update, M must be currently ranked above the winner-preferrer F_{pos}. Since the current ranking value of F_{pos} never goes below 0 as we just saw, M can only be demoted when its current ranking value is larger than or equal to 0. Therefore, the current ranking value of M cannot ever make it below -1. Finally, the ranking value corresponding to the general faithfulness constraint F_{gen} in (20) is always at least -2. In fact, F_{gen} can only be demoted through update by ERC 2. In order for that ERC to trigger an update, F_{gen} must be currently ranked above the winner-preferrer M. Since the current ranking value of M can only drop down to -1 as we just saw, F_{gen} can only be demoted when its current ranking value is larger than or equal to -1. Therefore, the current ranking value of F_{gen} cannot ever make it below -2. It turns out that this reasoning applies with full generality, so that inequality (24) holds in full generality, as stated in the following Theorem.[6]

THEOREM 1. *Consider a run of an EDRA (19) with a generic promotion/demotion re-ranking rule on a consistent training ERC set* $\mathbf{A}$ *(starting from null initial ranking values). Assume that the re-ranking rule only demotes the undominated loser-preferrers, as in (14). The current ranking vector* $\boldsymbol{\theta}$ *entertained at a generic time in the run considered satisfies*

[6] Theorem 1 is a restatement in terms of ranking vectors of TS's original Lemma (60).

the inequality $\boldsymbol{\theta} \geq \overline{\boldsymbol{\theta}}_{\mathbf{A}}$, namely it is always at least as large (component-wise) than the maximal consistent ranking vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ associated with the training ERC set **A**. ■

TS prove Theorem 1 for the case of the demotion-only re-ranking (15). But their proof actually extends from the demotion-only case to any promotion/demotion re-ranking rule that only demotes those constraints that really need to be demoted, namely only the undominated ones. TS's proof of Theorem 1 thus generalized is reviewed in Appendix B.

3.1.5. *Characterization of the final ranking vector.* So far, I have considered an arbitrary re-ranking rule (14). TS note that Theorem 1 has a particularly important corollary in the case of their demotion-only re-ranking rule (15), that demotes by 1 each undominated loser-preferrer but does not re-rank the winner-preferrers. As recalled in 2.3.1, this re-ranking rule is provably convergent. Thus, the EDRA will always stop on a final ranking vector $\boldsymbol{\theta}^{\text{final}}$. In the case of this special re-ranking rule, the final ranking vector $\boldsymbol{\theta}^{\text{final}}$ happens to belong to the consistency lattice $\mathfrak{L}_{\mathbf{A}}$ associated with the training ERC set **A** as in (21a). In fact, this final ranking vector $\boldsymbol{\theta}^{\text{final}}$ has non-positive integer components, as the ranking values start at zero and are decreased by 1 with each update. And it is furthermore consistent with the training ERC set **A**, by the very definition of convergence. Furthermore, the final ranking vector $\boldsymbol{\theta}^{\text{final}}$ satisfies the inequality $\boldsymbol{\theta}^{\text{final}} \geq \overline{\boldsymbol{\theta}}_{\mathbf{A}}$, namely it is at least as large (component-wise) as the maximal consistent vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$. In fact, Theorem 1 ensures that this inequality holds for every current ranking vector, and thus in particular it holds for the final ranking vector $\boldsymbol{\theta}^{\text{final}}$. In conclusion, the final ranking vector $\boldsymbol{\theta}^{\text{final}}$ belongs to the consistency lattice $\mathfrak{L}_{\mathbf{A}}$ and is at least as large as its largest ranking vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$. The final ranking vector $\boldsymbol{\theta}^{\text{final}}$ must therefore coincide with $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$, as stated in Corollary 1.[7]

COROLLARY 1. *The final ranking vector $\boldsymbol{\theta}^{T}$ entertained by the EDRA (13) with the demotion-only re-ranking rule (15) trained on a set* **A** *of consistent input ERCs (starting from the null initial ranking vector) always coincides with the maximal consistent ranking vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ associated to the training set* **A**. ■

Of course, the sequence of ranking vectors entertained by the demotion-only EDRA in a given run depends on the training ERC sequence, and in particular on the frequencies and the order of presentation of the various ERCs. Despite this variability, all sequences of intermediate rankings vectors converge on exactly the same final ranking vector. The final ranking vector is thus unique and completely determined by the relevant properties of the training ERC set **A**, summarized into its maximal consistent vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$. Unfortunately, the latter result does not extend to the case of update rules that perform constraint promotion as well as demotion. For instance, the final ranking vector in the run (20) does not coincide with the maximal consistent ranking vector computed in (23b). Thus, new tools are needed for the analysis of EDRAs that perform both constraint demotion and promotion. The next Subsection fills this gap.

3.2. **A new tool for the analysis of EDRAs.** As long as the constraint set is finite, there is only a finite number of ERCs. The training ERC set **A** is thus always finite. For convenience, I will assume that the ERCs in the set **A** have been numbered consecutively in some arbitrary way, just as the ERCs in the set (18b) have been numbered from 1 to 4. I can thus refer to a generic ERC in the training ERC set **A** as to the ith input ERC, where i ranges over the number of ERCs in the training set. Suppose that at the current iteration of the EDRA, the algorithm is fed with the ith input ERC from the training set **A**. If an inconsistency is detected between the current ranking vector and this current ERC, then the algorithm updates the former according to the generic update rule (14), repeated in (25). Here, I have subscripted the promotion amount p_i with the index i that identifies the current ERC triggering the update, to encode the fact that the promotion

[7] Corollary 1 is a restatement in terms of ranking vectors of TS's original Theorem (65).

amount might depend on the current ERC. For instance, that is the case of the update rule (17), whereby the promotion amount coincides with the number of winner-preferrers in the current ERC.

(25) a. Demote each undominated loser-preferrer by 1;
b. promote each winner-preferrer by the promotion amount $p_i \geq 0$.

This subsection develops a tool for the robust analysis of EDRAs that adopt the promotion/demotion re-ranking rule (25). To keep the discussion cleaner, I assume throughout this Subsection that all the input ERCs that the EDRA is trained on have a unique loser-preferrer. This assumption can be straightforwardly relaxed at the only expenses of a slightly more cumbersome notation, as shown in Appendix C.3.

3.2.1. *The robust analysis requires a robust bound on the number of updates triggered by each input ERC.* If the ith input ERC has an L corresponding to a certain constraint C_k, then it demotes C_k by 1 with each update.[8] If this ith ERC has triggered a total of α_i updates up to the time considered in the run considered, then it has overall demoted C_k by α_i. And the collection $\mathrm{L}(k)$ of those input ERCs that have an L corresponding to constraint C_k have overall demoted C_k by $\sum_{i\in\mathrm{L}(k)} \alpha_i$. Suppose next that the ith input ERC has instead a W corresponding to this constraint C_k, so that it promotes C_k by the corresponding promotion amount p_i with each update. If this ith ERC has triggered a total of α_i updates up to the time considered in the run considered, then it has overall promoted C_k by $p_i\alpha_i$. And the collection $\mathrm{W}(k)$ of those input ERCs that have a W corresponding to constraint C_k have overall promoted C_k by $\sum_{i\in\mathrm{W}(k)} p_i\alpha_i$. In the end, the current ranking value θ_k of constraint C_k at the time considered in the run considered can be expressed as in (26), namely as the corresponding initial ranking value θ_k^{init} increased by the total promotions and decreased by the total demotions.

$$(26) \qquad \underbrace{\theta_k}_{\substack{\text{current ranking value}\\ \text{of constraint } C_k}} = \underbrace{\theta_k^{\mathrm{init}}}_{\substack{\text{initial ranking value}\\ \text{of constraint } C_k}} + \underbrace{\sum_{i\in\mathrm{W}(k)} p_i\alpha_i}_{\substack{\text{total promotions}\\ \text{of constraint } C_k}} - \underbrace{\sum_{i\in\mathrm{L}(k)} \alpha_i}_{\substack{\text{total demotions}\\ \text{of constraint } C_k}}$$

By (26), the current ranking value θ_k entertained by the EDRA at a certain time in a certain run is completely determined by the following three factors: the choice of the *initial ranking value* θ_k^{init}; the choice of the *update rule*, and in particular of the promotion amounts p_i; and the *number of updates* α_i triggered by each ith input ERC up to the time considered. In conclusion, the robust analysis of EDRAs requires robust bounds on the number of updates triggered by each input ERC.

3.2.2. *Towards a robust bound on the number of updates triggered by each input ERC.* Intuitively, one might expect the number of updates triggered by an input ERC in a given run to depend in particular on the frequency with which that ERC is sampled from the training ERC set and fed to the algorithm. Thus, one might pessimistically expect not to be able to develop robust bounds on the number of updates triggered by the various input ERCs, namely bounds that only depend on the training ERC set, and not on the actual training ERC sequences. Fortunately, this pessimistic conclusion turns out not to be correct. In fact, the number of updates triggered by an input ERC depends not only on the number of times it is fed to the model, but also on the proportion of those times that the current ranking vector turns out to be inconsistent with that ERC at step (19b). Taking advantage of this fact, we can indeed get distribution-independent bounds

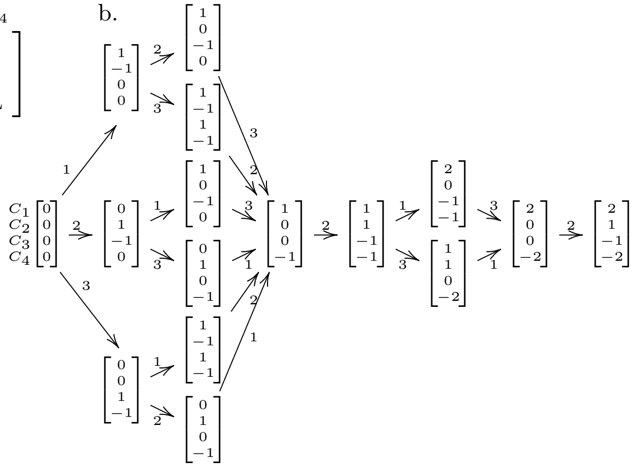
[8] This would not be true if this ERC contained multiple loser-preferrers. In that case, the ith input ERC could trigger an update despite the fact that its loser-preferrer C_k is currently dominated, and thus not demoted, as (25a) only demotes those loser-preferrers that are undominated. The assumption that input ERCs have a unique loser-preferring constraint allows me to collapse the number of times an ERC triggers an update with the number of times its loser-preferrer is demoted.

on the number of updates triggered by input ERCs, leading to distribution-independent characterizations of the current ranking vector through (26). Let me illustrate the idea with a simple example. Suppose that the training ERC set is (27a), consisting of only three ERCs. For concreteness, suppose the algorithm adopts the GLA update rule (16). I provide in (27b) *all* possible sequences of ranking vectors entertained by the algorithm when trained on this three input ERCs (starting from null initial ranking values).

(27) a.

$$\begin{array}{r|cccc|} & C_1 & C_2 & C_3 & C_4 \\ \text{ERC 1} & \text{W} & \text{L} & & \\ \text{ERC 2} & & \text{W} & \text{L} & \\ \text{ERC 3} & & & \text{W} & \text{L} \end{array}$$

b.



Of course, there is an infinite number of training ERC sequences that can be sampled from the training ERC set (27a) and fed to the algorithm. Nonetheless, the number of learning sequences described by the algorithm is very limited. Furthermore, the three input ERCs trigger the same number of updates in each learning sequence (ERCs 1 and 3 always trigger 2 updates; ERC 3 always triggers 3 updates). Why is that? For concreteness, suppose ERC 1 triggers the first update. Thus, C_1 is promoted by 1 and C_2 is demoted by 1, with the effect that they are now separated by 2. No matter how often ERC 1 is fed next, it cannot trigger another update until this ranking configuration $C_1 \gg C_2$ is disrupted. That requires C_2 to be promoted twice by ERC 2. Thus, no matter how often ERC 1 is fed to the algorithm, it will not trigger another update until ERC 2 has triggered at least 2 updates. In other words, the number of further updates triggered by ERC 1 is bound by twice the number of updates triggered by ERC 2, no matter the frequencies with which the three input ERCs are fed to the model. The rest of this Subsection formalizes this intuition in the general case.

3.2.3. *The general shape of the bound.* Let's focus on a specific $\bar{\imath}$th input ERC, that I will refer to as the *target* ERC. I want to obtain a bound on the number $\alpha_{\bar{\imath}}$ of updates triggered by this target $\bar{\imath}$th input ERC in an arbitrary run of the EDRA up to a certain arbitrary time. Let C_ℓ be the loser-preferrer of this target $\bar{\imath}$th ERC (recall that I am assuming that all input ERCs have a unique loser-preferrer; see Appendix C.3 for the straightforward extension to the general case); and let C_h be one of its (possibly many) winner-preferrers, as in (28).

(28)
$$\bar{\imath}\text{th ERC} = \begin{array}{c} \begin{array}{ccccc} \cdots & C_h & \cdots & C_\ell & \cdots \end{array} \\ \left[\begin{array}{ccccc} \ldots & \text{W} & \ldots & \text{L} & \ldots \end{array}\right] \end{array}$$

Suppose the initial ranking value $\theta_\ell^{\text{init}}$ of the loser-preferrer C_ℓ is larger than the initial ranking value θ_h^{init} of the winner-preferrer C_h. A certain number of updates by this $\bar{\imath}$th ERC are thus justified just in order to compensate for this bad choice of the initial ranking values. Each time the $\bar{\imath}$th ERC triggers an update according to the general re-ranking rule (25), the winner-preferrer C_h is promoted by the corresponding promotion amount $p_{\bar{\imath}}$ and the loser-preferrer C_ℓ is demoted by 1. The separation between C_ℓ and C_h thus decreases by $1 + p_{\bar{\imath}}$ with every update. After $\frac{\theta_\ell^{\text{init}} - \theta_h^{\text{init}}}{1+p_{\bar{\imath}}}$ updates, C_ℓ and C_h will thus have the same ranking value.[9] These considerations informally motivate the term (29a) in the bound on the number $\alpha_{\bar{\imath}}$ of updates triggered by this $\bar{\imath}$th target ERC. One more update is sufficient to bring the winner-preferrer C_h above the loser-preferrer C_ℓ, establishing a ranking configuration $C_h \gg C_\ell$ that ensures consistency with this $\bar{\imath}$th ERC. This further update corresponds to term (29b).

$$(29) \quad \alpha_{\bar{\imath}} \leq \underbrace{\frac{\theta_\ell^{\text{init}} - \theta_h^{\text{init}}}{p_{\bar{\imath}} + 1}}_{(a)} + \underbrace{1}_{(b)} + \underbrace{\frac{1}{s_{\bar{\imath}}} \sum_{i \in Dis_\ell^h} \delta_i \cdot \alpha_i}_{(c)}$$

Further updates by the target $\bar{\imath}$th ERC are only justified if this ranking configuration $C_h \gg C_\ell$ is disrupted by updates triggered by some other *disruptor* input ERC. The *generic* input ERC (different from this target $\bar{\imath}$th ERC) will be referred to as the ith ERC. And α_i will denote the number of updates triggered by this generic ith ERC in the run considered, up to the time considered. Let Dis_ℓ^h be the collection of those input ERCs that can disrupt the ranking configuration $C_h \gg C_\ell$ that guarantees consistency with the target $\bar{\imath}$th ERC (28). As we expect different disruptor ERCs to be able to disrupt the ranking configuration $C_h \gg C_\ell$ to different degrees, let δ_i denote the *amount of disruption* caused by the ith disruptor ERC. The number of further updates by the $\bar{\imath}$th ERC justified in order to re-establish the disrupted ranking configuration $C_h \gg C_\ell$ will plausibly be proportional to the number of updates α_i triggered by the disruptor ith ERC as well as to its amount of disruption δ_i. And it will also plausibly be inversely proportional to the *strength* with which the $\bar{\imath}$th ERC asserts this ranking configuration $C_h \gg C_\ell$, as expressed by some coefficient $s_{\bar{\imath}}$ that depends on the properties of the target $\bar{\imath}$th input ERC. These informal considerations motivate the term (29c). In order to make the bound conjectured in (29) explicit, I now need to define the *disruptor* ERCs (i.e., the set Dis_ℓ^h), the *amount of disruption* they cause (i.e., the coefficients δ_i) and the *strength* with which the target $\bar{\imath}$th ERC fights back (i.e., the coefficient$s_{\bar{\imath}}$).

3.2.4. *Disruptors and their amount of disruption.* The ranking configuration $C_h \gg C_\ell$ guarantees consistency with the target $\bar{\imath}$th ERC (28). Another ith input ERC might disrupt this ranking configuration if it falls into one of the three cases (30a), (30b), or (30c). The disruptor ERC (30a) has an L corresponding to constraint C_h, thus triggering a demotion of a constraint that is winner-preferring for the target $\bar{\imath}$th ERC. The disruptor ERC (30b) has a W corresponding to constraint C_ℓ, thus triggering a promotion of a constraint that is loser-preferring for the target $\bar{\imath}$th ERC. Finally, the disruptor ERC (30c) has both an L corresponding to C_h *and* a W corresponding to C_ℓ, thus combining both disruptions. The set Dis_ℓ^h of ERCs that *disrupt* the ranking configuration $C_h \gg C_\ell$ that ensures consistency with the target $\bar{\imath}$th ERC is thus defined as the collection of all input ith ERCs of type (30a), (30b) and (30c).

$$\begin{array}{llcccccccl}
 & & & \cdots & C_h & \cdots & C_\ell & \cdots & & \\
(30) & \text{a. } i\text{th ERC} = & [& \ldots & \text{L} & \ldots & \text{L}/e/\cancel{\text{W}} & \ldots &] & \text{a}'.\ \delta_i = 1 \\
 & \text{b. } i\text{th ERC} = & [& \ldots & e/\text{W}/\cancel{\text{L}} & \ldots & \text{W} & \ldots &] & \text{b}'.\ \delta_i = p_i \\
 & \text{c. } i\text{th ERC} = & [& \ldots & \text{L} & \ldots & \text{W} & \ldots &] & \text{c}'.\ \delta_i = p_i + 1
\end{array}$$

If the ith ERC is a disruptor of type (30a), then it demotes constraint C_h by 1, despite the fact that it is winner-preferring for the target $\bar{\imath}$th ERC. The *amount of disruption* δ_i it causes can thus be set equal to 1, as in (30a′). If the ith ERC is a disruptor of type

[9] Assume for concreteness that $\theta_\ell^{\text{init}} - \theta_h^{\text{init}}$ is divisible by $1 + p_{\bar{\imath}}$.

(30b), then it promotes constraint C_ℓ by p_i, despite the fact that it is a loser-preferrer for the target $\bar{\imath}$th ERC. The amount of disruption δ_i it causes can thus be set equal to p_i, as in (30b′). Finally, if the ith ERC is a disruptor of type (30c), then it demotes constraint C_h by 1 and promotes constraint C_ℓ by p_i, despite the fact that they are winner- and loser-preferrers respectively for the target $\bar{\imath}$th ERC. The amount of disruption δ_i it causes can thus be set equal to p_i+1, as in (30c′).

3.2.5. *The strength of the target ranking.* Suppose that the winner-preferrer C_h and the loser-preferrer C_ℓ currently have the same ranking value. If the target $\bar{\imath}$th ERC (31) is fed to the algorithm, it thus triggers an update. Because of this update, the winner-preferrer C_h is promoted by the promotion amount $p_{\bar{\imath}}$ and the loser-preferrer C_ℓ is demoted by the fixed demotion amount 1. Thus, the desired ranking configuration $C_h \gg C_\ell$ is established, with the ranking value of the winner-preferrer C_h surpassing the ranking value of the loser-preferrer C_ℓ with a separation of $1+p_{\bar{\imath}}$. If this separation is large (namely the promotion amount $p_{\bar{\imath}}$ is large), then it will take a large number of updates by disruptor ERCs in order to disrupt this ranking configuration $C_h \gg C_\ell$. This means that the target $\bar{\imath}$th ERC has established this ranking configuration very strongly. If instead this separation $1+p_{\bar{\imath}}$ is small (namely the promotion amount $p_{\bar{\imath}}$ is small), then a few updates by disruptor ERCs will be sufficient to disrupt it. This means that the target $\bar{\imath}$th ERC has established the ranking configuration $C_h \gg C_\ell$ only very weakly. These considerations suggest to define as in (31) the *strength* $s_{\bar{\imath}}$ with which the target $\bar{\imath}$th ERC (28) establishes the ranking configuration $C_h \gg C_\ell$.

$$(31) \quad s_{\bar{\imath}} = p_{\bar{\imath}} + 1$$

According to the bound in (29), the number of updates $\alpha_{\bar{\imath}}$ triggered by the target $\bar{\imath}$th ERC in an arbitrary run is inversely proportional to its strength $s_{\bar{\imath}}$.

3.2.6. *Putting the pieces together.* In 3.2.1, I have noted that the robust analysis of EDRAs requires robust bounds on the number of updates triggered by a certain input ERC. In 3.2.3, I have bound the number of updates as in (29), based on heuristic considerations. And I have defined the ingredients that appear in the latter bound in 3.2.4 and 3.2.5. This heuristic reasoning turns out to be correct, as stated in the following theorem 2. For a proof of (a slight generalization of) this theorem, see Appendix C. The bound holds for any choice of the promotion amounts, and in particular also for the demotion-only case whereby the promotion amounts are all set equal to zero. The extension to training ERC sets that have multiple L's is straightforward, as discussed in the Appendix C.3.

THEOREM 2. *Consider a run of the EDRA with the general update rule (25) on a training set of ERCs that have a unique* L *each. Focus on a target $\bar{\imath}$th input ERC as in (28), whose loser-preferrer is C_ℓ. Suppose this ERC triggers at least an update up to some arbitrary time in the run considered. Then, the number of updates $\alpha_{\bar{\imath}}$ triggered by this target $\bar{\imath}$th input ERC can be bound as in (29) for each one of its winner-preferring constraints C_h. Here, $\theta_\ell^{\text{init}}$ and θ_h^{init} are the initial raking values of these two constraints C_ℓ and C_h; the set Dis_ℓ^h of disruptor ERCs and their amount of disruption δ_i are defined as in (30); the strength $s_{\bar{\imath}}$ of the target $\bar{\imath}$th ERC is defined as in (31); and α_i is the number of updates triggered by the ith ERC in that same run up to the time considered.* ■

This theorem provides a distribution-independent bound on the number of updates triggered by the target $\bar{\imath}$th input ERC. As the current ranking vector depends on the number of updates triggered by the various ERCs through the identity (26), the Theorem can in turn be used as a new tool for the robust analysis of EDRAs. Various specific tricks-of-the-trade to do that are illustrated with some concrete examples in Section 4.

4. An application to modeling the child early acquisition of phonotactics

One of the most important applications of EDRAs concerns modeling the child early acquisition of phonotactics, as described in Subsection 4.1. But does the EDRA model manage to learn the target adult phonotactics it is trained on? Subsection 4.2 properly formulates this core question of the computational theory of the EDRA model of the acquisition of phonotactics. In Subsection 4.3, I then address this question focusing on three simple, abstract case studies, discussed in Prince and Tesar (2004). Crucially, these case studies require EDRAs that perform both constraint promotion and demotion. Tesar and Smolensky's (1998) analysis of demotion-only EDRAs from Subsection 3.1 is thus insufficient. The new tool for the robust analysis of promotion/demotion EDRAs developed in Subsection 3.2 is needed instead. And I illustrate how indeed the latter tool can be used to provide robust guarantees for EDRAs' success on these case studies.

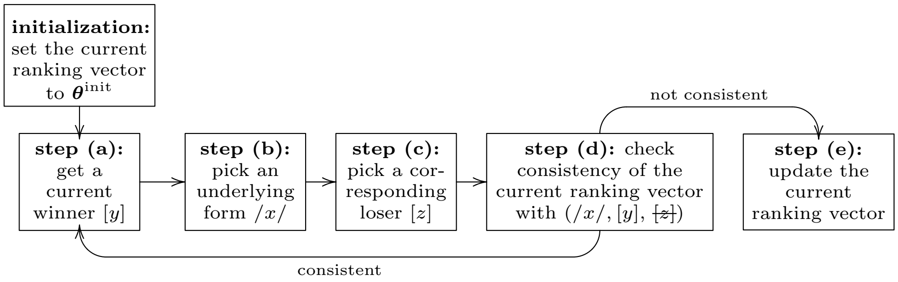
4.1. The EDRA model of the child early acquisition of phonotactics.

In carefully controlled experimental conditions, nine-month-old infants already react differently to licit and illicit sound combinations (Jusczyk et al. 1993, among others). They thus display knowledge of the target adult phonotactics at an early stage, when other linguistic abilities are still not fully developed. In particular, morphology is still lagging behind at this early age, so that the child has still no access to phonological alternations. As Hayes (2004) notes, "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." There is therefore an *early stage* of language acquisition, when the child manages to acquire the target adult phonotactics despite being blind to alternations. EDRAs have been endorsed in the OT acquisition literature as cognitively plausible models of this early developmental stage (Gnanadesikan 2004 and Boersma and Levelt 2000 among many others; but see Tessier 2009 for critical discussion, as well as Jesney and Tessier 2009 for a variant of the model within the HG framework). In fact, they describe a sequence of grammars that can be matched with child acquisition paths, providing a straightforward tool to model child acquisition gradualness. Furthermore, they don't keep track of previously seen forms and don't require a lexicon, so that they don't impose memory requirements that might be implausible at this early developmental stage. In this Subsection, I outline the EDRA model of the early acquisition of phonotactics.

4.1.1. *Algorithmic kernel.*

According to the original formulation (13), an EDRA is provided with a whole underlying/winner/loser form triplet $(/x/, [y], \cancel{[z]})$ at each iteration. This assumption is not viable in the context of modeling the early stage of the child acquisition of phonotactics. In fact, at this stage the child is blind to alternations and thus has no access to pairs of an underlying form $/x/$ and a corresponding (possibly different) winner surface form $[y]$. The child is really only exposed to a stream of surface forms, all licit according to the target phonotactics (abstracting away from the possibility of noise in the data). In this specific modeling application, the EDRA is thus only provided with a licit winner form [y] at each iteration, and needs to complete it into a triplet by reconstructing by itself both a corresponding underlying form $/x/$ and a corresponding loser form $[z]$. In other words, step (13a) in the original formulation of EDRAs is split up into the three steps (32a)-(32c).

(32)


initialization: set the current ranking vector to $\boldsymbol{\theta}^{\text{init}}$
step (a): get a current winner $[y]$
step (b): pick an underlying form $/x/$
step (c): pick a corresponding loser $[z]$
step (d): check consistency of the current ranking vector with $(/x/, [y], \cancel{[z]})$
step (e): update the current ranking vector
not consistent
consistent

The core algorithmic scheme (32) of the EDRA model of the child early acquisition of phonotactics needs to be completed with a discussion of four crucial implementation details, concerning the choice of the initial ranking vector $\boldsymbol{\theta}^{\text{init}}$; the subroutine that provides the underlying form $/x/$ at step (13b); the subroutine that provides the loser form $[z]$ at step (13c); and the re-ranking rule used at step (13e).

4.1.2. *Choice of the initial ranking vector.* Constraints in OT come in two varieties. *Faithfulness constraints* measure how well a certain candidate matches the corresponding underlying form. For instance, the constraint IDENT[VOICE] in (3b) is violated when an underlying form and one of its candidates differ in voicing. *Markedness constraints* measure how much a candidate violates well-formedness conditions. For instance, the constraint *[+VOICED, −SONORANT] in (3b) is violated by a candidate that contains a voiced obstruent. Fikkert and De Hoop (2009, p. 325) note that "the recurrent pattern in child language data is that children's output is considerably less marked than the corresponding adult target forms. [. . .]. Hence, the starting hypothesis in much research on phonological acquisition is that children begin with markedness constraints outranking faithfulness constraints." I thus assume that markedness constraints start out with a large positive initial ranking value while faithfulness constraints start out with a null initial ranking value. See Smolensky (1996b,a) for theoretical arguments in favor of this choice of the initial ranking vector; Jusczyk et al. (2002) for empirical evidence; Davidson et al. (2004) for a review; and Hale and Reiss (1998) for critical discussion.

4.1.3. *Choice of the underlying form.* Pater and Barlow (2003, p. 490) write that "the *input* in [. . .] child phonology [is] taken to correspond to the child's stored lexical representation" under the assumption that it "is likely [that] children do perceive and store [forms] accurately". A number of authors have thus suggested that the subroutine for the choice of the underlying form at step (32b) be defined as follows: the EDRA always posits an underlying form $/x/$ identical to the given surface form $[y]$ it is provided with.[10] Tesar (2008) provides a computational justification of this assumption: under mild restrictions on the constraint set, he shows that the set of underlying/winner/loser form triplets thus constructed at steps (32a)-(32c) is guaranteed to be consistent with at least a ranking (no matter how the loser form is chosen).

4.1.4. *Choice of the loser form.* To complete the construction of an underlying/winner/loser form triplet, the EDRA needs to pick a loser form at step (32c). Tesar and Smolensky (1998) suggest that the EDRA sets the loser $[z]$ equal to the winner currently predicted by the algorithm (i.e. predicted by an arbitrary refinement of the current ranking vector) for the current underlying form $/x/$. See Magri (2012c) for a justification of this assumption.

[10] Of course, this assumption only makes sense provided that the set of underlying forms coincides with the set of surface forms (namely $\mathcal{X} = \mathcal{Y}$), so that the same phonological structure can be construed both as an underlying and as a surface form.

4.1.5. *Choice of the re-ranking rule.* In Magri (2012b, Section 4), I have argued that the EDRA model cannot adopt at step (32e) a re-ranking rule that performs constraint demotion but no constraint promotion, such as the re-ranking rule (15) considered in 2.3.1. Here is the logic of the argument. I have assumed in 4.1.3 that the EDRA posits at step (32b) underlying forms that are fully faithful to the intended winners. Thus, the faithfulness constraints are never loser-preferring. In other words, the EDRA is trained on ERCs that have W's and e's but no L's corresponding to the faithfulness constraints. As demotion-only re-ranking rules only re-rank loser-preferrers, a demotion-only EDRA will never re-rank the faithfulness constraints throughout learning. And this cannot be right, for at least two reasons. The *first* reason is that the algorithm would fail to learn the target phonotactics in those cases where that phonotactics requires a specific relative ranking of the faithfulness constraints. The case studies from Prince and Tesar (2004) considered in Subsection 4.3 illustrate this scenario. The *second* reason is that the algorithm would fail to model child acquisition paths where the repair strategy for a given marked structure changes over time, as the choice of the repair strategy is determined by the relative ranking of the faithfulness constraints. In conclusion, a certain amount of constraint promotion is needed, both from a computational perspective (does the algorithm learn the target phonotactics?) as well as from a modeling perspective (does the algorithm model child acquisition paths?). In the rest of this Section, I focus on the former perspective.

4.2. **Computational soundness of the EDRA model of the acquisition of phonotactics.** The EDRA (32) is trained on a sequence of surface forms that are all licit according to the target phonotactics. Will the algorithm eventually learn the target phonotactics it is trained on? This Subsection formalizes this core question of the computational theory of the EDRA model of the acquisition of phonotactics

4.2.1. *Languages.* Consider again a 4-tuple of universal specifications $(\mathcal{X}, \mathcal{Y}, Gen, \mathcal{C})$ as in (3a). As discussed in 2.1.6, a ranking $\gg$ over the constraint set $\mathcal{C}$ induces a corresponding OT grammar $\mathsf{OT}_{\gg}$, which has the form of a function from the set $\mathcal{X}$ of underlying forms into the set $\mathcal{Y}$ of surface forms. The range of this function is called the *language* $L_{\gg}$ generated by the ranking $\gg$, as stated in (33a). In other words, the language $L_{\gg}$ is the set of surface forms that are licit according to (the OT grammar corresponding to) the ranking $\gg$. To learn the phonotactics induced by a target ranking $\gg$ thus means to learn the corresponding language $L_{\gg}$, as it represents the specific way that $\gg$ splits up the phonological forms into those that are licit (i.e., those in $L_{\gg}$) and those that are illicit (i.e., those in the complement of $L_{\gg}$).

(33) a. $L_{\gg} = \mathcal{R}ange\big(\mathsf{OT}_{\gg}\big)$ b. $L_{\gg} = \big\{[\text{ta}], [\text{da}], [\text{rat}]\big\}$

To illustrate, I provide in (33b) the language $L_{\gg}$ generated by the ranking $F_{\text{pos}} \gg M \gg F_{\text{gen}}$ of the constraint set in (3b). Indeed, it is the range of the corresponding grammar $\mathsf{OT}_{\gg}$ computed in (8a).

4.2.2. *Target and final language.* The EDRA (32) is trained on a sequence of surface forms that are licit according to a target ranking $\gg_{\text{target}}$, namely that are sampled from the corresponding target language $L_{\text{target}} = L_{\gg_{\text{target}}}$. Assume the EDRA adopts a convergent re-ranking rule at step (32e). Let $\boldsymbol{\theta}_{\text{final}}$ be the final ranking vector entertained by the algorithm in the run considered. Convergence means that the subset condition (34) holds: the language L_{final} corresponding to (any refinement of) the final ranking $\boldsymbol{\theta}_{\text{final}}$ is at least as large as the language L_{target} the algorithm has been trained on.

(34) $L_{\text{final}} \supseteq L_{\text{target}}$.

In fact, suppose by contradiction that there existed a form in L_{target} that did not belong to L_{final}. This means in turn that the final ranking vector $\boldsymbol{\theta}_{\text{final}}$ (admits a refinement that) is not consistent with that form. The algorithm would therefore still make mistakes on that form, contradicting the hypothesis that it has converged.

4.2.3. *Restrictiveness.* The problem of the acquisition of phonotactics is twofold. On the one hand, the learner needs to correctly rule-in each form which is licit according to the target adult phonotactics. On the other hand, the learner needs to correctly rule-out each form which is instead illicit. The convergence condition (34) ensures that the learner succeeds at the first half of the task, namely it recognizes as licit every form which is indeed licit according to the target adult phonotactics. But a convergent EDRA could still fail at the second half of the task, namely it could fail at ruling out forms which are illicit. For instance, the EDRA could converge to a final ranking that assigns all faithfulness constraints to the top, and thus incorrectly declares licit each single form. In order to rule out such cases, the EDRA needs to satisfy the additional *restrictiveness* condition (35): the language L_t corresponding to (any refinement of) the current ranking vector $\boldsymbol{\theta}_t$ entertained by the algorithm at a certain time t is never larger than the target language L_{target} the algorithm is trained on.

(35) $L_t \subseteq L_{\text{target}}$

Restrictiveness (35) holds in particular of the language L_{final} corresponding to the final ranking vector. Together with convergence (34), restrictiveness thus entails that $L_{\text{final}} = L_{\text{target}}$. In other words, convergent and restrictive EDRAs are guaranteed to correctly learn the target adult language L_{target} they are trained on. Tools for the robust analysis of EDRAs are thus needed in order to provide guarantees that the algorithm satisfies the restrictiveness condition (35).

4.2.4. *PT's reformulation.* Prince and Tesar (2004, henceforth: PT) offer a compact computational characterization of the restrictiveness condition (35). They note that faithfulness constraints work towards the preservation of the underlying contrasts while markedness constraints work against it. Thus, a large language is likely to arise when the faithfulness constraints are ranked high while a small language is likely to arise when it is the markedness constraints that are ranked high. To capture this intuition, they pair up a ranking $\gg$ with its *restrictiveness measure* (henceforth: R-measure) $\mu(\gg)$ defined in (36). In words, $\mu(\gg)$ is the number of pairs of a faithfulness and a markedness constraint such that the former is ranked above the latter. Thus, the measure $\mu(\gg)$ is small provided the faithfulness constraints are low $\gg$-ranked.

(36) $\mu(\gg) = \Big|\Big\{(F, M) \in \mathcal{F} \times \mathcal{M} \,\Big|\, F \gg M\Big\}\Big|$

PT therefore suggest to use this measure μ as a proxy for restrictiveness, in the sense that the language generated by a ranking $\gg$ is expected to be small provided that the corresponding measure $\mu(\gg)$ is small. And they thus restate the restrictiveness condition (35) as in (37): each refinement $\gg_t$ of the current ranking vector $\boldsymbol{\theta}_t$ entertained by the algorithm at an arbitrary time t has R-measure not larger than the target ranking $\gg_{\text{target}}$.

(37) $\mu(\gg_t) \leq \mu(\gg_{\text{target}})$

In 4.1.2, I have assumed that markedness constraints start out with a large, positive initial ranking value while faithfulness constraints start out with a null initial ranking value. This means that the EDRA starts from an initial ranking vector with minimal (namely null) R-measure. We are thus interested in the following question: starting from an initial ranking with minimal R-measure, will the EDRA be able to keep the faithfulness constraints low ranked and thus walk through a sequence of ranking vectors with small R-measure? In the next Subsection, I investigate this question on three case studies.

4.3. **Prince and Tesar's (2004) three case studies.**

4.3.1. *Overview of the three case studies.* To achieve a small R-measure, the faithfulness constraints need to be low ranked. A faithfulness constraint F can be ranked low if it is not needed to ensure consistency with the training ERC set. Intuitively, there are three reasons why that might be the case, summarized in (38). One reason is that F is never winner-preferrer in the training ERC set, and thus can never contribute to consistency. This means in particular that F is *inactive* (namely, always even), as it can never be loser-preferrer either, because of the assumption that underlying forms are faithful to the winners. Another reason why the faithfulness constraint F does not contribute to consistency with the training ERC set is that, despite it being active, its work can be done by a markedness constraint: in any input ERC where F is winner-preferrer and thus could contribute to consistency with that ERC, there is a markedness constraint that is winner-preferrer as well and could be ranked higher. A final reason why a faithfulness constraint F might not play any role towards consistency with the training ERC set is that, despite it being active, its work can be done by another faithfulness constraint: in any input ERC where F is winner-preferrer, there is another faithfulness constraint that is winner-preferring as well and is ranked higher.

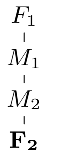
(38)
$$\begin{array}{r}F \text{ does not contribute to}\\ \text{consistency because}\end{array}\left\{\begin{array}{l}\text{it is } \textit{inactive}\\[1em] \text{it is } \textit{active} \text{ but} \left\{\begin{array}{l}\text{its work can be done by another}\\ \textit{markedness} \text{ constraint}\\ \text{its work can be done by another}\\ \textit{faithfulness} \text{ constraint}\end{array}\right.\end{array}\right.$$

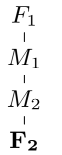
PT thus single out the three benchmark test cases (39) for restrictiveness, that illustrate these three scenarios (38), respectively. Note that the faithfulness constraints are never loser-preferrers in these three case studies (39), as a consequence of the assumption made in 4.1.3 that the EDRA always posits underlying forms identical to the winner forms, so that the faithfulness constraints can never prefer the loser.

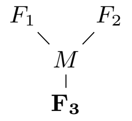
(39) a.
$$\begin{array}{cccc} F_1 & \mathbf{F_2} & M_1 & M_2 \\ \end{array}$$
$$\left[\begin{array}{cccc} & & \mathrm{W} & \mathrm{L} \\ \mathrm{W} & & \mathrm{L} & \mathrm{W} \end{array}\right]$$

b.
$$\begin{array}{cccc} F_1 & \mathbf{F_2} & M_1 & M_2 \\ \end{array}$$
$$\left[\begin{array}{cccc} \mathrm{W} & & \mathrm{L} & \\ & \mathrm{W} & \mathrm{W} & \mathrm{L} \end{array}\right]$$

c.
$$\begin{array}{cccc} F_1 & F_2 & \mathbf{F_3} & M \\ \end{array}$$
$$\left[\begin{array}{cccc} \mathrm{W} & & \mathrm{W} & \mathrm{L} \\ & \mathrm{W} & & \mathrm{L} \\ \mathrm{W} & & & \mathrm{L} \end{array}\right]$$

a′.


b′.


c′.


In the raining ERC set (39a), the faithfulness constraint F_2 is inactive, and thus useless for consistency. It can therefore be ranked at the bottom, as in the desired ranking (39a′), which is indeed the one with minimal R-measure among those consistent with this ERC set. In the training ERC set (39b), the faithfulness constraint F_2 is winner-preferrer in the second ERC. Yet, it is not really needed for consistency. In fact, the loser-preferrer M_2 in the second ERC can be shielded with the winner-preferrer markedness constraint M_1. The faithfulness constraint F_2 can thus be ranked at the bottom, as in the desired ranking (39b′), which is indeed the one with minimal R-measure among those consistent with this ERC set. Finally, in the training ERC set (39c), the faithfulness constraint F_3 is winner-preferrer in the first ERC. Yet it is not really needed for consistency. In fact, the faithfulness constraint F_1 needs to be ranked above the markedness constraint M no matter what. The faithfulness constraint F_3 can thus be ranked at the bottom, as in the desired ranking (39c′), which is indeed the one with minimal R-measure among those consistent with this ERC set.

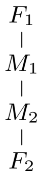
4.3.2. *Preview of the robust analysis of the three case studies.* In this Subsection, I provide robust guarantees for the restrictiveness condition (37) on these three case studies (39). An EDRA that adopts Tesar and Smolensky's (1998) demotion-only re-ranking rule (15) is doomed to fail. In fact, the faithfulness constraints would never be re-ranked by a demotion-only EDRA, as they are never loser-preferrers. They would thus stay put at their initial ranking value. And the algorithm would have no way to pull apart the useful faithfulness constraints that need to be ranked high from the redundant ones that can instead be ranked low. In order to re-rank the faithfulness constraints, some amount of constraint promotion is thus needed, as in the GLA's update rule (16) or my calibrated update rule (17). Yet, it is not *prima facie* obvious that the faithfulness constraints will be re-ranked in such a way that the crucial ones go on top while the redundant ones stay at the bottom. This is where the new tool for the robust analysis of EDRAs provided by Theorem 2 becomes useful. In fact, it will allow me to show that EDRAs that perform both constraint demotion and promotion do satisfy the restrictiveness condition (37) when run on the three case studies (39). To get a sense of why that is the case, consider for instance the case study (39b). In this case, we want F_2 to be ranked low because the work it could do in accounting for the loser-preferrer M_2 in the the second ERC can actually be done by the other winner-preferrer M_1. The two markedness constraints M_1 and M_2 start out with the same initial ranking value and stay close to each other throughout learning, no matter the training ERC sequence. Thus, many times the second ERC is fed to the EDRA, it will not be able to trigger any update, because M_1 currently shelters M_2, despite the fact that F_2 is still low ranked. In other words, the number of updates triggered by the second ERC is small no matter how often that ERC is fed to the algorithm, as can be easily established using Theorem 2. As the second ERC triggers few updates and as it is the only one that promotes F_2, then F_2 will raise slowly, yielding the desired ranking configuration (39b′). In short, we want F_2 to stay low because its work can be done by M_1. And indeed, the fact that its work can be done by M_1, correctly prevents F_2 from raising quickly.

4.3.3. *Robust analysis of the first case study.* The faithfulness constraint F_2 is inactive in the training ERC set (40a), as all its entries are e's. Among the rankings consistent with these ERCs (40a), the one that ranks the faithfulness constraints as low as possible and thus attains minimum R-measure is (40b), with F_2 bottom ranked. Other rankings consistent with these ERCs, such as (40c), might generate superset languages, as F_2 is ranked too high.

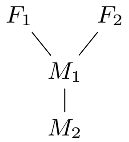
(40) a.

$$\begin{array}{r|cccc|} & F_1 & F_2 & M_1 & M_2 \\ \text{ERC 1} & & & \text{W} & \text{L} \\ \text{ERC 2} & \text{W} & & \text{L} & \text{W} \end{array}$$

b.



c.

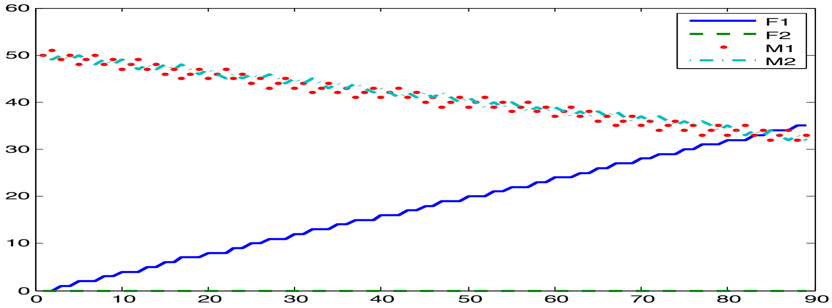


The challenge raised by this case study is thus for the EDRA to recognize that F_2 is useless and should therefore be ranked at the bottom, as in (40b). PT call this case study *Avoid the Inactive* and summarize it with the following injunction: "when placing faithfulness constraints into the hierarchy, if possible only place those that prefer some winner" (p. 260). How do EDRAs score on this case study (40)?

If the EDRA adopts Tesar and Smolensky's (1998) demotion-only re-ranking rule (15), then it cannot satisfy the PT's restrictiveness condition (37) and in particular converge to the desired ranking (40b). In fact, this demotion-only update rule only re-ranks loser-preferrers. Thus, it cannot distinguish between the winner-preferrer F_1 and the inactive F_2. Both stay put at their common initial ranking value. As M_1 and M_2 have to drop below F_1 for consistency, they will also drop below F_2. The algorithm thus converges to the incorrect ranking configuration (40c).

Things are very different when the EDRA performs both constraint demotion and promotion. To start, I provide in (41) the dynamics of the current ranking values in a run of the EDRA with the calibrated promotion/demotion update rule (17) and the two input ERCs (40a) sampled uniformly.[11]

(41)



The two faithfulness constraints F_1 and F_2 start out equally ranked, underneath the two markedness constraints M_1 and M_2. Of course, F_2 stays put at its initial ranking value, as it is always inactive and thus never gets promoted by the promotion component of the update rule (17). On the contrary, F_1 raises every time ERC 2 triggers an update. In the meantime, M_1 and M_2 drop. As soon as M_1 crosses F_1, the current ranking vector becomes consistent with ERC 2. A few more updates suffice to bring M_2 underneath M_1, ensuring consistency with ERC 1 as well. Learning thus ceases, and M_1 and M_2 find themselves squeezed in between F_1 and F_2, as in the desired ranking (40b).

In the simulation reported in (41), the two input ERCs are sampled and fed to the model with the same frequency. Yet, the success obtained in (41) does not in any way depend on the input frequencies, as stated in the following claim 1, that provides a first distribution-independent result on restrictiveness of promotion-demotion EDRAs.

CLAIM 1. *Consider a run of the EDRA model on the training set of input ERCs (40a). As prescribed in 4.1.2, assume that the initial ranking values of the faithfulness constraints are null while the initial ranking values of the markedness constraints are set equal to a large positive constant θ^{init}. Furthermore, assume that the two input ERCs trigger the update (79) repeated from above, where p_1 and p_2 are the promotion amounts corresponding to the two input ERCs 1 and 2 ($p_1, p_2 \geq 0$).*

(42) a. Demote each undominated loser-preferrer by 1;

b. promote each winner-preferrer by the promotion amount p_i.

As long as the promotion amount p_2 of ERC 2 is larger than zero, the EDRA always satisfies PT's restrictiveness condition (37), no matter how the two input ERCs are sampled and fed to the algorithm. In particular, the unique refinement of the final ranking vector is always (40b). ■

The proof of claim 1 is provided in Appendix D. The core idea is as follows. The target ranking (40b) has R-measure 2, as two markedness constraints are ranked underneath the faithfulness constraint F_1 and none underneath the faithfulness constraint F_2. Suppose by contradiction that the EDRA did not satisfy PT's restrictiveness condition (37). This means that some current ranking vector admits a refinement whose R-measure is 3. This means in turn that at least one of the two markedness constraints is ranked underneath the

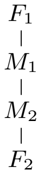
[11] The simulations reported in this paper have been performed with the Matlab file `ErrorDrivenRankingAlgorithm`, available on the author's website.

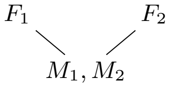
faithfulness constraint F_2. As F_2 stays put at its initial null ranking value, this means in turn that the ranking value of one of the two markedness constraints must have dropped down to zero or below. Yet, M_1 cannot have dropped that low. In fact, only ERC 2 demotes M_1. And the bound on the number of updates provided by Theorem 2 ensures that ERC 2 cannot trigger the too many updates that would be needed to demote M_1 down to zero. Nor can M_2 have dropped down to zero. In fact, only ERC 1 demotes M_2. Since M_1 is winner-preferrer in that ERC, then M_2 never makes it far from M_1. And since M_1 stays well above zero, then M_2 stays well above zero as well.

4.3.4. *Robust analysis of the second case study.* The faithfulness constraint F_1 in the training ERC set (43a) needs to be ranked above the markedness constraint M_1 for consistency with ERC 1. Consistency with ERC 2 can be guaranteed by ranking M_1 above the other markedness constraint M_2. Consistency with this training ERC set can thus be obtained without the other faithfulness constraint F_2 playing any role. Among the consistent rankings, the one that ranks the faithfulness constraints as low as possible and thus attains minimum R-measure is (43b), with F_2 bottom ranked. Other rankings consistent with these ERCs, such as (43c), might generate superset languages, as F_2 is ranked too high.

(43) a.
$$\begin{array}{r} \\ \text{ERC 1} \\ \text{ERC 2} \end{array} \begin{array}{cccc} F_1 & F_2 & M_1 & M_2 \\ \end{array}$$

$$\begin{array}{r|cccc} & F_1 & F_2 & M_1 & M_2 \\ \hline \text{ERC 1} & \text{W} & & \text{L} & \\ \text{ERC 2} & & \text{W} & \text{W} & \text{L} \end{array}$$

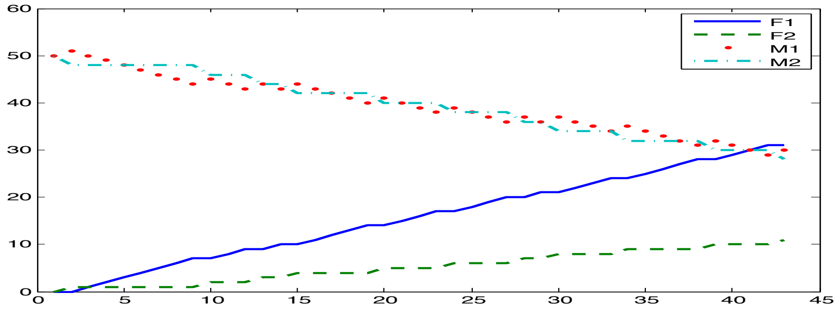
b.


c.


The challenge raised by this case study is to recognize that F_2 is useless although active, and should therefore be ranked at the bottom, as in (43b). PT call this case study *Richest Markedness Cascade* and summarize it with the following injunction: "when placing faithfulness constraints into the hierarchy, [. . .] place the [one] that yields the largest set of markedness constraints in contiguous subsequent strata, i.e. until another faithfulness constraint has to be ranked" (p. 268). How do EDRAs score on this case study (43)?

Again, if the EDRA adopts Tesar and Smolensky's (1998) demotion-only re-ranking rule (15), then it cannot satisfy the PT's restrictiveness condition (37) and in particular converge to the desired ranking (43b). The reasoning is identical to that in 4.3.3 above. As this demotion-only update rule does not re-rank winner-preferrers, it cannot distinguish between the crucial winner-preferrer F_1 and the redundant winner-preferrer F_2. Thus, both stay put at their common initial ranking value. As M_1 and M_2 drop below F_1 for consistency, they also drop below F_2. The model thus converges to the incorrect ranking configuration (43c).

Once again, things are very different when the EDRA performs both constraint demotion and promotion. To start, I provide in (44) the dynamics of the ranking values in a run of the EDRA with the calibrated promotion/demotion update rule (17) and the two input ERCs (43a) sampled uniformly.

(44)


The two markedness constraints M_1 and M_2 drop with approximately the same speed, as shown by their overlapping trajectories. Almost throughout the entire run, every time ERC 1 is fed to the algorithm, it is inconsistent with the current ranking vector, as the winner-preferrer F_1 is well below the loser-preferrer M_1. Thus, ERC 1 always triggers an update and promotes F_1. Things are different for ERC 2. Many of the times that ERC 2 is fed to the algorithm, the current ranking vector is already consistent with ERC 2, as the winner-preferrer M_1 happens to be ranked above the loser-preferrer M_2, without any need for the other winner-preferrer F_2 to do any work. Thus, ERC 2 often does not trigger an update and thus does not promote F_2. As a consequence, F_1 raises faster than F_2. At the time M_1 and M_2 cross F_1, the latter is well above F_2. From this moment on, the current ranking vector will be consistent with ERC 1, that will therefore trigger no further updates. Furthermore, since M_1 and M_2 are so close, the former will get above the latter right away, with no need for M_2 to drop all the way down below F_2. As soon as M_1 gets above M_2, the current ranking vector becomes consistent with ERC 2 as well. Learning thus ceases and M_1 and M_2 find themselves squeezed in between F_1 and F_2, as in the desired ranking configuration (43b). Recall PT's intuition that F_2 should be bottom ranked because it is not really needed for consistency with the training ERC set (43a), as its work can be done instead by M_1. The promotion/demotion EDRA automatically implements this intuition: precisely because of the fact that the work of F_2 in ERC 2 can be done most of the times by M_1, the faithfulness constraint F_2 is promoted little and thus stays low.

In the simulation reported in (44), the two input ERCs have been fed to the EDRA with the same frequency. One might expect this fact to play a key role in the success. Indeed, ERC 2 is easier to account for, because of the winner-preferrer M_1. Thus, it does not trigger an update many of the times it is fed to the EDRA. If ERCs 1 and 2 are sampled with comparable frequencies, then one might expect this fact to entail that ERC 2 will indeed trigger less updates than ERC 1 and that F_2 will thus be promoted less than F_1, yielding the desired ranking of F_1 above F_2. But what if ERC 2 is fed to the algorithm much more often than ERC 1? In this case, one might expect that ERC 1 and ERC 2 will trigger a comparable number of updates and will thus in the end promote F_1 and F_2 at the same speed, failing at ranking F_1 above F_2. Quite surprisingly, that turns out not to be the case. The success obtained in the simulation (44) does not in any way depend on how the input ERCs are fed to the model, as stated in the following claim 2, that provides a second distribution-independent result on the PT's restrictiveness of promotion-demotion EDRAs. Condition (46) requires the promotion amount p_1 corresponding to ERC 1 not to be too small compared to the promotion amount p_2 corresponding to ERC 2. In particular, it requires p_1 to be non-null, so that ERC 1 needs indeed to perform some constraint promotion. This condition (46) is satisfied both in the case of the GLA update rule (16), whereby $p_1 = p_2 = 1$; as well as in the case of the convergent variant (17), whereby $p_1 = c < 1$ and $p_2 = c/2$.

CLAIM 2. *Consider a run of the EDRA model on the training set of input ERCs (43a). As prescribed in 4.1.2, assume that the initial ranking values of the faithfulness constraints are null while the initial ranking values of the markedness constraints are set equal to a large positive constant θ^{init}. Furthermore, assume that the two input ERCs trigger the update (87) repeated from above, where p_1 and p_2 are the promotion amounts corresponding to the two input ERCs 1 and 2 ($p_1, p_2 \geq 0$).*

(45) a. *Demote each undominated loser-preferrer by 1;*
b. *promote each winner-preferrer by the promotion amount p_i.*

Assume that the promotion amount p_1 of ERC 1 satisfies the strict inequality (46).

(46) $$p_1 > \frac{p_2}{1 + p_2}$$

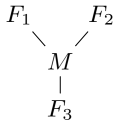
Then, the EDRA always satisfies PT's restrictiveness condition (37), no matter how the two input ERCs are sampled and fed to the algorithm. In particular, the unique refinement of the final ranking vector is always (43b). ■

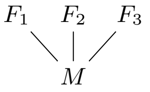
The proof of claim 2 is provided in Appendix E. The core idea is as follows. F_1 is only promoted by ERC 1 and F_2 is only promoted by ERC 2. Thus, in order for F_1 to be ranked high and F_2 low as in the target ranking (43b), ERC 1 needs to trigger many updates while ERC 2 needs to trigger few updates. Consider for instance the GLA update rule (16), whereby $p_1 = p_2 = 1$. Indeed, Theorem 2 can then be used to bound the number of updates triggered by the two ERCs, showing that ERC 2 triggers less than half of the updates triggered by ERC 1, no matter how the two ERCs are sampled and fed to the algorithm (see the crucial inequality (93b) in Appendix E, with $p_2 = 1$). The intuitive reason for why that is the case is as follows. One initial update by ERC 2 according to the GLA re-ranking rule promotes M_1 by 1 and demotes M_2 by 1, ensuring a ranking configuration consistent with ERC 2, whereby the winner-preferrer M_1 outranks the loser-preferrer M_2 with a separation of 2 (recall that all markedness constraints start out with the same initial ranking value). Thus, ERC 2 will not trigger any further update until this ranking configuration is disrupted. And it takes two updates by ERC 1 in order to disrupt this ranking configuration, as ERC 1 demotes M_1 by 1.

4.3.5. *Robust analysis of the third case study.* The faithfulness constraints F_1 and F_2 in the training ERC set (47a) need to be ranked above the markedness constraint M, in order to ensure consistency with ERCs 2 and 3. Consistency with ERC 1 is thus entailed, and the faithfulness constraint F_3 thus plays no role. Among the rankings consistent with the ERC set (47a), the one that ranks the faithfulness constraints as low as possible and thus attains minimum R-measure is (47b), with F_3 bottom ranked. Other consistent rankings, such as (47c), might generate superset languages, as they rank F_3 too high.

(47) a.

$$\begin{array}{l} & F_1 & F_2 & F_3 & M \\ \text{ERC 1} & \left[\begin{array}{cccc} \text{W} & & \text{W} & \text{L} \\ & \text{W} & & \text{L} \\ \text{W} & & & \text{L} \end{array}\right] \\ \text{ERC 2} \\ \text{ERC 3} \end{array}$$

b. 

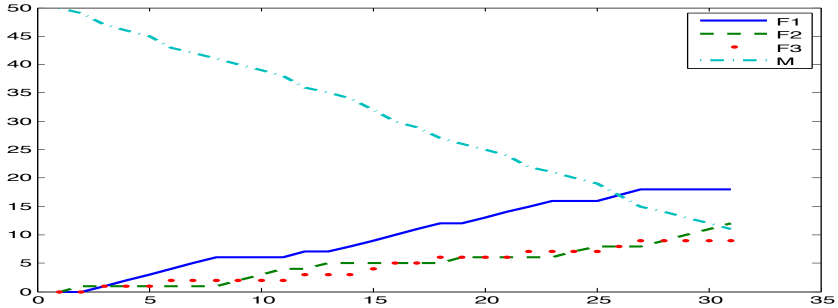
c. 

The challenge raised by this case study is thus to recognize that F_3 is useless for consistency and should therefore be ranked at the bottom, as in (47b). PT call this case study *Smallest Effective F-sets* and summarize it with the following injunction: "when placing faithfulness constraints into the hierarchy, place the smallest set of faithfulness constraints [namely $\{F_1, F_2\}$ rather than $\{F_1, F_2, F_3\}$] that frees up some markedness constraint [namely M]" (p. 267). How do EDRAs score on this case study (47)?

Once again, there is no way of converging to the desired ranking (47b) with the demotion-only re-ranking rule (15). As this update rule only demotes loser-preferrers, it does not distinguish between the crucial winner-preferrers F_1 and F_2 on the one hand and the redundant winner-preferrer F_3 on the other hand. Thus, all three of them stay put at their common initial ranking value. As M drops below F_1 and F_2 for consistency, it also drops below F_3. The algorithm thus converges to the incorrect ranking (47c).

Things are very different when the EDRA performs constraint promotion as well as demotion. To start, I provide in (48) the dynamics of the ranking values in a run of the EDRA with the calibrated promotion/demotion update rule (17) and the three input ERCs (47a) sampled uniformly.

(48)



The three faithfulness constraints F_1, F_2 and F_3 start out equally ranked, underneath the markedness constraint M. Crucially, F_1 raises faster than F_3, as the former is promoted by both ERCs 1 and 3, while the latter is only promoted by ERC 1. The first faithfulness constraint intercepted by M in its free fall is thus F_1. As soon as F_1 and M cross, the current ranking vector becomes consistent with ERCs 1 and 3, that thus cease to trigger any update. A few more updates by ERC 2 allow M to cross F_2 as well. In the end, M finds itself squeezed underneath F_1 and F_2 and above F_3, as in the desired ranking configuration (47b).

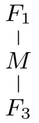
In the simulation (48), I have sampled the input ERCs uniformly. For the two preceding case studies (40) and (43), this assumption of uniform sampling turned out not to play any role: claims 1 and 2 ensure that it is not possible to construct any training ERC sequence that fools the algorithm. This third case study (47) is different, as some restrictions on the way the input ERCs are sampled and fed to the algorithm are needed. Interestingly, these restrictions can be determined analytically, thus obtaining a complete understanding of the behavior of the algorithm.

To get started, consider a simplified version of this third case study, whereby there is no faithfulness constraint F_2. The training ERC set (47a) thus simplifies to (49a), and the target ranking is (49b) with the useless faithfulness constraint F_3 ranked at the bottom, rather than (49c).

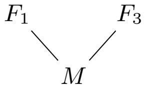
(49) a.

$$\begin{array}{l} & F_1 & F_3 & M \\ \text{ERC 1} & \left[\begin{array}{ccc} \text{W} & \text{W} & \text{L} \\ & & \\ \text{W} & & \text{L} \end{array}\right] \\ \text{ERC 2} & \\ \text{ERC 3} & \end{array}$$

b.

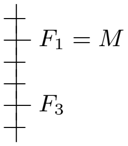


c.

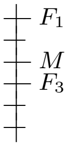


In order for the EDRA to converge to the desired final ranking (49b), it is sufficient that the promotion amount p_3 corresponding to ERC 3 is strictly larger than 0 and that ERC 3 triggers more than $2/p_3$ updates. In fact, the two faithfulness constraints F_1 and F_3 start out equally ranked; the former is promoted by both ERCs 1 and 3; while the latter is only promoted by ERC 1. Thus, the current ranking value of F_1 will always be larger than or at least equal to the current ranking value of F_3, with the difference between the two current ranking values at any given time being equal to $p_3\alpha_3$, where α_3 is the number of updates triggered by ERC 3 up to that time. In particular, if the number α_3 of updates triggered by ERC 3 is strictly larger than $2/p_3$, then F_1 will always be ranked above F_3 by more than 2. This is just enough space for the markedness constraint M to drop below F_1 without crossing F_3. In fact, right before the last update that brings M underneath F_1, M can be at most as low as F_1, as in (50a).

(50) a.



b.



With this last update, M will drop by 1 (because the demotion amount is always set equal to 1) while F_3 will raise by at most 1 (because the promotion amount is at most 1). A separation of more than 2 between F_1 and F_3 is thus just enough to ensure that after this last update M is still above F_3, as in (50b).

Now, let's go back to the original truing ERC set (47a), and ask the following question: which further conditions on the training ERC sequence fed to the algorithm are needed in order for it to converge to the desired final ranking (47b), besides this condition just noted that ERC 3 triggers more than $2/p_3$ updates? This question is answered by the following claim 3, proved in Appendix F.

CLAIM 3. *Consider a run of the EDRA model on the training ERC set (47a). As prescribed in 4.1.2, assume that the initial ranking values of the faithfulness constraints are null while the initial ranking values of the markedness constraints are set equal to a large positive constant θ^{init}. Furthermore, assume that the three input ERCs trigger the update (51) repeated from above, where p_1, p_2, and p_3 are the promotion amounts corresponding to the three input ERCs 1, 2, and 3 (with $0 \leq p_1, p_2, p_3 \leq 1$).*

(51) a. Demote each undominated loser-preferrer by 1;
b. promote each winner-preferrer by the promotion amount p_i.

Assume that p_3 is strictly larger than 0, so that ERC triggers some constraint promotion. Assume furthermore that ERC 3 has triggered more than $2/p_3$ updates in that run. Then, the EDRA satisfies PT's restrictiveness condition (37) provided that condition (52) holds, where α_i^T with $i = 1, 2, 3$ is the number of updates triggered by the ith input ERC in the run considered up to the earliest time T at which the current ranking vector has become consistent with at least one of the three input ERCs.

(52) $$p_1\alpha_1^T < p_2^2\alpha_2^T + p_2p_3\alpha_3^T - 1 - p_2$$

In particular, condition (52) guarantees that the unique refinement of the final ranking vector entertained at the end of the run is the desired ranking (47b). ■

Note that condition (52) requires the promotion amount p_2 to be non-null, namely it requires ERCs 2 to trigger some constraint promotion. For concreteness, suppose that the EDRA adopts the GLA update rule (16), whereby all winner-preferrers are always promoted by 1, so that $p_1 = p_2 = p_3 = 1$. Condition (52) thus simplifies as in (53). The latter condition says that ERC 1 triggers few updates relative to ERCs 2 and 3. This condition thus makes good sense: if ERC 1 triggered no updates at all, then it could be ignored altogether and F_3 would effectively be inactive.

(53) $$\alpha_1^T < \alpha_2^T + \alpha_3^T - 2$$

Condition (53) is stated in terms of this special time T, which is the earliest time at which the current ranking vector has become consistent with one of the three input ERCs in the run considered. The fact that condition (53) is stated in terms of this special time T makes this condition very handy. In fact, what is special about this time T is that the current ERC fed to the algorithm up to this time T is never consistent with the current ranking vector and thus always triggers an update. In other words, what is special about this time T is that the identity (54) holds.

(54) $$\begin{array}{c}\text{number of updates triggered by}\\ \text{the } i\text{th ERC up to time } T\end{array} = \begin{array}{c}\text{number of times the } i\text{th ERC has been}\\ \text{fed to the algorithm up to time } T\end{array}$$

The number of updates triggered by the ith ERC up to time T that appears in the left-hand side of (54) is denoted by α_i^T. Suppose that at each time the ith ERC in (47a) is sampled and fed to the model with probability $\pi_i \in [0, 1]$, with $\pi_1 + \pi_2 + \pi_3 = 1$. Then, the number of times that the ith ERC is fed to the algorithm up to time T is approximately $\pi_i \cdot T$ (with the quality of the identity increasing as T increases). Thus, the identity (54) entails the approximate identity (55) for every $i = 1, 2, 3$.

(55) $\alpha_i^T \simeq \pi_i \cdot T$

Using (55), condition (53) thus becomes $\pi_1 < \pi_2 + \pi_3 - 2/T$. If the initial ranking value θ^{init} of the markedness constraint is large, the time T that it will take for a markedness constraint to drop below a faithfulness constraint will be large, and the term $2/T$ can thus be ignored as a small correction. Condition (53) thus effectively becomes condition (56).

(56) $\pi_1 < \pi_2 + \pi_3$

In conclusion, the EDRA is restrictive provided the probability π_1 of sampling ERC 1 is smaller than the probability $\pi_2 + \pi_3$ of sampling one of the other two ERCs. We have thus obtained a complete analytical characterization of the conditions for EDRAs restrictiveness on case study (47). In particular, the case of uniform sampling corresponds to $\pi_1 = \pi_2 = \pi_3 = \frac{1}{3}$ and thus satisfies (56), explaining the success obtained in the simulation (48) with uniform sampling.

5. Conclusions

In the design of error-driven learning algorithms, we only have control over the initial grammar the algorithm starts from and over the update rule used to move from the current grammar to the updated one, whenever the current grammar is inconsistent with the current piece of data. The sequence of grammars predicted by the algorithm, and in particular the final learned grammar, thus appear to be crucially governed by the sequence of data that the algorithm is trained on. One might then worry that error-driven learners behave as a leaf in the wind of data, with little guarantees on the quality of the final learned grammar.

In this paper, I have focused on error-driven learning algorithms within the framework of OT, namely the EDRAs introduced in detail in Section 2. What is special about this specific class of error-driven learners is that the notion of consistency between the current grammar and the current piece of data has a particularly simple combinatorial nature, stated as condition (7) or equivalently (12) above. Exploiting this fact, it is possible to show that EDRAs to not really wonder around as a leaf in the wind of data but rather follow a straight path determined by the target grammar, largely independent of the stream of data. More precisely, it is possible to characterize the sequence of grammars predicted by the algorithm (and in particular the final learned grammar) based on the target grammar alone, independently from the specific way in which the data are sampled from the target grammar and organized into a training sequence. This is the idea behind the two tools for the *robust* analysis of EDRAs provided by Theorems 1 and 2 in Section 3.

In Section 4, I have illustrated the implications of these results, by looking at an important application of EDRAs to modeling the child early stage of the acquisition of phonotactics. Does an EDRAs manage to learn the phonotactics it is trained on? Or does it instead to a grammar that is not *restrictive* enough, namely that fails at recognizing as illicit many forms that are indeed illicit according to the target phonotactics? I have focused on three simple, abstract case studies for phonotactics learning, discussed in Prince and Tesar (2004). And I have shown how tools for the robust analysis of EDRAs can be used to obtain robust guarantees of their restrictiveness on these case studies.

Appendix A. Proof of Lemma 1

According to condition (12), a ranking vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ is consistent with an ERC $\mathbf{a}$ provided there is a winner-preferring constraint whose ranking value is strictly larger than the ranking value of every loser-preferring constraint. Let $W(\mathbf{a})$ and $L(\mathbf{a})$ be the sets of winner- and loser-preferring constraints relative to the ERC $\mathbf{a}$. This consistency condition can then be made explicit as in (57): the largest ranking value over winner-preferrers is strictly larger than the largest ranking value over loser-preferrers.

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

Once the notion of consistency has been made explicit as in (57), the proof of Lemma 1, repeated below, turns out to be completely straightforward.

LEMMA 1. *Let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ be the component-wise maximum between two ranking vectors $\boldsymbol{\theta}' = (\theta'_1, \ldots, \theta'_n)$ and $\boldsymbol{\theta}'' = (\theta''_1, \ldots, \theta''_n)$, i.e. $\theta_k = \max\{\theta'_k, \theta''_k\}$ for every component $k = 1, \ldots, n$. If $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$ are both consistent with an ERC* **a**, *then their component-wise maximum $\boldsymbol{\theta}$ is consistent with* **a** *as well.* ■

Proof. The proof consists of the chain of inequalities in (58). Here, I have reasoned as follows. In step (58a), I have used the definition of the ranking vector $\boldsymbol{\theta}$ as the component-wise maximum of the two ranking vectors $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$. In step (58b), I have commuted the two maximum operators. In step (58c), I have used the hypothesis that both ranking vectors $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$ are consistent with the ERC **a**, namely satisfy the corresponding conditions (57). In step (58d), I have commuted again the two maximum operators. And in step (58e), I have used again the definition of the ranking vector $\boldsymbol{\theta}$ as the component-wise maximum of the two ranking vectors $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$.

(58)
$$\begin{aligned}
\max_{h \in W(\mathbf{a})} \theta_h &\overset{(a)}{=} \max_{h \in W(\mathbf{a})} \underbrace{\max\left\{\theta'_h, \theta''_h\right\}}_{\theta_h} \\
&\overset{(b)}{=} \max\left\{\max_{h \in W(\mathbf{a})} \theta'_h, \max_{h \in W(\mathbf{a})} \theta''_h\right\} \\
&\overset{(c)}{>} \max\left\{\max_{k \in L(\mathbf{a})} \theta'_k, \max_{k \in L(\mathbf{a})} \theta''_k\right\} \\
&\overset{(d)}{=} \max_{k \in L(\mathbf{a})} \underbrace{\max\left\{\theta'_k, \theta''_k\right\}}_{\theta_k} \\
&\overset{(e)}{=} \max_{k \in L(\mathbf{a})} \theta_k
\end{aligned}$$

The chain of inequalities (58) shows that the ranking vector $\boldsymbol{\theta}$ satisfies condition (57) and is therefore consistent with the ERC **a**. □

Appendix B. Proof of Theorem 1

TS prove Theorem 1, repeated below, for the case of the demotion-only update rule (15). But their proof does not in any way hinge on the assumption that the EDRA performs no constraint promotion. Rather, it hinges on the assumption that the EDRA only demotes those loser-preferrers that need to be demoted, namely the currently *undominated* ones, that are not already ranked underneath a winner-preferrer. This Section reviews TS's proof, underscoring the fact that it indeed iextends beyond demotion-only.

THEOREM 1. *Consider a run of an EDRA (19) with a generic promotion/demotion re-ranking rule on a consistent training ERC set* **A** *(starting from null initial ranking values). Assume that the update rule only demotes undominated loser-preferrers, as in (59).*

(59) a. Demote each undominated *loser-preferring constraint by 1;*

b. promote each winner-preferring constraint by a promotion amount $p \geq 0$.

Then, the current ranking vector $\boldsymbol{\theta}$ entertained at a generic time by the EDRA satisfies the condition $\boldsymbol{\theta} \geq \overline{\boldsymbol{\theta}}_{\mathbf{A}}$, namely it is at least as large (component-wise) as the maximal consistent ranking vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ associated with the training ERC set **A** *according to (23a)* ■

Proof. As the maximal consistent vector $\overline{\boldsymbol{\theta}}_{\mathbf{A}}$ is the component-wise maximum over the ranking vectors in the consistency lattice $\mathfrak{L}_{\mathbf{A}}$, it is sufficient to prove that the current ranking vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ entertained by the EDRA is at least as large (component-wise) as any vector $\mathbf{v} = (v_1, \ldots, v_n)$ in the consistency lattice $\mathfrak{L}_{\mathbf{A}}$, as stated in (60).

(60) $\theta_k \geq v_k$

Let $\gg$ be a refinement of this ranking vector $\mathbf{v}$. Without loss of generality, assume that it is $C_1 \gg C_2 \gg \cdots \gg C_n$. Let me prove (60) by induction on k. The inequality (60) holds for $k = 1$. In fact, since the training ERC set $\mathbf{A}$ is consistent with a ranking that assigns C_1 to the top stratum, then the no ERC in $\mathbf{A}$ can have an L corresponding to C_1. Constraint C_1 is therefore never demoted and its current ranking value θ_1 never gets smaller than its initial null ranking value. Inequality (60) for $k = 1$ thus follows from the fact that v_1 is at most zero, by definition of the consistency lattice. Let me now prove that (60) holds for a generic k, using the inductive hypothesis that it holds for each h such that $h < k$. In other words, let me prove that, if the kth current ranking value θ_k has dropped down to v_k, then constraint C_k will never be demoted any further. In other words again, let me prove that, if C_k is a loser-preferring constraint according to the current ERC $\mathbf{a}$, then there is some winner-preferring constraint C_h such that the current ranking value θ_h of C_h is larger than the current ranking value θ_k of C_k. The proof consists of the chain of implications in (61).

(61) the ranking vector $\mathbf{v}$ is consistent with $\mathbf{a} \Longrightarrow$

$$\begin{aligned}
&\overset{(a)}{\Longrightarrow} \quad \text{there exists } C_h \in W(\mathbf{a}) \text{ such that } v_h > v_k \\
&\overset{(b)}{\Longrightarrow} \quad \text{there exists } C_h \in W(\mathbf{a}) \text{ such that } v_h > v_k \text{ and } h \in \{1, \ldots, k-1\} \\
&\overset{(c)}{\Longrightarrow} \quad \text{there exists } C_h \in W(\mathbf{a}) \text{ such that } v_h > \theta_k \text{ and } h \in \{1, \ldots, k-1\} \\
&\overset{(d)}{\Longrightarrow} \quad \text{there exists } C_h \in W(\mathbf{a}) \text{ such that } \theta_h > \theta_k
\end{aligned}$$

In step (61a), I have noted that, since the ranking vector $\mathbf{v}$ is by hypothesis consistent with this input ERC $\mathbf{a}$ (because it belongs to the consistency lattice, and is therefore consistent with each ERC in the training set $\mathbf{A}$), then there has got to exist some winner-preferring constraint $C_h \in W(\mathbf{a})$ that has a larger ranking value v_h than the ranking value v_k of the loser-preferring constraint C_k, by the consistency condition (57). In step (61b), I have noted that, since $C_1 \gg C_2 \gg \cdots \gg C_n$ is a refinement of $\mathbf{v}$ and since $v_h > v_k$, then it must be $h < k$. In step (61c), I have used the hypothesis that $\theta_k = v_k$. And in step (61d), I have used the inductive hypothesis that $\theta_h \geq v_h$ for every $h = 1, \ldots, k-1$. □

Appendix C. Proof of Theorem 2

In Subsection C.1, I prove a slight generalization of Theorem 2. In Subsection C.2, I then show how to derive the original formulation of the Theorem, as stated in Subsection 3.2 of the paper. Both formulations assume that the ERCs the algorithm is trained on have a unique L each. In Subsection C.3, I show that the latter restriction can be straightforwardly relaxed, at the only exponses of a more cumbersome notation.

C.1. A slight generalization of Theorem 2.

Suppose that the EDRA (19) is trained on ERCs that have a unique L each; I will relax this restrictive assumption in Subsection C.3. Suppose that it starts from the initial ranking vector $\boldsymbol{\theta}^{\text{init}} = (\theta_1^{\text{init}}, \ldots, \theta_n^{\text{init}})$. And that it makes use of the re-ranking rule (62), whereby the ith input ERC demotes its loser-preferrer by 1 and promotes its winner-preferrers by p_i.

(62) a. Demote the loser-preferring constraint by 1;
b. promote each winner-preferring constraint by p_i.

Let's focus on a certain target $\bar{\imath}$th input ERC. Let C_ℓ be its loser-preferrer and let C_h be one of its winner-preferrers, as in (63). Thus, the ranking configuration $C_h \gg C_\ell$ is sufficient for consistency with this target $\bar{\imath}$th ERC.

(63) $$\text{target } \bar{\imath}\text{th input ERC} = \begin{array}{c} \begin{array}{ccccc} \cdots & C_h & \cdots & C_\ell & \cdots \end{array} \\ \left[\begin{array}{ccccc} \ldots & \text{W} & \ldots & \text{L} & \ldots \end{array} \right] \end{array}$$

Theorem 2 in Subsection 3.2 in the paper bounds the number $\alpha^t_{\bar\imath}$ of updates triggered by this designated $\bar\imath$th ERC in a generic run up to time t as in (29), repeated in (64).[12]

$$\text{(64)}\quad \alpha^t_{\bar\imath} \;\leq\; \underbrace{\frac{\theta_\ell^{\text{init}} - \theta_h^{\text{init}}}{1+p_{\bar\imath}}}_{(a)} + \underbrace{1}_{(b)} + \underbrace{\frac{1}{1+p_{\bar\imath}} \sum_{i \in Dis^h_\ell} \delta_i \cdot \alpha^t_i}_{(c)}$$

The terms (64a) and (64b) capture the number of updates that are needed for this $\bar\imath$th ERC to establish the ranking configuration $C_h \gg C_\ell$ in case the two constraint are initially ranked in the reverse order. And the term (64c) captures the number of further updates that are needed to this $\bar\imath$th ERC to maintain this ranking configuration $C_h \gg C_\ell$ against other input ERCs that might disrupt it. A generic ith input ERC counts as a disruptor provided that it demotes the winner-preferrer C_h, as (65a); or it promotes the winner preferrer C_ℓ, as in (65b); or both, as in (65c). The collection of all these input ERCs that can disrupt the ranking configuration $C_h \gg C_\ell$ is denoted by Dis^h_ℓ. If the ith input ERC is indeed a disruptor, then its *amount of disruption* is quantified as the coefficient δ_i in (65a$'$), (65b$'$), and (65c$'$).

$$\text{(65)}\quad \begin{array}{lcccccccl}
 & & \cdots & C_h & \cdots & C_\ell & \cdots & & \\
\text{a.} & \Big[& \ldots & \text{L} & \ldots & \text{L}/e/\cancel{\text{W}} & \ldots & \Big] & \text{a}'.\ \delta_i = 1 \\
\text{b.} & \Big[& \ldots & e/\text{W}/\cancel{\text{L}} & \ldots & \text{W} & \ldots & \Big] & \text{b}'.\ \delta_i = p_i \\
\text{c.} & \Big[& \ldots & \text{L} & \ldots & \text{W} & \ldots & \Big] & \text{c}'.\ \delta_i = p_i + 1
\end{array}$$

Crucially this bound (64) only considers those input ERCs (65) that *disrupt* the ranking configuration $C_h \gg C_\ell$. And it captures the intuition that updates by disruptor ERCs buy updates by the target $\bar\imath$th ERC, as the latter has to do extra work to enforce the ranking configuration $C_h \gg C_\ell$ against these disruptors. Yet, the bound does not take into account those other input ERCs that instead *contribute* to this ranking configuration $C_h \gg C_\ell$. And it does not capture the symmetric intuition that updates by these allied ERCs should be discounted from the number of updates triggered by the target $\bar\imath$th ERC, as the latter has to do less work to enforce the ranking configuration $C_h \gg C_\ell$ thanks to the contribution of these allies.

In order to formalize this latter intuition, let's say that the generic ith input ERC *contributes* to establishing the ranking configuration $C_h \gg C_\ell$ provided that it helps promoting the winner-preferrer C_h, as in (66a); or it helps demoting the loser-preferrer C_ℓ, as in (66b); or it helps with both, as in (66c). The collection of all these ERCs that contribute to establish the ranking configuration $C_h \gg C_\ell$ is denoted by Con^h_ℓ. If the ith input ERC does indeed contribute, then its *amount of contribution* is quantified as the coefficient γ_i in (66a$'$), (66b$'$), and (66c$'$).

$$\text{(66)}\quad \begin{array}{lcccccccl}
 & & \cdots & C_h & \cdots & C_\ell & \cdots & & \\
\text{a.} & \Big[& \ldots & \text{W} & \ldots & \text{W}/e/\cancel{\text{L}} & \ldots & \Big] & \text{a}'.\ \gamma_i = p_i \\
\text{b.} & \Big[& \ldots & \text{L}/e/\cancel{\text{W}} & \ldots & \text{L} & \ldots & \Big] & \text{b}'.\ \gamma_i = 1 \\
\text{c.} & \Big[& \ldots & \text{W} & \ldots & \text{L} & \ldots & \Big] & \text{c}'.\ \gamma_i = p_i + 1
\end{array}$$

Let me now revise the initial bound (64) as in (67), by adding the new term (67d). This new term only depends on those input ERCs that contribute to establish the ranking configuration $C_h \gg C_\ell$, on the number of updates they trigger and on their amount of contribution. This new term (67d) thus captures the intuition that the updates triggered by these contributing ERCs (properly weighted by their amount of contribution) should

[12] In the paper, I omitted the superscript "t" to indicate that I am considering the number of updates up to a certain time t. I use the superscript in this Appendix, as I will have to distinguish between the numbers of updates triggered up to two different times t and t'.

be discounted from the updates that the target $\bar\imath$th ERC is allowed to trigger in order to establish the ranking configuration $C_h \gg C_\ell$. Because of this third term (67d), the bound (67) is tighter than the original bound (64).

$$(67)\quad \alpha_{\bar\imath}^t \leq \underbrace{\frac{\theta_\ell^{\text{init}} - \theta_h^{\text{init}}}{1+p_{\bar\imath}}}_{(a)} + \underbrace{1}_{(b)} + \underbrace{\frac{1}{1+p_{\bar\imath}} \sum_{i\in Dis_\ell^h} \delta_i \cdot \alpha_j^t}_{(c)} - \underbrace{\frac{1}{1+p_{\bar\imath}} \sum_{i\in Con_\ell^h} \gamma_i \cdot \alpha_i^t}_{(d)}$$

The following variant of Theorem 2 says that the sharper bound (67) indeed holds, but under the additional assumption that the target $\bar\imath$th ERC is the one that triggers the update at time t. In Subsection C.2, I explain in detail why this additional assumption is needed. And I derive as a special case the fact that the weaker bound (64) holds also without this extra assumption, as stated in the original formulation of Theorem 2 in Subsection 3.2.

THEOREM 2 (SLIGHTLY GENERALIZED). *Consider a run of the EDRA (19) starting from the initial ranking vector* $\boldsymbol{\theta}^{\text{init}} = (\theta_1^{\text{init}}, \ldots, \theta_n^{\text{init}})$*, with the general update rule (62), on a training set of ERCs that have a unique* L *each. Suppose that the input ERC that triggers the update at a certain time* t *is the* $\bar\imath$*th input ERC (63), whose loser-preferrer is* C_ℓ*. Then, the number of updates* $\alpha_{\bar\imath}^t$ *triggered by this* $\bar\imath$*th input ERC up to this time* t *can be bound as in (67) for each one of its winner-preferrers* C_h*.* ■

Proof. As the $\bar\imath$th input ERC is the one that triggers the update at time t, then the number $\alpha_{\bar\imath}^{t-1}$ of updates triggered by it up to time $t-1$ and the number $\alpha_{\bar\imath}^t$ of updates triggered by it up to time t differ only by 1, as stated in (68a). Furthermore, the current ranking vector $\boldsymbol{\theta}^{t-1} = (\theta_1^{t-1}, \ldots, \theta_n^{t-1})$ at time $t-1$ cannot have been consistent with the $\bar\imath$th input ERC (63) in order for it to trigger an update. This means in particular that the current ranking value θ_h^{t-1} of the winner-preferring constraint C_h at time $t-1$ was not larger than the current ranking value θ_ℓ^{t-1} of the loser-preferring constraint C_ℓ at time $t-1$, as stated in (68b). Finally, none of the other ith input ERCs has triggered an update at time t. Thus the number α_i^{t-1} of updates triggered by any other ith input ERC up to time $t-1$ coincides with the number α_i^t of updates it has triggered up to time t, as stated in (68c).

(68) a. $\alpha_{\bar\imath}^t = \alpha_{\bar\imath}^{t-1} + 1$;
b. $\theta_h^{t-1} \leq \theta_\ell^{t-1}$;
c. $\alpha_i^t = \alpha_i^{t-1}$ for any ith input ERC different from the target $\bar\imath$th one.

For any constraint C_k, let W(k) and L(k) be the sets of those input ERCs where the constraint C_k is winner- and loser-preferring respectively, as stated in (69).

$$(69)\quad \begin{aligned} \mathrm{W}(k) &= \text{set of input ERCs where } C_k \text{ is winner-preferring;} \\ \mathrm{L}(k) &= \text{set of input ERCs where } C_k \text{ is loser-preferring.} \end{aligned}$$

Using the general expression (26), the ranking value θ_h^{t-1} of constraint C_h at time $t-1$ can be expressed as in (70a), namely as the sum of four factors: (a) its initial ranking value θ_h^{init}; (b) the contribution of the $\bar\imath$th input ERC, which is the number $\alpha_{\bar\imath}^{t-1}$ of updates triggered by that ERC up to time $t-1$ multiplied by the corresponding promotion amount $p_{\bar\imath}$, as that ERC promotes C_h by $p_{\bar\imath}$; (c) the contribution of those other ERCs in W(h) where C_h is winner-preferrer, which is the sum of the numbers of updates α_i^{t-1} that each such ERC has triggered up to time $t-1$ each multiplied by the corresponding promotion amount p_i, as each such ERC promotes C_h by p_i; (d) the contribution of those ERCs in L(h) where C_h is loser-preferrer, which amounts to subtracting the sum of the number of updates α_i^{t-1} that each such ERC has triggered up to time $t-1$, as each such ERC demotes C_h by 1. The ranking value θ_ℓ^{t-1} of constraint C_ℓ at time $t-1$ can be expressed analogously, as in (70b).

$$\text{(70)}\quad \text{a.}\quad \theta_h^{t-1} = \underbrace{\theta_h^{\text{init}}}_{(a)} + \underbrace{p_{\bar{\imath}}\alpha_{\bar{\imath}}^{t-1}}_{(b)} + \underbrace{\sum_{\substack{i\,\in\,\mathrm{W}(h)\\ i\,\neq\,\bar{\imath}}} p_i\alpha_i^{t-1}}_{(c)} - \underbrace{\sum_{i\in\mathrm{L}(h)} \alpha_i^{t-1}}_{(d)}$$

$$\text{b.}\quad \theta_\ell^{t-1} = \theta_\ell^{\text{init}} - \alpha_{\bar{\imath}}^{t-1} + \sum_{i\in\mathrm{W}(\ell)} p_i\alpha_i^{t-1} - \sum_{\substack{i\,\in\,\mathrm{L}(\ell)\\ i\,\neq\,\bar{\imath}}} \alpha_i^{t-1}$$

Using expressions (70a) and (70b) for the ranking values θ_h^{t-1} and θ_ℓ^{t-1} of constraints C_h and C_ℓ at time $t-1$, the inequality (68b) can be rewritten as in (71). This is an inequality in the numbers of updates triggered by the various input ERCs up to time $t-1$ (plus, of course, the initial raking values).

$$\text{(71)}\quad \theta_h^{\text{init}} + p_{\bar{\imath}}\alpha_{\bar{\imath}}^{t-1} + \sum_{\substack{i\,\in\,\mathrm{W}(h)\\ i\,\neq\,\bar{\imath}}} p_i\alpha_i^{t-1} - \sum_{i\in\mathrm{L}(h)} \alpha_i^{t-1} \leq$$

$$\leq \theta_\ell^{\text{init}} - \alpha_{\bar{\imath}}^{t-1} + \sum_{i\in\mathrm{W}(\ell)} p_i\alpha_i^{t-1} - \sum_{\substack{i\,\in\,\mathrm{L}(\ell)\\ i\,\neq\,\bar{\imath}}} \alpha_i^{t-1}$$

By (68a), I can replace $\alpha_{\bar{\imath}}^{t-1}$ with $\alpha_{\bar{\imath}}^{t} - 1$ in (71). By (68b), I can furthermore replace α_i^{t-1} with α_i^t for any i different from $\bar{\imath}$. Reordering, I thus end up with the inequality (72). This is an inequality in the numbers of updates triggered by the various input ERCs up to time t.

$$\text{(72)}\quad \alpha_{\bar{\imath}}^{t} \leq \underbrace{\frac{\theta_\ell^{\text{init}} - \theta_h^{\text{init}}}{1+p_{\bar{\imath}}}}_{(a)} + \underbrace{1}_{(b)} + \underbrace{\frac{1}{1+p_{\bar{\imath}}}\left(\sum_{i\in\mathrm{W}(\ell)} p_i\alpha_i^t + \sum_{i\in\mathrm{L}(h)} \alpha_i^t\right)}_{(c)}$$

$$\underbrace{- \frac{1}{1+p_{\bar{\imath}}}\left(\sum_{\substack{i\,\in\,\mathrm{W}(h)\\ i\,\neq\,\bar{\imath}}} p_i\alpha_i^t + \sum_{\substack{i\,\in\,\mathrm{L}(\ell)\\ i\,\neq\,\bar{\imath}}} \alpha_i^t\right)}_{(d)}$$

To conclude the proof, I just need to show that the terms (72c) and (72d) coincide with the terms (67c) and (67d). Look for example at (72c). This term consists of two summations: one over the set $\mathrm{W}(\ell)$ of ERCs where constraint C_ℓ is winner-preferrer and another one over the set $\mathrm{L}(h)$ of ERCs where constraint C_h is loser-preferrer. Thus, let me distinguish the three cases (73).

(73) The ith input ERC:

a. belongs to $\mathrm{L}(h)$ but not to $\mathrm{W}(\ell)$;

b. belongs to $\mathrm{W}(\ell)$ but not to $\mathrm{L}(h)$;

c. belongs to both $\mathrm{W}(\ell)$ and $\mathrm{L}(h)$.

The ith ERC falls into case (73a) provided it is a disruptor of type (65a). And its number of updates α_i^t then appears in the term (72c) multiplied by the coefficient 1, which is indeed the amount of disruption (65a′) caused by this type of disruptors. The ith ERC falls into case (73b) provided it is a disruptor of type (65b). And its number of updates α_i^t then appears in the term (72c) multiplied by the coefficient p_i, which is indeed the amount of disruption (65b′) caused by this type of disruptors. Finally, the ith ERC falls into the case (73c) provided it is a disruptor of type (65c). And its number of updates α_i^t then appears in the term (72c) twice, once multiplied by the coefficient 1 and another time multiplied by the coefficient p_i, so that in the end it is multiplied by the coefficient $1+p_i$, which is indeed the amount of disruption (65c′) caused by this type of disruptors.

In the end, the term (72c) coincides with the term (67c). Analogous considerations show that also the term (72d) coincides with the term (67d), completing the proof. □

C.2. **Back to the original formulation of Theorem 2.** The *original* formulation of Theorem 2 in Subsection 3.2 of the paper and the *revised* alternative formulation presented above provide bounds on the number of updates $\alpha^t_{\bar{\imath}}$ triggered by the $\bar{\imath}$th target ERC up to time t. There are two differences between these two formulations. The *first* difference is that the bound (67) provided by the revised formulation of the theorem is tighter than the bound (64) provided by the original formulation, as the former discounts the number of updates performed by allied ERCs, through the additional term (67d). Yet, this stronger bound comes at a price. In fact, the *second* difference between the two formulations is that the original formulation of the theorem guarantees that the weaker bound (64) holds at time t no matter how we choose the target $\bar{\imath}$th input ERC, while the revised formulation of the theorem guarantees that the sharper bound (67) holds at time t only for that $\bar{\imath}$th input ERC that has triggered the last update. Let me explain why the sharper bound (67) needs this restriction and why the weaker bound (64) does not, thus in particular establishing the latter bound as well.

Suppose we want to bound the number of updates triggered by a certain target $\bar{\imath}$th input ERC up to a certain time t. Let t' be the last time before time t when this $\bar{\imath}$th ERC has triggered an update. As t' is the last time that this ERC has triggered an update, the number $\alpha^{t'}_{\bar{\imath}}$ of updates it has triggered up to time t' and the number $\alpha^t_{\bar{\imath}}$ of updates it has triggered up to time t differ by 1, as stated in (74a). Furthermore, the current ranking vector $\boldsymbol{\theta}^{t'}$ at the time t' when this ERC has triggered an update must have been inconsistent with this ERC. Assume again that this $\bar{\imath}$th ERC looks like (63). Then, the current ranking value $\theta^{t'}_h$ of the winner-preferrer C_h at time t' cannot have been larger than the current ranking value $\theta^{t'}_\ell$ of the loser-preferrer C_ℓ, as stated in (74b). Finally, as time t follows time t', then the number of updates triggered up to time t by any ERC can only be larger than or equal to the number of updates triggered by that ERC up to time t', as stated in (74c). Note that (74a) and (74b) coincide with (68a) and (68b) respectively, only with $t-1$ replaced by a generic time t', as I have dropped the hypothesis that it is the $\bar{\imath}$th ERC that has triggered the last update. Yet, (74c) is much weaker than (68c): as the times t' and t can be far apart, other ERCs different from the target $\bar{\imath}$th ERC might have triggered various of updates in between those two times.

(74) a. $\alpha^t_{\bar{\imath}} = \alpha^{t'}_{\bar{\imath}} + 1$

b. $\theta^{t'}_h \leq \theta^{t'}_\ell$

c. $\alpha^t_i \geq \alpha^{t'}_i$ for any ith input ERC different from the target $\bar{\imath}$th one.

The derivation of the inequality (71) above only relied on the two facts (68a) and (68b). By reasoning in the same way using the two analogous facts (74a) and (74b), I thus obtain the inequality (75), which is of course analogous to (71), only with t' in place of $t-1$.

(75)
$$\begin{aligned} &\theta^{\text{init}}_h + p_{\bar{\imath}}\alpha^{t'}_{\bar{\imath}} + \underbrace{\sum_{\substack{i\,\in\,\mathrm{W}(h)\\ i\,\neq\,\bar{\imath}}} p_i\alpha^{t'}_i}_{(*)} - \sum_{i\in\mathrm{L}(h)} \alpha^{t'}_i \leq \\ &\leq\ \theta^{\text{init}}_\ell - \alpha^{t'}_{\bar{\imath}} + \sum_{i\in\mathrm{W}(\ell)} p_i\alpha^{t'}_i - \underbrace{\sum_{\substack{i\,\in\,\mathrm{L}(\ell)\\ i\,\neq\,\bar{\imath}}} \alpha^{t'}_i}_{(**)} \end{aligned}$$

At this point, I have a bound (75) stated in terms of the number of updates $\alpha^{t'}_{\bar{\imath}}, \alpha^{t'}_i$ triggered by the various input ERCs up to time t'. But I want a bound at time t. In the preceding proof, I could easily convert the bound (71) at time $t-1$ into a bound at time t, by exploiting (68a) and (68c). Now, I cannot as easily go from the bound (75) at time t' to a bound at time t, because I only have (74c). This explains why the preceding proof fails in this general case and why I need the assumption that t' is indeed $t-1$, namely that the target $\bar{\imath}$th ERC is the one that triggers the update at time t.

In order to get a bound at time t out of the bound (75) at time t' without making further assumptions on t', I need to first weaken the latter bound, by dropping the terms (*) and (**). This yields the weakened inequality (76) that has the crucial property that all coefficients α_j are multiplied by a positive constant.

$$\text{(76)}\quad \alpha_{\bar{\imath}}^{t'} \leq \frac{\theta_\ell^{\text{init}}-\theta_h^{\text{init}}}{1+p_{\bar{\imath}}} + \frac{1}{1+p_{\bar{\imath}}}\left(\sum_{j\in \mathrm{W}(\ell)} p_j\alpha_j^{t'} + \sum_{j\in \mathrm{L}(h)} \alpha_j^{t'}\right)$$

The latter property allows me to further weaken the inequality (76) by replacing the number of updates $\alpha_i^{t'}$ triggered by the ith ERC up to time t' with the number of updates α_i^t it has triggered up to time t, as the former is smaller than or equal to the latter, by (74c). Furthermore, I can replace $\alpha_{\bar{\imath}}^{t'}$ with $\alpha_{\bar{\imath}}^t - 1$, by (74a). In the end, I thus obtain the inequality (77).

$$\text{(77)}\quad \alpha_{\bar{\imath}}^{t} \leq \underbrace{\frac{\theta_\ell^{\text{init}}-\theta_h^{\text{init}}}{1+p_{\bar{\imath}}}}_{(a)} + \underbrace{1}_{(b)} + \underbrace{\frac{1}{1+p_{\bar{\imath}}}\left(\sum_{j\in \mathrm{W}(\ell)} p_j\alpha_j^{t} + \sum_{j\in \mathrm{L}(h)} \alpha_j^{t}\right)}_{(c)}$$

Again, the term (77c) can be easily seen to coincide with the term (64c), so that the inequality (77) coincides with the bound (64).

C.3. **Relaxing the assumption that training ERCs have a unique L each.** So far, I have sticked to the assumption that the input ERCs fed to the EDRA throughout training all have a unique L each. Now I want to relax this assumption. The additional difficulty that comes along is as follows. If the ith ERC has a unique loser-preferrer, that loser-preferrer is demoted by 1 every time the ith ERC triggers an update. Thus, the number of updates α_i^t triggered by the ith ERC coincides with the number of times that loser-preferrer is demoted by the ith ERC. If instead the ith ERC has two loser-preferrers, then some updates by the ith ERC can demote one of the two loser-preferrers and some other updates can demote the other (or both), depending on which one of the two (or both) is currently undominated. Thus, the number of updates α_i^t triggered by the ith ERC does not coincide anymore with the number of times its loser-preferrers are demoted by the ith ERC. But this difficulty can be straightforwardly overcome, at the expenses of only a slightly more cumbersome notation. Let m be the total number of input ERCs. Suppose that the ith ERC has ℓ_i loser-preferrers $C_{k_1}, \ldots, C_{k_{\ell_i}}$. Let $\mathcal{C}_1^i, \ldots, \mathcal{C}_{2^{\ell_i}-1}^i$ be all $2^{\ell_i}-1$ non-empty subsets of the set $\{C_{k_1}, \ldots, C_{k_{\ell_i}}\}$ of loser-preferrers. For every such subset $\mathcal{C}_j^i$, let $\alpha_{i,j}^t$ be the number of updates triggered by this ith ERC up to time t because all and only the loser-preferrers in the set $\mathcal{C}_j^i$ were currently undominated. Obviously, $\alpha_i^t = \sum_j \alpha_{i,j}^t$. The current ranking vector can then be expressed as in (26) through these non-negative coefficients $\alpha_{i,j}^t$, rather than the old coefficients α_i^t. And the reasoning developed in this Subsection trivially extends to analogous bounds stated in terms of these refined coefficients $\alpha_{i,j}^t$.

Appendix D. Proof of claim 1

This Subsection offers a proof of claim 1 repeated below, that provides a robust analysis of promotion-demotion EDRAs on PT's first case study from the perspective of restrictiveness.

Claim 1. *Consider a run of the EDRA model on the training ERC set (78a).*

(78) *a.*

$$\begin{array}{r|cccc|} & F_1 & F_2 & M_1 & M_2 \\ \text{ERC 1} & & & \text{W} & \text{L} \\ \text{ERC 2} & \text{W} & & \text{L} & \text{W} \end{array}$$

b.

$$\begin{tikzcd} F_1 \arrow[d, dash] \\ M_1 \arrow[d, dash] \\ M_2 \arrow[d, dash] \\ F_2 \end{tikzcd}$$

As prescribed in 4.1.2, assume that the initial ranking values of the faithfulness constraints are null while the initial ranking values of the markedness constraints are set equal to a large positive constant θ^{init}. Furthermore, assume that the two input ERCs trigger the update (79) repeated from above, where p_1 and p_2 are the promotion amounts corresponding to the two input ERCs 1 and 2 ($p_1, p_2 \geq 0$).

(79) *a. Demote each undominated loser-preferrer by 1;*

b. promote each winner-preferrer by the promotion amount p_i.

As long as the promotion amount p_2 of ERC 2 is larger than zero, the EDRA always satisfies the PT's restrictiveness condition (37), no matter how the two input ERCs are sampled and fed to the algorithm. In particular, the unique refinement of the final ranking vector is always (78b). ■

Proof. The target ranking (78b) has R-measure 2, as two markedness constraints are ranked underneath the faithfulness constraint F_1 and none underneath the faithfulness constraint F_2. Assume by contradiction that the claim were false. This means that we can construct a run of the EDRA such that the current ranking vector entertained by the model at some time t admits a refinement with R-measure 3. Namely a refinement that ranks one of the two markedness constraints M_1 or M_2 underneath the faithfulness constraint F_2. Thus, the current ranking value of M_1 or M_2 at time t cannot be larger than the current ranking value of the faithfulness constraint F_2. As F_2 starts out with a null initial ranking value and is never promoted (because inactive), it stays put at its initial null ranking value. And the condition that the current ranking value $\theta^t_{M_1}$ of M_1 or the current ranking value $\theta^t_{M_2}$ of M_2 at time t is not larger than the current ranking value $\theta^t_{F_2}$ of F_2 means that either condition (80a) or (80b) holds.

(80) a. $\theta^t_{M_1} \leq 0$ b. $\theta^t_{M_2} \leq 0$

Let me show that both conditions (80) are impossible.

The intuitive idea for why condition (80a) is impossible is as follows: the markedness constraint M_1 is only demoted by ERC 2; that ERC cannot trigger too many updates; hence M_1 cannot drop all the way down to zero. Here are the details. Theorem 2 applied to ERC 2 and its winner-preferrer F_1 yields the bound (81) on the number of updates α^t_2 it can trigger up to time t.

(81) $$\alpha^t_2 \leq \frac{\theta^{\text{init}}}{p_2 + 1} + 1 + \frac{1}{p_2 + 1} p_1 \alpha^t_1$$

The chain of inequalities in (82) thus shows that the current ranking value $\theta^t_{M_1}$ of the markedness constraint M_1 is always above zero.

(82)
$$\begin{aligned}\theta^t_{M_1} &\overset{(a)}{=} \theta^{\text{init}} + p_1\alpha_1 - \alpha_2\\ &\overset{(b)}{\geq} \theta^{\text{init}} + p_1\alpha_1 - \left(\frac{\theta^{\text{init}}}{p_2+1} + 1 + \frac{1}{p_2+1}p_1\alpha_1^t\right)\\ &= \frac{p_2}{p_2+1}\theta^{\text{init}} - 1 + \underbrace{\frac{p_1(p_2+1)-p_1}{p_2+1}\alpha_1^t}_{(*)}\\ &\overset{(c)}{\geq} \frac{p_2}{p_2+1}\theta^{\text{init}} - 1\\ &\overset{(d)}{>} 0\end{aligned}$$

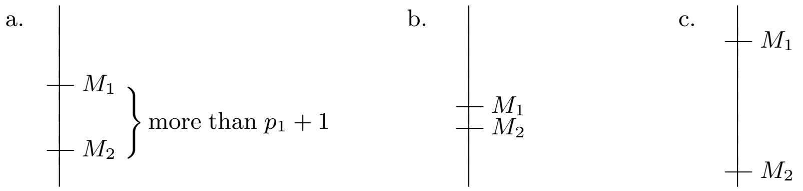
In step (a), I have expressed the current ranking value $\theta^t_{M_1}$ of M_1 at time t as in (26), namely as the sum of three terms: its initial ranking value θ^{init}; the contribution $p_1\alpha_1^t$ of ERC 1; and the contribution $-\alpha_2^t$ of ERC 2. In step (b), I have lower bounded using the inequality (81). In step (c), I have lower bounded by dropping the term (*), which is always non-negative. Finally, step (d) holds, provided that p_2 is not null (namely ERC 2 triggers some promotion) and that θ^{init} is large enough (namely $\theta^{\text{init}} > \frac{p_2+1}{p_2}$, which makes sense because of the hypothesis that $p_2 \neq 0$).

The intuitive idea for why condition (80b) is impossible is as follows: M_1 and M_2 start out with the same initial ranking value; the markedness constraint M_2 is only demoted by ERC 1; and M_1 is a winner-preferrer in ERC 1; thus M_2 cannot drop far away from M_1; as M_1 is always above zero (as we just saw), M_2 is thus above zero as well. Here are the details. To start, let me show that the inequality (83) holds at any time t: the markedness constraint M_2 can never drop below the other markedness constraint M_1 by more than $1 + p_1$.

(83) $\theta^t_{M_2} \geq \theta^t_{M_1} - (1 + p_1)$.

In fact, suppose by contradiction that (83) were false. Thus, at some time t, M_2 is ranked below M_1 and they are separated by more than $1 + p_1$, as in (84a).

(84)



Which ERC has triggered the last update that lead to this ranking configuration (84a)? It cannot have been ERC 1. In fact, ERC 1 promotes M_1 by p_1 and demotes M_2 by 1. In other words, ERC 1 increases the separation between the winner-preferrer M_1 and the loser-preferrer M_2 by $1 + p_1$. Thus, in order for M_1 to outrank M_2 by more than $1 + p_1$ after the update as in (84a), it ought to have been the case that M_1 was already ranked above M_2 before the update, as in (84b). But in the latter configuration (84b), ERC 1 triggers no update, as the winner-preferrer M_1 is already ranked above the loser-preferrer M_2. Thus, it must have been ERC 2 that triggered the last update that lead to the ranking configuration (84a). ERC 2 demotes M_1 by 1 and promotes M_2 by p_2. Thus, in order for an update by ERC 2 to lead to the ranking configuration (84a), the starting point should have been (84c). In the latter configuration, M_2 is ranked again below M_1 with a separation larger than the separation they have in (84a). I can thus repeat the same reasoning once more, and conclude that it was again ERC 2 that triggered the last update that lead to the ranking configuration (84c). And so on, contradicting the fact that M_1 and M_2 start out equally ranked.

The chain of inequalities in (85) finally shows that the current ranking value $\theta^t_{M_2}$ of the markedness constraint M_2 is always larger than zero.

$$(85)\qquad \theta^t_{M_2} \overset{(a)}{\geq} \theta^t_{M_1} - (1+p_1) \overset{(b)}{\geq} \left(\frac{p_2}{p_2+1}\theta^{\text{init}} - 1\right) - 1 - p_1 \overset{(c)}{>} 0$$

In step (a), I have used the bound (83) just proved. In step (b), I have used the bound computed in (82c) for the ranking value $\theta^t_{M_1}$ at an arbitrary time t. Finally, step (c) holds, provided that θ^{init} is large enough (namely $\theta^{\text{init}} > (2+p_1)\frac{p_2+1}{p_2}$).

As both conditions (80) are impossible, neither markedness constraint M_1 and M_2 can ever drop below F_2. Hence, the current ranking vector can never have R-measure smaller than 2. And the algorithm thus always satisfies PT's restrictiveness condition (37). At convergence, the current ranking vector is consistent with the training ERC set (78a). Consistency with ERC 1 entails $M_1 \gg M_2$ for every refinement $\gg$ of the final ranking vector. And consistency with ERC 2 thus entails $F_1 \gg M_1$. Since furthermore M_1 and M_2 never drop below F_2, then we also have $M_1, M_2 \gg F_2$. In conclusion, the only refinement $\gg$ of the final ranking vector is (78b), completing the proof of the claim. □

Appendix E. Proof of claim 2

This Subsection offers a proof of claim 2 repeated below, that provides a robust analysis of promotion-demotion EDRAs on PT's second case study from the perspective of restrictiveness.

Claim 2. *Consider a run of the EDRA mode; on the training ERC set (86a).*

(86) *a.*

$$\begin{array}{c} \\ \text{ERC 1} \\ \text{ERC 2} \end{array}\begin{array}{c} \begin{array}{cccc} F_1 & F_2 & M_1 & M_2 \end{array} \\ \left[\begin{array}{cccc} \text{W} & & \text{L} & \\ & \text{W} & \text{W} & \text{L} \end{array}\right] \end{array}$$

b.

$$\begin{tikzcd} F_1 \arrow[d, dash] \\ M_1 \arrow[d, dash] \\ M_2 \arrow[d, dash] \\ F_2 \end{tikzcd}$$

As prescribed in 4.1.2, assume that the initial ranking values of the faithfulness constraints are null while the initial ranking values of the markedness constraints are set equal to a large positive constant θ^{init}. Furthermore, assume that the two input ERCs trigger the update (87) repeated from above, where p_1 and p_2 are the promotion amounts corresponding to the two input ERCs 1 and 2 ($p_1, p_2 \geq 0$).

(87) *a. Demote each undominated loser-preferrer by 1;*

b. promote each winner-preferrer by the promotion amount p_i.

Assume that the promotion amount p_1 of ERC 1 satisfies the strict inequality (88), which entails in particular that p_1 is not null, namely that ERC 1 triggers some constraint promotion.

$$(88)\qquad p_1 > \frac{p_2}{1+p_2}$$

Then, the EDRA always satisfies PT's restrictiveness condition (37), no matter how the two input ERCs are sampled and fed to the algorithm. In particular, the unique refinement of the final ranking vector is always (86b). ■

Proof. Assume by contradiction that the claim were false. By reasoning as in the proof of the preceding claim 1, this contradictory assumption means that it is possible to construct a run of the EDRA such that the current ranking vector at some time t ranks either M_1 or M_2 underneath F_2, so that at least one of the two conditions (89a) or (89b) holds.

(89) a. $\theta^t_{M_1} \leq \theta^t_{F_2}$ b. $\theta^t_{M_2} \leq \theta^t_{F_2}$

Using the general identity (26), the current ranking values of M_1, M_2 and F_2 at time t can be expressed as in (90) in terms of the numbers of updates α_1^t and α_2^t triggered by the two input ERCs 1 and 2 up to time t. The identity (90a) holds because the faithfulness constraint F_2 starts out with a null initial ranking value and is promoted by p_2 every time ERC 2 triggers an update. The identity (90b) holds because the markedness constraint M_1 starts from the positive initial ranking value θ^{init}, is demoted by 1 whenever ERC 1 triggers an update and is promoted by p_2 whenever ERC 2 triggers an update. Finally, the identity (90c) holds because the markedness constraint M_2 starts from θ^{init} and is demoted by 1 whenever ERC 2 triggers an update.

$$(90)\quad \begin{array}{lrcl} \text{a.} & \theta^t_{F_2} & = & p_2\alpha_2^t \\ \text{b.} & \theta^t_{M_1} & = & \theta^{\text{init}} - \alpha_1^t + p_2\alpha_2^t \\ \text{c.} & \theta^t_{M_2} & = & \theta^{\text{init}} - \alpha_2^t \end{array}$$

Using (90), the two conditions (89) on the current ranking values can then be restated as the two conditions (91) on the current number of updates α_1^t and α_2^t triggered by the input ERCs 1 and 2, respectively.

$$(91)\quad \text{a. } \alpha_1^t \geq \theta^{\text{init}} \qquad\qquad \text{b. } \alpha_2^t \geq \frac{1}{1+p_2}\theta^{\text{init}}$$

I the rest of the proof, I use theorem 2 in order to show that the two input ERCs cannot perform the too many updates required to satisfy conditions (91).

By applying Theorem 2 to ERC 1, I get the bound (92) on the number α_1^t of updates it can trigger up to time t. And by applying the Theorem to the two winner-preferrers of ERC 2, I get the two bounds (93).

$$(92)\quad \alpha_1^t \leq \frac{\theta^{\text{init}}}{1+p_1} + 1 + \frac{1}{1+p_1}p_2\alpha_2^t$$

$$(93)\quad \text{a. } \alpha_2^t \leq \frac{\theta^{\text{init}}}{1+p_2} + 1 \qquad\qquad \text{b. } \alpha_2^t \leq 1 + \frac{1}{1+p_2}\alpha_1^t$$

The chain of inequalities (94) shows that condition (91a) cannot hold, namely that ERC 1 cannot trigger the too many updates needed to bring M_1 below F_2.

$$(94)\quad \begin{array}{rcl} \alpha_1^t & \overset{(a)}{\leq} & \dfrac{\theta^{\text{init}}}{1+p_1} + 1 + \dfrac{1}{1+p_1}p_2\alpha_2^t \\ & \overset{(b)}{\leq} & \dfrac{\theta^{\text{init}}}{1+p_1} + 1 + \dfrac{1}{1+p_1}p_2\left(\dfrac{\theta^{\text{init}}}{1+p_2} + 1\right) \\ & \overset{(c)}{=} & \underbrace{\left(\dfrac{1}{1+p_1} + \dfrac{1}{1+p_1}\dfrac{p_2}{1+p_2}\right)}_{A}\theta^{\text{init}} + \underbrace{\dfrac{1+p_1+p_2}{1+p_1}}_{B} \\ & \overset{(d)}{<} & \theta^{\text{init}} \end{array}$$

In step (a), I have used the inequality (92). In step (b), I have used the inequality (93a). The hypothesis (88) entails that the coefficient A in (94c) is strictly smaller than 1. Thus, step (c) holds, provided the initial ranking value θ^{init} of the markedness constraints is large enough (namely provided that $\theta^{\text{init}} > \frac{B}{1-A}$, which makes sense because of the fact that $0 \leq A < 1$).

The chain of inequalities (95) shows that condition (91b) cannot hold, namely that ERC 2 cannot trigger the too many updates needed to bring M_2 below F_2.

$$(95)\qquad \begin{aligned} \alpha_2^t &\overset{(a)}{\leq} 1+\frac{1}{1+p_2}\alpha_1^t \\ &\overset{(b)}{\leq} 1+\frac{1}{1+p_2}\Big(A\theta^{\text{init}}+B\Big) \\ &= \frac{1}{1+p_2}\Big(A\theta^{\text{init}}+1+B\Big) \\ &\overset{(c)}{<} \frac{1}{1+p_2}\theta^{\text{init}} \end{aligned}$$

In step (a), I have used the inequality (93b). In step (c), I have used the inequality (94c). As the coefficient A is already smaller than 1, step (d) holds provided that the initial ranking value θ^{init} of the markedness constraints is large enough (namely provided that $\theta^{\text{init}} > \frac{1+B}{1-A}$).

As both conditions (89) are impossible, neither markedness constraint M_1 or M_2 can ever drop below F_2. Furthermore, consistency with ERC 1 entails that at convergence we have $F_1 \gg M_1$ for any refinement $\gg$ of the final ranking vector. As M_1 and M_2 never drop below F_2, then $M_1, M_2 \gg F_2$. And consistency with ERC 2 thus entails that at convergence $M_1 \gg M_2$. In conclusion, the only refinement of the final ranking vector is (86b), completing the proof of the claim. □

Appendix F. Proof of claim 3

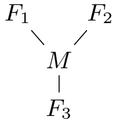
This Subsection offers a proof of claim 3 repeated below, that provides a robust analysis of promotion-demotion EDRAs on PT's third case study from the perspective of restrictiveness.

Claim 3. *Consider a run of the EDRA model on the training ERC set (96a).*

(96) *a.*

$$\begin{array}{r c} & \begin{array}{cccc} F_1 & F_2 & F_3 & M \end{array} \\ \begin{array}{r} \text{ERC 1} \\ \text{ERC 2} \\ \text{ERC 3} \end{array} & \left[\begin{array}{cccc} \text{W} & & \text{W} & \text{L} \\ & \text{W} & & \text{L} \\ \text{W} & & & \text{L} \end{array}\right] \end{array}$$

b.



As prescribed in 4.1.2, assume that the initial ranking values of the faithfulness constraints are null while the initial ranking value of the markedness constraint is set equal to a large positive constant θ^{init}. Furthermore, assume that the three input ERCs trigger the update (97) repeated from above, where p_1, p_2, and p_3 are the promotion amounts corresponding to the three input ERCs 1, 2, and 3 (with $0 \leq p_1, p_2, p_3 \leq 1$).

(97) *a. Demote each undominated loser-preferrer by 1;*
b. promote each winner-preferrer by the promotion amount p_i.

Assume that p_3 is strictly larger than 0, so that ERC triggers some constraint promotion. Assume furthermore that ERC 3 has triggered more than $2/p_3$ updates in the run considered. Then, the EDRA satisfies the PT's restrictiveness condition (37) provided that condition (98) holds, where α_i^T with $i = 1,2,3$ is the number of updates triggered by the ith input ERC in the run considered up to the earliest time T at which the current ranking vector has become consistent with at least one of the three input ERCs.

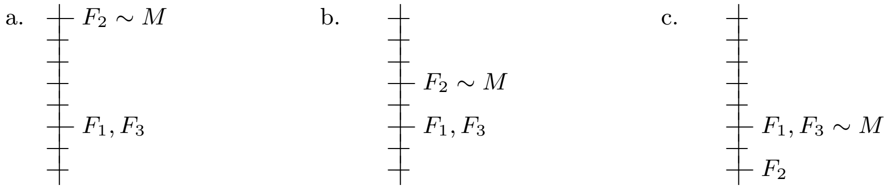
(98) $p_1\alpha_1^T < p_2^2\alpha_2^T + p_2p_3\alpha_3^T - 1 - p_2$

In particular, condition (98) guarantees that the unique refinement of the final ranking vector entertained at the end of the run is the desired ranking (96b). ■

Proof. Throughout learning, the faithfulness constraints raise and the markedness constraint drops. Hence, it is enough to show that PT's restrictiveness condition (37) holds at the end of learning, in order for it to hold at any time throughout the run. The current ranking value of the faithfulness constraint F_1 can never be smaller than the current ranking value of the faithfulness constraint F_3, as the former is promoted by both ERCs 1 and 3 while the latter only by ERC 1. Thus two scenarios need to be considered, depending on whether at time T it happens that F_1 and F_3 have the same ranking value or else F_1 is ranked above F_3.

To start, consider the former scenario, where F_1 and F_3 have the same ranking value at time T. Three cases need to be distinguished, depending on whether F_2 is ranked well above $F_1 = F_3$ at time T, as in (99a); or F_2 barely outranks $F_1 = F_3$ at time T, as in (99b); or else F_2 is ranked below $F_1 = F_3$ at time T, as in (99c). Let me consider each of these three cases in turn.

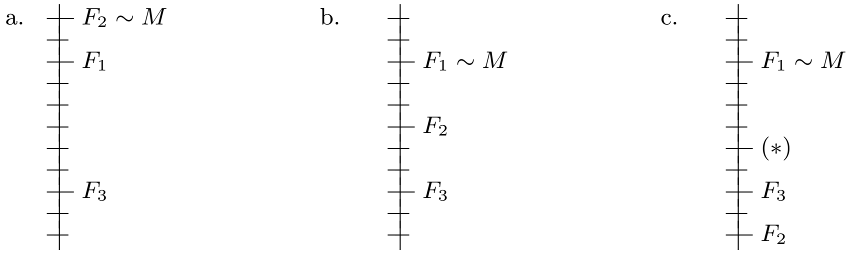
(99)



In case (99a), F_2 is top ranked at the earliest time T when the current ranking vector has become consistent with an input ERC. This means that the markedness constraint M is ranked right below F_2 at time T and that ERC 2 will not trigger any further updates. As F_1 and F_3 are currently equally ranked, ERC 3 has so far triggered no update. It will thus have to trigger more than $2/p_3$ updates in the remaining portion of the learning path. As F_2 is well above $F_1 = F_3$ and since M is right below F_2, then M is well above $F_1 = F_3$ as well. Thus, there is enough separation between M and $F_1 = F_3$ in order for ERC 3 to indeed trigger more than $2/p_3$ updates. After F_1 has been promoted by $p_3 > 0$ for more than $2/p_3$ times, F_1 will be ranked above F_3 by more than 2, leaving enough space for M to drop below F_1 while still being above F_3 (see Subsection 4.3.5 for details on this step of the reasoning). The cases in (99b) and (99c) can instead never arise under the hypothesis that ERC 3 triggers more than $2/p_3$ updates. In fact, as F_1 and F_3 are equally ranked at time T, ERC 3 has not triggered any update up to time T; and it will not be able to trigger its required updates in the remaining portion of the learning path, as the markedness constraint M is too close to $F_1 = F_3$ in case (99b) and is already outranked by $F_1 = F_3$ in case (99c). Note that condition (98) has not played any role in the discussion of these three cases (99).

Consider next the scenario where F_1 is ranked above F_3 at time T. Again, three cases need to be distinguished, depending on whether F_2 is ranked above both F_1 and F_3, as in (100a); or in between them, as in (100b); or else below both of them, as in (100c).

(100)



If F_2 is top ranked at time T as in (100a), then M is ranked right underneath F_2. From now on, ERC 2 is accounted for and only ERCs 1 and 3 will trigger updates. As long as ERC 3 triggers more than $2/p_3$ updates (either before or after time T), the two faithfulness constraints F_1 and F_3 will be separated enough in order for M to drop below F_1 but not below F_3. Analogously, if F_1 is top ranked at time T as in (100b), then M is ranked right underneath F_1. From now on, ERCs 1 and 3 are accounted for and only ERC 2 will trigger updates. The markedness constraint M will drop a bit in order to cross F_2. As M and F_2 will cross in a position that is above the position of F_2 at time T, then M will never cross F_3. Note that condition (98) has not played any role in the discussion of the two cases (100a) and (100b).

Let me finally consider the more delicate case (100c), where F_2 is ranked below both F_1 and F_3 at time T. Since F_1 is top ranked at time T, the markedness constraint M is just below F_1. From now on, ERCs 1 and 3 are accounted for and only ERC 2 will trigger updates. Each update by ERC 2 promotes F_2 by p_2 and demotes M by 1. Updates will continue until the two constraints F_2 and M meet at some point marked as (*) in (100c). Let θ_{meeting} be the ranking value corresponding to this meeting point (*). After M and F_2 have met at (*), M can still drop by 1, in order to cross F_2, at which point learning ceases. Thus, the final ranking value of M cannot be smaller than $\theta_{\text{meeting}} - 1$ at the end of learning. While at the end of learning, F_3 sits at the same position where it seated at time T. The condition that M be above F_3 at the end of learning thus boils down to the condition (101).

(101) $\theta_{\text{meeting}} - 1 > \theta^T_{F_3}$

The ranking value θ_{meeting} of the point (*) in (100c) where M and F_2 meet can be characterized analytically as follows. Let x be the number of updates by ERC 2 required after time T in order to demote M and promote F_2 until they meet at (*). Thus, x satisfies condition (102). In fact, the left and side $\theta^T_{F_2} + p_2 x$ is the ranking value of F_2 after x promotions by p_2 triggered by ERC 2 starting from the ranking value $\theta^T_{F_2}$ of F_2 at time T; and the right hand side $\theta^T_M - x$ is the ranking value of M after x demotions by 1 triggered by ERC 2 starting from the ranking value θ^T_M of M at time T.

(102) $\theta^T_{F_2} + p_2 x = \theta^T_M - x$

The ranking value θ_{meeting} of the point (*) in (100c) where M and F_2 meet can thus be characterized as in (103a), namely as the sum between the ranking value $\theta^T_{F_2}$ of F_2 at time T plus the extra way $p_2 x$ that F_2 needs to raise in order to meet M. And the latter expression can be made explicit as in (103b), using the explicit expression for the number of updates x obtained by solving equation (102).

(103)
$$\begin{aligned}\theta_{\text{meeting}} &\overset{(a)}{=} \theta^T_{F_2} + p_2 x\\ &\overset{(b)}{=} \theta^T_{F_2} + p_2 \underbrace{\frac{1}{1+p_2}\Big(\theta^T_M - \theta^T_{F_2}\Big)}_{x}\end{aligned}$$

Using the explicit characterization of θ_{meeting} in (103), condition (101) can be made explicit as in (104), in terms of the current ranking values of F_2, F_3 and M at time T.

(104)
$$\underbrace{\theta^T_{F_2} + \frac{p_2}{1+p_2}\Big(\theta^T_M - \theta^T_{F_2}\Big)}_{\theta_{\text{meeting}}} -1 > \theta^T_{F_3}$$

The ranking value $\theta^T_{F_2}$ of constraint F_2 at time T is $p_2\alpha^T_2$; the ranking value $\theta^T_{F_3}$ of constraint F_3 at time T is $p_1\alpha^T_1$. The markedness constraint M is ranked right underneath the faithfulness constraint F_1 at time T in (100c), namely $\theta^T_M = \theta^T_{F_1} - 1$. As the ranking value $\theta^T_{F_1}$ of F_1 at time T is $p_1\alpha^T_1 + p_3\alpha^T_3$, then the ranking value of M at time T cannot be smaller than $p_1\alpha^T_1 + p_3\alpha^T_3 - 1$. Condition (104) is thus entailed by condition (105).

(105)
$$p_2\alpha^T_2 + \frac{p_2}{1+p_2}\Big(p_1\alpha^T_1 + p_3\alpha^T_3 - 1 - p_2\alpha^T_2\Big) - 1 > p_1\alpha^T_1$$

Finally, the latter condition (105) simplifies to the desired condition(98).

□

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 *Proceedings of the Institute of Phonetic Sciences (IFA) 21*, ed. Rob van Son, 43–58. University of Amsterdam: Institute for Phonetic Sciences.

Boersma, Paul. 1998. Functional phonology. Doctoral Dissertation, University of Amsterdam, The Netherlands. 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, and Bruce Hayes. 2001. Empirical tests for the Gradual Learning Algorithm. *Linguistic Inquiry* 32:45–86.

Boersma, Paul, and Clara Levelt. 2000. Gradual constraint-ranking learning algorithm predicts acquisition order. In *Proceedings of the 30th Child Language Research Forum*, ed. Eve V. Clark, 229–237. Stanford University: CSLI. Corrected version available as ROA 361.

Cesa-Bianchi, Nicolò, and Gábor Lugosi. 2006. *Prediction, learning, and games*. Cambridge University Press.

Curtin, Suzanne, and Kie Zuraw. 2002. Explaining constraint demotion in a developing system. In *Boston University Conference on Language Development (BUCLD) 26*, ed. Barbora Skarabela, Sarah Fish, and Anna H.-J. Do, volume 1, 118–129. Cascadilla Press.

Davidson, Lisa, Peter W. Jusczyk, and Paul Smolensky. 2004. The initial and final states: Theoretical implications and experimental explorations of richness of the base. In *Constraints in phonological acquisition*, ed. R. Kager, J. Pater, and W. Zonneveld, 158–203. Cambridge University Press.

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.

Hale, Mark, and Charles Reiss. 1998. Formal and empirical arguments concerning phonological acquisition. *Linguistic Inquiry* 29.4:656–683.

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.

Jesney, Karen, and Anne-Michelle Tessier. 2009. Biases in harmonic grammar: the road to restrictive learning. *Natural Language and Linguistic Theory* 29:251–290.

Jusczyk, P. W., A. D. Friederici, J. M. I. Wessels, V. Y. Svenkerud, and A. Jusczyk. 1993. Infants' sensitivity to the sound patterns of native language words. *Journal of Memory and Language* 32:402–420.

Jusczyk, Peter, Paul Smolensky, and Theresa Allocco. 2002. How english-learning infants respond to markedness and faithfulness constraints. *Language Acquisition* 10:31–73.

Kager, René. 1999. *Optimality Theory*. Cambridge, United Kingdom: Cambridge University Press.

Keller, Frank, and Ash Asudeh. 2002. Probabilistic learning algorithms and Optimality Theory. *Linguistic Inquiry* 33.2:225–244.

Kivinen, Jyrki. 2003. Online learning of linear classifiers. In *Advanced lectures on machine learning (lnai 2600)*, ed. S. Mendelson and A.J. Smola, 235–257. Berlin Heidelberg: Springer-Verla.

Magri, Giorgio. 2011. On the performance of error-driven ranking algorithms on prince & tesar's (2004) case studies. In *Proceedings of CLS 47: the 47th annual conference of the Chicago Linguistics Society*.

Magri, Giorgio. 2012a. Constraint promotion: not only convergent, but also efficient. In *Proceedings of the 48th annual conference of the Chicago Linguistics Society*.

Magri, Giorgio. 2012b. Convergence of error-driven ranking algorithms. *Phonology* 29:213–269.

Magri, Giorgio. 2012c. A note on the GLA's choice of the current loser from the perspective of factorizability. Accepted for publication at the *Journal of Logic, Language, and Information*.

Magri, Giorgio. in press. HG has no computational advantages over OT: towards a new toolkit for computational OT. *Linguistic Inquiry* .

Pater, Joe. 2008. Gradual learning and convergence. *Linguistic Inquiry* 39.2:334–345.

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

Prince, Alan. 2002. Entailed ranking arguments. Ms., Rutgers University. Also available as 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. Also available as ROA 537 version.

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.

Smolensky, Paul. 1996a. The initial state and richness of the base in optimality theory. John Hopkins Technical Report.

Smolensky, Paul. 1996b. On the comprehension/production dilemma in child language. *Linguistic Inquiry* 27.4:720–731.

Stemberger, Joseph Paul, and Barbara Handford Bernhardt. 1999. The emergence of faithfulness. In *The emergence of language*, ed. B. MacWhinney, 417–446. Mahweh, NJ: Erlbaum.

Stemberger, Joseph Paul, and Barbara Handford Bernhardt. 2001. U-shaped learning in language acquisition, and restrictions on error correction. Poster presented at the 2001 Biennial Meeting of the Society for Research in Child Development.

Stemberger, Joseph Paul, Barbara Handford Bernhardt, and Carolyn E. Johnson. 1999. U-shaped learning in the acquisition of prosodic structure. Poster presented at the sixth International Child Language Congress.

Tesar, Bruce. 2008. Output-driven maps. Ms., Rutgers University. Also available as ROA-956.

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

Tessier, Anne-Michelle. 2009. Frequency of violation and constraint-based phonological learning. *Lingua* 119.1:6–38.

Wexler, Kenneth, and Peter W. Culicover. 1980. *Formal principles of language acquisition*. Cambridge, MA: MIT Press.