Three Formal Extensions to Primitive Optimality Theory

Daniel M. Albro
Linguistics Department, UCLA
3125 Campbell Hall
Los Angeles, CA 90095-1543, USA
albro@humnet.ucla.edu

Abstract

This paper proposes an expansion of set of primitive constraints available within the Primitive Optimality Theory framework (Eisner, 1997a). This expansion consists of the addition of a new family of constraints—existential implicational constraints, which allow the specification of faithfulness constraints that can be satisfied at a distance—and the definition of two ways to combine simple constraints into complex constraints, that is, constraint disjunction (Crowhurst and Hewitt, 1995) and local constraint conjunction (Smolensky, 1995).

1 Introduction

Primitive Optimality Theory (OTP) (Eisner, 1997a), and extensions to it (*e.g.*, Albro (1998)), can be useful as a formal system in which phonological analyses can be implemented and evaluated. However, for certain types of constraints, translation into the primitives of OTP (Eisner (1997b)) can only be accomplished by adding to the grammar a number of ad hoc phonological tiers. Because these tiers serve no phonological purpose other than to allow calculation of the constraints without adding new primitives, and because the addition of phonological tiers to an OTP grammar can have a dramatic negative impact on the efficiency of OTP implementations[1], it is preferable to avoid the addition of ad hoc tiers by adding new primitives to the system. This paper looks at three types of constraints employed throughout the Optimality Theoretic literature that cannot be translated in to the primitives of OTP without reference to ad hoc tiers, and proposes a formalization of these constraints that is compatible with the finite state model described in Eisner (1997a) and Albro (1998). These are constraints of existential implication (that is, of faithfulness without the requirement of alignment), constraint disjunction, and local constraint conjunction.

[1]The computation time for an Optimality Theoretic derivation within the implementation of Albro (1998) increases exponentially with the number of tiers. The same is true for the implementation described in Eisner (1997a), although a proposal is given there for a method that might improve the situation.

2 Existential Implication

2.1 Motivation

OTP as described in Eisner (1997a) provides some support for correspondence constraints (input-output only). These may be defined by means of *implication* constraints of the form $P \rightarrow \underline{P}$ or $\underline{P} \rightarrow P$, which can be interpreted as requiring, in the first case, that each surface constituent representing property P be aligned with an underlying constituent representing that property, and in the second case that every underlying constituent representing property P be aligned with a surface constituent representing that property. Constraints of this type may be employed to require correspondence between the underlying representation and the surface representation where corresponding constituents must be aligned with one another. However, natural languages also seem to follow weaker constraints requiring only that for each underlying constituent there be a corresponding surface constituent, regardless of the position of that constituent relative to its position in the underlying representation. For example, in Sanskrit roots with at least two voiced stops, where the root ends in a voiced aspirated stop, the underlying aspiration of the root-final stop can be realized upon that stop in the surface representation only when the root is followed by a suffix beginning with a vocoid or a nasal (data from Whitney (1889)):

/dag$^{\text{ɦ}}$/	reach to	[dag$^{\text{ɦ}}$iʂjanti]	(Fut.)
/bud$^{\text{ɦ}}$/	know, wake	[bod$^{\text{ɦ}}$i]	(Aor.)
/dab$^{\text{ɦ}}$/	harm	[dab$^{\text{ɦ}}$ati]	(Pres.)

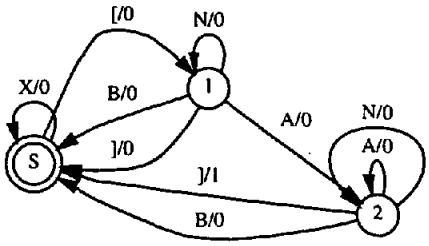
Otherwise, the aspiration is realized on the preceding stop:

/dag$^{\text{ɦ}}$/	reach to	[d$^{\text{ɦ}}$ak]	(root noun)
/bud$^{\text{ɦ}}$/	know, wake	[b$^{\text{ɦ}}$ut]	(root noun)
/dab$^{\text{ɦ}}$/	harm	[d$^{\text{ɦ}}$ap]	(root noun)

In these forms it is clear that aspiration is being preserved, but that it is surfacing in a position that cannot overlap with the underlying form. Another example is the Bantu language Chizigula (Kenstowicz and Kisseberth, 1988), in which roots with underlying high vowels appear on the surface with a single high tone in the penultimate syllable of the word, where this syllable could belong to a suffix. Additionally, if a prefix with an underlying high tone is prefixed to a root with no underlying high tone, the high tone of the prefix appears in the penultimate syllable of the resulting word. The existence of a high tone in the underlying form implies the existence of a high tone in the surface form, but the position where that high tone occurs in the underlying form has nothing to do with where the tone appears on the surface.

2.2 Formalization

Existential implication constraints can be used to drive correspondence effects such as the above. These constraints take the form "If X appears within domain D, then Y must appear within D as well." Using the terms of OTP, this family of constraints can be written as $\alpha_1 \wedge \ldots \wedge \alpha_m \rightarrow \beta_1 \vee \ldots \vee \beta_n / \gamma_1 \wedge \ldots \wedge \gamma_p$. Here each α_i or β_j is a constituent interior or edge, one of *tier*,]*tier*, or [*tier*, where *tier* represents a constituent type, and each γ_k must be a constituent interior (*tier*). The constraint represented by this notation outputs a violation for each domain γ, where γ represents the intersection of the domains γ_k, in which the time slice represented by the α_i occurs, but no β_j occurs. Using the FST notation of Eisner (1997a), the implementation for this constraint would be the following FST:



where X represents "Σ^*−((in or begin all γ_k) - (in all γ_k))," N represents "((in all γ_k) $\wedge$ ¬(in all α_i) $\wedge$ ¬(in some β_j))," B represents "((in all γ_k) $\wedge$ (in some β_j))," A represents "((in all γ_k) $\wedge$ (in all α_i) $\wedge$ ¬(in some β_j))," [represents "((in or begin all γ_k) - (in all γ_k))," and] represents "((in or end all γ_k) - (in all γ_k))." That is, the machine moves from state S to state 1 if the domain γ is entered. It moves from there back to state S if the end of the domain appears before α does, or if any β appears. If α appears, the machine moves from state 1 to state 2. From state 2, if β appears, the machine returns to the start state without outputting a violation, but if the end of the domain appears without any β having appeared, the machine outputs a violation.

3 Constraint Disjunction

Crowhurst and Hewitt (1995) cite a number of instances in which it appears that multiple *simple* constraints must be combined via disjunction (there called conjunction) into complex constraints. Here a *simple* constraint is a function that takes an input, surface pair as its input and returns *true* if a particular disallowed phonological structure or lack of correspondence is present in the pair, otherwise *false*. A constraint disjunction would thus be a function that returns the disjunction of the outputs of its component constraints. Thus a constraint defined by disjunction of component constraints outputs a violation whenever any one of its components does.

Formalization of constraint disjunction requires reference only to intersection of weighted finite state machines. Specifically, if constraint C_1 is defined as a weighted finite state machine $T_1 = \langle \Sigma_1, \Sigma_2, Q_1, F_1, s_1, E_1 \rangle$, where Σ_1 is

the alphabet of labels, Σ_2 is the alphabet of weights, drawn from the natural numbers, Q_1 is the set of states of the machine, $F_1 \subseteq Q_1$ is the final states, s_1 is the start state, and $E_1 \subseteq Q_1 \times \Sigma_1 \times \Sigma_2 \times Q_1$ is the set of edges, and constraint C_2 is another weighted deterministic finite state machine $T_2 = \langle \Sigma_1, \Sigma_2, Q_2, F_2, s_2, E_2 \rangle$, then the disjunction of the two constraints may be defined as follows:

$$\begin{aligned} T = \langle \Sigma_1, \Sigma_2, Q_1 \times Q_2, F_1 \times F_2, \langle s_1, s_2 \rangle, E \rangle, \\ \langle \langle q_{1,1}, q_{2,1} \rangle, a, n, \langle q_{1,2}, q_{2,2} \rangle \rangle \in E \text{ iff} \\ \langle q_{1,1}, a_1, n_1, q_{1,2} \rangle \in E_1 \wedge \\ \langle q_{2,1}, a_2, n_2, q_{2,2} \rangle \in E_2 \wedge \\ a = a_1 \cap a_2 \wedge \\ n = (n_1 \vee n_2) \end{aligned}$$

A possible notation for the disjunction of two constraints C_1 and C_2 is $C_1 \vee C_2$, for example "(vce $\rightarrow$ vce) $\vee$ (cont $\rightarrow$ cont)".

A similar concept is that of mutually unranked primitive constraints. For any given input, a complex constraint defined as a group of mutually unranked primitive constraints returns the sum of the violations that the primitive constraints returned. Although it has been argued that the formal power provided by allowing new constraints to be defined by grouping mutually unranked primitive constraints is too great, constraints so defined are fairly prevalent in the literature. For example, Steriade (1996) makes use of a constraint **Paradigm Uniformity (PU) Stress** which requires that all features within stressed syllables in one member of a paradigm must be preserved in the corresponding syllable of other members of that paradigm. **PU Stress** is equivalent to a set of mutually unranked paradigm uniformity constraints for all phonological features. The empirical prediction of **PU Stress** is that changes in any one feature are as important as changes in any other. If **PU Stress** were instead to be considered a block of ranked constraints for the individual features, the prediction would be that in the comparison between one candidate in which the top-ranked feature is identical between stressed syllables of the paradigm members, but all other features are different, and another candidate in which only a lower-ranked feature is different, the first candidate would prevail. The data seems to bear out the prediction of the definition using mutually unranked constraints. Another possible definition of **PU Stress** would be to make use of constraint disjunction. In this definition, all features would be equally important, but the number of non-identical features would not matter—candidates differing in three features would be equal to candidates differing in one feature. Once again, the definition using mutually unranked constraints seems better borne out by the data.

Leaving aside constraints such as **PU Stress**, we will see that complex constraints defined as combinations of mutually unranked constraints are useful as inputs to local constraint conjunctions. The formal definition of a complex constraint in terms of mutually unranked subconstraints is identical to the definition of a constraint disjunction, except that the weight n of a new edge is defined as the sum of the weights of the input edges n_1 and n_2 rather than the disjunction:

$$\begin{aligned} T = \langle \Sigma_1, \Sigma_2, Q_1 \times Q_2, F_1 \times F_2, \langle s_1, s_2 \rangle, E \rangle, \\ \langle \langle q_{1,1}, q_{2,1} \rangle, a, n, \langle q_{1,2}, q_{2,2} \rangle \rangle \in E \text{ iff} \\ \langle q_{1,1}, a_1, n_1, q_{1,2} \rangle \in E_1 \wedge \\ \langle q_{2,1}, a_2, n_2, q_{2,2} \rangle \in E_2 \wedge \\ a_1 \cap a_2 = a \wedge \\ n_1 + n_2 = n \end{aligned}$$

A possible notation for a complex constraint C combining mutually unranked constraints C_1 and C_2 is $C_1 + C_2$, for example "(vce $\rightarrow$ vce) + (cont $\rightarrow$ cont)".

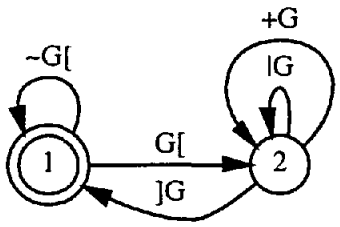
4 Local Constraint Conjunction

Smolensky (1995) and Kirchner (1996) propose a different method for combining constraints: local conjunction. A local conjunction of constraints is defined as a constraint that outputs a violation for each domain of a specified type in which all of the component constraints are violated. A constraint may be locally conjoined with itself, in which case the resulting conjunction outputs a violation whenever there are two violations of the component constraint within the specified domain. The conjunction of a constraint C_1 with itself within a domain γ may be notated "$\wedge(C_1)/\gamma$."

The following algorithm computes the local conjunction of constraint C_1, where C_1 is represented by the weighted finite state machine $T_1 = \langle \Sigma_1, \Sigma_2, Q_1, s_1, F_1, E_1 \rangle$, with itself within

a domain γ defined as the intersection of the domains $\gamma_1 \wedge \ldots \wedge \gamma_n$:

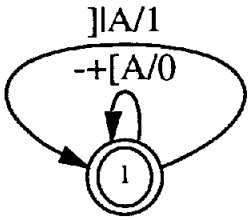
1. *Weaken* C_1 to a new constraint C_1' such that for any utterance to which C_1 assigns a non-zero number n of violations, C_1' assigns $n-1$ violations. This may be accomplished as follows:
   (a) Copy T_1 as T_2, renumbering the states of T_2 so that there is no ambiguity.
   (b) Combine T_1 and T_2 into $T = \langle \Sigma_1, \Sigma_2, Q_1 \cup Q_2, s_1, F_1 \cup F_2, E = E_1 \cup E_2 \rangle$.
   (c) For each edge $\langle q_i, a, w, q_j \rangle \in E_1$, where $w > 0$, modify the edge to $\langle q_i, a, w-1, s_2 \rangle$, where s_2 is the state corresponding to the start state of T_2. T represents constraint C_1'.
2. Define a finite state machine M as follows:



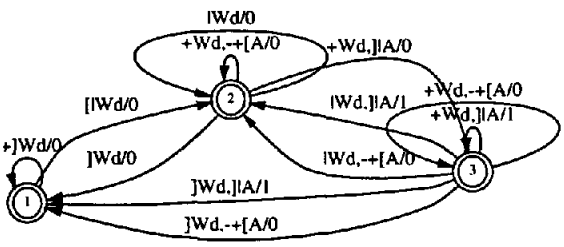
   where $G[$ represents the beginning of domain γ, $\overline{G[}$ represents anything other than $G[$, $+G$ represents the interior of the domain, $|G$ represents a boundary between two γ domains, and $]G$ represents the end of the γ domain.
3. The machine M will be used to limit the evaluation of constraint C_1' to the domain γ. To accomplish this, we need to define the behavior at the edges of the γ domain. Outside the γ domain, violations of C_1' will have no effect. At the left edge of the γ domain, violations that do not involve the left edge of constituents will have no effect. At the right edge of the γ domain, violations that do not involve the right edge of constituents will have no effect. The final weighted finite state machine L representing the local conjunction of C_1 with itself is produced by intersecting M with T, with the following modifications made to the intersection algorithm. Edges from T that are intersected with the edge $G[$, or edges from T that are intersected with the edge $\overline{G[}$ and contain no reference to a left edge, or edges from T that are intersected with the edge $]G$ and contain no reference to a right edge, are assigned a weight of 0, and if their destination within T was state s_2, their destination in T is treated as having been s_1. This has the effect of limiting the constraint violations of C_1' to the domain γ. Edges from T that are intersected with edge $|G$ keep their original weight, but are treated as though their destination within T was s_1. This has the effect of resetting C_1' to zero violations at the beginning of a γ domain immediately following another.

The constraint $\wedge(C_1)/\gamma$ produced by the above algorithm outputs a violation for every violation of C_1 after the first within domain γ. Thus $\wedge(C_1)/\gamma$ penalizes two or more violations of C_1 within γ, but does not penalize single violations of C_1.

For example, the constraint $A \perp A$ is represented as the following weighted finite state machine:



The result of the above algorithm is the following machine:



While this algorithm does not allow definition of local conjunction of different constraints,

it can be given nearly equivalent power by applying it to the output of complex constraints formed from mutually unranked subconstraints.

References

Daniel M. Albro. 1998. Evaluation, implementation, and extension of Primitive Optimality Theory. Master's thesis, UCLA.

Megan Crowhurst and Mark Hewitt. 1995. Conjunctive constraints and templates in Optimality Theory. Ms.

Jason Eisner. 1997a. Efficient generation in primitive Optimality Theory. In *Proceedings of the ACL.*

Jason Eisner. 1997b. What constraints should OT allow? Handout for talk at LSA, Chicago, January.

Michael Kenstowicz and Charles Kisseberth. 1988. Chizigula tonology—the word and beyond. Ms.

Robert Kirchner. 1996. Synchronic chain shifts in optimality theory. *Linguistic Inquiry.*

Paul Smolensky. 1995. On the internal structure of the constraint component Con of UG. Handout of talk given at UCLA, April.

Donca Steriade. 1996. Paradigm uniformity and the phonetics-phonology boundary. In *5th Conference in Laboratory Phonology,* Evanston, Illinois.

W. D. Whitney. 1889. *Sanskrit grammar.* Harvard University Press, Cambridge.