

Four Rules of Inference for Ranking Argumentation

Bruce P. Hayes

UCLA

April 17, 2003

1. Setting

I think any scholar working in Optimality Theory (Prince and Smolensky 1993, and much subsequent work) would agree that rigorous language-particular analysis within this framework should provide solid arguments for the ranking of constraints. It is sometimes underappreciated just what is involved in developing such arguments. I will suggest that arguments from ordinary human reasoning are (probably) not generally as reliable as arguments produced by algorithm. A sample algorithm of the appropriate type, which uses four rules of inference, is laid out here. I also suggest that algorithmic ranking argumentation has some potential theoretical as well as practical consequences.

2. Fundamentals of Ranking

Assume an analysis employing Optimality Theory, with a suitable set of candidates for each input to test the analysis.¹ Given this, the logical basis of ranking works in a quite simple way, laid out clearly by Tesar and Smolensky (1996) and summarized as follows.

In principle, each pair of the form {Winning candidate, defeated Rival candidate} (abbreviation: **W**, **R**) gives us information on the following basis:

(1) Let $C_w = \{C_1, C_2, C_3, \dots\}$ = the set of constraints that are violated more times by **W** than by **R**.

Let $C_r = \{C'_1, C'_2, C'_3, \dots\}$ = the set of constraints that are violated more times by **R** than by **W**.

Then **every** constraint in C_w must be dominated by **at least one** constraint in C_r .

The reasoning here should be clear: in the “culling out” procedure that in OT defines the choice of winning candidate, **R** must be culled out before **W**. This will happen if, for each constraint that potentially could erroneously cull **W** early, there is at least some constraint ranked high enough to cull **R** instead.

¹ Often this is itself a difficult issue; just how a set of candidates can be found that tests the constraint system to a full degree of certainty is a problem that to my knowledge has not been solved in the general case. See Albro (1997) for some progress on this issue.

The hard part is that in actual practice, it is very often that case that there are multiple choices for *which* constraint in C_r saves the day.

Therefore, the “raw material” from which we will usually have to construct ranking arguments will typically involve **sets**, as in the first three of the following representative hypothetical starting point (A, B, ... Z are constraints; throughout, square brackets surround ranking arguments):

- (2) [A, B, or C >> D]
 [E or F >> G]
 [H, I, J, or K >> L]
 [M >> N]

I define a **set argument** as anything like the top three lines of (2): it emerges from comparing a winner’s constraint violations with a rival’s, and requires that at least one constraint in a set (with more than one member) dominate some other constraint. A **pairwise argument** is what we (as analysts) would like to achieve: it simply states that one constraint must outrank another. The last line of (2) shows a pairwise argument.

My experience in analysis and pedagogy with OT indicates that the starting point for ranking argumentation is indeed very often a very large, tangled web, consisting for the most part of set arguments with only a few pairwise arguments.² Careful ranking argumentation must go beyond merely locating those pairwise arguments that (through good fortune or wise candidate choices) emerged at the very beginning; to be truly rigorous it must sort through the entire set of arguments to see what can be learned.

3. Reasoning Through the Tangle

It is to a fair degree possible to untangle these initial dense webs of argumentation, through the application of simple rules of inference, the presentation of which is the main purpose of this squib. The rules of inference tend gradually to reduce the of set arguments, ultimately (in many, but not all cases) creating in the end a straightforward set of pairwise arguments as their final output.

There are four inference rules.

² Careful choice of rival candidates, letting them differ only minimally from the winner, will normally improve the number of pairwise arguments found at the outset.

3.1 Superset Redundancy

Suppose that the set of ranking arguments obtained from the violation pattern includes a pair like this:

- (3) [A, B, or C >> D]
[A or B >> D]

It should be plain that the first ranking argument in such a case is completely redundant. Any grammar that respects the second condition in (3) will also respect the first. Thus, in such an instance, it is safe to discard the first argument.

Stating this as a rule of inference, we have:

(4) **Superset Redundancy**

If, in two ranking arguments, the dominees are the same, and the dominator set of one is a superset of the dominator set of the other, discard the ranking argument that includes the superset.

Naturally, a special case of the above is when the subset ranking argument is a pairwise argument.

3.2 Reverse-Ranking Elimination

Second, suppose our list of raw ranking arguments contains a pair like this:

- (5) [A, B or C >> D]
[D >> A]

It should be plain that, of the three constraints A, B, and C that might *in principle* dominate D, A is not the one, because if it were, it would contradict the independently known pairwise ranking [D >> A]. Thus, we can infer:

- (6) [B, C >> D]

and we trim down the number of possibilities for the ranking arguments.

In general then, a rule of inference that may be employed is:

(7) **Reverse-Ranking Elimination**

Eliminate from the dominator set of a set argument any constraint C for which a pairwise ranking argument exists which subordinates C to the dominee of the set argument.

3.3 Part-For-Whole Selection

Consider the scenario in (8):

- (8) [A or B >> C]
[A >> B]

We can reason here as follows. Suppose that, of the possibilities { A, B } for the constraint that dominates C, the actual dominator were B. If so, by transitivity of domination,³ A must also dominate C. On the other hand, if it is A that dominates C, we know nothing about the ranking of B and C. Local conclusion: either both A and B dominate C, or A alone dominates C. Therefore, we *know* that A dominates C, and we can safely replace the argument [A or B >> C] with the argument [A >> C].

I will call this mode of inference “Part-for-Whole Selection”: a part of the set { A, B } turns out to be “doing all the work” of domination, so the rest of the set may be discarded.

Part-for-whole selection need not be confined to cases where the part-whole relation is defined by a simple pairwise argument. For example, if we have [A, B, or C >> D] and [A or B >> C], then it is safe to reduce [A, B, or C >> D] to the simpler [A or B >> D]. The same reasoning is applicable.

The part-for-whole inference is stated as follows:

(9) **Part-for-Whole Selection**

Let **S** be a set of constraints, **s** be a proper subset of **S**, **C** a constraint included in **S** but not **s**, and **S >> D** be an ordinary set argument (“at least one constraint in **S** must dominate **D**”). If **s >> C** is also a ranking argument, then replace **S >> D** with the simpler ranking argument **S' >> D**, where **S'** is the set obtained by removing **C** from **S**.

3.4 Transitivity

This is the simplest mode of reasoning. It is the only mode that actually adds arguments, rather simplifying or eliminating them.

(10) **Transitivity**

If [A >> B], and [B >> C], then [A >> C].

This assumes that transitivity is indeed a valid assumption for constraint ranking. For more on this, see below.

³ For more on transitivity, see below.

4. Iterated Reasoning; Implementation

It is easy to see that these various rules of inference could in principle “feed” one another. For example, knowing $[A \text{ or } B \gg C]$ and $[A \gg B]$, we obtain $[A \gg C]$, by Part-for-Whole Selection. If we also have $[C > D]$ (say, from a winner-rival pair that yielded it at the outset), we could obtain $[A \gg D]$ by Transitivity. The newly obtained $[A \gg D]$ could then be used to eliminate another argument, $[A, E, \text{ or } F \gg D]$ from the argument list by Superset Redundancy.

To state this as an algorithm: suppose we repeatedly examine the full set of arguments (set- and otherwise), attempting to apply each rule of inference in succession, in all possible locations; and not stop until we have reached a state where none of the rules of inference can apply further. In principle, we could ultimately obtain a set of ranking arguments that was far simpler than what we started out with. If good fortune prevails, most or all of these arguments will be pairwise.

This procedure is plainly tedious to carry out by hand, but it is not at all hard to program.⁴

My judgment is that, assuming I have found all the valid rules of inference, ranking arguments obtained by machine following my algorithm are quite likely to be more reliable than ranking arguments done by hand. In my own experience (both teaching and research), ranking by algorithm does seem to be able to reduce a tangled web of initial ranking arguments rather effectively, often eliminating every one of the large group of set-ranking arguments with which it began. The chain of inference can be rather impressive in length, even for relatively simple problems.

5. A Transitivity Problem

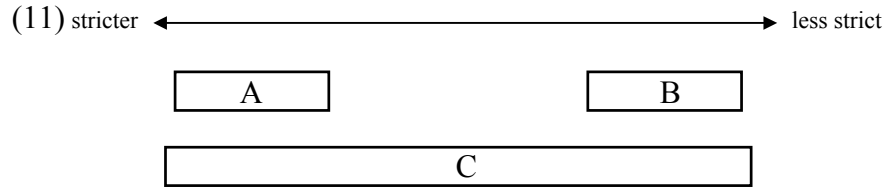
A setback for the approach set out here is that there is a pervasively-applicable circumstance under which the assumption of transitivity of ranking is simply false, namely where a single input produces two outputs in free variation. It is an assumption widely adopted enough in OT to be considered folklore⁵ that free variation is to be generated by ranking a subset of the constraints freely. In such cases, to know what the grammar can generate one must examine all rankings of the constraints that are permitted and collect the resulting outputs.

Now, in such analyses, we usually find that a particular input will have two outputs, generated along the following scheme: output O requires that constraint A dominate constraint B; while O' requires $[B \gg A]$. In such a case, the procedure given above will often produce both $[A \gg B]$ and $[B \gg A]$ as ranking arguments, assuming that both O and O' appear in the forms from which ranking is inferred.

Consider now a hypothetical situation which is perhaps best depicted graphically:

⁴ My own amateur's rendition can be downloaded from <http://www.humnet.ucla.edu/humnet/linguistics/people/hayes/otsoft/otsoft.htm>, where it resides as part of a package of analytic OT software.

⁵ But whoever thinks they invented the idea first is most welcome to contact me with a correction!



Let the bars depict “ranges of strictness” for constraints on an arbitrary scale.⁶ Here, the ranges are merely meant to illustrate the following possibility for ranking argumentation. On the left side of the diagram, the overlap indicates that somewhere in the data set, there are output forms that require the ranking $A \gg C$; and that due to free variation, there are also output forms that demand the ranking $C \gg A$. Similar facts require $B \gg C$ and $C \gg B$.

In principle, we could take *two* out of these four ranking arguments and develop from them an erroneous conclusion, as follows: $B \gg C$, $C \gg A$, therefore by Transitivity $A \gg B$. As the chart shows, this is quite false; the actual grammar at hand *never* permits outputs to arise in which $B \gg A$.

The conclusion here is that ranking-argument reasoning based on the transitivity of dominance cannot be reliably employed in a grammar that generates free variation.

Moreover, it will be noticed that the rationale of Part-for-Whole Selection in 3.3 above likewise relies crucially on transitivity; thus this rule of inference apparently must also be abandoned in free-variation grammars. [*Help! Is this also true for Reverse Ranking Elimination?*].

Both conclusions are unfortunate, since it would seem (at least to me) that every grammar of a natural language is a free-variation grammar. However, the Part-for-Whole Selection and Transitivity inference rules remain usable for the artificially circumscribed data sets with which linguists often deal, such as problem sets.

⁶ For actual theoretical development of this concept, see Boersma (forthcoming), Hayes (forthcoming).

6. Conclusion: Why This Matters

[stuff on ranking algorithms: Why they are some important; why Tesar/Smolensky's is too tight; why Boersma's is too lax; the future lies in **carefully tuned** algorithms that use as much reasoning as they possibly can.

[Also: Maybe there are more inference rules, especially for free variation cases. Further inquiry is cordially solicited.]

References

- Albro, Dan 1997 UCLA MA on GEN-less computational OT
Boersma, Paul (forthcoming) *Pointing Finger* volume
Hayes, Bruce (forthcoming) *Pointing Finger* volume
Tesar, Bruce and Paul Smolensky, all three Constraint Demotion papers