Output-Driven Maps*

Bruce Tesar
Linguistics Department and the Rutgers Center for Cognitive Science
Rutgers University, New Brunswick
3/15/08

"*Who do we ask for help when we don't know which way to go? The map!*"
-- Dora the Explorer

1 Introduction

1.1 Output-Orientedness: Maps and Generalizations

A long-standing issue of interest in phonology is the extent to which phonological maps from underlying forms to phonetic forms (in the terms of generative phonology) can be characterized in terms of conditions on the output forms (Chomsky 1964, Kiparsky 1971, Kiparsky 1973, Kisseberth 1970, for a broader overview of the issue see McCarthy 2007a). The intuition might be expressed like this: a map will by default preserve the input in the output, and any disparities between the input and the output are motivated by restrictions on the output form. I will call this intuitive property **output-orientedness**. While the statement "all restrictions motivating input-output disparities are output restrictions" may seem intuitive and concrete, formalizing it in a satisfactory way is not so straightforward. Any direct formalization requires defining exactly what kind of thing an output restriction is, and what exactly it means for a restriction (or set of restrictions) to "motivate" an input-output disparity (motivations tend to be murky things). Furthermore, it is not immediately obvious how to give such a characterization without delving deeply into the mechanics of a particular theory.

Two component intuitions are important to the overall intuition of output-orientedness that is being pursued here. One component intuition is that if input-output disparities are only motivated by output restrictions, then inputs that are equivalent to well-formed outputs should map to themselves (with no disparities). This is the identity map property (Prince and Tesar 2004): a well-formed output, when used as an input, maps to itself. The other component intuition is that disparities are introduced consistently to satisfy output restrictions. There should be a non-arbitrary connection between the disparities introduced and the output restrictions that motivate them, non-arbitrary in terms of patterning across the different input-output mappings of the map.

(1) Guiding intuitions about output-orientedness

a. Well-formed outputs should map to themselves (the identity map property).

b. The pattern relating disparities their motivating restrictions should be non-arbitrary.

* YOUR NAME COULD APPEAR HERE! Send comments to tesar@rutgers.edu. The author wishes to thank Eric Baković, Paul de Lacy, Jane Grimshaw, Fernando Guzman, Brett Hyde, Alan Prince, and Paul Smolensky for helpful discussions, as well as the Rutgers Optimality Reading Group, the audience at the Mathematics Association of America Seaway Section and New York Association of Two Year Colleges Fall 2007 Meeting, and the audience at the first meeting of the Northeast Computational Phonology Circle. Special thanks to Alan Prince for suggesting the name "output-driven maps." All factual errors and bizarre-sounding conceptions are the sole responsibility of the author.

A traditional approach to the issue of output-orientedness, within the theory of SPE (Chomsky and Halle 1968), is to attribute input-output disparities to phonological processes that cause the disparities, state the conditions under which the processes apply, and see the extent to which those conditions on process application can be characterized in terms of the output. In this way, phonological opacity has been characterized as a property of phonological processes relative to phonological maps (Kiparsky 1971, Kiparsky 1973). Roughly put, a process is said to be opaque if it contributes meaningfully to the analysis of a phonological map (either by applying or by not applying), but the conditions for its (non)application are not surface apparent. A diagnosis that a phonological map is not fully output-oriented is then dependent on the particular processes used in the analysis. Using phonological processes to assess output-orientedness can be particularly awkward in a theory like Optimality Theory (Prince and Smolensky 1993/2004), in which processes are not primitives of the theory, but are at best descriptive commentaries on the theory, subject to equivocation.

One can describe two directly conflicting intuitions about Optimality Theory and its capacity for realizing maps that are and are not output-oriented. One intuition is that OT is inherently output-oriented: there are two kinds of constraints, faithfulness constraints which simply attempt to preserve the input, and markedness constraints which only evaluate the output and thus are by definition output-oriented. This characterization of OT is sometimes labeled classic OT.[1] This intuition holds that OT, at least classic OT, should only be able to realize output-oriented maps, because the only constraints that can cause disparities are markedness constraints (which only evaluate outputs). The other intuition builds on a general notion of opacity as a matter of generalizations that are not surface-true. This is the view put forth by Idsardi (2000): "An opaque generalization is a generalization that does crucial work in the analysis, but which does not hold of the output form." This view of output-orientedness (via opacity) leads to the intuition that OT is inherently **not** output-oriented: generalizations in OT are expressed by constraints, constraints are inherently violable, and in practice nearly any worthwhile OT analysis involves at least some constraint violation. The view that, at least, OT markedness constraints which are violated in surface forms constitute generalizations which do not hold of the output form has been stated explicitly by McCarthy (1999, p. 332) and Idsardi (2000, p. 342).

The view that markedness constraints which are violated in grammatical forms constitute generalizations that don't hold of surface forms is straightforward. However, restricting expressed generalizations to those that are surface-true doesn't seem to do justice to the intuition of "output-oriented". To take an example from McCarthy, he offers, as an example involving a generalization that isn't surface-true but can be handled by OT, a pattern in which syllables always have onsets except for syllables occurring word-initially.[2] To fully describe the map, we have to state what happens to inputs that don't satisfy the pattern. Suppose, for the sake of discussion, that consonants are inserted into onset position to ensure that non-initial syllables have onsets. Thus, the map includes mappings like /tika/ → [ti.ka], /ika/ → [i.ka], /tia/ → [ti.ka], and /ia/ → [i.ka]. While the word-initial syllables certainly constitute violations of the constraint ONSET, the map itself nevertheless seems rather output-oriented: consonants are inserted (creating disparities) only when necessary to ensure that non-initial syllables have onsets (an output condition). The pattern seems output-oriented in a way that patterns like chain shifts are not.

A possible source for these conflicting intuitions is the distinction between properties of phonological maps themselves and properties of generalizations made about maps. The first intuition, that OT is inherently output-oriented, appears to focus more on the map itself. Disparities only happen in order to satisfy markedness constraints, and markedness constraints only evaluate outputs. This intuition makes no reference to the nature of markedness constraints, save for the fact that they only evaluate output forms.

---

[1] The term "classic OT" appears in (McCarthy 1999), for example.
[2] This is a piece of a more complex illustration given by McCarthy (1999, fn. 1) illustrating different kinds of non-surface-true generalizations.

In particular, it says nothing about whether markedness constraints are actually "surface true" in a language, this is, whether or not they are actually violated in a language. The second intuition, that OT is inherently not output-oriented, is focused on whether markedness constraints are violated in grammatical forms; this intuition is much more concerned with the status of the generalizations that the markedness constraints are taken to express.

The discussions on opacity in the work by Kiparsky, McCarthy, and Idsardi all focus on properties of the generalizations. Since phonology is in the business of characterizing generalizations about phonological systems, such a focus is both expected and justified. What is missing, however, is a more abstract characterization of output-oriented phonological maps themselves. There are intuitions that some phenomena are or are not inherently output-oriented, independent of the narrow confines of a particular theoretical analysis. The intuitions of output-orientedness involving identity maps and the consistent introduction of disparities concern properties of maps themselves, not properties of processes or constraints.

1.2 Characterizing Output-Oriented Phonological Maps

If interesting properties of maps can be identified, then various theoretical devices can be evaluated with respect to those properties. This is similar to what is accomplished by the Chomsky hierarchy of formal languages (Chomsky 1959): language-generating devices can be evaluated for their generative capacity, based on their (in)ability to generate the languages of the independently defined language classes of the hierarchy. The main goals of this paper are to state a formal property that captures intuitions of output-orientedness of maps, and then evaluate some theoretical devices (specifically, components of Optimality Theoretic systems) with respect to their capacity for helping to define output-oriented and non-output-oriented maps.

The first task pursued in this paper is to characterize output-orientedness in purely representational terms, that is, as a property of the phonological map independent of any particular analysis in terms of processes or constraints.

(2) A phonological map will be said to be **output-driven** if for any mapping from an input to an output, any other input that has greater similarity to the output also maps to the same output.

The separate terminology is employed to distinguish the intuition from the formal concept: "output-orientedness" refers to the general intuition of disparities being motivated by output conditions, while "output-driven" refers to a formally defined property of maps.

The concept of output-driven map fits the intuitions about output-orientedness given in (1). For any grammatical mapping from input to output with some set of disparities, the input identical to that output must map to that same output: a form (as an input) clearly has greater similarity to itself (as an output) than any other input form has. Thus, output-driven maps will necessarily have the identity map property, as described in (1)a. If the disparities between an input and its output are "motivated" by restrictions on output forms, then any input which has only a subset of those disparities with the same output should map to that same output; there are a strict subset of obstacles to the same destination. This is the sense in which output-driven maps satisfy the intuition in (1)b: the non-arbitrariness lies in the fact that inputs capable of reaching the same output with a subset of the disparities in fact do so, rather than mapping to a different output via a different set of disparities (no matter how well-formed that other output might be). In fact, requiring maps to be output-driven is a stronger condition than merely requiring them to have the identity map property. An example of a phonological map that has the identity map property but is not output-oriented is given in section 2; others are given later in the paper. The distinction between output-drivenness and the identity map property lies in the patterning of the non-identity mappings: output-drivenness imposes conditions on the mapping of all inputs, not just those that are identical to well-formed outputs.

The concept of output-driven map developed here is independent of the theory used to define maps relating the inputs to the outputs (it does not presume SPE-type ordered rules, Optimality Theory, or any other such theory). It **is** dependent on the representational commitments used to define the linguistic inputs and outputs, including correspondence relations between the inputs and outputs related by maps. The dependence on representational commitments includes the characterization of input/output disparities. Thus, it would be a large mistake to suggest that evaluating a map as output-driven or not is “theory-independent” in the more general sense of theory; the (non)output-drivenness of a phonological map will be heavily dependent on representational commitments. Section 3 gives a specific set of representational commitments, for purposes of illustration, and then gives a precise definition of output-driven maps in terms of those commitments.

The concept of output-driven maps is not a representational way of arriving at the same conclusions as process opacity; the two diverge in interesting ways. This is discussed in section 4, with focus on an illustration in which the same phonological map can be characterized in terms of processes as opaque or not, depending on the individual processes used in the analysis. Both analyses use the same representational assumptions, so the map itself cannot be both output-driven and not; only a single judgment is rendered. However, it is the case that many of the phenomena analyzed in this paper as not output-driven have also been previously analyzed in terms of opaque processes.

Formal conditions that characterize entire phonological maps, in addition to being of interest for phonological theory, have great potential significance for theories of language learning. Interest in identity maps arose in the study of phonotactic learning: if it could be assumed that the target grammar mapped each well-formed (grammatical) linguistic form to itself, then a learner could start out assuming that the underlying form for each observed surface form was identical to said surface form, and thereby learn non-trivial things about the map. Output-drivenness is a stronger condition, and has significant implications for later, non-phonotactic stages of learning, in particular the learning of underlying forms, as discussed in section 8.4.

1.3 Output-Orientedness and Optimality Theory

Once we have a formal characterization of output-orientedness, in the form of the concept of output-driven map, we can evaluate different Optimality Theoretic systems in terms of their ability to generate output-driven and non-output-driven maps. As will be seen in later sections, this evaluation of OT systems will effectively be an extension of the theoretical understanding of faithfulness in OT. If we accept output-driven maps as a formal characterization of output-orientedness, then neither of the simple intuitions about Optimality Theory holds up; grammars defined by Optimality Theoretic systems can describe output-driven maps and non-output-driven maps.

The present work derives properties of Optimality Theoretic systems that are sufficient to ensure that all maps defined by such a system are output-oriented. The conditions ensuring that the phonological maps of an OT system are output-driven are derived directly from the definition of Optimality Theory in section 5. These conditions break into separate conditions on GEN (the candidates) and on CON (the constraints). If GEN meets the sufficient conditions, it is said to be **correspondence uniform** (the property is defined in section 5.2.1). Of particular interest is the fact that the conditions on the constraints apply separately to each constraint. A constraint which meets the sufficient conditions is said to be **output-driven preserving**, or **ODP** (the property is defined in section 5.2.2.). The maps definable by an OT system are guaranteed to be output-driven if GEN is correspondence uniform and all of the constraints are ODP. Proofs that a variety of basic OT constraints are ODP are given in section 6 and in the appendix.

If a constraint is not ODP, then it must exhibit one of three possible non-ODP behaviors. If a map is not output-driven, and GEN is correspondence uniform, the OT system must have at least one constraint that is not ODP, and the failure of the map to be output-driven must be a consequence of the non-ODP behavior of at least one non-ODP constraint in the grammar. The properties of correspondence uniformity (for GEN) and output-driven preserving (for CON) provide a unified understanding of various proposals for handling within OT maps that are not output-oriented, and provides the desired theoretical understanding behind intuitions concerning OT and output-oriented maps. Section 7 includes several examples of maps that are not output-driven, and in each case the non-output-drivenness is traced to a specific non-ODP constraint behavior. The characterization of constraints that are not ODP unifies our understanding of a number of different proposals within Optimality Theory that address phenomena analyzed in terms of process opacity, including constraint conjunction and sympathy theory.

1.4 Overview of This Paper

To summarize the goals of this paper:

- Define the concept of output-driven map, formally characterizing the notion of output-orientedness in a purely representational fashion.
- Derive properties of Optimality Theoretic systems that are sufficient to ensure that all of the maps defined by such systems are output-driven.
- Show that constraints previously proposed to achieve “opacity effects” that are not output-driven always achieve such effects by virtue of behavior that violates the derived sufficient conditions on constraints.

The intuitions behind the concept of output-driven maps are further discussed in section 2. A general definition of output-driven maps is given in section 3, along with a more detailed specification based upon specific representational assumptions. Section 4 demonstrates the difference between process opacity and output-drivenness of phonological maps.

Much of the rest of the paper focuses specifically on Optimality Theory. Section 5 derives sufficient conditions on OT systems ensuring that all grammars definable in the system are output-driven. Section 6 discusses “basic” constraints, demonstrating that some are ODP, while others are not. Supporting formal analysis for the results of section 6 can be found in the appendix. The results reveal a principled distinction between markedness constraints and faithfulness (more generally, input-referring) constraints with respect to output-drivenness. Markedness constraints are inherently ODP; any non-output-driven behavior attributed to the constraints must be a consequence of the behavior of input-referring constraints. The basic faithfulness constraints MAX, DEP, and IDENT are proven to be ODP. The significance of value-restricted faithfulness constraints, such as IDENT constraints restricted to apply only to input segments with a particular value for a feature, is also demonstrated.

Section 7 investigates non-ODP constraint behavior in more detail, and shows that a number of constraints proposed to achieve “opacity effects” do so by virtue of exhibiting non-ODP behavior. The kinds of constraints discussed in this section include conjoined constraints (both faithfulness conjoined with faithfulness, and faithfulness conjoined with markedness), disjoined constraints, simple antifaithfulness constraints, positional faithfulness constraints, and sympathy constraints.

Section 8 provides discussion of a number of issues. Included are ideas for extending the formal results to cases with weaker representational assumptions, involving non-unique correspondence (coalescence and breaking), autosegmental representations, and different possible relationships between individual maps and entire grammars. The possible role of output-drivenness in language learning is also discussed. The take-away moral for language learning is that for learning to be possible, the linguistic theory must have some kind of non-trivial structure connecting the candidates that can be exploited by a learner. The definition of output-driven maps is a first cut at identifying that structure, permitting analysis of a wide range of basic phonological phenomena while also contributing significantly to efficient learning.

2 Output-driven Maps: The Basic Idea

2.1 Maps, Mappings, and Grammars

A **phonological map** is a set of structured representations comprising an input/output relation. Each member of the set is called a **mapping**, and has an input structure, an output structure, and a correspondence relation between elements of the input and the output. A phonological grammar is a phonological map. However, a grammar could additionally be defined in terms of several component maps, for example via the composition of several maps.

A phonological grammar defined in terms of an ordered set of rules is a phonological map. Additionally, each rule of the grammar itself defines a phonological map, relating inputs (the representations before the application of the rule) to outputs (the representations after the application of the rule). The grammar results from the composition of the rules in a particular procedural fashion. The term map here is intended to apply to both the individual rules and the overall composition. An Optimality Theoretic system is often used to define a complete grammar, but a grammar can also be defined as a composition of Optimality Theoretic systems (Itô and Mester 2001, Kiparsky 2003, McCarthy and Prince 1993). Optimality Theoretic systems will here be analyzed as maps, whether they are treated as defining grammars in full or are components of a larger grammatical system.

The concept of output-driven map given here can be applied to any phonological map.

2.2 Inputs of Greater Similarity Yield the Same Output

The following illustration characterizes vowels in terms of two height features: +/–low and +/–hi. Following Chomsky & Halle (1968, p. 305), the feature combination [+low, +hi] is ruled out representationally. Thus, there are three forms in the illustration: *i* [–low, +hi], *e* [–low, –hi], and *a* [+low, –hi].

Output-drivenness is based on a notion of similarity between representations. In particular, we want to be able to say when one input representation has greater similarity to an output representation than another input has. In this illustration, similarity is based on disparities between feature values. A **disparity** in feature value is an instance where corresponding input and output segments disagree in the value of a feature. *e* has greater similarity to *i* than *a* has to *i*, because *a* and *i* disagree on every feature that *e* and *i* disagree on: there is a disparity between *e* and *i* on the feature [hi], a disparity that also exists between *a* and *i*. Further, *a* and *i* have an additional disparity between them, on the feature [low], a disparity that does not exist between *e* and *i*. Similarity here is **not** simply a matter of the **number** of disparities; see section 2.3 for further explanation.

The intuitive notion of output-drivenness is this: if a given input is mapped to an output, then any other input which has greater similarity to that output must also be mapped to that output. Suppose *a* maps to *i*: /*a*/→[*i*]. If the map is output-driven, then *e* necessarily also maps to *i*: /*e*/→[*i*]. This is because *e* has greater similarity to *i* than *a* is. The map given in (3) is output-driven.

(3) Output-driven: /*a*/→[*i*] /*e*/→[*i*] /*i*/→[*i*]

The map in (4) is also output-driven. Here, there is no required relationship between /*a*/ and /*e*/. Given that /*a*/ maps to [*a*], /*e*/ does not have greater similarity to [*a*] than /*a*/ has, so the mapping /*a*/→[*a*] has no implications for the output that /*e*/ is mapped to (with respect to output-drivenness). Similarly, given that

/*e*/ maps to [*i*], /*a*/ does not have greater similarity to [*i*] than /*e*/ has, so the mapping /*e*/→[*i*] has no implications for the output of /*a*/.

(4) Output-driven: /*a*/→[*a*] /*e*/→[*i*] /*i*/→[*i*]

Note that the relationship between the mappings in an output-driven map is implicational. There is nothing about /*e*/→[*i*] by itself that makes it obligatory in an output-driven map. The map in (5) is also output-driven. Output-drivenness here is consistent with /*e*/ not mapping to [*i*] because /*a*/ does not map to [*i*].

(5) Output-driven: /*a*/→[*a*] /*e*/→[*e*] /*i*/→[*i*]

One immediate consequence of this conception is that all output-driven maps have the identity map property (Prince and Tesar 2004): any representation which is the output for some input is the output for itself. This is because any representation necessarily has greater similarity to itself than any other representation has to it. If we have /*e*/→[*i*] in an output-driven map, then the mapping /*i*/→[*i*] follows from the definition of output-driven map. This means that output-driven maps automatically disallow chain shifts; any map exhibiting chain shifts does not have the identity map property, and is not output-driven. An example of such a non-output-driven map is given in (6).

(6) Non-output-driven: /*a*/→[*e*] /*e*/→[*i*] /*i*/→[*i*]

A chain-shift is a case where the "greater similarity" input is identical to the output of the first form. There are also non-output-driven mappings in which the "greater similarity" input is distinct from both the input and the output of the first form. An example is given in (7).

(7) Non-output-driven: /*a*/→[*i*] /*e*/→[*e*] /*i*/→[*i*]

In this map, /*a*/ maps to [*i*]. /*e*/ has greater similarity to [*i*] than /*a*/ has, yet /*e*/ does not map to [*i*]. This example is significant because the map **does** have the identity map property; the grammatical outputs, [*i*] and [*e*], each map to themselves. Thus, output-drivenness is a stronger property than the identity map property. All output-driven maps have the identity map property, but not vice-versa.

The definition of output-driven map is designed to formalize the intuition that the maps in (6) and (7) are not output-oriented in the way that (3)-(5) are. Chain shifts, like the one in (6), are canonical examples of maps that aren't output-oriented because they lack the identity map property: if deviations from identity of outputs to inputs were motivated solely by conditions on the well-formedness of outputs, then any well-formed output should map to itself. If /*a*/→[*e*], it follows that [*e*] is a well-formed output, so nothing should stand in the way of /*e*/→[*e*]. The map in (7) is an identity map, but isn't purely output-oriented, because the extent of the disparity between input and output in /*a*/→[*i*] is gratuitous given that [*e*] is a well-formed output. The intuition of output-orientedness holds that the output deviates from the input **only to the extent necessary** to satisfy restrictions on the output. Since [*e*] is a well-formed output, and /*a*/ and [*e*] differ only in the value of low, while /*a*/ and [*i*] differ not only on the value of low but on the value of hi as well, the change in the value of hi in the mapping /*a*/→[*i*] is not motivated by conditions on the output. Both of these intuitions, the intuition of identity map and the intuition of deviation only to the extent necessary, follow as consequences of the definition of output-driven map.

2.3 Similarity is a Relational Notion

A key point must be emphasized here: similarity here is **not** based on the number of disparities between two representations. Similarity here is not a numeric distance metric. It is instead a relational notion: B has greater similarity to C than A has to C if every disparity between B and C has a matching disparity between A and C, but not vice-versa. To change A to something with greater similarity to C, you can only remove disparities; you cannot merely remove more disparities than you add.

Recall the vowel mapping in (4), repeated below.

(4) Output-driven: /*a*/→[*a*] /*e*/→[*i*] /*i*/→[*i*]

Now consider larger forms containing those vowels (presume for the moment that there is no consonant-vowel interaction). The mappings in (8) and (9) would be predicted.

(8) /tapeke/ → [tapiki]

(9) /tepiki/ → [tipiki]

Of interest here is the relation between the two mappings, and the fact that they do not contradict the definition of output-driven map. Specifically, we want to see why it is possible for /tepiki/ to map to an output form other than [tapiki], despite the fact that [tapiki] is the optimal output form in (8), and /tepiki/ has fewer disparities with [tapiki] than the input in (8), /tapeke/, does. In other words, we need to explain the non-grammaticality of the mapping in (10), in light of the grammaticality of the mapping in (8).

(10) * /tepiki/ → [tapiki]

In (8), the input differs from the output in the fourth and sixth segments, on the feature hi on each vowel. The disparities of mapping (8) are given in (11). The notation uses subscripts to indicate which segment of a form is being referred to: i_4 means the fourth segment (counting from the beginning) of the input form, while o_6 means the sixth segment of the output form. The disparity i_4(–hi):o_4(+hi) for the mapping in (8) indicates that the fourth segment of the input is in correspondence with the fourth segment of the output, and the correspondents disagree on the value of the feature hi, with the input correspondent having the value –hi and the output correspondent having the value +hi.

(11) Disparities of mapping (8): i_4(–hi):o_4(+hi) i_6(–hi):o_6(+hi)

The mapping in (10) has the same output as the mapping in (8), but a different input. The input in (10) matches the output in the fourth and sixth segments, but differs in the second segment on the feature low. The disparities of mapping (10) are given in (12).

(12) Disparities of mapping (10): i_2(–low):o_2(+low)

Mapping (10) has fewer feature value disparities than mapping (8); there are fewer disparities between /tepiki/ and [tapiki] than there are between /tapeke/ and [tapiki]. Yet, the input of (9) and (10), /tepiki/, does not have "greater similarity" to the output of (8), [tapiki], in the sense relevant here, because mapping (10) has a disparity that (8) lacks: the differing value of low in the second segment. Thus, the fact that the output of (9) is not the same as the output of (8) does not violate the requirements of an output-driven map, because mapping (8) does not entail mapping (10).

Similarity is based on an implicit correspondence relation between disparities in mappings. This can be further illustrated by examining an input whose output is entailed in an output-driven map by the mapping in (8). The mapping in (13) is entailed by (8) in an output-driven map.

(13) /tapeki/ → [tapiki]

The disparities of (13) are given in (14).

(14) Disparities of mapping (13): i_4(–hi):o_4(+hi)

The single disparity of (13) has a direct corresponding disparity in (8). The disparities correspond because they involve the same segment of the (shared) output, o_4, and they are the same type of disparity, a mismatch in the values of feature hi. Importantly, the disparities are also featurally the same, both having –hi in the input segment and +hi in the output segment. The input in (13) allows a simple reduction in the number of disparities with output [tapiki] relative to the input in (8). In an output-driven map, mapping (8) entails mapping (13).

2.4 The Importance of Input-Output Correspondence

The discussion given in section 2.2 is simple to follow, because the input and output forms consist of a single segment, and the input segment of a mapping always corresponds to the output segment of that mapping. The discussion in section 2.3 assumes 1-to-1, order-preserving input-output correspondences (no insertion or deletion of segments): for a given input-output pair, only one possible correspondence is discussed. In general, it is desirable to allow more freedom of correspondence between inputs and outputs in mappings. As a consequence, the correspondence relation between input and output must be a factor when characterizing output-driven maps. Intuitive notions of "similarity" between forms inevitably appeal (implicitly or explicitly) to a correspondence between the forms.

While the terminology of correspondence is found most explicitly in the OT literature, the concept is equally important to any generative theory. There is a correspondence relation implicit in every SPE-style rule (Chomsky and Halle 1968). A rule has an input representation (before the rule applies) and an output representation (after the rule applies), and a correspondence relation is assumed between them. When a rule devoices an obstruent word-finally, the word-final obstruent of the output representation corresponds to the word-final obstruent of the input representation, and the other (unaltered) segments of the output correspond to the positionally analogous segments of the input. For a deletion rule, the targeted segment of the input has no correspondent in the output. An implicit correspondence holds between the underlying representation at the start of a derivation and the surface representation at the end of the derivation, via the composition of the correspondence relations for each of the rules that apply during the derivation.

The notion of correspondence is indispensable to linguistic theorizing in general. A mapping cannot be properly characterized solely by its input and output; the correspondence relation must be specified as well. This is illustrated here with an example that draws upon Optimality Theory, but corresponding examples could likely be constructed for any plausible generative theory. Consider an underlying form /ɡa/ which surfaces as [ʔa]. This same input-output pair can give rise to rather different conclusions about a grammar, depending upon what correspondence relation is assumed.

(15) /$ɡ_1a_2$/ → [$ʔ_1a_2$]

(16) /$ɡa_2$/ → [$ʔa_2$]

The mapping in (15), with underlying /ɡ/ corresponding to surface [ʔ], violates constraints evaluating feature identity between corresponding segments, such as IDENT[F] (McCarthy and Prince 1995). The mapping in (16), on the other hand, does not have any feature value mismatches between IO correspondents, but does violate constraints against segmental insertion and deletion, such as MAX and DEP (McCarthy and Prince 1995). Each of these mappings requires different ranking relations among such constraints (assuming that these constraints are included in the linguistic system in question). In the general case, reasoning over spaces of candidates requires reasoning over not only input-output pairs, but over candidates resulting from the possible correspondence relations holding between possible input-output pairs.

Note that in an SPE-style rule system, (15) and (16) require very different derivations using very different rules, the former requiring rules for feature change and the latter requiring rules for segmental deletion and insertion. The differences between the analyses does not lie in the input and output, which are the same for both; the grammatically significant difference lies in the correspondence relation. The linguistic analysis and the correspondence relation are mutually dependent.

To see why correspondence is important for the concept of output-drivenness, consider the mappings in (17), (18) and (19). The correspondence relation for each is indicated with the subscripted indices.[3] All three mappings have the same output form.

(17) /$p_1a_2k_3a_4$/ → [$b_1a_2g_3a_4$]

(18) /$pa_2k_3a_4$/ → [$ba_2g_3a_4$]

(19) /$a_2k_3a_4$/ → [$ba_2g_3a_4$]

First compare the mappings (17) and (18). In (17), there are two disparities, the disagreements in voicing between the correspondents with index 1 and between the correspondents with index 3. In (18), there are three disparities: the disagreement in voicing between the correspondents with index 3, the /p/ in the input with no output correspondent, and the [b] in the output with no input correspondent. If the input and output representations are taken as representations on their own, with no correspondence indices, the inputs of the two mappings are identical, as are the outputs. Yet they are different mappings.

(20) Disparities of mapping (17): i_1(–voi):o_1(+voi) i_3(–voi):o_3(+voi)

(21) Disparities of mapping (18): i_1:_ _:o_1 i_3(–voi):o_3(+voi)

(22) Disparities of mapping (19): _:o_1 i_3(–voi):o_3(+voi)

The differences between (17) and (18) have non-trivial implications for the notion of "one input having greater similarity to an output than another." Compare (17) and (19). Both have the same output form. In (19), there are two disparities: the disagreement in voicing between the correspondents with index 3, and the [b] in the output with no input correspondent. Each of (17) and (19) has a disparity that the other lacks: (17) has a disagreement in voicing between the correspondents with index 1, while (19) has an output segment with no input correspondent. Looked at in this fashion, there is no obvious sense in which one of the inputs of these mappings has "greater similarity" than the other to the shared output; each differs in a different way.

Now compare (18) and (19). Mapping (18) shares both of the disparities of (19), but also has an additional one: the input segment /p/ with no output correspondent. Looked at in this fashion, the input of (19) has greater similarity than the input of (18) to the shared output. The challenge posed by this is that (17) and (18) have the same input form. Does /aka/ have "greater similarity" to [baga] than /paka/ has? You cannot answer that question as stated, because the necessary correspondence relations aren't provided. A different question needs to be asked, one that accounts for correspondence. Does (19) have a greater similarity between input and output than (18)? That question is answerable (in the affirmative) if we can formalize the intuition appealed to above of different mappings having corresponding disparities. We want to be able to support the idea that (19) has greater similarity than (18), while neither of (17) and (19) has greater similarity than the other. A formal realization of this idea is developed in section 3.

3 Definition of an output-driven Map

3.1 General Definition of Output-driven Maps

A **candidate** is an input, an output, and an input-output (IO) correspondence relation between the input and the output. A candidate with input form in_a, IO correspondence relation R_k, and output form out_x will

[3] Each subscripted index has scope only within a single mapping. However, the index numbers in each mapping have been selected to emphasize certain relationships between the three mappings.

be denoted with the term *akx*.[4] When the correspondence relation is considered to be "obvious", candidates will sometimes be denoted with a colon between the input and output, for example /in_a/:[out_x]. For the purposes of evaluating maps, some reference set of possible candidates must be used; it defines the possible linguistic representations to be considered. The set of possible candidates is here labeled the **reference representation space** (RRS). The set of possible inputs consists of those input forms that appear in at least one candidate in the RRS, while the set of possible outputs consists of those output forms that appear in at least one candidate in the RRS.

A **map** M is a function from inputs to sets of candidates.[5] The domain of M is the set of possible inputs. The codomain of M is the power set of the RRS (the possible subsets of the RRS). An important restriction on M is that the set of candidates assigned to an input by M may only contain candidates containing that input.[6] M(in_a) cannot contain a candidate with input in_b if $in_a \neq in_b$. Any candidate that is contained in the candidate set assigned to its own input is labeled a grammatical candidate. That is, *akx* is **grammatical** if and only if $akx \in$ M(in_a).[7]

Output-drivenness involves a comparison of the similarity of pairs of candidates with identical output forms. Formally, what is required is a relation, **relative similarity**, containing pairs of candidates with identical output forms, where each pair is interpreted as meaning that the second candidate of the pair has **greater internal similarity** than the first candidate of the pair. The term internal similarity signifies that the similarity between the input and output of a candidate is being referred to (similarity is "internal" to a candidate). Relative similarity is the comparison of the internal similarity of one candidate to the internal similarity of another.

Although the term "greater internal similarity" is used for convenience, the relative similarity relation is actually a partial order: reflexive, antisymmetric, and transitive. This means that each candidate is paired with itself in the relation.[8] The partial order factors into suborders, with each suborder consisting of candidates sharing an output form. Figure 1 depicts the relative similarity relation for the vowel height candidates described in section 2.2.

---

[4] The candidate label includes a separate index for the IO correspondence relation to explicitly recognize that the same input-output pair can have more than one IO correspondence relation defined on it (each relation defining a different candidate).

[5] While it might seem natural to formalize the map as a function from inputs to candidates, for the sake of generality, each input is mapped to a set of candidates, thus allowing for the possibility that an input has more than one output (or none at all). Maps for which each input has exactly one output are a special case of primary interest, where each input is mapped to a set containing exactly one candidate.

[6] Here, a candidate 'contains' an input in the sense that the input is a part of the full candidate, **not** in the sense that elements of the input are directly contained in the output. The analysis of Optimality Theory pursued here assumes a correspondence theory of faithfulness (McCarthy and Prince 1995), not a containment-based theory of faithfulness (Prince and Smolensky 2004).

[7] The term grammatical is here used for convenience. More generally, a map might be used as part of a larger grammar, such that a candidate mapped to by M might be an intermediate form, rather than a form that is grammatical in the sense of the overall grammar.

[8] The slightly more accurate but much more cumbersome phrase "internal similarity which is at least as great" is being avoided here.

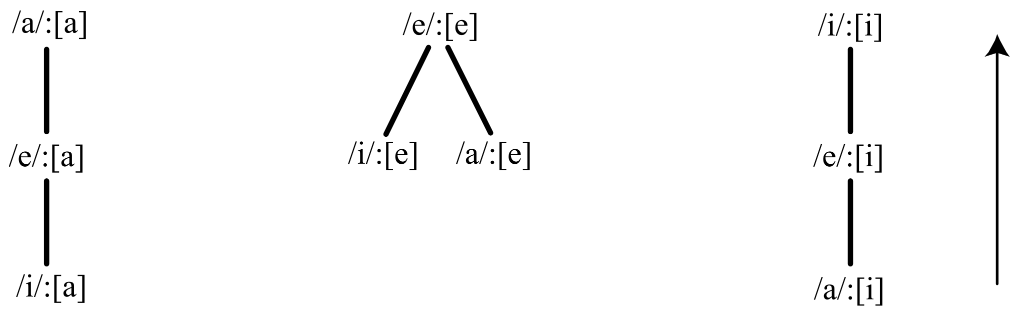

Figure 1 Relative similarity relation (upward is greater internal similarity)

Given a reference representation space RRS and a relative similarity relation RSIM, an output-driven map is defined as follows. M is an **output-driven map** if, for all candidates, the grammaticality of *akx* entails the grammaticality of every candidate *bmx* in RRS with greater internal similarity than *akx*. More tersely, letting input(c) denote the input representation for candidate c, M is output-driven if

(23) $\forall c \in RRS, \forall d \in RRS\ [(c \in M(input(c))\ \&\ (c,d) \in RSIM) \rightarrow (d \in M(input(d))]$

For convenience, some of the terminology used here comes straight from the OT literature. However, the concepts are not parochial to OT. As explained in section 2.4, any generative theory works in terms of inputs, outputs, and correspondences between them. The label "candidate" might suggest a comparative evaluation mechanism, but does not require it.

Given this formulation, a map like (3) might be written as shown in (24).

(24) /*a*/→{/*a*/:[*i*]} /*e*/→{/*e*/:[*i*]} /*i*/→{/*i*/:[*i*]}

To simplify the presentation (and match standard notation), in this paper mappings will be denoted with an arrow from an input to the output of the single grammatical candidate for that input, as is done in (3).

The general definition of output-drivenness given in this section leaves unspecified the reference representation space and the relative similarity relation. The RRS will vary depending upon the linguistic system being investigated. A fully defined RSIM is dependent upon the reference representation space employed. However, more can be specified about the RSIM without full knowledge of the RRS, along the intuitive lines presented in section 2. Such a specification is given in sections 3.2 and 3.3.

3.2 Identity-Based Relative Similarity

In this section a more specific notion of similarity is developed, based on identity relations between particular representational elements in candidates. The **internal similarity** of a candidate is an evaluation of the similarity between the input and the output of the candidate. It will actually be more natural to think in terms of **disparities** in a candidate: the ways in which the output of the candidate differs from the input of the candidate. Disparities in a candidate are differences between corresponding input and output elements, and divergences of the IO correspondence relation from a bijection (representational elements not having exactly one correspondent).

The **relative similarity** of pairs of candidates can then be derived from the internal similarity of each individual candidate. Such a comparison is based on a constructed correspondence relation between the disparities of both candidates. Candidate *bmx* has **greater internal similarity** than *akx* if (i) the candidates have identical output forms; (ii) every disparity in *bmx* has an "appropriate" corresponding disparity in *akx*. This is the relational conception of similarity discussed in section 2.3. Section 3.3 gives a detailed definition of correspondence between the disparities of the two candidates.

I here propose an **identity-based relative similarity** in which the "appropriateness" of corresponding disparities between two candidates is based on identity of the disparities. An identity-based relative similarity requires that each disparity in the greater internal similarity candidate be identical to the corresponding disparity in the other candidate. This avoids notions of similarity in which one might speak of a single disparity in one candidate being "less than" its corresponding disparity in another candidate; in an identity-based relative similarity, if two corresponding disparities are not identical, then they cannot sustain a relationship of greater internal similarity between their respective candidates.[9]

Recall the map in (3), repeated here.

(3) Output-driven: /*a*/→[*i*] /*e*/→[*i*] /*i*/→[*i*]

This map has three grammatical candidates, /*a*/:[*i*], /*e*/:[*i*], and /*i*/:[*i*]. Each vowel has two features, low and hi. Since the correspondence relations are all bijections (each input segment has exactly output correspondent, and vice-versa), the only disparities in the candidates of this map are feature identity disparities (disagreeing feature values). The candidate /*a*/:[*i*] has two disparities: the +low:–low feature disagreement and the –hi:+hi feature disagreement. The candidate /*e*/:[*i*] has one disparity: –hi:+hi. The two candidates have a corresponding disparity: –hi:+hi. The corresponding disparities are identical: they have the same feature value on the input side, –hi, and the same feature value on the output side, +hi.

An identity-based relative similarity assumes an **inventory of disparities**: the representational configurations that constitute individual instances of disparity.[10] The preceding discussion has implicitly assumed that disparities are differing values of individual features, differing between corresponding input and output segments, as well as individual segments of the input or output that lack IO correspondents. That is the inventory of disparities that will be used throughout most of this paper. It is worth pointing out, however, that the choice of an inventory of disparities can have a non-trivial effect on whether a map is judged to be output-driven or not.

The importance of the inventory of disparities can be illustrated by revisiting the situations presented in examples (3)-(7) using a different representational theory. Suppose segments are characterized atomically, without any reference to features. In this representational theory, *a*, *e*, and *i* are three distinct segments with no internal structure to refer to. In such a theory, the possible disparities involving corresponding input and output segments is quite limited: two segments are either identical, or they are not. The disparities involving corresponding elements are defined solely with reference to segments. Thus, with this inventory of disparities, the candidate /*a*/:[*i*] has only one disparity, the corresponding non-identical segments *a*:*i*. The candidate /*e*/:[*i*] also has one disparity, but it is not identical to the (single) disparity of /*a*/:[*i*]. Disparity *a*:*i* is not the same as disparity *e*:*i*; the two disparities have different values on the input side.

Under this representational theory, maps (3)-(5) are output-driven, like before. Map (6) is non-output-driven, like before: /*a*/ maps to [*e*], and /*e*/ has greater similarity to [*e*] than /*a*/ has, yet /*e*/ does not map to [*e*]. Chain shifts are unavoidably non-output-driven, so long as a phonological form (like [*e*]) has greater similarity to itself than any other form has. However, map (7), repeated below, is output-driven under the new representational theory.

(7) /*a*/→[*i*] /*e*/→[*e*] /*i*/→[*i*]

[9] In later sections, it will prove useful in contexts apart from the evaluation of relative similarity to have non-identical disparities correspond to each other.

[10] There is a similarity of spirit between the representational concept of disparity as used here, and McCarthy's (McCarthy 2007b) notion of basic faithfulness constraint violations, which are the presupposed basis for localized unfaithful mappings (LUMs) in OT-CC.

/*a*/:[*i*] is grammatical, but /*e*/ does not have greater similarity to [*i*] than /*a*/ has, both are simply non-identical to [*i*]. Neither of the candidates /*a*/:[*i*] and /*e*/:[*i*] has greater internal similarity than the other. One could consider the disparities *a*:*i* and *e*:*i* to be corresponding, as they involve the same output segment, but they are not **identical** disparities. The disparity *e*:*i* of /*e*/:[*i*] does not have an identical corresponding disparity in /*a*/:[*i*]. Therefore, under the new representational theory, the /*a*/→[*i*] mapping is not relevant to the mapping for /*e*/ with respect to output-drivenness.

The two schemes for relative similarity just described are both based on identity of disparities. They differ on the inventories of disparities. The first approach individuates disparities between corresponding elements on the level of features, while the second approach individuates disparities between corresponding elements on the level of segments. It seems to natural to expect that the inventory of disparities would derive directly from the linguistic representational theory being used. If features are included in linguistic representations, in particular if features play a linguistically efficacious role in the evaluation of input-output correspondences, then one would expect disparities to be individuated with respect to feature values (at least in part). If a linguistic theory includes input-output correspondences directly between autosegments, then one would expect that the inventory of disparities would include individual autosegments (independent of any segmental affiliation) with no IO correspondents.

3.3 Segment-Based Input-Output Correspondence

For the sake of concreteness, the analyses in the rest of this paper will assume that the conditions laid out in section 3.3.1 are true of the reference representation space. An inventory of disparities is derived directly from that space of representations, which in turn provides the basis for a fully concrete characterization of output-driven maps, given in sections 3.3.2 and 3.3.3.

3.3.1 Representational Conditions and an Inventory of Disparities

Recall that the reference representation space contains the representations (candidates) that will be referred to in the evaluation of maps. Input-output correspondence is here assumed to be segment-based: the input-output correspondence relation for a candidate relates input segments to output segments, following McCarthy and Prince (1995); for an alternative view, see Lombardi (2001). Nothing here conflicts with prosodic structure in the output, so long as nonsegmental prosodic elements do not stand in input-output correspondence.[11] The basic assumed conditions are given in (25). Issues concerning representational theory are further discussed in section 8.1.

[11] There is no conflict with prosodic structure in the input, either, provided the same restriction on correspondence holds; however, some complications could arise. If the grammar does not make reference to prosodic structure in the input, then such structure is completely inert and has no impact on the grammar. If the grammar does potentially make reference to such input structure, then it could distinguish candidates with identical corresponding disparities (the candidates would be distinguished by the input structure that does not stand in IO correspondence, and therefore cannot constitute disparities). This is not necessarily a problem, but it would mean that the relative similarity relation would no longer be a partial order, because it would not be antisymmetric: two non-identical candidates could each have greater internal similarity than the other.

(25) Basic conditions on the reference representation space

- Inputs and outputs each contain a string of segments.
- Segments are characterized by features.
- Candidates have only a single correspondence relation between the input and the output, one that relates only segments (input and output segments can stand in correspondence, but no other representational elements can).
- Any type of segment (characterized in terms of its feature values) in the input can correspond to any type of segment in the output.

To keep the formal analysis tractable for purposes of this paper, the following additional conditions are also assumed. Both are restrictions on input-output correspondence.

(26) Additional conditions relating to input-output correspondence

- Segments (both input and output) can have at most one correspondent (no coalescence or breaking).
- For any pair of output segments with input correspondents, the order of the segments in the output must match the order of their input correspondents (no metathesis).

These conditions are not intended to be the final word on representational theory; they are intended as a good starting point for investigating output-driven maps. The additional complexities involved in relaxing these conditions are discussed in section 8.1.2.

The representational conditions serve in part to limit the kinds of disparities that can occur in candidates. The types of disparities are given in (27).

(27) The types of disparities

- deletion: an input segment with no output correspondent.
- insertion: an output segment with no input correspondent.
- feature identity: a difference in the values of some feature for corresponding input and output segments.

The inventory of disparities consists of the possible disparities of any of these types. For the deletion type of disparity, there will be one disparity in the inventory for each possible segment. A segment /*a*/ without a correspondent is a distinct (non-identical) disparity from a segment /*e*/ without a correspondent, although both are of the same type with respect to (27). Analogously, for the insertion type of disparity, there will be one disparity in the inventory for each possible segment. For the feature identity type of disparity, there will be one disparity in the inventory for each possible combination of non-identical input and output values for a feature.

3.3.2 Input-Input Correspondence and Disparity Instance Correspondence

Suppose we are given a pair of candidates *bmx* and *akx* with identical output form out_x, and input forms in_b and in_a respectively. An analysis of the relative similarity between *akx* and *bmx* requires that a correspondence be established between the disparities of the two candidates. The anchor of the relationship between the two candidates is the shared output form.[12] In order to relate the disparities of the candidates, we have to relate the segments of the candidates. Since the two candidates have identical output forms, it is easy to relate the output segments of the candidates: each segment of the output in one candidate relates to "itself" in the other candidate.

Relating the disparities of the two candidates involves comparing the fates of corresponding segments in the two candidates. How an output segment is handled in one candidate will be directly related to how that same output segment is handled in the other candidate. If an output segment is inserted in one candidate, say *akx*, that constitutes an insertion disparity in *akx*. That insertion disparity will have a corresponding disparity in *bmx* if and only if the same output segment is also inserted in *bmx*. The two insertion disparities can correspond only if they affect the same output segment.

Deletion disparities concern input segments, and feature identity disparities concern both input and output segments. Therefore, comparing these kinds of disparities of the two candidates involves creating an input-input correspondence relation between the inputs of the candidates. The input-input correspondence relation, which will be denoted R_{II}, is not a part of a grammar, nor is it a part of any candidate; it is a structure constructed as part of an analysis of output-drivenness. In general, the input forms of the candidates will not be identical, so defining the correspondence between the input forms is not as trivial as the one between the (identical) output forms. Relative similarity is intrinsically concerned with how two different candidates relate to the same output form, so the input-input correspondence is based upon the input-output relations between each of the input forms and the output form.

Segments of the respective inputs can only be input-input correspondents if they relate to the output form in the same way. Thus, input segments can be input-input correspondents only if they either (a) both have the same output segment as their respective output correspondents, or (b) both lack output correspondents. The input-input correspondence is important to the establishment of a correspondence between disparities: a pair of disparities from the two candidates can correspond only if the disparities involve segments that correspond between the two candidates. Deletion disparities can only correspond if the respective deleted input segments are input-input correspondents. Feature identity disparities can only correspond if they involve corresponding input segments (input-input correspondents) and corresponding output segments (an identical segment of the shared output form).

The input-input correspondence R_{II} between in_b and in_a must satisfy several conditions; these are listed in (28).

[12] Technically, an order-preserving bijective correspondence is assumed between the identical outputs of the two candidates, matching the first segments to each other, and so forth. When comparing candidates with identical outputs, I will write as if there were a single output, to cut down on the wordiness of the discussion.

(28) Conditions on the input-input correspondence relation R_{II}

- As with the input-output correspondence, the relative order of corresponding segments in the input-input correspondence cannot be reversed, and segments can have at most one correspondent.
- An input segment with an output correspondent in one candidate has an input-input correspondent **if and only if** that output segment has an input correspondent in the other candidate. For instance, segment s_b of in_b with output correspondent s_x of out_x in *bmx* has input-input correspondent s_a in in_a if and only if s_a has output correspondent s_x in *akx*. Otherwise, s_b has no input-input correspondent in in_a. Put concisely:
  - if $s_b R_m s_x$, then ($s_a R_{II} s_b$ if and only if $s_a R_k s_x$).
  - if $s_a R_k s_x$, then ($s_a R_{II} s_b$ if and only if $s_b R_m s_x$).
- A segment s_b of in_b with no output correspondent in *bmx* can only have an input-input correspondent s_a of in_a if s_a has no output correspondent in *akx*.

Given an input-input correspondence satisfying the conditions in (28), a correspondence between the disparities of the two candidates can be constructed as described in (29).

(29) Constructing a correspondence between the disparities of *akx* and *bmx*

- Let s_b:_ be a deletion disparity in *bmx*. This disparity has a corresponding disparity s_a:_ in *akx* if and only if s_b has input-input correspondent s_a in *akx* (and thus s_a necessarily has no output correspondent in *akx*, by the conditions on input-input correspondence).
- Let _:s_x be an insertion disparity in *bmx*. This disparity has a corresponding disparity _:s_x in *akx* if and only if s_x has no input correspondent in *akx*.
- Let s_x be an output segment of *bmx* with an input correspondent s_b such that s_b and s_x differ on the value of feature F. This disparity in *bmx* has a corresponding disparity in *akx* if and only if s_x has an input correspondent s_a in *akx* such that s_a and s_x differ on the value of feature F (s_b and s_a are then necessarily input-input correspondents by the conditions on input-input correspondence).

These constructions are illustrated in the following example. To simplify the example, I will only be concerned with the segmental features hi and low (as in the earlier examples of this paper). In order to adequately describe the example, some further notational conventions need to be introduced. For a string of segments denoted by form index *x*, denote the first segment as x_1, the second segment as x_2, etc. Thus, if output form out_x = [*tibi*], denote the segments of out_x as $x_1 = t$, $x_2 = i$, $x_3 = b$, $x_4 = i$. This indexing of the segments is useful for describing the various correspondence relations that hold between different forms.

Let *akx* be a grammatical candidate with the components listed in (30).

(30) in_a = /*agbik*/ out_x = [*tibi*] $R_k = \{(a_1,x_2), (a_3,x_3), (a_4,x_4)\}$

Let *bmx* be the candidate with the components listed in (31).

(31) in_b = /*tebik*/ out_x = [*tibi*] $R_m = \{(b_1,x_1), (b_2,x_2), (b_3,x_3), (b_4,x_4)\}$

Figure 2 shows the candidates, with their IO correspondences indicated by lines. At the bottom, the input-input correspondence between *akx* and *bmx*, described next, is also shown.

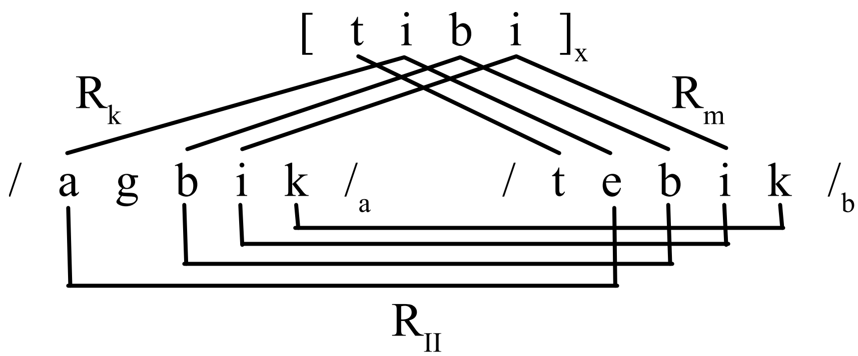

Figure 2 Candidates *akx* and *bmx*, and their input-input correspondence.

To support the claim that *bmx* has greater internal similarity than *akx*, a correspondence between the disparities of the candidates is required. That, in turn, requires the construction of an input-input correspondence R_{II} between in_a and in_b. R_{II} is shown in (32).

(32) $R_{II} = \{(a_1,b_2), (a_3,b_3), (a_4,b_4), (a_5,b_5)\}$

The first three pairs of R_{II} are based on common output correspondents: both a_1 and b_2 correspond to x_2, etc. The last pair of R_{II}, (a_5,b_5), is possible because neither segment has an output correspondent and their correspondence doesn't violate the ordering requirements of (28). Further, the two segments are identical (both are *k*), which is necessary for this input-input correspondence to result in identical corresponding disparities.

Examining the correspondence between the disparities of *akx* and *bmx* requires yet more notation. Let a_1:_ denote the disparity in which input segment a_1 has no output correspondent. Let _:x_1 denote the disparity in which output segment x_1 has no input correspondent. Let a_1(–hi):x_2(+hi) denote the disparity in which input segment a_1 has the feature value –hi and its output correspondent x_2 has the feature value +hi.

Candidate *akx*, as described in (30), has the disparities listed in (33).

(33) _: x_1 a_1(–hi):x_2(+hi) a_1(+low):x_2(–low) a_2:_ a_5:_

Candidate *bmx*, as described in (31), has the disparities listed in (34).

(34) b_2(–hi):x_2(+hi) b_5:_

Candidate *akx* has five disparities, while *bmx* has two. Following (29), A correspondence between the disparities of the two candidates can now be constructed, given in (35).

(35) Correspondence between the disparities of *akx* and *bmx*

- a_1(–hi):x_2(+hi) of *akx* corresponds to b_2(–hi):x_2(+hi) of *bmx*
- a_5:_ of *akx* corresponds to b_5:_ of *bmx*

Disparity a_1(–hi):x_2(+hi) of *akx* corresponds to b_2(–hi):x_2(+hi) of *bmx*, because a_1 and b_2 are input-input correspondents ($a_1 R_{II} b_2$), and both disparities are feature identity disparities involving the feature hi. Disparity a_5:_ of *akx* corresponds to b_5:_ of *bmx*, because a_5 and b_5 are input-input correspondents ($a_5 R_{II} b_5$), and both disparities are deletion disparities.

To support a claim of greater internal similarity, corresponding disparities must be "featurally" the same. Consider two disparities both involving input segments that disagree in the feature F with their (common) output correspondent. If feature F has more than two possible values, it is not sufficient that the pair of disparities each be such that the feature value on the input side is different from the value on the output side; the two disparities must have the same value for F on the input side. Similarly, two disparities consisting of input segments with no output correspondents can only correspond if the input segments have the same values for all features (they are the same segment type). Disparities that are not featurally the same could be distinguished solely on the basis of output conditions, so treating them as equivalent for purposes of evaluating relative similarity would run counter to the motivations for the definition of output-driven maps. For instance, a disparity in which an output coda segment with coronal place has an input correspondent with dorsal place is distinct from a disparity in which the output coda segment with coronal place has an input correspondent with labial place. Both disparities could involve the same output segment but still involve different output conditions (a ban on dorsals in coda position vs. a ban on labials in coda position). By contrast, if two disparities are featurally the same, and relate in the same way to the same output form, then any output condition must relate to both disparities in the same way.

The conditions on input-input correspondence in (28) are motivated by the role they play in defining correspondence between disparities. The second condition reflects the fact that the primary basis for correspondence between the two candidates is their shared output form. If a segment in in_a and a segment in in_b both have the same output segment as their output correspondents, they should be input-input correspondents; they are playing corresponding roles in the two candidates by virtue of the fact that they correspond to the output in the same way. Any feature identity disparity involving feature F for one of the input segments should correspond to a feature identity disparity involving feature F for the other input segment; making the input segments input-input correspondents accords with the correspondence between the disparities sharing the same output segment. The third condition restricts input segments with no output correspondents to input-input correspondence with other segments with no output correspondents: the segments then have the same relation to their respective outputs (lack of output correspondents), and constitute the same sort of disparity. Input-input correspondence between segments lacking output correspondents supports a correspondence between deletion disparities in the two candidates. The first condition simply keeps input-input correspondence in accord with the conditions assumed on input-output correspondence; if one were to relax this condition on input-output correspondence, then presumably one would do the same for input-input correspondence. Since disparities in candidates are based on input-output correspondence, the correspondence between the disparities of the two candidates should respect the same restrictions on ordering and multiple correspondence.

It is worth noting that the conditions on input-input correspondence do not always uniquely determine an input-input correspondence for a pair of candidates sharing the same output. The indeterminacy involves input-input correspondence between segments lacking output correspondents. While the conditions limit the input-input correspondence of segments lacking output correspondents to other segments lacking output correspondents, it does not oblige such segments to have input-input correspondents, even if a possible input-input correspondent is available. Of course, if each of a pair of candidates has an input segment lacking an output correspondent and lacking an input-input correspondent, then each candidate has a disparity with no corresponding disparity in the other candidate, and neither candidate can have greater internal similarity than the other with respect to the selected input-input correspondence. In order to sustain a “greater internal similarity” relation between two candidates, the selected input-input correspondence much be such that all input segments of the greater internal similarity candidate lacking output correspondents must have input-input correspondents, so that all of the deletion disparities of the greater internal similarity candidate have corresponding disparities in the lesser internal similarity candidate.

Under special conditions, more than one input-input correspondence can sustain a “greater internal similarity” relation for the same pair of candidates. This can be illustrated by slightly altering the previous illustration. Let *akx* and *bmx* be as follows.

(36) in_a = /*agbikk*/ out_x = [*tibi*] $R_k = \{(a_1,x_2), (a_3,x_3), (a_4,x_4)\}$

(37) in_b = /*tebik*/ out_x = [*tibi*] $R_m = \{(b_1,x_1), (b_2,x_2), (b_3,x_3), (b_4,x_4)\}$

The only difference is the additional segment at the end of in_a. Importantly, this segment has no output correspondent, and is identical to the preceding segment, which also has no output correspondent. This creates the opportunity for two equivalent but non-identical input-input correspondences, because the final segment of in_b, b_5, can have either a_5 or a_6 as an input-input correspondent. Either way, the disparity b_5:_ in *bmx* has a corresponding disparity in *akx*: it corresponds to either a_5:_ or a_6:_.

Because of the above observations, the definition of "greater internal similarity" given below is expressed in terms of the existence of an appropriate input-input correspondence, rather than in terms of a uniquely defined input-input correspondence.

3.3.3 Specific Characterization of Output-drivenness

(38) Definition: *bmx* has **greater internal similarity** than *akx* if there exists a correspondence between the disparities of the two candidates, satisfying the conditions in (29), such that every disparity in *bmx* has an **identical** corresponding disparity in *akx*.

This definition of greater internal similarity has some consequences that may not be immediately obvious, but are important.

(39) If an output segment s_x has an input correspondent s_b in *bmx* but not in *akx*, then *bmx* may have greater internal similarity than *akx* only if s_b and s_x are identical. If s_b and s_x aren't identical, then the feature value mismatches distinguishing the segments constitute feature identity disparities that have no corresponding disparities in *akx*.

(40) If *bmx* has greater internal similarity than *akx*, then every segment in in_a with an output correspondent has an input-input correspondent in in_b. If segment s_a of in_a had an output correspondent s_x in out_x but no input-input correspondent, then s_x would have no input correspondent in *bmx*, which would constitute a disparity in *bmx* with no correspondent in *akx*, contradicting the premise that *bmx* has greater internal similarity than *akx*. Contrapositively speaking, every segment in in_a without an input-input correspondent in in_b has no output correspondent.

(41) If *bmx* has greater internal similarity than *akx*, then for every segment s_a in in_a without an output correspondent, either s_a has an input-input correspondent s_b that also has no output correspondent and is identical to s_a, or s_a has no input-input correspondent. If s_a had a non-identical input-input correspondent s_b, then the corresponding deletion disparities would be non-identical, contradicting the premise that *bmx* has greater internal similarity than *akx*.

For the candidates *akx* and *bmx* given in (30) and (31), *bmx* has greater internal similarity than *akx* because both of the disparities for *bmx* have corresponding disparities for *akx*, as described in (35).

Consistent with the general definition given in (23), a map M is **output-driven** if, for each grammatical candidate *akx*, it is true that for every candidate *bmx* that has greater internal similarity than *akx*, *bmx* is also grammatical.

Because the pairs of candidates being compared in the definition of output-driven maps have identical outputs, they differ in their inputs and in their input-output correspondence relations. Intuitively, *bmx* has greater internal similarity than *akx* if it is possible to get the input form for *bmx* by taking the input form for *akx* and changing it in ways that give it greater similarity to the output form.[13] Because similarity is

[13] In principle, it is possible for two candidates with the same input form to stand in a greater internal similarity relationship. This requires that the greater similarity candidate have a pair of identical IO corresponding segments

characterized in terms of corresponding disparities, changing the input must have the effect of eliminating disparities without introducing any new ones. Thus, we can itemize the kinds of "changes" that can result in greater similarity by identification with the disparities that those changes eliminate. Removing an input segment that has no output correspondent eliminates a deletion disparity. Inserting an input segment such that it corresponds to an output segment previously lacking an input correspondent eliminates an insertion disparity, and if the added input segment is identical to its output correspondent then no new feature identity disparities are introduced. If the value for a feature of an input segment is changed to match the value of its output correspondent, that eliminates a feature identity disparity.

In an output-driven map, any grammatical mapping forces any input that has greater similarity to the output of that mapping to map to that same output.

3.3.4 Discussion: Internal similarity and Output-Orientedness

The definition of greater internal similarity given in (38) requires that corresponding disparities be featurally the same (identical disparities). A corresponding pair of feature identity disparities are only identical if both disparities have the same value of the relevant feature on the input side and the output side.[14] A corresponding pair of deletion disparities is only identical if the two input segments are identical (have all of the same feature values). Featural identity between corresponding disparities is important to capturing the notion of output-orientedness.

To see the significance, consider the following illustration. Let candidate /*tek*/:[*ti*] be grammatical. The third input segment, *k*, has no output correspondent in this candidate, constituting the disparity *k*:_. Now consider another candidate /*tip*/:[*ti*], where the third input segment, *p*, has no output correspondent. Under the definition of greater internal similarity, /*tip*/:[*ti*] does **not** have greater internal similarity than /*tek*/:[*ti*]. The deletion disparity in /*tip*/:[*ti*] of the final input segment, *p*:_, is not identical to the deletion disparity in /*tek*/:[*ti*] of the final input segment, *k*:_. If /*tip*/:[*ti*] were considered to have greater internal similarity than /*tek*/:[*ti*], then /*tip*/:[*ti*] would have to be grammatical if the map were to satisfy the definition of output-driven. But a different output for input /*tip*/ could be accounted for by conditions on the output. An output condition banning dorsals in codas could account for the deletion of input-final *k* in /*tek*/:[*ti*], while allowing the grammar to choose /*tip*/:[*tip*] rather than /*tip*/:[*ti*]. Allowing the deletion disparities to satisfy greater internal similarity would require that a map admitting both /*tek*/:[*ti*] and /*tip*/:[*tip*] be considered non-output-driven, despite the clear availability of an output-based characterization of the pattern. While the segments being deleted in the candidates /*tek*/:[*ti*] and /*tip*/:[*ti*] do not appear in the outputs of the candidates, their feature values have relevant relationships to possible output conditions. Output conditions can be responsible for deletion disparities in a way that is sensitive to the feature values of the deleted segments.

The lack of a greater internal similarity relation between /*tek*/:[*ti*] and /*tip*/:[*ti*] contrasts with a case like /*tik*/:[*ti*] having greater internal similarity than /*tek*/:[*ti*]. An output-oriented map should not be able to distinguish the fate of *k* in one input from the other when the rest of the input surfaces identically. The only difference between the two inputs, /*tek*/ and /*tik*/, is the height of the vowel. The vowel height for

---

that both lack IO correspondents in the lesser similarity candidate: the greater similarity candidate lacks both an insertion disparity and a deletion disparity relative to the lesser similarity candidate. While technically all that is different between the two candidates is the IO correspondence relation, the "input changing" metaphor would be sustained in terms of deleting the input segment lacking an output correspondent, and then inserting an identical input segment in IO correspondence with the output segment.

[14] That such disparities, if they correspond, will have the same value of the relevant feature on the output side needn't be directly asserted in the definition, because it is a consequence of the definition. Both disparities must have the same feature value on the output side, because both disparities must involve the same output segment: disparities involving IO feature nonidentity can only correspond if the involve the same output segment.

/*tek*/ is neutralized to +hi in the output of the grammatical candidate, /*tek*/:[*ti*]. The only way for a map to treat the final *k* differently is to make reference to something other than the *k* itself and the output: the input quality of the vowel. A map admitting /*tek*/:[*ti*] but not /*tik*/:[*ti*] does not conform to our intuitions about output-oriented, and does not satisfy the definition of output-driven map.

4 The Relation to Opaque Processes

The commonly cited characterization of phonological opacity is the one given by Kiparsky (1971, 1973):

(42) A process P of the form A → B / C__D is opaque to the extent that there are surface representations of the form:

a. A in the environment C__D, or

b. B derived by P in environments other than C__D.

Note that in this characterization, opacity is a property of processes in the context of grammars: a process is opaque to the extent that the process relates to mappings of the grammar in certain ways. In this view, opacity is not a property of phonological maps; it is a property of processes as they relate to particular phonological maps. The concept of output-drivenness developed in this paper characterizes phonological maps without reference to a presupposed analysis in terms of processes; output-drivenness here is a property of phonological maps, not of processes.

Some maps that have been characterized in terms of opaque processes are non-output-driven. Chain shifts are obvious cases. I will illustrate this here with data from Etxarri Basque vowel raising in hiatus, taken from the description by Kirchner (1995). Much more will be said about this example, and Kirchner's analysis, in section 7.2. For now, simply consider the vowel mappings shown in (43).

(43) Etxarri Basque vowel raising in hiatus
/a/ → [a] /e/ → [i] /i/ → [i^y] /i^y/ → [i^y]

A chain shift is necessarily non-output-driven. In this instance, /e/:[i] is grammatical. /i/:[i] has greater internal similarity than /e/:[i], by virtue of removal of the disparity in the value of the feature hi, yet /i/:[i] is not grammatical. Kirchner analyzes this in terms of an opaque relation among raising processes: the application condition of the process raising high vowels to high and raised is met by the output [i] in /e/:[i], yet the process has not applied.

Non-output-drivenness of maps will not line up with process opacity (using Kiparsky's characterization of opaque processes) for every analysis, however. The matter can be forced by examining a case where the same phonological map can be analyzed in two different ways, with each analysis using a different phonological process. The key is that one of the processes is opaque relative to the map, while the other process is not. This makes particularly clear that process opacity is a property of processes, not maps alone. Map output-drivenness, on the other hand, does not make reference to processes, so it renders a single judgement on the phonological map, independent of any analysis into processes.

This can be illustrated with a set of prefix alternations in Lithuanian involving epenthesis and voice assimilation. I follow the presentation and discussion given by Baković (2007) (see also Bakovic 2005). All of the Lithuanian data used by Baković come from (Ambrazas 1997, Dambriunas et al. 1966, Kenstowicz and Kisseberth 1971, Kenstowicz 1972, Mathiassen 1996).

The data concern two distinct verbal prefixes /at/ and /ap/. The consonants of the prefixes assimilate to adjacent stem-initial consonants in voicing and palatalization; the focus here is on the voicing assimilation.

(44) Voicing assimilation in Lithuanian

at-ko:p^{j}t^{j}i	'to rise, climb up'	ap-kal^{j}b^{j}et^{j}i	'to slander'
ad-gaut^{j}i	'to get back'	ab-gaut^{j}i	'to deceive'
at^j-p^{j}aut^{j}i	'to cut off'	ap^j-t^{j}em^{j}d^{j}i:t^{j}i	'to obscure'
ad^j-b^{j}ek^{j}t^{j}i	'to run up'	ab^j-g^{j}i:d^{j}i:t^{j}i	'to cure (to some extent)'

If the prefix-final consonant and the initial consonant of the stem are the same apart from possible differences in voicing and palatalization, then the prefixes surface as [at^{j}i] and [ap^{j}i], respectively.[15]

(45) Vowel epenthesis in Lithuanian

at^{j}i-taik^{j}i:t^{j}i	'to make fit well'	ap^{j}i-put^{j}i	'to grow rotten'
at^{j}i-t^{j}eis^{j}t^{j}i	'to adjudicate'	ap^{j}i-p^{j}i:l^{j}t^{j}i	'to spill something on'
at^{j}i-duot^{j}i	'to give back, return'	ap^{j}i-bar^{j}t^{j}i	'to scold a little bit'
at^{j}i-d^{j}et^{j}i	'to delay, postpone'	ap^{j}i-b^{j}er^{j}t^{j}i	'to strew all over'

Baković cites an analysis by Odden (2005) which analyzes this in terms of two processes. One is a regressive voicing process in which an obstruent is voiced when it appears immediately before a voiced obstruent. The second process epenthesizes vowels between adjacent obstruent stops with identical place of articulation. Baković focuses on the epenthesis process, and notes that it serves to epenthesize precisely in those situations where assimilation would have otherwise created adjacent identical obstruent stops. In line with this, he proposes an alternative epenthesis process, one that epenthesizes vowels between adjacent identical obstruent stops.[16]

My present goal is not to comment on which analysis is preferable (see Baković (2007) for further discussion). Instead, note that the first version of the epenthesis process is not an opaque process in the Lithuanian data: it only epenthesizes vowels between consonants that satisfy the conditions of application for the process. The second version of the epenthesis process, however, is opaque: it epenthesizes vowels between consonants that are not, on the surface, identical to each other. Process opacity lies in the eye of the analyst proposing the process.

Map output-drivenness does not make reference to processes. It does make reference to all of the mappings of a map, and thus does require a specification of the outputs for all relevant inputs. The relevant mappings, based on the description of Lithuanian, are shown in (46). The mappings for two attested forms, based on the analysis above of the underlying form for the prefix, are given. For each mapping of the original analysis, the greater internal similarity mappings are shown: these are inferred from the description of the language. The map in (46) is map output-driven. In the sense characterized by output-driven, the phonological map can be characterized by output conditions. This matches Baković's qualitative description: a ban on adjacent obstruent stops with identical place of articulation is an output condition.

(46) Vowel epenthesis in Lithuanian is output-driven

Attested: /at-t^{j}eis^{j}t^{j}i/ → at^{j}i-t^{j}eis^{j}t^{j}i	Greater int. similarity:	/at^j-t^{j}eis^{j}t^{j}i/ → at^{j}i-t^{j}eis^{j}t^{j}i
		/ati-t^{j}eis^{j}t^{j}i/ → at^{j}i-t^{j}eis^{j}t^{j}i
		/at^{j}i-t^{j}eis^{j}t^{j}i/ → at^{j}i-t^{j}eis^{j}t^{j}i
Attested: /at-d^{j}et^{j}i/ → at^{j}i-d^{j}et^{j}i	Greater int. similarity:	/at^j-d^{j}et^{j}i/ → at^{j}i-d^{j}et^{j}i
		/ati-d^{j}et^{j}i/ → at^{j}i-d^{j}et^{j}i
		/at^{j}i-d^{j}et^{j}i/ → at^{j}i-d^{j}et^{j}i

[15] The consonants palatalize before front vowels, including the epenthetic front vowels here (Bakovic 2007).

[16] The latter process is a "process" in the sense that it involves a conditioned change that plays a role in the analysis. If turned into a rule as stated, it won't work in an SPE-style analysis, which is part of Baković's point. It does, however, satisfy Kiparsky's definition of opaque process, given the role it plays in Baković's analysis.

Some comments are in order. First, output-drivenness is a property of a phonological map, not just data forms. Thus, it is only independent of a choice between analyses to the extent that the differing analyses predict the same phonological map. Analyses that differ in their assignment of underlying forms to surface forms, or that make different predictions about inputs not assigned to attested data, differ in the actual phonological maps they predict.

Apart from the map itself, all that is additionally required to evaluate output-drivenness is the inventory of disparities. The inventory of disparities is something like a set of primitives that could be used to describe the **possible** changes that can be made by processes, but without the specification of **which** changes will actually be used, and without the **conditions** for the application of processes. Specifying the inventory of disparities is unavoidable; one cannot describe phonology without giving some characterization of input-output disparities. The dependence of the judgement of (non)output-drivenness on the chosen inventory of disparities is similarly unavoidable. One can render any map output-driven by selecting an inventory of disparities that contains all possible entire input/output mappings, and no others; this is akin to restricting one's phonological theory to a list of processes each of which applies to only a single input. The natural move is to let the representational theory, specifically the theory of input-output correspondence, dictate the inventory of disparities utilized in an analysis of output-drivenness.

The (non)output-drivenness of phonological maps is identified not by attempting to individuate processes and look for individual conditioning environments in an output, but by looking to see if the systematic reduction (via changes to the input) of disparities between the input and output always yields the same output form.

5 Properties That Ensure Output-driven Maps in Optimality Theory

This section examines the relationship between output-driven maps and Optimality Theory, and gives sufficient conditions ensuring that an Optimality Theoretic system defines only output-driven maps. For concreteness, all discussion will assume the representational conditions specified in section 3.3.1.

5.1 Output-driven Maps and Optimality Theory

5.1.1 Output-driven Maps and Optimization

In Optimality Theory, a candidate is optimal if it is at least as harmonic as all of the other candidates sharing the same input form. The function GEN identifies sets of candidates sharing inputs; GEN(in_a) denotes the candidates that have in_a as their input form. The expression $akx \succcurlyeq aqz$ denotes the proposition that *akx* is at least as harmonic as *aqz*, and the expression $akx \succ aqz$ denotes the proposition that *akx* is strictly more harmonic than *aqz*. The condition that *akx* is grammatical can be expressed as in (47), where *aqz* is here a variable representing any member of GEN(in_a).

(47) $\forall aqz(akx \succcurlyeq aqz)$

The condition that *bmx* is grammatical can be similarly expressed as in (48), where *bpy* is here a variable representing any member of GEN(in_b).

(48) $\forall bpy(bmx \succcurlyeq bpy)$

A map defined by an Optimality Theoretic system is output-driven if, for any pair *akx* and *bmx* such that *bmx* has greater internal similarity than *akx*, the condition in (49) is true.

(49) $[\forall aqz(akx \succcurlyeq aqz)] \rightarrow [\forall bpy(bmx \succcurlyeq bpy)]$

The same condition can be expressed by taking the contrapositive.

(50) $\neg[\forall bpy(bmx \succcurlyeq bpy)] \rightarrow \neg[\forall aqz(akx \succcurlyeq aqz)]$

Applying each negation operator to the expression within its scope yields the result in (51).

(51) $[\exists bpy(bpy \succ bmx)] \rightarrow [\exists aqz(aqz \succ akx)]$

In words, in an output-driven map, if *bmx* has greater internal similarity than *akx*, then the existence of a candidate (in GEN(in_b)) more harmonic than *bmx* entails the existence of a candidate (in GEN(in_a)) more harmonic than *akx*.

5.1.2 A designated competitor: *aoy*

The condition in (51) states that, for any candidates *bmx* with greater internal similarity than *akx*, the existence of a candidate *bpy* such that $bpy \succ bmx$ entails the existence of **some candidate** *aqz* such that $aqz \succ akx$. To make the analysis more tractable, I will now adopt a stronger requirement, leading to sufficient conditions for output-driven maps. This stronger condition, given in (52), requires the existence of **a particular candidate**, *aoy*, defined relative to *bpy*, and requires that for any *bpy* such that *bpy* is more harmonic than *bmx*, the corresponding *aoy* is more harmonic than *akx*. The definition of *aoy* is given below in (53).

(52) $\forall bpy\ [(bpy \succ bmx) \rightarrow (aoy \succ akx)]$

This allows the analysis to focus on only one competitor to *akx* relative to *bpy*. The definition of *aoy* is not arbitrary; I will argue that under ordinary expectations about candidate spaces, this candidate will always exist, and that for most if not all output-driven maps of interest this candidate will always satisfy the condition in (52). An advantage of this approach is that the sufficient conditions for output-driven maps can be broken in to separate conditions on GEN (the candidate space) and on CON (the constraints).

The candidate *aoy* is defined relative to *akx*, *bmx*, and *bpy*, where *bmx* has greater internal similarity than *akx*. The premise that *bmx* has greater internal similarity than *akx* requires that an appropriate correspondence exist between the disparities of the candidates, which in turn requires that an appropriate input-input correspondence R_{II} exists between the inputs in_a and in_b, 'appropriate' meaning that R_{II} must satisfy the conditions given in (28). Given a candidate *bpy* with input in_b and output out_y, candidate *aoy* has input in_a, output out_y, and IO correspondence relation R_o, based upon R_{II} and *bpy*, defined in (53).

(53) The correspondence relation R_o for candidate *aoy*

- For each segment s_a in in_a that has an input-input correspondent s_b in in_b, s_a has the same output correspondent in *aoy* that s_b has in *bpy*.
- Each segment s_a in in_a that does not have an input-input correspondent in in_b necessarily has no output correspondent in *akx* (see (40)), and has no output correspondent in *aoy*.

This fully determines the correspondence relation R_o, which is summarized in (54).

(54) $s_a R_o s_y$ iff $\exists s_b\ [s_a R_{II} s_b$ and $s_b R_p s_y]$

The candidate *aoy* relates to *bpy* in a way analogous to the way *akx* relates to *bmx*. The input-input correspondence relates segments of inputs in_a and in_b that have matching roles in their respective candidates *akx* and *bmx*. The definition of *aoy* preserves this "matching role" property by assigning, to each segment of in_a, the same role *aoy* that its input-input correspondent has in *bpy*. The point of defining *aoy* in this way is that any reduction of disparities that *bpy* has relative to *bmx* will be mirrored by a corresponding reduction of disparities for *aoy* relative to *akx*. To the extent that eliminating disparities improves harmony, such a relationship supports (52) above: for any *bpy* that can be defined with fewer disparities than *bmx*, the related *aoy* will lack equivalent disparities relative to *akx*. If those disparities are sufficient to make *bpy* more harmonic than *bmx*, then they will also make *aoy* more harmonic than *akx*.

The "analogy" between the *aoy*/*bpy* and the *akx*/*bmx* relationships is illustrated by building on the example in section 3.3.2. Repeated below are candidates *akx* and *bmx*, as well as the derived input-input correspondence R_{II}.

(30) in_a = /*agbik*/ out_x = [*tibi*] $R_k = \{(a_1,x_2), (a_3,x_3), (a_4,x_4)\}$

(31) in_b = /*tebik*/ out_x = [*tibi*] $R_m = \{(b_1,x_1), (b_2,x_2), (b_3,x_3), (b_4,x_4)\}$

(32) $R_{II} = \{(a_1,b_2), (a_3,b_3), (a_4,b_4), (a_5,b_5)\}$

The disparities of *akx* and *bmx* are repeated below in (33) and (34), respectively, along with the correspondence between the disparities (35).

(33) _: x_1 a_1(–hi):x_2(+hi) a_1(+low):x_2(–low) a_2:_ a_5:_

(34) b_2(–hi):x_2(+hi) b_5:_

(35) Correspondence between the disparities of *akx* and *bmx*

- a_1(–hi):x_2(+hi) of *akx* corresponds to b_2(–hi):x_2(+hi) of *bmx*
- a_5:_ of *akx* corresponds to b_5:_ of *bmx*

Now consider a new output form, out_y = [tebi]. Consider also a candidate with input in_b, output out_y, and IO correspondence relation R_p. This candidate, *bpy*, is summarized in (55), and its disparities are listed in (56).

(55) in_b = /*tebik*/ out_y = [*tebi*] $R_p = \{(b_1,y_1), (b_2,y_2), (b_3,y_3), (b_4,y_4)\}$

(56) b_5:_

Relative to *bpy*, the definition in (53) yields the candidate *aoy* summarized in (57), which has the disparities listed in (58).

(57) in_a = /*agbik*/ out_y = [*tebi*] $R_o = \{(a_1,y_2), (a_3,y_3), (a_4,y_4)\}$

(58) _:y_1 a_1(+low):y_2(–low) a_2:_ a_5:_

Figure 3 repeats the candidates *akx* and *bmx* from Figure 2, and additionally shows the candidates *aoy* and *bpy*, along with their correspondence relations and the input-input correspondence relation R_{II}. The input-input correspondence is (by definition) identical for both pairs of candidates. The candidates *aoy* and *bpy* differ from *akx* and *bmx* in that they involve output form out_y = [*tebi*].

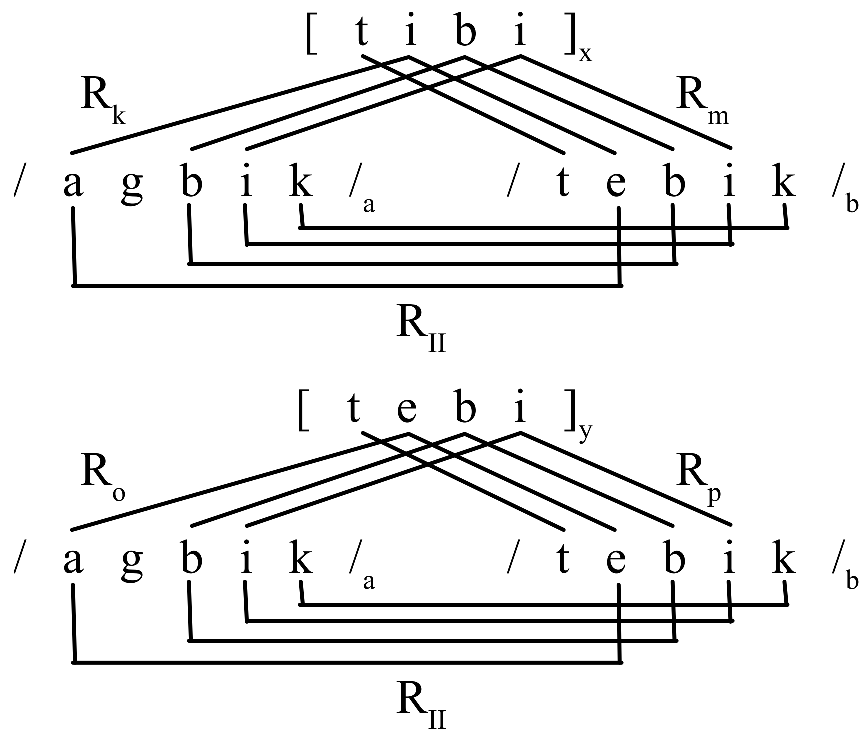

Figure 3 Candidates *akx*, *bmx*, *aoy* and *bpy*.

The use of the input-input correspondence R_{II} in the definition of the correspondence relation R_o patterns the relationship between the IO correspondences of *aoy* and *bpy* after the relationship between the IO correspondences of *akx* and *bmx*. Candidate *bpy* relates input segment b_2 to its output correspondent y_2. Because b_2 has input-input correspondent a_1, segment a_1 has output correspondent y_2 in *aoy*. In candidate *bpy*, input segment b_5 has no output correspondent; because b_5 has input-input correspondent a_5, segment a_5 has no output correspondent in *aoy*. Segment a_2, which has no input-input correspondent, has no output correspondent in *aoy*. This is not arbitrary, as a_2 necessarily has no output correspondent in *akx*; a_2 thus has the same role in *aoy* as in *akx* (and a_2 has no corresponding input element in either *bpy* or *bmx*).

A correspondence can now be constructed between the disparities of *aoy* and *bpy*, based on R_{II}, just as was described for *akx* and *bmx* in (29).

(59) Correspondence between the disparities of *aoy* and *bpy*

- a_5:_ of *aoy* corresponds to b_5:_ of *bpy*

Of the four disparities of *aoy*, only one stands in correspondence with the lone disparity of *bpy*. However, the other three disparities of *aoy* have clear counterparts in *akx*. To distinguish the different kinds of relationships between disparities of different candidates, we will refer to related disparities between candidates *akx* and *aoy* as **analogous disparities**.

(60) Analogous disparities of *aoy* and *akx*

- _:y_1 of *aoy* is analogous to _:x_1 of *akx*
- a_1(+low):y_2(–low) of *aoy* is analogous to a_1(+low):x_2(–low) of *akx*
- a_2:_ of *aoy* is analogous to a_2:_ of *akx*
- a_5:_ of *aoy* is analogous to a_5:_ of *akx*

A key general property is that every disparity of *aoy* will have either a corresponding disparity in *bpy* or an analogous disparity in *akx*. In this particular example, every disparity of *aoy* has an analog in *akx*, but that will not be true in general. Further, **the disparities of *aoy* lacking corresponding disparities in *bpy* all have analogous disparities in *akx* that lack corresponding disparities in *bmx***. This is the rigorous sense in which *aoy* relates to *bpy* in the same way that *akx* relates to *bmx*. The first three disparities listed in (60) have no corresponding disparities in *bpy* (59) and their analogous disparities in *akx* have no corresponding disparities in *bmx* (35). The definition of *aoy* insures that every disparity of *aoy* will have either a corresponding disparity in *bpy* or an analogous disparity in *akx*. A proof of these properties of the definition of *aoy* may be found in the analysis of the relationships between disparities in section 10.1.

Note that it is possible for a disparity in *aoy* to have both a corresponding disparity in *bpy* and an analogous disparity in *akx*. In the example above, this is the case for disparity a_5:_ of *aoy*: it has corresponding disparity b_5:_ in *bpy*, and analogous disparity a_5:_ in *akx*. Note further, however, that the analogous disparity a_5:_ in *akx* also has a corresponding disparity b_5:_ in *bmx*. The final *k* of the inputs, a_5 for in_a and b_5 for in_b, is deleted in all four candidates, and the four deletion disparities are collective counterparts, via disparity correspondence and disparity analogy.

While the discussion presumes that *bmx* has greater internal similarity than *akx*, it is **not** necessarily the case that *bpy* has greater internal similarity than *aoy*. It is possible for *aoy* and *bpy* to have corresponding disparities that are non-identical, and it is possible for *bpy* to have a disparity with no corresponding disparity in *aoy*. Demonstrations of these possibilities can be found in section 10.1.

While I am labeling as analogous the relationship between certain disparities of *aoy* and *akx*, the relationship does constitute another kind of correspondence. The second, third, and fourth analogs listed in (60) are based on the fact that the analogous disparities have the same input segment, and that is the basis for establishing analogous deletion and feature identity disparities in general. The first analog listed in (60) is between insertion disparities, and the output segments are necessarily from different output forms. The basis for the relationship between these disparities is that y_1 has input correspondent b_1 in *bpy* (recall that disparity _:y_1 in *aoy* has no corresponding disparity in *bpy*). Input segment b_1 has output correspondent x_1 in *bmx*, and x_1 in turn is the inserted output segment of the analogous disparity _:x_1 of *akx*. The disparities _:x_1 and _:y_1 for the candidates with input in_a are analogous because both output segments have the same input correspondent (b_1) in the candidates with input in_b, *bpy* and *bmx*. This is what makes the defined relationship between the disparities of *aoy* and *akx* "analogous": the relationship between the output segments of *aoy* and *akx* is partly based on the relationship between the same output segments in candidates *bpy* and *bmx*.

The definition of analogous disparities between *aoy* and *akx* is based on the shared input form between the two candidates, and the disparity correspondences between *akx* and *bmx* and between *aoy* and *bpy*, which in turn are based on the input-input correspondence R_{II}. The definition is given in (61).

(61) Analogous disparities between *aoy* and *akx*.

- Let s_a:_ be a deletion disparity of *aoy*. This disparity in *aoy* has an analogous disparity in *akx* if and only if s_a has no output correspondent in *akx*.
- Let _:s_y be an insertion disparity in *aoy*. This disparity has an analogous disparity _:s_x in *akx* if and only if s_y has an input correspondent s_b in *bpy*, and s_b has output correspondent s_x in *bmx*.
- Let $s_a(\alpha)$:$s_y(\beta)$ be a feature identity disparity of *aoy* for feature F ($\alpha \neq \beta$). This disparity in *aoy* has an analogous disparity in *akx* if and only if s_a has an output correspondent s_x in *akx* such that s_a and s_x differ on the value of feature F.

Note the relationship in the above example between the disparities of *bpy* in (56) and the disparities of *bmx* in (34). *bpy* is such that it has "fewer" disparities than *bmx*; every disparity in *bpy* has a counterpart in *bmx*. Neither candidate assigns an output correspondent to input segment b_5. However, in *bmx*, input segment b_2 has a mismatch with its output correspondent for the feature hi, while in *bpy* segment b_2 is identical to its output correspondent.[17] Now note the relationship between the disparities of *aoy* in (58) and the disparities of *akx* in (33). In *akx*, input segment a_1 has a mismatch with its output correspondent for the feature hi, while in *aoy* segment a_1 has the same value for hi as its output correspondent y_2.

The significance of these observations about disparities is that, for each disparity that *bpy* lacks relative to *bmx*, *aoy* lacks a related disparity relative to *akx*. Recall the explicit correspondence in disparities between *akx* and *bmx*, given in (35). This correspondence between disparities was necessary to support the claim that *bmx* has greater internal similarity than *akx*. In (35), disparity a_1(–hi):x_2(+hi) of *akx* corresponds to disparity b_2(–hi):x_2(+hi) of *bmx*. These corresponding disparities are exactly the ones missing in *aoy* and *bpy*, respectively.

The general condition being approximated here is the one given in (51): if *bpy* is more harmonic than *bmx*, then some candidate is more harmonic than *akx*. *aoy* is defined so that in an output-driven map, when *bpy*≻*bmx*, we would expect *aoy*≻*akx*. If *bpy*≻*bmx* because of an output condition preferring out_y to out_x, clearly that condition will also prefer *aoy* to *akx*. If *bpy*≻*bmx* because *bpy* lacks a disparity that *bmx* possesses, then *aoy* will have a similar advantage over *akx* because it will lack the same disparity relative to *akx*.

5.1.3 Output-driven Maps and Constraints

Recall the general condition on harmonic relations for output-driven maps given in (52), repeated here. This condition applies to every pair of candidates *akx* and *bmx* such that *bmx* has greater internal similarity than *akx* (no assumptions are made about the grammaticality of *akx* and *bmx*). Every candidate *bpy* is a member of GEN(in_b) (as is *bmx*), and *aoy* is defined relative to each *bpy* as specified in (53).

(52) $\forall$ *bpy* [(*bpy*≻*bmx*) → (*aoy*≻*akx*)]

In Optimality Theory, harmony is defined by a set of constraints ranked in a strict dominance hierarchy. The implication in (52) can be understood as an implicational relation between elementary ranking conditions (Prince 2002). The theory of entailed ranking arguments can be used to derive the conditions on individual constraints ensuring that all maps are output-driven.

In Optimality Theory, a constraint renders one of three possible evaluations of a comparison between two candidates. Following Prince (2002), a comparison between two candidates *w* and *l* is denoted *w*~*l*. The first of the two candidates (here, *w*) is assigned the role labeled 'winner', while the second candidate (here, *l*) is assigned the role labeled 'loser'. The three possible evaluations are the possible preferences between the pair: the constraint can prefer the winner, it can prefer the loser, or it can have no preference. The possible evaluations of candidate comparisons are denoted as shown in (62).

(62) Notation for constraint evaluations of candidate comparsions ('C' here is a constraint)

C[*w*~*l*] = W indicates that C prefers *w* to *l*.

C[*w*~*l*] = e indicates that C has no preference between *w* and *l*.

C[*w*~*l*] = L indicates that C prefers *l* to *w*.

An **elementary ranking condition**, or ERC, is a collection of the evaluations of a candidate comparison by all of the constraints of the system. The ERC for the comparison between *w* and *l* (with *w* as the winner) is denoted [*w*~*l*]. An ERC expresses the conditions under which the winner of the ERC is more

[17] It is important to realize that the relation of greater internal similarity **cannot** hold between *bpy* and *bmx*: the two candidates have different output forms.

harmonic than the loser of the ERC; the ERC [*w~l*] is true precisely when $w \succ l$. Thus, (52) can be cast in terms of ERCs, as shown in (63).

(63) $\forall bpy$ ([*bpy~bmx*] → [*aoy~akx*])

The theory of ERC entailment is based on entailment relations among the three possible evaluations, L→e→W (where each evaluation also entails itself as well). In other words, L entails each of {L,e,W}, e entails each of {e,W}, and W entails W. One ERC entails another in the linguistic system if, for each constraint, the evaluation of the first ERC entails the evaluation of the second; see Prince (2002, 2003) for details. Thus, to prove the implication in (63), it is sufficient to show that the condition in (64) holds.

(64) $\forall bpy$ $\forall$C∈CON (C[*bpy~bmx*] → C[*aoy~akx*])

Note that the condition in (64) applies separately to each individual constraint; each constraint can be independently evaluated for the implication relation between its evaluations of the two ERCs.

The implicational condition imposed on each constraint in (64) can be interpreted in light of the implicational relation among the three possible evaluations that a constraint can make of a candidate comparison, L→e→W. If C[*bpy~bmx*]=L, then the implication will be satisfied no matter what evaluation C assigns to [*aoy~akx*]. If C[*bpy~bmx*]=e, then C[*aoy~akx*] must be either e or W. If C[*bpy~bmx*]=W, then C[*aoy~akx*] must be W. The latter two cases impose non-trivial restrictions on C[*aoy~akx*], and are summarized in (65).

(65) Output-driven conditions on constraints (ERC version). Given constraint C, candidates *bmx* with greater internal similarity than *akx*, *bpy* in GEN(in_b), and *aoy* as defined in (53):

- C[*bpy~bmx*]=W → C[*aoy~akx*]=W
- C[*bpy~bmx*]=e → C[*aoy~akx*]=e or W

These conditions can in turn be interpreted in terms of constraint preference on candidate comparisons, using the definitions in (62), and yielding the version of the output-driven conditions given in (66).

(66) Output-driven conditions on constraints (preference version). Given constraint C, candidates *bmx* with greater internal similarity than *akx*, *bpy* in GEN(in_b), and *aoy* as defined in (53):

- (C prefers *bpy* to *bmx*) entails (C prefers *aoy* to *akx*)
- (C has no preference between *bpy* and *bmx*) entails (C either prefers *aoy* to *akx* or has no preference between them)

Constraint preferences between candidates arise as a consequence of the number of violations assessed by a constraint to each of the candidates: a constraint prefers one candidate to a second if it assesses strictly fewer violations to the first than to the second, and a constraint has no preference between two candidates if it assesses an equal number of violations to both. The output-driven conditions in (66) can be translated into the terms of violations counts, as given in (67). C(*cand*) denotes the number of violations assessed to candidate *cand* by constraint C.

(67) Output-driven conditions on constraints (violation counts version). Given constraint C, candidates *bmx* with greater internal similarity than *akx*, *bpy* in GEN(in_b), and *aoy* as defined in (53):

- C(*bpy*)<C(*bmx*) entails C(*aoy*)<C(*akx*)
- C(*bpy*)=C(*bmx*) entails C(*aoy*)≤C(*akx*)

The condition in (64) is sufficient to ensure the condition in (63). Two exceptional cases prevent it from being strictly necessary. The exceptions stem from the two types of ‘trivial’ ERCs: logically valid ERCs, and logically invalid ERCs. Logically valid ERCs contain no L’s, and are satisfied under every constraint ranking. Logically invalid ERCs contain no W's and at least one L, and are not satisfied under any constraint ranking. If [*aoy~akx*] is logically valid, then it is entailed by anything, including [*bpy~bmx*] even if there exists a constraint that does not satisfy (64). Similarly, if [*bpy~bmx*] is logically invalid, then it entails everything, including [*aoy~akx*] even if there exists a constraint that does not satisfy (64). I will adopt (64), and the equivalent form in (67), as a sufficient condition on constraints in the analysis below (section 5.2.2).

5.2 Sufficient Conditions for Output-driven Maps

5.2.1 Properties of GEN: correspondence uniformity

For an OT system to define output-driven maps, GEN must generate the relevant candidates. For instance, if GEN generates a candidate *akx* with input in_a and output out_x, but does not generate some candidate *bmx* that has greater internal similarity than *akx* (where *bmx* is contained in the RRS), then the map could be non-output-driven as a consequence of GEN; the candidates required to be grammatical by the definition of an output-driven map wouldn't be made available by GEN (let alone be optimal).

A GEN function will be said to be **correspondence uniform**, relative to a reference representation space RRS, if it satisfies the conditions in (68).

(68) Conditions for GEN to be correspondence uniform

- For each candidate *akx* of the RRS generated by GEN, every candidate *bmx* in the RRS that has greater internal similarity than *akx* must also be generated by GEN.
- For every *bmx* with greater internal similarity than *akx* generated by GEN, for each competitor *bpy* generated by GEN, the corresponding candidate *aoy* defined in (53) must also be generated by GEN.

These requirements of GEN are fully consistent with the standard "freedom of analysis" view of GEN, in which any representation in the reference representation space is generated by GEN. Thus, far from requiring anything unusual of GEN, the conditions stated here merely focus on the properties of GEN that are essential for guaranteeing output-driven maps. I distinguish the RRS from GEN in this analysis so that the RRS can establish the analyst's expectations about the possible behaviors of maps, and the RRS can then be used to evaluate different linguistic theories, including different OT systems (which might have different definitions of GEN). An example of an intentional distinction between the RRS and GEN would be the harmonic serialism variant of Optimality Theory (Prince and Smolensky 1993/2004). In harmonic serialism, the outputs of the set of candidates assigned to an input may only differ from the input in at most one way,[18] and a derivation consists of a series of such optimizations, each using as its input the output of the previous optimization, until a form is reached which maps to itself. Such restrictions on GEN can result in maps (defined by the beginning and ending representations of derivations) that are not output-driven; see McCarthy (2006) for relevant discussion of harmonic serialism.

A GEN function meeting the conditions in (68) is said to be correspondence uniform because it must possess a certain uniformity of possible correspondence. The uniformity runs along the lines of the possible types of disparities. Given that GEN generates a candidate *akx*, GEN must also generate all candidates in the RRS with the same output and a strict subset of disparities. With correspondence uniformity, the existence of a single candidate with numerous disparities can automatically compel the existence of a whole subspace of candidates with the same output and different inputs, each input strictly eliminating some of the disparities of the original candidate. Further, given that GEN generates a

[18] Such a condition might well be formalized, in the terms of the present work, as a restriction that GEN can only generate candidates containing at most one disparity.

candidate *bmx* with greater internal similarity than *akx*, the existence of competitors to *bmx* automatically entails the existence of analogous competitors for *akx*.

The wide scope of correspondence uniformity becomes more apparent when you consider that the existence of competitors for one candidate requires the existence of analogous competitors for every candidate with the same output and strictly more disparities. Suppose that GEN generates an identity mapping *xqx*, a candidate with no disparities, and competitors for the input in_x. Every other input form in_c that has a generated candidate with output out_x (presuming that every such candidate will contain at least one disparity) must, under correspondence uniformity, also have competitors analogous to every competitor of *xqx*. If GEN is such that every possible input has at least one candidate with output out_x, then every competitor of *xqx* must have an analogous candidate for every other input. Under correspondence uniformity, it can be possible for the competitors for a single identity mapping to automatically entail most (if not all) of the competitors for all other inputs.

Correspondence uniformity fits naturally with a conception of GEN in which any input can stand in correspondence with any output. However, it is also consistent with a "partitioned" GEN, with separate sets of inputs each forming candidates with separate sets of outputs. For example, if one banned insertion and deletion in GEN (as well as coalescence and breaking), then an input could only stand in correspondence with outputs containing the same number of segments as that input. Thus, all inputs of length two could stand in correspondence with all outputs of length two but none of the outputs of length three, while all of the inputs of length three could stand in correspondence with all of the outputs of length three but none of the outputs of length two, etc. Such a "partitioned" GEN function could still be correspondence uniform.

5.2.2 Properties of Constraints: output-driven preserving

Let *akx* be a candidate with input in_a and output out_x, and let *bmx* be a candidate with input in_b and output out_x such that *bmx* has greater internal similarity than *akx*, based on correspondence R_{II} between in_a and in_b. For each candidate *bpy*, with input in_b and output out_y, let *aoy* denote the candidate with input in_a, output out_y, and correspondence R_o as defined in (53). A constraint C is **output-driven preserving** (ODP) if it has the properties previously listed in (67), for every such pair *akx* and *bmx*, and every competitor *bpy* of each *bmx*. The properties are separately labeled here as (69) and (70).

(69) C(*bpy*) < C(*bmx*) entails C(*aoy*) < C(*akx*)

(70) C(*bpy*) = C(*bmx*) entails C(*aoy*) ≤ C(*akx*)

5.2.3 Proof of sufficient conditions for output-driven maps

Notation: the expression "*akx* ≽ *aoy*" denotes the proposition that *akx* is at least as harmonic as *aoy*, while the expression "*akx* ≻ *aoy*" denotes the proposition that *akx* is strictly more harmonic than *aoy*.

Suppose an Optimality Theoretic system has a GEN function that is correspondence uniform, and all of the constraints are ODP. Then all grammars realizable in that system define output-driven maps.

Proof[19]

Let M be the map defined by an arbitrary grammar in the OT system. Let *akx* (with input in_a and output out_x) be any candidate that is optimal for M. Let *bmx* be any candidate that has greater internal similarity than *akx*. Because GEN is correspondence uniform, it must generate *bmx*. To prove that M is output-

[19] The proof technique used here, of reasoning from the highest-ranked constraint with a preference in at least one of two related candidate comparisons, is adapted from the proof technique used by Tesar (2006) to reason about contrast.

driven, it is sufficient to prove that *bmx* is at least as harmonic as any candidate *bpy* for input in_b: for all candidates *bpy*, $bmx \succcurlyeq bpy$.

The proof is by contradiction. Suppose, to the contrary, that there exists a candidate *bpy* such that $bpy \succ bmx$. It will be shown that this unavoidably leads to a contradiction. Note that no particular commitment is made here about the input-output correspondence relation in *bpy*.

Let *aoy* be as defined in (53). Because the OT system's GEN is correspondence uniform, *aoy* must be generated by GEN. *akx* is optimal, so it must be the case that $akx \succcurlyeq aoy$. Let C_p be the highest-ranked constraint with a preference in at least one of the two candidate comparisons, $akx \succcurlyeq aoy$ and $bpy \succ bmx$. Note that the comparison $bpy \succ bmx$ requires that at least one constraint distinguish them, so C_p must exist. All constraints of the system are ODP, meaning that they have the properties given in (69) and (70), repeated here:

(69) $C(bpy) < C(bmx)$ entails $C(aoy) < C(akx)$

(70) $C(bpy) = C(bmx)$ entails $C(aoy) \leq C(akx)$

With respect to the comparison of $C_p(bpy)$ and $C_p(bmx)$, there are three cases to consider that exhaust all possibilities.

Case 1: $C_p(bmx) < C_p(bpy)$

This entails that $bmx \succ bpy$, directly contradicting the hypothesis that $bpy \succ bmx$. **Contradiction**.

Case 2: $C_p(bpy) < C_p(bmx)$

By (69), this entails that $C_p(aoy) < C_p(akx)$. That in turn entails that $aoy \succ akx$, directly contradicting the premise that *akx* is optimal. **Contradiction**.

Case 3: $C_p(bpy) = C_p(bmx)$

By (70), this entails that $C_p(aoy) \leq C_p(akx)$. Because C_p does not have a preference in the comparison between *bpy* and *bmx*, it must have a preference between *aoy* and *akx*, and therefore it must be the case that $C_p(aoy) < C_p(akx)$. This in turn entails that $aoy \succ akx$, directly contradicting the premise that *akx* is optimal. **Contradiction**.

All possibilities have resulted in contradiction, so the hypothesis, that there exists a candidate *bpy* such that $bpy \succ bmx$, must be false. It follows that *bmx* must be an optimal candidate.

End of Proof

6 Evaluating Constraints for ODP: Basic Constraints

6.1 Terminology: Faithfulness and input-referring constraints

The correspondence between input and output in candidates is normally evaluated by constraints commonly referred to as faithfulness constraints (McCarthy and Prince 1999, Prince and Smolensky 2004). The conventional view is that faithfulness constraints are violated by various failures of the output of a candidate to be faithful to the input of the candidate. Instances of "failure to be faithful" can be viewed as disparities in the sense used in this paper, and many common faithfulness constraints are violated by particular classes of disparities. However, there is another use of the term "faithfulness constraint", one that refers to any constraint which makes reference to the input. This latter usage is more common in the literature on learning in OT, particularly with respect to discussions of the relative ranking of markedness and faithfulness constraints (Demuth 1995, Gnanadesikan 1995, Hayes 2004, Prince and Tesar 2004, Smolensky 1996). More generally, the term faithfulness constraint is used to refer to a constraint which evaluates a correspondence between two forms (usually by penalizing disparities between the forms), be it input-output (McCarthy and Prince 1999), base-reduplicant (McCarthy and Prince 1999), output-output (Benua 1997), or some other correspondence.

To avoid confusion here, the term **input-referring constraints** will be used, meaning constraints that make reference of any kind to the input or to the input-output correspondence relation. Input-referring constraints are of particular interest to the investigation of output-driven maps, whether they penalize specific classes of disparities or other sorts of correspondence configurations.

6.2 Markedness Constraints

A markedness constraint is one that evaluates a candidate solely on the basis of the output form of the candidate, ignoring the input and the IO correspondence. All markedness constraints are trivially ODP. If C_m is a markedness constraint and $C_m(bpy) < C_m(bmx)$, then any candidate with output out_y will have fewer violations of C_m than any candidate with output out_x, and thus $C_m(aoy) < C_m(akx)$. The analogous observation holds whenever $C_m(bpy) = C_m(bmx)$.

It is constraints other than markedness that must be further scrutinized to the determine if they are ODP.[20]

6.3 Value-Independent Input-Referring Constraints

The basic set of input-referring constraints for correspondence faithfulness proposed by McCarthy and Prince (1995) are ODP. They are MAX, DEP, and IDENT[F] (for each segmental feature F). Proofs that these constraints are output-driven-preserving can be found in the appendix (section 10). These constraints evaluate the basic types of disparity adopted in (27). MAX is violated by any deletion disparity: one in which an input segment has no output correspondent. DEP is violated by any insertion disparity: one in which an output segment has no input correspondent. IDENT[F] is violated by any feature identity disparity: one in which input-output corresponding segments have non-identical values for the feature F.

Intuitively, one can understand the ODP status of these constraints to follow from the relationships of disparities between (*bpy* and *bmx*) and (*aoy* and *akx*). If MAX (for instance) prefers *bpy* to *bmx*, it must be because *bpy* has fewer deletion disparities. Thus, there must be deletion disparities that *bpy* lacks relative to *bmx*. The definition of *aoy* is constructed so that *aoy* should also lack, relative to *akx*, disparities corresponding to the ones that *bpy* lacks relative to *bmx*. Thus, if *bpy* ends up with fewer deletion disparities than *bmx*, *aoy* will end up with fewer deletion disparities than *akx*, and therefore MAX will also prefer *aoy* to *akx*.

Elaborating slightly on the deletion disparity case, recall that every disparity in *bmx* has a corresponding disparity in *akx*, by the definition of greater internal similarity. Therefore, every deleted input segment in *bmx* has an input-input correspondent that is deleted in *akx*. Since *bpy* has fewer violations of MAX, it must be the case that at least one of the segments that is deleted in *bmx* has an output correspondent in *bpy*. By the definition of *aoy*, the input-input correspondent of each such segment is assigned the same output correspondent segment in *aoy*. Because those segments (in input in_a) are input-input correspondents of segments (in input in_b) that are deleted in *bmx*, those segments (in input in_a) are deleted in *akx*, and thus constitute deletion disparities (and MAX violations) that *akx* has but *aoy* does not.

[20] Candidates are restricted here to inputs, outputs, and input-output correspondence. If restrictions are relaxed to include other forms of correspondence, such as output-output correspondence, output-drivenness might well be an issue for the constraints evaluating those forms of correspondence as well.

6.4 Value-Restricted Constraints

To the extent that corresponding and analogous disparities are identical, one would expect that any constraint that is violated solely by a particular class of disparities in isolation would be ODP. Here, “in isolation” means that a particular disparity constitutes a violation solely on the basis of the form of the disparity, without concern for the representational context in which that disparity occurs. By the definition of greater internal similarity, all corresponding disparities between *akx* and *bmx* are identical. However, it is possible for corresponding disparities between *aoy* and *bpy* to be non-identical, and it is also possible for analogous disparities between *aoy* and *akx* to be non-identical. Such non-identical corresponding disparities create opportunities for constraints to be non-ODP even as they evaluate disparities in isolation. To be non-ODP, such a constraint would need to be sensitive to the feature values that distinguish the non-identical corresponding disparities.

Input-referring constraints include what will here be called “value restricted” constraints, where the constraint is violated by a type of disparity only if distinguished segments of the disparity have one of a constraint-specified set of values for a particular feature. An example is value restricted IDENT constraints, where the constraint only evaluates IO correspondents for agreement on the value of a feature if one of the corresponding segments has one of a specific set of values for the feature. Pater (1999) first proposed constraints that were like IDENT, but restricted to correspondences in which the input segment had a particular value.[21] de Lacy (2002) proposed IDENT constraints restricted to correspondences in which the input segment had a value belonging to a subset of the possible values for a feature, in the context of markedness scales and scale category conflation. Such constraints are here labeled IDENT[$F_{in}\in V$]: IDENT[$F_{in}\in V$] is violated by any pair of IO correspondents such that the input correspondent’s value of feature F is a member of the set of values V, and the output correspondent’s value for F is different from the input correspondent’s value for F (regardless of whether the output correspondent’s value for F is a member of V). Such constraints are one possible generalization of IDENT[F]: IDENT[F] can be understood as IDENT[$F_{in}\in V$] where V is equal to the set of all possible values for F.

6.4.1 IDENT[$F_{in}\in V$] and IDENT[$F_{out}\in V$]

Constraints of this sort have a value restriction on one of the correspondents. These constraints are ODP; they cannot induce non-output driven maps. A proof of this is given for IDENT[$F_{in}\in V$] in section 10.4, and for IDENT[$F_{out}\in V$] in section 10.5. These constraints are value-restricted, but defined such that non-vacuous satisfaction of the constraint requires identity of the feature value between the correspondents. For IDENT[$F_{in}\in V$], if an input segment has a value for feature F in V, and an output correspondent, then the constraint can only be satisfied for that segment if its output correspondent has the same value for feature F. For IDENT[$F_{out}\in V$], if an output segment has a value for feature F in V, and an input correspondent, then the constraint can only be satisfied for that segment if its input correspondent has the same value for feature F.

6.4.2 IDENT[$F_{in}\in\{\alpha\}$, $F_{out}\in\{\delta\}$]

Constraints of this sort have value restrictions on both the input and output correspondents. Note that if the input and output correspondents are restricted to have the same single value of the feature, the constraint will never be violated. To get a constraint of this sort to be non-ODP requires that the feature have at least three distinct values. This is illustrated in Table 1, using a place feature pl with three values: coronal, labial, and dorsal.

[21] Pater used the notation IDENTI→O[F] for such constraints.

Table 1 IDENT[pl_{in}∈{dor}, pl_{out}∈{cor}] induces a chain shift.

		MAX	*DOR	IDENT[pl_{in}∈{dor}, pl_{out}∈{cor}]	*LAB	IDENT[pl]
☞	/tik/:[tip]				*	*
	/tik/:[tik]		*!			
	/tik/:[tit]			*!		*
	/tik/:[ti]	*!				
	/tip/:[tip]				*!	
	/tip/:[tik]		*!			*
☞	/tip/:[tit]					*
	/tip/:[ti]	*!				

in_a = /tik/ in_b = /tip/ out_x = [tip] out_y = [tit]
akx = /tik/:[tip] *bmx* = /tip/:[tip] *aoy* = /tik/:[tit] *bpy* = /tip/:[tit]
bmx has greater internal similarity than *akx*, yet it is not grammatical.

The key behavior of the constraint IDENT[pl_{in}∈{dor}, pl_{out}∈{cor}] lies in the shaded cells. In the first competition, the constraint penalizes the output coronal coda relative to the output labial coda, while in the second competition the constraint does not distinguish the two. In both competitions, the output coronal coda does not match its input correspondent in the value of place, but the disparity only violates the constraint in the first competition, when the input correspondent has dorsal place. This dependence on the value of an input feature that is not faithfully preserved makes the map non-output-driven.

One way to think about this example is with respect to a place markedness scale, with dorsal more marked than labial more marked than coronal. If dorsal place in the input is altered solely to avoid the markedness of dorsal, one would by default expect the place feature to be changed to the least marked value, coronal. The constraint IDENT[pl_{in}∈{dor}, pl_{out}∈{cor}], however, explicitly penalizes a change in place from most marked to least marked (on this simple three-valued scale). The constraint causes the place feature to change to a more marked value, labial. When an input segment has labial place, however, it vacuously satisfies IDENT[pl_{in}∈{dor}, pl_{out}∈{cor}], and place is allowed to change to coronal.

If the constraint is viewed as penalizing candidates which change a place feature "two steps" along the place markedness scale, then the constraint resembles Gnanadesikan's IDENT-ADJ[F] constraint (Gnanadesikan 1997), which is violated whenever input and output correspondents have feature values that are more than one step apart on the scale of values for that feature.

6.4.3 MAX[F_{in}∈{α}]

The effect, with respect to ODP, of restrictive conditions on the evaluation of a constraint depend in non-trivial ways on the interaction of the conditions with the rest of the definition of the constraint. This can be illustrated by contrasting two different input-referring constraints, both of which are conditioned by a value of a feature of an input segment.

IDENT[voi_{in}∈{+voi}] is ODP, as discussed in section 6.4.1. What bears emphasizing here is that the constraint is constrained so that it can only be violated by input segments that have the feature value +voi and that have output correspondents, and for those input segments it is only satisfied if the conditioning feature value, +voi, is identical in the output correspondent. Thus, conditions in the input referred to by the constraint, +voi, are required to be preserved in the output (provided the input segment has any output correspondent at all).

Things are very different for a constraint here called MAX[voi$_{in}$∈{+voi}]. This constraint is like MAX, but can only be violated by input segments that have the feature value +voi and do not have output correspondents. The key difference between MAX[voi$_{in}$∈{+voi}] and IDENT[voi$_{in}$∈{+voi}] is that, for an input segment s_{in} with the feature value +voi, MAX[voi$_{in}$∈{+voi}] is satisfied if s_{in} has an output correspondent **even if the output correspondent does not have the value +voi**. In terms of the disparities that constitute violations, every disparity that violates MAX[voi$_{in}$∈{+voi}] also violates MAX, just as every disparity that violates IDENT[voi$_{in}$∈{+voi}] also violates IDENT. But the failure to require identity of the conditioning feature value makes MAX[voi$_{in}$∈{+voi}] not ODP.

MAX[voi$_{in}$∈{+voi}] is very different from the constraint MAXLARYNGEAL proposed by Lombardi (Lombardi 2001). Although Lombardi describes this constraint in terms of a direct correspondence between features (specifically, between autosegments in the input and in the output), in practice the constraint appears to behave like a combination of IDENT and MAX: if a segment in the input is +voi, then that segment must have an output correspondent **and** the output correspondent must be +voi. The latter condition distinguishes MAXLARYNGEAL from MAX[voi$_{in}$∈{+voi}].

The failure of MAX[voi$_{in}$∈{+voi}] to be ODP can be illustrated in the context of coda conditions on voicing. As shown in Table 2, MAX[voi$_{in}$∈{+voi}] is violated when an underlyingly voiced segment, /g/, has no output correspondent, but is satisfied if /g/ has a voiceless output correspondent. When a candidate with greater internal similarity is formed by devoicing this segment to /k/ in the input, MAX[voi$_{in}$∈{+voi}] is satisfied when /k/ has no output correspondent, because it is not underlyingly voiced. This results in a chain shift: tig → tik → ti.

Table 2 MAX[voi$_{in}$∈{+voi}] induces a chain-shift.

		NOVOICODA	DEP	MAX[voi$_{in}$∈{+voi}]	NOCODA	MAX	IDENT[voi]
☞	/tig/:[tik]				*		*
	/tig/:[tig]	*!			*		
	/tig/:[ti]			*!		*	
	/tig/:[ti.ga]		*!				
	/tik/:[tik]				*!		
	/tik/:[tig]	*!			*		*
☞	/tik/:[ti]					*	
	/tik/:[ti.ka]		*!				

in_a = /tig/ in_b = /tik/ out_x = [tik] out_y = [ti]
akx = /tig/:[tik] *bmx* = /tik/:[tik] *aoy* = /tig/:[ti] *bpy* = /tik/:[ti]
bmx has greater internal similarity than *akx*, yet it is not grammatical.

While this particular example constraint is conditioned on the feature value +voi, there is nothing specific to voicing about the non-output-driven effect. In principle, the effect could be reproduced with any feature given the appropriate markedness constraints (here, NOVOICODA and NOCODA). Constraints like

MAX[C] (C for 'consonant') and MAX[V] (V for 'vowel')[22] have the same potential, provided that GEN permits candidates in which input vowels can correspond to output consonants (and vice-versa), and the constraints are satisfied by such correspondences.[23] Note that if the candidates with such correspondents simply aren't permitted by GEN, then MAX[C] and MAX[V] will likely be ODP (all other things being equal). The non-output-driven map caused by MAX[F_{in}∈{+voi}] depends crucially on the existence of candidate /tig/:[tik], where the conditioning voiced input segment has a voiceless output correspondent. If such a candidate isn't permitted, then it cannot be optimal. Assumptions about the structure of GEN often receive less attention, but they are no less important.

6.4.4 DEP[F_{out}∈{α}]

IDENT[voi_{out}∈{–voi}] is ODP, as discussed in section 6.3. What bears emphasizing here is that the constraint is constrained so that it can only be violated by output segments that have the feature value –voi and that have input correspondents, and for those output segments it is only satisfied if the conditioning feature value, –voi, is identical in the input correspondent. Thus, conditions in the output referred to by the constraint, –voi, are required to be supported by the input (provided the output segment has any input correspondent at all).

Things are very different for a constraint here called DEP[voi_{out}∈{–voi}]. This constraint is like DEP, but can only be violated by output segments that have the feature value –voi and do not have input correspondents. The key difference between DEP[voi_{out}∈{–voi}] and IDENT[voi_{out}∈{–voi}] is that, for an output segment s_{out} with the feature value –voi, DEP[voi_{out}∈{–voi}] is satisfied if s_{out} has an input correspondent **even if the input correspondent does not have the value –voi**. In terms of the disparities that constitute violations, every disparity that violates DEP[voi_{out}∈{–voi}] also violates DEP, just as every disparity that violates IDENT[voi_{out}∈{–voi}] also violates IDENT. But the failure to require identity of the conditioning feature value makes DEP[voi_{out}∈{–voi}] not ODP.

The failure of DEP[voi_{out}∈{–voi}] to be ODP can be illustrated in the context of coda conditions on voicing. The constraint FINALC (Prince and Smolensky 1993/2004) requires that the final segment in a word be a consonant. As shown in Table 3, DEP[voi_{out}∈{–voi}] is violated when an output voiceless segment, /k/, has no input correspondent, but is satisfied if /k/ has a voiced input correspondent. When a candidate with greater internal similarity is formed by adding the surface segment /g/ to the input, DEP[voi_{out}∈{–voi}] is satisfied by a competing candidate with final consonant /k/ when /k/ has an input correspondent, even though the input correspondent is underlyingly voiced. This results in a chain shift: ti → tig → tik.

---

[22] The origins of MAX[C] and MAX[V] can be traced to their pre-correspondence theory counterparts in containment faithfulness theory, PARSE^C and PARSE^V (Prince and Smolensky 1993/2004, p. 256).

[23] Note that if the conditions defining 'C' and 'V' aren't properties of segments themselves (such as if they were defined solely in terms of syllabic positions in the output), then the constraints become incoherent: you cannot identify if a segment lacking an output correspondent is a 'C' or a 'V', and thus cannot determine if the segment constitutes a violation or not.

Table 3 DEP[voi_{out}∈{–voi}] induces a chain-shift.

		FINALC	MAX	DEP[voi_{out}∈{–voi}]	NOVOICODA	DEP	IDENT[voi]
☞	/ti/:[tig]				*	*	
	/ti/:[ti]	*!					
	/ti/:[tik]			*!		*	
	/ti/:[ ]		*!*				
	/tig/:[tig]				*!	*	
	/tig/:[ti]	*!	*				
☞	/tig/:[tik]					*	*
	/tig/:[ ]		*!**				

in_a = /ti/ in_b = /tig/ out_x = [tig] out_y = [tik]
akx = /ti/:[tig] *bmx* = /tig/:[tig] *aoy* = /ti/:[tik] *bpy* = /tig/:[tik]
bmx has greater internal similarity than *akx*, yet it is not grammatical.

The key difference between IDENT[F_{out}∈{–voi}] and DEP[voi_{out}∈{–voi}] is the requirement of input-output identity on the value of the conditioning feature voi. The non-output-driven map is crucially made possible by the fact that DEP[voi_{out}∈{–voi}] does not require the support of the feature value –voi in the input correspondent.

7 Complex Constraints and Non-Output-Driven Maps

Constraints that cause non-output-driven maps must not be output-driven-preserving; they are non-ODP constraints. This section characterizes the relationships between non-ODP constraint behavior and non-output-driven maps.

7.1 Non-ODP Constraints and Non-output-driven Maps

Recall that for a constraint to be output-driven preserving (ODP), it must have the properties given in (69) and (70), repeated here:

(69) C(*bpy*) < C(*bmx*) entails C(*aoy*) < C(*akx*)

(70) C(*bpy*) = C(*bmx*) entails C(*aoy*) ≤ C(*akx*)

Any constraint which is not output-driven preserving (non-ODP) must have some instance in which it violates one of these properties. A constraint which has such an instance can be said to exhibit non-ODP behavior in that instance. Based upon the properties in (69) and (70), there are three kinds of non-ODP constraint behavior.

(71) [C(*bpy*) = C(*bmx*)] and [C(*aoy*) > C(*akx*)] violates (70).

(72) [C(*bpy*) < C(*bmx*)] and [C(*aoy*) > C(*akx*)] violates (69).

(73) [C(*bpy*) < C(*bmx*)] and [C(*aoy*) = C(*akx*)] violates (69).

Constraints that are non-ODP have the **potential** to cause non-output-driven maps. The mere presence of a non-ODP constraint in a constraint hierarchy does not ensure that the hierarchy defines a non-output-driven map, of course. For instance, if the constraint is dominated by other constraints such that it is never active on any candidate competition, it cannot cause non-output-drivenness for that ranking. It is also possible that a non-ODP constraint is active in a particular ranking, but interacts with other constraints such that output-drivenness holds. The discussion below will connect the definitions of different non-ODP constraints to examples of non-ODP behavior. However, no claim is made here about exclusivity: I explicitly leave open the possibility that the same non-ODP constraint could give rise to more than one type of non-ODP behavior in different circumstances.

All instances of non-output-drivenness have the following general format:

- *akx* is optimal.
- *bmx* has greater internal similarity than *akx*.
- *bmx* is not optimal.

If *bmx* is not optimal, then some other candidate for in_b, *bpy*, is optimal.

Different instances of non-output-drivenness can involve different kinds of non-ODP constraint behavior. The three kinds of non-ODP behavior can be distinguished in terms of the competitions in which they distinguish key pairs of candidates in the instance of non-output-drivenness. The non-ODP behavior described in (71) distinguishes the competitors for in_a, but not in_b. Thus, (71) involves **distinction only at lesser similarity**. The non-ODP behavior described in (73) distinguishes the competitors for in_b, but not in_a, hence the label **distinction only at greater similarity**. The non-ODP behavior described in (72) distinguishes the competitors for both in_a and in_b, but in opposite ways relative to the output forms, hence the label **distinction conflict**.

Observe that the only kind of distinction between *bmx* and *bpy* involved in non-ODP behavior is C(*bpy*)<C(*bmx*). If it were the case that C(*bmx*)<C(*bpy*), then the constraint could not play a role in causing *bmx* to lose to *bpy*. Thus, “distinction only at greater similarity” presumes that the constraint doesn’t just distinguish the competitors for in_b, but does so to the effect that C(*bpy*)<C(*bmx*).

Further observe that the only kind of distinction between *akx* and *aoy* involved in non-ODP behavior is C(*akx*)<C(*aoy*). If it were the case that C(*aoy*)<C(*akx*), then the constraint could not play a role in causing *akx* to beat *aoy*; it would need to be crucially dominated by some other constraint with the potential to decide both competitions. Thus, “distinction only at lesser similarity” presumes that the constraint doesn’t just distinguish the competitors for in_a, but does so to the effect that C(*akx*)<C(*aoy*).

7.2 Distinction only at lesser similarity

Kirchner analyzes Etxarri Basque vowel raising in hiatus (Kirchner 1995). Representationally, there are three vowel features involved: low, hi, and raised. In his analysis, the vowel [i^y] is both +hi and +raised, and is higher than the vowel [i] which is +hi and –raised. The non-output-drivenness takes the form of a chain-shift on vowel raising in hiatus: a → a, e → i → i^y. To keep the discussion and notation simple, I will focus on the vowel behavior, and leave the conditioning phonological context implicit. In the analysis, the sensitivity of raising to the hiatus context is contained in the markedness constraint which prefers the candidates exhibiting the appropriate raising (HIATUS RAISING).

Kirchner originally constructed his analysis using Parse-feature constraints. However, they can be translated into correspondence faithfulness terms; the translation used here is the same as that employed by Kager (1999, p. 392-397). The key constraint for realizing the chain shift, IDENT[hi&rai], is non-ODP.

(74) Correspondence-based statement of Kirchner's constraints

IDENT[hi] corresponding vowels must match in hi (Kirchner's PARSE_{hi})

IDENT[low] corresponding vowels must match in low (Kirchner's PARSE_{low})

IDENT[raised] corresponding vowels must match in raised (Kirchner's PARSE_{raised})

IDENT[hi] & IDENT[raised], abbreviated IDENT[hi&rai]

corresponding vowels should not differ in both hi and raised

HIATUS RAISING

markedness constraint penalizing non-maximally high vowels by degree

('a' 3 viols, 'e' 2 viols, 'i' 1 viol, 'i^y' 0 viol)

The constraint IDENT[hi&rai] is an example of a conjoined constraint (Smolensky 1995). A conjoined constraint is based upon two (or more) component constraints (here, IDENT[hi] and IDENT[raised]), and is violated when both of the component constraints are violated within a defined domain (here, the domain is a vowel). Of interest is the fact that the component constraints are both input-referring; each requires identity in the value of a feature between corresponding input and output vowels. Although the conjoined constraint refers to both features (hi and raised), it is satisfied even if only one of them is identical. This is the key to the non-output-drivenness in this analysis.

Table 4 Conjoined faithfulness exhibits distinction only at lesser similarity

		IDENT[low]	IDENT[hi&rai]	HIATUSRAISING	IDENT[hi]	IDENT[raised]
☞	/a/:[a]			***		
	/a/:[e]	*!		**		
	/a/:[i]	*!		*	*	
	/a/:[i^y]	*!	*		*	*
	/e/:[a]	*!		***		
	/e/:[e]			**!		
☞	/e/:[i]			*	*	
	/e/:[i^y]		*!		*	*
	/i/:[a]	*!		***	*	
	/i/:[e]			*!*	*	
	/i/:[i]			*!		
☞	/i/:[i^y]					*
	/i^y/:[a]	*!	*	***	*	*
	/i^y/:[e]		*!	**	*	*
	/i^y/:[i]			*!		*
☞	/i^y/:[i^y]					

Mappings: a → a, e → i → i^y

The map is non-output-driven because /e/:[i] is grammatical, and /i/:[i] has greater internal similarity than /e/:[i], yet /i/:[i] is not grammatical; instead /i/:[i^y] is grammatical. The situation can be expressed in terms of the analysis of this paper as follows.

in_a = /e/	in_b = /i/	out_x = [i]	out_y = [i^y]
akx = /e/:[i]	*bmx* = /i/:[i]	*aoy* = /e/:[i^y]	*bpy* = /i/:[i^y]

The map is non-output-driven because candidate *akx* is grammatical, and *bmx* has greater internal similarity than *akx*, yet *bmx* is not grammatical.

The key to understanding the non-output-driven map is understanding why the candidate *aoy*, /e/:[i^y], isn't grammatical in the first place. /e/:[i^y] is precisely the candidate for input /e/ eliminated by IDENT[hi&rai] in favor of *akx*, /e/:[i]. That is, IDENT[hi&rai](*aoy*) > IDENT[hi&rai](*akx*). But when the input is changed to /i/, which is even more like [i] than /e/ is, IDENT[hi&rai] goes silent, failing to distinguish between /i/:[i] and /i/:[i^y]. The key cells are shaded in the tableau in Table 4. Thus, the competition between /i/:[i] and /i/:[i^y] falls to the lower-ranked HIATUS RAISING, which has the opportunity the decide against the "output-driven" /i/:[i] (an opportunity it was denied by IDENT[hi&rai] in the comparison between /e/:[i] and /e/:[i^y]). In other words, IDENT[hi&rai](*bpy*) = IDENT[hi&rai](*bmx*) while IDENT[hi&rai](*aoy*) > IDENT[hi&rai](*akx*). Thus, IDENT[hi&rai] satisfies the condition in (71) (repeated below).

(71) [C(*bpy*) = C(*bmx*)] and [C(*aoy*) > C(*akx*)]

The behavior of IDENT[hi&rai] in this instance constitutes distinction only at lesser similarity: it distinguishes candidates for input /e/ with lesser similarity to the output [i^y], but does not distinguish the corresponding candidates for input /i/ with greater similarity the output [i^y].

The non-ODP behavior of IDENT[hi&rai] can be traced to its conjunction structure. IDENT[hi&rai] can be satisfied without enforcing identity to all of the elements that the constraint evaluates on input-output correspondence. Candidate /e/:[i^y] mismatches in both hi and raised, incurring a violation of IDENT[hi&rai]. /e/:[i] satisfies the constraint, even though one of the referenced features, hi, still doesn't match. In a sense, the constraint evaluates the disparity on the feature hi differently depending on context (whether or not the feature raised also doesn't match). The definition of IDENT[hi&rai] makes this relationship symmetric: the constraint also evaluates a disparity on the feature raised differently depending on the context (whether or not the feature hi also doesn't match).

7.3 Distinction only at greater similarity

Recall the condition for the non-output-driven constraint behavior "distinction only at greater similarity", repeated here:

(73) [C(*bpy*) < C(*bmx*)] and [C(*aoy*) = C(*akx*)]

This is "distinction only at greater similarity" because the constraint doesn't distinguish the candidates *akx* and *aoy*, two candidates for the input with lesser similarity to output out_x, but does distinguish the candidates *bmx* and *bpy*, two candidates for the input with greater similarity to output out_x.

"Distinction only at greater similarity" can be exhibited by a constraint that will here be described as an instance of constraint disjunction. Such a constraint is described in (75).

(75) IDENT[low] OR IDENT[hi], abbreviated IDENT[low|hi]: violated **once** for each IO correspondent pair that differs in the value of low or in the value of hi.

What is called constraint disjunction here is essentially the same as what Hewitt and Crowhurst (1996) proposed under the label "constraint conjunction", not to be confused with the sense of constraint conjunction proposed by Smolensky and discussed in section 7.1.[24] The tableau in Table 5 illustrates the behavior of IDENT[low|hi] relative to the individual constraints IDENT[low] and IDENT[hi], as well as to the conjoined (in the Smolenskian sense) constraint ID[low&hi].

Table 5 Constraint disjunction

	IDENT[low]	IDENT[hi]	IDENT[low&hi]	IDENT[low|hi]
/a/:[a]				
/a/:[e]	*			*
/a/:[i]	*	*	*	*
/e/:[a]	*			*
/e/:[e]				
/e/:[i]		*		*
/i/:[a]	*	*	*	*
/i/:[e]		*		*
/i/:[i]				

The shaded cells highlight the differing non-ODP behavior for constraint conjunction and constraint disjunction. For both, recall that /e/:[e] has greater internal similarity than /a/:[e]. The constraint formed by conjunction, IDENT[low&hi], exhibits distinction only at lesser similarity, distinguishing the candidates for /a/ (in favor of /a/:[e], crucially) but not for /e/. The constraint formed by disjunction, IDENT[low|hi], exhibits distinction only at greater similarity, distinguishing the key candidates for /e/ (in favor of /e/:[e], crucially), but not for /a/. Note that the issue is one of distinction, not violation vs. non-violation: both key candidates for /a/ violate IDENT[low|hi], but they violate it an equal number of times.

This distinction only at greater similarity behavior is active in the non-output-driven map shown in Table 6.

[24] The requirement that the constraint assess at most one violation for each locus of evaluation is important here, and it was important to the proposal of Hewitt and Crowhurst, who used the device to mask the gradient evaluation of one of the constraints being combined. The particular constraints constructed by Hewitt and Crowhurst in (Hewitt and Crowhurst 1996) are of no particular interest here: they are constructed from markedness constraints, so the constructed constraints are markedness constraints, and therefore output-driven preserving.
The confusing constraint nomenclature is a consequence of differing perspectives on constraint characterization. Smolenskian constraint conjunction defines the conditions for **violation** of the constructed constraint as the logical conjunction of the conditions for **violation** of the factor constraints. Hewitt and Crowhurst style conjunction defines the conditions for **satisfaction** of the constructed constraint as the logical conjunction of the conditions for **satisfaction** of the factor constraints. A simple application of the appropriate DeMorgan theorem reveals that [satisfaction of the constructed constraint as the **conjunction** of conditions for satisfaction of the factor constraints] is equivalent to [violation of the constructed constraint as the **disjunction** of conditions for violation of the factor constraints]. I opt for characterization of the constraints in terms of the conditions under which they are violated.

Table 6 Non-output-driven map due to distinction only at greater similarity

		*[+low]	IDENT[low|hi]	*[–hi]	IDENT[low]	IDENT[hi]
	/a/:[a]	*!		*		
	/a/:[e]		*	*!	*	
☞	/a/:[i]		*		*	*
	/e/:[a]	*!	*	*	*	
☞	/e/:[e]			*		
	/e/:[i]		*!			*
	/i/:[a]	*!	*	*	*	*
	/i/:[e]		*!	*		*
☞	/i/:[i]					

The non-output-drivenness of the map is demonstrated with the following candidate label assignments.

in_a = /a/	in_b = /e/	out_x = [i]	out_y = [e]
akx = /a/:[i]	*bmx* = /e/:[i]	*aoy* = /a/:[e]	*bpy* = /e/:[e]

The map is /a/→[i], /e/→[e], /i/→[i]. The map is non-output-driven: /a/:[i] is grammatical, and /e/ is closer to [i] than /a/ is, but /e/:[i] is not grammatical. Note that the map does not exhibit a chain shift: the well-formed outputs are [e] and [i], and each of those maps to itself. In an output-driven map the grammaticality of /a/:[i] would entail the grammaticality of /e/:[i], but the constraint IDENT[low|hi] "punches a hole" in the space of inputs, a hole located at input /e/.

The "distinction only at greater similarity" behavior is crucially a consequence of the fact that the disjoined constraint IDENT[low|hi] does not assess two violations when both of its disjuncts are violated at a single locus of violation (a single pair of input-output corresponding segments). In a candidate like /a/:[i], where the locus of violation (the vowel) has non-identity for the values of both low and hi, the constraint still only assesses a single violation, rendering it incapable of distinguishing violations of both disjuncts from violation of only a single disjunct. This failure to distinguish is what prevents distinction at lesser similarity (for the input /a/).

The conjoined constraint IDENT[low&hi] and the disjoined constraint IDENT[low|hi] have some similarities in their non-ODP behavior. Both are violated by mappings with differing values of both low and hi (maximally dissimilar with respect to the two features), and both are satisfied by an identity mapping with differing values of neither low nor hi (maximally similar with respect to the two features). The conjoined constraint is unable to distinguish between one differing value and none, while the disjoined constraint is unable to distinguish between two differing values and one. Both of them "conflate" regions along the similarity path, but they conflate different regions.

In an interesting way, the disjoined constraint IDENT[low|hi] combines the two features into a single composite feature, and evaluates segments with respect to identity on the composite. In an analysis where these are the only two features used, the effect is similar to that of basing internal similarity solely on segment identity, as was discussed in section 3.2: from the point of view of the constraint, segments are either identical or they are not, with no further distinctions. The potential for non-output-drivenness arises when an input-referring constraint like IDENT[low|hi], that makes no further distinctions, is combined with a theory of relative similarity that does make further distinctions.

7.4 Distinction Conflict

The condition for distinction conflict is given in (72), repeated here.

(72) [C(*bpy*) < C(*bmx*)] and [C(*aoy*) > C(*akx*)]

A simple example of a constraint exhibiting this sort of behavior is an input-output antifaithfulness constraint.[25] The constraint NONIDENT[low], defined in (76), requires that an output segment be **non-identical** to its input correspondent in the value of the feature low.

(76) NONIDENT[low] IO correspondents should differ in the value of low.

The tableaux in Table 7 show how NONIDENT[low] is able, when sufficiently active, to cause a non-output-driven map.

Table 7 Tableaux showing distinction conflict for NONIDENT[low]

		NONIDENT[low]	IDENT[hi]
	/a/:[a]	*!	
☞	/a/:[e]		
	/a/:[i]		*!
☞	/e/:[a]		
	/e/:[e]	*!	
	/e/:[i]	*!	*
☞	/i/:[a]		*
	/i/:[e]	*!	*
	/i/:[i]	*!	

The non-output-drivenness of the map is demonstrated with the following candidate label assignments.

in_a = /a/	in_b = /e/	out_x = [e]	out_y = [a]
akx = /a/:[e]	*bmx* = /e/:[e]	*aoy* = /a/:[a]	*bpy* = /e/:[a]

bmx has greater internal similarity than *akx* because *bmx* lacks the mismatch in the value of feature low which *akx* has (there are no other disparities in these candidates). The map is non-output-driven because candidate *akx* is optimal, and *bmx* has greater internal similarity than *akx*, yet *bmx* is not optimal.

This map includes a circular chain shift: a→e, e→a. Note that the input-referring constraint under consideration, NONIDENT[low], requires non-identity between input and output, rather than identity. Moreton (2004) proved that in Optimality Theoretic systems in which all input-referring constraints assess no violations to identity map candidates (candidates with no disparities), circular chain shifts are impossible.

NONIDENT[low] exhibits "distinction conflict" behavior in this instance: the constraint prefers *akx* to *aoy*, but prefers *bpy* to *bmx*.

[25] Input-output antifaithfulness is different from transderivational antifaithfulness (Alderete 1999a), which does not evaluate input-output correspondence but instead evaluates an output-output correspondence (Benua 1997) between a morphologically derived form and its base form. Output-output correspondence is beyond the scope of this paper.

It is possible to get this kind of behavior out of certain other constraints via constraint interaction. This has been demonstrated by Wolf (2006) for constraints constructed by material implication between a faithfulness constraint and a markedness constraint. A complex constraint constructed via material implication has the logical form $C_1 \rightarrow C_2$: for a specified locus of evaluation (such as a segment), if C_1 is satisfied, then C_2 must also be satisfied (Archangeli et al. 1998, Balari et al. 2000).[26] Wolf points out that it is possible to get anti-faithfulness type behavior out of a constraint of the form F→M, where F is a faithfulness constraint and M is a markedness constraint.

Employing a strategy that is similar to Wolf's, I here show how to create the effects of NONIDENT[low] with an implicationally defined constraint [IDENT[low]→*[–hi]] in interaction with a dominating markedness constraint *[+hi]. Seeing how is a little easier if you consider the logically equivalent disjunctive form of an implication: P→Q is the same as ¬P OR Q. Putting the implicational constraint in this form yields ¬[IDENT[low]] OR *[–hi]. In other words, for each output segment with an input correspondent, the segment must either **disagree** with its input correspondent on the feature low or be +hi. The expression ¬[IDENT[low]] is equivalent to NONIDENT[low]. Thus, satisfying the implicationally defined constraint [IDENT[low]→*[–hi]] requires satisfaction of either NONIDENT[low] or *[–hi]. The effects of NONIDENT[low] can be derived by having the constraint *[+hi] dominate, which effectively eliminates the possibility of satisfying the implicational constraint via satisfaction of *[–hi]. The use of these constraints to achieve the circular chainshift map is shown in Table 8.

Table 8 Antifaithfulness effects with an implicationally defined constraint

		*[+hi]	[IDENT[low]→*[–hi]]	IDENT[hi]
	/a/:[a]		*!	
☞	/a/:[e]			
	/a/:[i]	*!		*
☞	/e/:[a]			
	/e/:[e]		*!	
	/e/:[i]	*!		*
☞	/i/:[a]			*
	/i/:[e]		*!	*
	/i/:[i]	*!		

The map defined in Table 8 is identical to the one in Table 7. The demonstration that [IDENT[low]→*[–hi]] is not output-driven-preserving is also identical to that for NONIDENT[low], because the two constraints evaluate the same four candidates { *akx* , *bmx* , *aoy* , *bpy* } identically. The two candidates evaluated differently by [IDENT[low]→*[-hi]] and NONIDENT[low], /e/:[i] and /i/:[i], are handled by the dominating constraint *[+hi].

7.5 Faithfulness Conditioned on Output Context

Faithfulness constraints have been proposed that evaluate correspondence relations conditioned upon properties of the output. An example is HEAD-DEPENDENCE (HEAD-DEP), a version of DEP that is

[26] The definition of construction via implication discussed here is distinct from the definition proposed by Crowhurst and Hewitt (1997); see (Balari et al. 2000, Wolf 2006) for explanatory discussion.

conditioned to be violable only by output segments that are in prosodic head positions (Alderete 1999b). Prosodic heads are frequent targets of stress assignment, and indeed much of the data motivating this constraint involves languages that actively avoid stressing epenthetic vowels. A stressed epenthetic vowel constitutes a violation of HEAD-DEP because the assignment of stress requires that the vowel be in a prosodic head position, and being epenthetic means having no input correspondent, that is, a violation of DEP.

HEAD-DEP: every segment contained in a prosodic head has an input correspondent.

The non-output-driven effects of HEAD-DEP are here illustrated with data from Dakota, shown in (77). The analysis of the data comes from Alderete (1999b); the data originate with Shaw (1976, 1985).

(77) Stress in Dakota (epenthesized vowels are underlined)

č^{h}ikté	‘I kill you’	/ček/ → [čéka]	‘stagger’
mayákte	‘you kill me’	/khuš/ → [khúša]	‘lazy’
wičháyakte	‘you kill them’	/čap/ → [čápa]	‘trot’
owíčhayakte	‘you kill them there’		

Main stress in Dakota falls on the second syllable from the beginning by default.[27] However, if the vowel of the second syllable is epenthetic, then main stress shifts to the first syllable. This requires that the grammar distinguish between surface vowels that are epenthetic and those that are not. The result is a non-output-driven map, as is demonstrated with the following candidate label assignments.

in_a = /čap/	in_b = /čapa/	out_x = [čápa]	out_y = [čapá]
akx = /čap/:[čápa]	*bmx* = /čapa/:[čápa]	*aoy* = /čap/:[čapá]	*bpy* = /čapa/:[čapá]

bmx has greater internal similarity than *akx*, yet it is not grammatical; *bpy* is grammatical in Dakota.

The interaction between epenthesis and stress is realized with HEAD-DEP, as shown in Table 9. The constraint MAINFOOTLEFT requires that the head foot of the word (which assigns main stress) appear at the beginning of the word; this constraint dominates all constraints that might otherwise pull the head foot away from the beginning of the word (like IDENT[stress] and WEIGHTTOSTRESS), thus keeping main stress on one of the first two syllables in longer words in Dakota.

Table 9 Non-output-driven map in Dakota

		MAINFOOTLEFT	HEAD-DEP	IAMBIC
☞	/čap/:[čápa]			*
	/čap/:[čapá]		*!	
	/čapa/:[čápa]			*!
☞	/čapa/:[čapá]			

HEAD-DEP exhibits distinction only at lesser similarity here. It prefers *akx* to *aoy*, but has no preference between *bmx* and *bpy*.

HEAD-DEP is an example of a positional faithfulness constraint (Beckman 1999): a faithfulness constraint that is restricted to the evaluation of segments that appear in a specified (normally prominent) position in

[27] If the word is monosyllabic, stress falls on the first (and only) syllable of the word: /kte/ → kté ‘s/he, it kills’ (Shaw 1976, Shaw 1985).

the output. There are two ways to satisfy a potential violation of a positional faithfulness constraint. One is to avoid the relevant faithfulness violation for the segment in the prominent position. This is the kind of effect that normally motivates positional faithfulness constraints: the observation of greater contrast in prominent positions (Beckman 1999). However, a positional faithfulness constraint violation can also be avoided by not permitting the output segment with the relevant faithfulness violation to appear in the relevant kind of prominent position. This is the kind of effect exhibited in the analysis of Dakota: the segment exhibiting the faithfulness violation (the epenthetic vowel) is prevented from appearing in the prominent position (the prosodic head bearing main stress), accomplished by shifting the prosodic head position to the first syllable. The effect is achieved in the analysis of Dakota because of the existence of another constraint, IAMBIC, that is violated by the output in which the prominent position is shifted to avoid the faithfulness-violating segment (IAMBIC prevents the prosodic head from simply always being the first syllable).

It bears emphasizing that the non-ODP quality of positional faithfulness constraints is not crucially dependent on the lack of constraints which enforce faithfulness to the specification of particular prosodic positions in the input. In fact, lexical stress languages require that the linguistic system allow the capability for the specification of main stress in inputs and constraints capable of preserving lexically specified stress in the output. Faithfulness to underlying stress is effectively faithfulness to a lexically specified prosodic position. In the analysis of Dakota above, the relevant constraint(s) that prefers candidates that preserve lexical stress must be dominated in the ranking by the constraints determining the default stress pattern (here, MAINFOOTLEFT and IAMBIC). For further discussion of the interaction between constraints enforcing predictable stress, lexical stress, and stress-epenthesis interaction, see Alderete & Tesar (2002).

In the analysis of Dakota, the non-ODP constraint was a version of DEP with evaluation restricted to certain prominent positions. Other positional faithfulness constraints have been proposed involving restrictions of IDENT[F] to prominent positions. IDENT-ONSET[voice] (Lombardi 1995, Lombardi 2001, Padgett 1995) restricts evaluation of voice identity to input-output correspondents in which the output segment is in a syllable onset. This constraint is non-ODP and can, under the right circumstances, induce non-output-driven mappings.

Positional faithfulness constraints restrict the evaluation of faithfulness to prominent positions, but the same kind of effect can in principle be achieved by any output-based restriction on the evaluation of faithfulness. This is shown by Itô and Mester's discussion of constraints formed by the conjunction of markedness and faithfulness constraints (Itô and Mester 2003). Such a constraint is violated by a segment in the output that is the simultaneous locus of a violation of the conjunct markedness constraint (an output condition) and a violation of the conjunct faithfulness constraint. The locus of violation of the markedness constraint effectively defines the "position" that the faithfulness constraint evaluation is restricted to.

In general, non-output-drivenness due to output conditioning need not necessarily be a consequence of restricting the evaluation of disparities to those within prominent prosodic positions, nor of disparities involving independently marked output structures. It is a possible effect of constraints that only evaluate disparities of output segments meeting conditions that are independent of the disparities themselves.

7.6 Reference to Other Forms: Sympathy

Some OT constraints derive opacity effects by making reference to additional forms other than the input or the output. Sympathy constraints (McCarthy 1999) are an example. I will give only a very brief description here. A sympathy constraint, when evaluating a candidate, makes reference to an independent candidate, the ❀ (flower) candidate. The ❀ candidate is the **most harmonic candidate** that fully satisfies some designated faithfulness constraint. The sympathy constraint then evaluates all candidates with respect to a correspondence relation between the outputs of those candidates and the output of the ❀ candidate, typically evaluating a different dimension of correspondence than the one used to determine the ❀ candidate.

Sympathy constraints pose challenges for analysis in terms of output-drivenness, because they are **non-stationary**: their evaluation of a candidate depends not only on the candidate, but on an assumed ranking of the constraints.[28] Changing the ranking can change the number of violations a sympathy constraint assesses to a candidate. This is due to the fact that the ❀ candidate referred to by a sympathy constraint is determined with reference to harmony (which is defined by a full ranking); changing the ranking can change the ❀ candidate. Thus, the criteria for ODP cannot be applied to the constraint in isolation.

However, one can observe a sympathy constraint in a particular context, with respect to an assumed ranking. If the sympathy constraint plays a key role in determining a non-output-driven mapping, it is because it exhibits a non-ODP behavior in that context. This is illustrated here using the Tiberian Hebrew example from McCarthy (1999). The two key mappings for the example are given in (78) and (79).

(78) /dešʔ/→[deše]

(79) /deš/→[deš]

This is a non-output-driven situation, because /deš/:[deše] has greater internal similarity than the grammatical /dešʔ/:[deše], but is not itself grammatical. Simplifying the analysis to just the relevant elements yields the tableaux in Table 10.

Table 10 The sympathy candidate exhibits distinction only at lesser similarity.

		❀MAX-V$_{\text{MAX-C}}$	CODA-COND	DEP-V
☞	/dešʔ/:[deše]			*
	/dešʔ/:[deš]	*!		
❀	/dešʔ/:[dešeʔ]		*!	*
	/deš/:[deše]			*!
☞ ❀	/deš/:[deš]			

in_a = /dešʔ/ in_b = /deš/ out_x = [deše] out_y = [deš]
akx = /dešʔ/:[deše] *bmx* = /deš/:[deše] *aoy* = /dešʔ/:[deš] *bpy* = /deš/:[deš]
bmx has greater internal similarity than *akx*, yet it is not grammatical.

The shaded cells indicate the non-ODP behavior: the sympathy constraint ❀MAX-V$_{\text{MAX-C}}$ is exhibiting distinction only at lesser similarity, giving rise to the non-output-driven map. The lesser similarity input, /dešʔ/, differs by having the final glottal stop. The ❀ candidate is the one which best satisfies MAX-C, so that glottal stop will not be deleted in the ❀ candidate; instead, a vowel is epenthesized. The sympathy constraint evaluates a correspondence between each of the candidates and the ❀ candidate with respect to MAX-V (requiring each vowel of the output of the ❀ candidate to have a correspondent in the output of the candidate being evaluated). Crucially, the ❀ constraint can be satisfied without preserving the glottal stop that is necessarily preserved in the ❀ candidate, so the glottal stop does not appear in the output of the optimal candidate, [deše].

[28] Other non-stationary constraints pose the same challenges of analysis, including output-output faithfulness constraints (Benua 1997) and targeted constraints (Wilson 2001).

The greater similarity input, /deš/, does not have the glottal stop, therefore MAX-C can be fully satisfied without any vowel epenthesis in the ❀ candidate. The two candidates shown for /deš/ are not distinguished by the sympathy constraint, because the ❀ candidate has one vowel, with a correspondent in the outputs of each of the two candidates. The greater similarity is due to the lack of the glottal stop, the lack of the glottal stop results in the lack of a second vowel in the ❀ candidate, and the lack of a second vowel in the ❀ candidate results in a failure of the sympathy constraint to distinguish the two candidates.

7.7 Kinds of Non-Output-driven Situations

The non-output-driven situations discussed above are described as chain-shift and non-chain-shift situations. The different patterns are distinguished by the relationships between the input and output forms of the key candidates. Chain-shifts are situations in which candidate *bmx*, with in_b and out_x, has no disparities. In other words, the candidate with greater internal similarity that fails to be grammatical is the one with no disparities; the input is "the same" as the output, at least as far as correspondence is concerned. The Etxarri Basque example in section 7.2 is such a case. In non-chain-shift situations, the candidate with greater internal similarity that fails to be grammatical does not have zero disparities; the input is **not** "the same" as the output. This is true of the examples discussed in section 7.3, as well as the Tiberian Hebrew example discussed in section 7.6.

There is no simple correlation between the type of non-ODP behavior exhibited by a constraint and the kind of non-output-driven situation it causes. In both the chain-shift analysis of Etxarri Basque and the non-chain-shift analysis of Tiberian Hebrew, the key non-ODP constraint exhibits distinction only at lesser similarity. In the situation described for Dakota stress in section 7.5, the constraint HEAD-DEP exhibits distinction only at lesser similarity, but the situation is not a chain shift: *bmx* = /čapa/:[čápa] has a single disparity, the stress on the first vowel. However, the same constraint, again exhibiting distinction only at lesser similarity, can also cause a chain-shift, in the same language, if input /čápa/ is chosen: the optimal candidates /čap/:[čápa] and /čápa/:[čapá] form the basis for a chain-shift.

Table 11 HEAD-DEP causes a chain-shift.

		MAINFOOTLEFT	HEAD-DEP	IAMBIC
☞	/čap/:[čápa]			*
	/čap/:[čapá]		*!	
	/čápa/:[čápa]			*!
☞	/čápa/:[čapá]			

8 Discussion

8.1 Representational Issues

The formal analyses given in this paper are based on the representational assumptions given in (25). The aspects of representation that are important to any characterization of output-drivenness are those involving input-output correspondence: the kinds of representational elements that can be correspondents. The significance of segment-based representations for the analyses given here is that segments are the sole basis for input-output correspondence. The analyses are also based on the prescribed ways of evaluating input-output correspondences: these are the types of disparities (27). Obviously, any inventory of disparities is dependent on the nature of the input-output correspondence employed.

8.1.1 Nonidentical Corresponding Representational Elements

The initial intuition behind output-drivenness is that, if an input is non-faithfully mapped to an output, you continue to get the same output as you change to input to be more like the output. This intuition rests on the intuition that inputs and outputs are fundamentally built out of the same kinds of "things". For example, inputs and outputs both consist of segments, specifically the same kinds of segments, ones with the same features. Stepping away from that coarse intuition, what is really assumed is that the elements of inputs and outputs that stand in correspondence are the same kinds of things. The analyses of output-drivenness given in this paper had no trouble with grammars in which output forms had prosodic structures like syllables that inputs lacked, because it was assumed that such prosodic elements do not stand in IO correspondence relations.[29]

It is possible to relax the assumption that elements standing in IO correspondence are exactly the same types of objects. What is required is a specification of which IO correspondence relationships qualify as "faithful" ones. An example of this would be an analysis in which input segments have the feature accent, whereas output segments have instead the features stress, tone, and pitch_accent. In such an analysis, an input segment with the feature value +accent would not have a feature value disparity if its output correspondent had any of the values +stress, +tone, or +pitch_accent. Note that the term "feature" could be construed rather generically here as denoting any representational element: an output segment having the feature value +stress might actually be realized in the output representation as the output segment having a certain kind of projection on a prominence grid. What matters is that the analysis specify which pairs of respective properties of the IO corresponding elements count as faithful, that do not constitute disparities.

8.1.2 Non-unique Correspondence

The assumptions in (25) include the assumption that correspondence is unique, in the sense that any element has at most one IO correspondent. Expanding the analysis of output-driven maps to include non-unique correspondence (coalescence and breaking) will be non-trivial. Adding violations of uniqueness to the inventory of disparities requires specifying precisely how to individuate the instances. The individuation is necessary to establish a correspondence between the disparities of two candidates having the same output form, to determine if one candidate has greater internal similarity than the other.

Individual segmental correspondences alone cannot constitute individuated disparities: a correspondence between an input segment and an output segment isn't a non-uniqueness disparity on its own, only when combined with another segmental correspondence involving either the same input segment or the same output segment. A segment having more than one IO correspondent could constitute an individuated non-uniqueness disparity; the disparities would be localized to single segments. This might prove inadequate, at least in an identity-based relative similarity, because segments with non-identical sets of correspondents would simply constitute non-identical disparities. The potential inadequacy lies in the intuition that if one disparity involves a segment with three IO correspondents, and another disparity involves the same segment with two of the three IO correspondents, the latter disparity should somehow count as "greater similarity", rather than simply non-identity; one of the offending IO correspondences has been removed, and that ought to result in greater internal similarity (other things being equal). If preserving this intuition ultimately proves desirable, it might help to shift from a purely identity-based relative similarity to one in which distinct disparities can stand in an ordered similarity relation, so that one non-uniqueness disparity can count as "greater similarity" than another if the first disparity involves IO correspondence with a set of segments that is a strict subset (short of the empty set) of the set of corresponding segments in the second disparity.

[29] Moreton calls such elements 'nonhomogeneous elements' (Moreton 2004).

Further pursuit of these issues is left to future research.

8.1.3 Autosegmental Representation

The implications of autosegmental representations for the analysis of output-drivenness given here depend upon precisely how such representations are used in an analysis. Output-drivenness concerns IO correspondence. If output representations are realized with root nodes linked to autosegments as an expression of feature values, and the only IO correspondents are input segments and output root nodes (segments, effectively), then the segment-based analysis of output-drivenness given here could apply without alteration.

On the other hand, the output-driven analysis would need to be extended to handle representational theories in which elements other than segments can be IO correspondents. This would be the case if feature values were realized as autosegments in both the input and output, and autosegments could correspond independent of segmental affiliation. Within Optimality Theory, the work of Lombardi (2001), among others, presumes the existence of multiple correspondence relations of this sort.

Zoll proposes a more flexible view of a single IO correspondence relation in her work on floating features, proposing that both inputs and outputs can contain floating features as well as full segments, and that floating features in the input can correspond to either full output segments or to output floating segments (Zoll 1998, p. 40). Analyzing output-drivenness with those representational assumptions would require addressing a number of issues, such as the implications for internal similarity when input floating features are replaced by full segments (and vice-versa).

8.2 Non-output-driven Maps and Non-output-driven Grammars

This paper has focused on output-drivenness as a property of maps, and in particular on maps defined by Optimality Theoretic systems. It is possible to define an overall grammar, which is itself a map, as the composition of other maps. This has multiple precedents in phonological theory. A system of ordered rules (Chomsky and Halle 1968) is a composition of maps, where each rule that applies is a map. Lexical phonology (Kiparsky 1982, Kiparsky 1985) breaks a grammar into composed maps along morphological lines.

Thus, two ways of introducing non-output-drivenness into a grammar can be identified. One is to have a basic map be non-output-driven. This is the case when OT constraints which are not ODP cause a map to be non-output-driven, as described in section 7. The other way is to have a grammar consist of the composition of maps (each possibly output-driven) which are non-identical. The latter approach would attribute the non-output-drivenness of an overall grammar to interactions between the composed maps of the grammar.

Note that the composition of two or more identical output-driven maps is always equivalent to a single instance of that map, because output-driven maps are idempotent: a possible output of the map always maps to itself. But two distinct output-driven maps, when composed, can result in an overall map that is not output-driven.

Kiparsky’s theory of stratal OT (2003, to appear) conceives of a grammar as a serial composition of maps, each of which is realized by an Optimality Theoretic map. Bermudez-Otero has argued that each of the individual OT maps in stratal OT should be non-opaque, in the sense of process-based phonological opacity, and that opacity should arise only as the consequence of interaction between non-identical OT strata within the grammar (Bermudez-Otero 2003). An alternative approach, also employing Optimality Theory, is proposed by Itô and Mester. They also envision a grammar consisting of the composition of at least two maps each of which is defined by OT rankings. However, they suggest that different kinds of phonological opacity should be handled in different ways, with some the result of opaque maps defined by individual OT rankings, and others the result of interaction between the composed rankings of the grammar (Itô and Mester 2001, Itô and Mester 2003).

It is not hard to imagine analogous claims for the structure of linguistic theory being made with respect to output-drivenness as defined in this paper. A proposal analogous to that of Bermudez-Otero would claim that all input-referring constraints are ODP, and that non-output-drivenness only arises as a consequence of interactions between composed stratal maps.

8.3 Diagnosing Non-Output-driven Maps

The result in section 5.2.3 ensures that, for OT systems that are correspondence uniform, a map can be non-output-driven only through the activity of at least one constraint that is not ODP. Thus, the potential for non-output-driven maps in an OT system can be localized to specific constraints, and further localized to specific elements of the behavior of those constraints. Presuming correspondence uniformity, any instance of non-output-drivenness in a map is attributable to a constraint exhibiting one of the three kinds of non-ODP behavior described in section 7.1.

The mere presence of a non-ODP constraint in a ranking does not by itself ensure that the map defined by the ranking is non-output-driven. The non-ODP constraint must be ranked appropriately with respect to other constraints. First, the constraint must be active; the “potential” for non-output-drivenness will remain unrealized if the non-ODP constraint never has an opportunity to eliminate candidates. More specifically, the non-ODP constraint must be active on the candidates that characterize the non-ODP behavior, for at least one instance of that non-ODP behavior.

The non-ODP behaviors “distinction only at lesser similarity” and “distinction only at greater similarity” require cooperation from at least one lower-ranked constraint in order to actually realize non-output-drivenness. Both distinguish candidates for two outputs with respect to one input, eliminating one of the candidates from that competition, but do not distinguish the corresponding candidates for the same two outputs with respect to the other input, leaving it to a lower-ranked constraint to decide between them. Non-output-drivenness requires that the lower ranked constraint choose, for the latter input, in favor of the candidate for the output whose candidate was eliminated by the non-ODP constraint for the former input. In a sense, the non-ODP constraint must work in cooperation with a lower-ranked constraint that chooses “the other way” in the case that the non-ODP constraint does not decide. Further, it must be the case that neither of these candidates is eliminated by any other constraint further down in the hierarchy (in favor of some other candidate still active in its respective competition).

In the example of “distinction only at lesser similarity” given in Table 4, the non-ODP constraint, IDENT[hi&rai], chooses [i] over [i^y] for input /e/, but does not distinguish between [i] and [i^y] for input /i/. The lower-ranked constraint, HIATUSRAISING, chooses “the other way” for input /i/, choosing [i^y] over [i]. In the example of “distinction only at greater similarity” given in Table 6, the non-ODP constraint, IDENT[low|hi], chooses [e] over [i] for input /e/, but does not distinguish between [e] and [i] for input /a/. The lower-ranked constraint, *[–hi], chooses “the other way” for input /i/, choosing [i] over [e]. Notice that in both cases, if the lower-ranked constraint were instead ranked above the non-ODP constraint, the resulting map would be output-driven: the choice between the candidates would be decided the same way for both inputs.

8.4 Structure in the Input Space and the Relation to Learning

If a phonological map is output-driven, then the map’s input space is guaranteed to have a certain structure, relative to the map. Every phonotactically well-formed input (more properly put, every input forming a candidate with no disparities with a phonotactically well-formed output) identity-maps, and is the anchor for a "basin of attraction", where the basin consists of all inputs that map to that same output.[30] The further away (measured in disparities) an input is from the output to which it maps, the further out it is in the basin, and the greater the number of other inputs whose mapping it determines by implication (all the inputs forming candidates with greater internal similarity must map to the same output). The boundaries between these basins in input space constitute the thresholds of contrast in the map: each basin constitutes a distinct contrastive form. Every contrastive output form corresponds to a contiguous region of the input space, anchored on an input forming a candidate with minimal disparities.

Requiring maps to be output-driven is a stronger condition than merely requiring them to have the identity map property. This is demonstrated by maps like the ones discussed in section 7.3 involving distinction only at greater similarity, which are not output-driven but do have the identity map property. The identity map property requires that if any input form maps to a surface form, then the input identical to the surface form must map to that surface form, but says nothing further about the relationship among the input forms that map to the same output. Output-drivenness requires more of the map, such that lesser internal similarity mappings entail greater internal similarity mappings, with the identity map property being a special case of such entailment. Requiring that a map be output-driven imposes structure on the input space, relative to the map. It places restrictions on relations between different inputs and their outputs, including inputs that do not surface identically, whereas the identity map property only places restrictions on inputs that are identical to grammatical outputs.

The identity map property has played a non-trivial role in language learning, in particular in the learning of phonotactics (Hayes 2004, Prince and Tesar 2004). Work on phonotactic learning has posited that learners presume the identity map property, and adopt underlying forms that are identical to observed surface forms for the purposes of learning about phonotactic restrictions, prior to the acquisition/utilization of information about morphemic identity and alternation. To the extent that the identity map property actually holds of the target language, this is safe, because even if the actual underlying form for a word is in fact non-identical to the surface form, the map nevertheless does map the surface-identical underlying form to that surface form; the constructed mapping from the form to itself is in fact a mapping of the overall map. This allows phonotactic learning to learn some properties of the map prior to determination of morphemic identity and underlying forms for morphemes.

There is much less of a direct role for the identity map property to play in the learning of underlying forms. The primary contribution comes via phonotactic learning: the information about the map obtained via phonotactic learning can contribute to the learning of underlying forms (Tesar and Prince to appear) (see also (Jarosz 2006)). The limitation comes from the fact that the identity map property says nothing about inputs that are non-identical to grammatical outputs, save that they map to some or other grammatical output.

The structure in the space of possible inputs imposed upon output-driven maps could have an important role in the learning of phonological underlying forms, a more profound role than the identity map property is capable of. The role that this structure could play would depend upon the assumptions one makes about the relationship of output-driven maps to overall phonology. Assuming that the entire grammar consists of an output-driven map provides particularly strong structure for a learner to capitalize on. Output-drivenness is defined in section 3.3.3 as an implication from candidates of lesser internal similarity to candidates of greater internal similarity: if the lesser internal similarity candidate is well-formed, then the greater internal similarity candidate must also be well-formed. However, the same condition can be run in the contrapositive direction: if a greater internal similarity candidate is not well-

[30] More generally, one could imagine grammatical systems in which at least some well-formed outputs have no inputs that form candidates with zero disparities; in such a case, a basin would be anchored on a candidate with a minimal number of disparities.

formed, then a lesser internal similarity candidate is not well-formed either. Just as an input that **does** map to an output requires that every input closer to the output **must** map to that output, an input that **does not** map to an output requires that every input strictly farther away from the output **must not** map to that output. This property can greatly limit the search of the input space for possible underlying forms, if a learner can presume that an individual phonological map must be output-driven. If a learner is testing possible underlying forms for a morpheme in a given context, and determines that a given underlying form is not a possible one, then the learner needn't bother with evaluating any underlying forms that are strictly further away (would form lesser internal similarity candidates) relative to the observed form.

If phonological underlying forms are to be efficiently learnable (and human phonological performance suggests that they are), learners must be able to capitalize on structured relationships among mappings within possible phonological maps. These investigations into output-driven maps are a step in the process of uncovering such structured relationships.

9 Conclusion

Defining properties of phonological maps themselves makes it possible to study such maps independently of any particular characterization in terms of processes. While still requiring representational commitments, the properties stand apart from any particular theory relating input representations to output representations, providing independent landmarks for evaluating both data and theoretical proposals. This paper has proposed one such property, that of being output-driven, and argued that it captures familiar intuitions about output-orientedness in phonological maps. It is likely that other properties of maps themselves wait to be discovered, perhaps defining less restrictive classes of maps, that are relevant for phonological theory.

The analysis given in this paper derives, from the definition of Optimality Theory itself, sufficient conditions for an Optimality Theoretic system to define only output-driven maps. The key conditions are that GEN must be correspondence uniform and that all of the constraints must be output-driven-preserving. This also determines necessary conditions for getting non-output-driven behavior from OT systems. If GEN is correspondence uniform, then a system must include at least one non-ODP constraint in order to define a non-output-driven map. The definition of non-ODP constraint behavior unifies our understanding of a number of different proposals addressing phenomena that have been analyzed in terms of process opacity, including proposals as disparate as constraint conjunction and sympathy theory.

A better understanding of the properties of phonological maps, and their implications for specific theories, can only be of benefit to an evaluation of the relative strengths of competing theories. Such understanding will prove indispensible to theories of language learning.

10 Appendix: Proofs for Output-driven Preserving Constraints

This section examines certain input-referring constraints, and proves that they are ODP, as defined in section 5.2.2.

10.1 Analysis of Relationships Between Disparities

Let *akx* be a candidate with input in_a and output out_x, let *bmx* be a candidate with input in_b and output out_x, and let R_{II} be a correspondence relation between in_a and in_b such that *bmx* has greater internal similarity than *akx*. Let *bpy* be a candidate with input in_b and output out_y. For a constraint to be ODP, it must be the case that for any such *akx*, *bmx*, R_{II}, and *bpy*, the corresponding candidate *aoy* satisfies the conditions (69) and (70) with respect to the constraint. The candidate *aoy* was defined in section 5.2.1 with the IO correspondence relation R_o given in (54), repeated here.

(54) $s_a R_o s_y$ iff $\exists s_b\ [s_a R_{II} s_b$ and $s_b R_p s_y]$

The conditions (69) and (70) that *aoy* must satisfy in order for a constraint to be ODP are repeated here.

(69) C(*bpy*) < C(*bmx*) entails C(*aoy*) < C(*akx*)

(70) C(*bpy*) = C(*bmx*) entails C(*aoy*) ≤ C(*akx*)

The correspondence between the disparities of *akx* and *bmx* defined in (29), repeated below, is based upon the input-input correspondence R_{II} supporting the claim that *bmx* has greater internal similarity than *akx*, and the fact that the two candidates share the same output. The same definition can be used to construct a correspondence between the disparities of *aoy* and *bpy*, using the same input-input correspondence R_{II}.

(29) Constructing a correspondence between the disparities of *akx* and *bmx*

- Let s_b:_ be a deletion disparity in *bmx*. This disparity has a corresponding disparity s_a:_ in *akx* if and only if s_b has input-input correspondent s_a in *akx* (and thus s_a necessarily has no output correspondent in *akx*, by the conditions on input-input correspondence).
- Let _:s_x be an insertion disparity in *bmx*. This disparity has a corresponding disparity _:s_x in *akx* if and only if s_x has no input correspondent in *akx*.
- Let s_x be an output segment of *bmx* with an input correspondent s_b such that s_b and s_x differ on the value of feature F. This disparity in *bmx* has a corresponding disparity in *akx* if and only if s_x has an input correspondent s_a in *akx* such that s_a and s_x differ on the value of feature F.

The definition of analogous disparities between *akx* and *aoy*, given in (61), is repeated here. The candidates being related in this case share the same input form (in_a). The shared input is the basis for the relation between deletion disparities in *akx* and *aoy*, and between feature identity disparities in *akx* and *aoy*. The relation between insertion disparities in *akx* and *aoy* is based on the treatment of the corresponding output segments in *bmx* and *bpy*.

(61) Analogous disparities between *aoy* and *akx*

- Let s_a:_ be a deletion disparity of *aoy*. This disparity in *aoy* has an analogous disparity in *akx* if and only if s_a has no output correspondent in *akx*.
- Let _:s_y be an insertion disparity in *aoy*. This disparity has an analogous disparity _:s_x in *akx* if and only if s_y has an input correspondent s_b in *bpy*, and s_b has output correspondent s_x in *bmx*.
- Let $s_a(\alpha F){:}s_y(\beta F)$ be a feature identity disparity of *aoy* ($\alpha \neq \beta$). This disparity in *aoy* has an analogous disparity in *akx* if and only if s_a has an output correspondent s_x in *akx* such that s_a and s_x differ on the value of feature F.

The motivation for the definition of *aoy* lies in the relations between the disparities of the four candidates *akx*, *bmx*, *aoy*, and *bpy*. It is shown that every disparity of *aoy* either has a corresponding disparity in *bpy* or has an analogous disparity in *akx*. It is further shown that every disparity of *aoy* that lacks a corresponding disparity in *bpy* has an analogous disparity in *akx* that has no corresponding disparity in *bmx*.

A complete analysis is given here for the possible disparities of candidates *akx*, *bmx*, *aoy*, and *bpy*. Various results of this analysis will be used in the constraint-specific proofs that follow.

10.1.1 Deletion Disparities

Input segments with no input-input correspondents

Segments of in_a with no input-input correspondent must have no output correspondent in *akx*. By the definition of R_o, segments of in_a with no input-input correspondent must have no output correspondent in *aoy*. By the definition of correspondence between disparities (29), a deletion disparity involving a segment with no input-input correspondent cannot have a corresponding disparity.

(80) Each deletion disparity s_a:_ in *akx*, where s_a has no input-input correspondent, has an identical analogous disparity s_a:_ in *aoy*, and no corresponding disparity in *bmx*.

(81) Each deletion disparity s_a:_ in *aoy*, where s_a has no input-input correspondent, has an identical analogous disparity s_a:_ in *akx*, and no corresponding disparity in *bpy*.

Combining (80) and (81) yields the result in (82).

(82) Each deletion disparity s_a:_ in *aoy*, where s_a has no input-input correspondent, has an identical analogous disparity s_a:_ in *akx* with no corresponding disparity in *bmx*.

Each segment of in_b with no input-input correspondent must have an identical output correspondent in *bmx*, and therefore cannot be part of a deletion disparity in *bmx*.

(83) There are no deletion disparities s_b:_ in *bmx* such that s_b has no input-input correspondent.

Each deletion disparity of *bpy* for a segment s_b with no input-input correspondent has no corresponding disparity in *aoy*, because there is no input-input correspondent for s_b.

(84) Each deletion disparity s_b:_ in *bpy*, where s_b has no input-input correspondent, has no corresponding disparity in *aoy*.

Note that each such deletion disparity has no counterpart in *bmx*, because segment s_b cannot be part of a deletion disparity in *bmx*.

Input segments with input-input correspondents

A segment s_a with input-input correspondent s_b has no output correspondent in *akx* if and only if (s_b has no output correspondent in *bmx* and s_a=s_b), by the definition of greater internal similarity. Every such deletion disparity s_a:_ in *akx* has a corresponding disparity s_b:_ in *bmx*, and vice-versa. Because s_a=s_b, each corresponding pair of disparities is identical.

(85) Each deletion disparity s_a:_ of *akx*, where s_a has input-input correspondent s_b, has an identical corresponding disparity s_b:_ in *bmx*.

(86) Each deletion disparity s_b:_ of *bmx*, where s_b has input-input correspondent s_a, has an identical corresponding disparity s_a:_ in *akx*.

A segment s_a with input-input correspondent s_b has no output correspondent in *aoy* if and only if s_b has no output correspondent in *bpy*, by the definition of R_o. Every such deletion disparity s_a:_ in *aoy* has a corresponding disparity s_b:_ in *bpy*, and vice-versa. However, the corresponding disparities will be non-identical when $s_a \neq s_b$. Note that when s_a and s_b are input-input correspondents, $s_a \neq s_b$ only when both s_a and s_b have an output correspondent s_x in *akx* and *bmx*, respectively.

(87) Each deletion disparity s_a:_ of *aoy*, where s_a has input-input correspondent s_b, and s_a=s_b, has an identical corresponding disparity s_b:_ in *bpy*.

(88) Each deletion disparity s_a:_ of *aoy*, where s_a has input-input correspondent s_b, and $s_a \neq s_b$, has a **non-identical** corresponding disparity s_b:_ in *bpy*.

(89) Each deletion disparity s_b:_ of *bpy*, where s_b has input-input correspondent s_a, and s_a=s_b, has an identical corresponding disparity s_a:_ in *aoy*.

(90) Each deletion disparity s_b:_ of *bpy*, where s_b has input-input correspondent s_a, and $s_a \neq s_b$, has a **non-identical** corresponding disparity s_a:_ in *aoy*.

Each deletion disparity of *aoy* either has a corresponding disparity in *bpy* or an analogous disparity in *akx*. There is the possibility of overlap. Given an input segment s_a with input-input correspondent s_b such that s_a:_ is a disparity of both *akx* and *aoy*, s_b:_ must be a disparity of both *bmx* and *bpy*. In that case, s_a:_ in *aoy* has both a corresponding disparity in *bpy* and an analogous disparity in *akx*.

10.1.2 Insertion Disparities

Insertion disparities in candidates *bmx* and *bpy*

By the definition of greater internal similarity, every disparity of *bmx* has an identical corresponding disparity in *akx*. Therefore, each segment s_x of out_x that lacks an input correspondent in *bmx* has a corresponding disparity in *akx*. The corresponding disparities are identical, each involving the same output segment s_x.

(91) Each insertion disparity _:s_x of *bmx* has an identical corresponding disparity in *akx*.

By the definition of *aoy*, any segment s_y of out_y that lacks an input correspondent in *bpy* must also lack an input correspondent in *aoy*. Therefore, each insertion disparity in *bpy* has a corresponding disparity in *aoy*. The corresponding disparities are identical, each involving the same output segment s_y.

(92) Each insertion disparity _:s_y of *bpy* has an identical corresponding disparity in *aoy*.

Insertion disparities in candidates *akx* and *aoy*

Because every insertion disparity in *bmx* has a corresponding disparity in *akx*, it remains to account for any insertion disparities in *akx* with no corresponding disparity in *bmx*. For each disparity _:s_x of *akx* with no corresponding disparity in *bmx*, it must be the case that s_x has an identical input correspondent s_b in *bmx*, such that s_b has no input-input correspondent. If s_b has no output correspondent in *bpy*, then the disparity _:s_x in *akx* has no analog in *aoy*. If s_b has an output correspondent s_y in *bpy*, then s_y has no input correspondent in *aoy*, because s_b has no input-input correspondent. Therefore, the disparity $\_:s_y$ in *aoy* will be analogous to the disparity $\_:s_x$ in *akx*.

Because every insertion disparity in *bpy* has a corresponding disparity in *aoy*, it remains to account for any insertion disparities in *aoy* with no corresponding disparity in *bpy*. For each disparity $\_:s_y$ of *aoy* with no corresponding disparity in *bpy*, it must be the case that s_y has an input correspondent s_b in *bpy*. By the definition of R_o, s_b must not have an input-input correspondent. Therefore, s_b must have an identical output correspondent s_x in *bmx*, and s_x must have no input correspondent in *akx*. The disparity $\_:s_y$ in *aoy* is therefore analogous to the disparity $\_:s_x$ in *akx*. Because s_x has input correspondent s_b in *bmx*, $\_:s_x$ in *akx* has no corresponding disparity in *bmx*.

The analogous disparities $\_:s_x$ in *akx* and $\_:s_y$ in *aoy* are possibly not identical, as there is nothing to require that s_x=s_y.

(93) Each insertion disparity $\_:s_x$ of *akx*, where $\_:s_x$ has no corresponding disparity in *bmx*, s_x has input correspondent s_b in *bmx*, and s_b has no output correspondent in *bpy*, has no analogous disparity in *aoy*.

(94) Each insertion disparity $\_:s_y$ of *aoy*, where $\_:s_y$ has no corresponding disparity in *bpy*, s_y has input correspondent s_b in *bpy*, s_b has output correspondent s_x in *bmx*, and s_x=s_y, has an **identical** analogous disparity $\_:s_x$ in *akx*, where $\_:s_x$ in *akx* lacks a corresponding disparity in *bmx*.

(95) Each insertion disparity $\_:s_y$ of *aoy*, where $\_:s_y$ has no corresponding disparity in *bpy*, s_y has input correspondent s_b in *bpy*, s_b has output correspondent s_x in *bmx*, and $s_x \neq s_y$, has a **non-identical** analogous disparity $\_:s_x$ in *akx*, where $\_:s_x$ in *akx* lacks a corresponding disparity in *bmx*.

Each insertion disparity of *aoy* either has a corresponding disparity in *bpy* or an analogous disparity in *akx*.

10.1.3 Feature Identity Disparities

In order for a pair of identity disparities to correspond, the output segments of the disparities must be the same segment, and thus the input segments of the disparities must be input-input correspondents. In order for a pair of identity disparities to be analogous, the input segments of the disparities must be the same segment (we don't concern ourselves with output-output correspondence for *akx* ~ *aoy*).

Input segments without input-input correspondents

If a segment s_a of in_a has no input-input correspondent, then it has no output correspondent in *akx*, and thus cannot participate in an identity disparity in *akx*. By the definition of *aoy*, if s_a has no input-input correspondent then it has no output correspondent in *aoy*, and thus cannot participate in an identity disparity in *aoy*.

(96) No feature identity disparities of *akx* involve an input segment s_a without an input-input correspondent.

(97) No feature identity disparities of *aoy* involve an input segment s_a without an input-input correspondent.

If a segment s_b of in_b has no input-input correspondent, then it must have an identical output correspondent in *bmx*, and thus cannot participate in an identity disparity in *bmx*. If s_b has a non-identical output correspondent in *bpy*, the resulting identity disparities have no correspondents in *aoy* (s_b has no input-input correspondent).

(98) No feature identity disparities of *bmx* involve an input segment s_b without an input-input corresponent.

(99) Each feature identity disparity $s_b(\alpha)$:$s_y(\delta)$ in *bpy*, where s_b has no input-input correspondent, has no corresponding disparity in *aoy*.

Note that each such identity disparity in *bpy* has no correspondent in *bmx* (s_b is identical to its output correspondent in *bmx*).

Input segments with input-input correspondents

If s_a and s_b are input-input correspondents, they must have the same output correspondent s_x in *akx* and *bmx*, respectively, in order to possibly have any identity disparities in either of those candidates. Similarly, they must have the same output correspondent s_y in *aoy* and *bpy*, respectively, in order to possibly have any identity disparities in either of those candidates. There are three kinds of situations of interest:

- s_b has an output correspondent in *bmx*, but not in *bpy*.
- s_b has an output correspondent in *bpy*, but not in *bmx*.
- s_b has an output correspondent in both *bmx* and *bpy*.

Each of the three will be considered in turn.

s_b has an output correspondent in *bmx*, but not in *bpy*.

In this case, the possible values for feature F that could be assigned to corresponding segments s_a, s_b, and s_x are considered. What matters for purposes of distinguishing disparities is whether or not two values for feature F are the same. In the tables that follow, distinct greek letters signify necessarily distinct values for feature F.

Table 12 Feature identity disparities when s_b has an output correspondent in *bmx*, but not in *bpy*

F(s_a)	F(s_b)	F(s_x)	*akx*	*bmx*	Observations
α	α	α	α→α	α→α	no disparities
α	α	γ	α→γ	α→γ	corresponding disparities *akx*:*bmx*
α	β	α	α→α	β→α	NOT POSSIBLE, by def. of greater internal similarity
α	β	β	α→β	β→β	*akx* disparity has no correspondent in *bmx*
α	β	γ	α→γ	β→γ	NOT POSSIBLE, by def. of greater internal similarity

The logical possibilities are listed in Table 12. Recall that, in the present case, s_a and s_b are input-input correspondents, and they have output correspondent s_x in *akx* and *bmx*, respectively. Two of the rows are shaded; these rows represent cases that do not satisfy the requirement that *bmx* have greater internal similarity than *akx*. In the first shaded row, *bmx* has disparity $s_b(\beta)$:$s_x(\alpha)$ on feature F, while *akx* has no corresponding disparity. In the second shaded row, both *akx* and *bmx* have disparities on feature F for the corresponding segments, but the disparities are not identical, and therefore cannot be corresponding disparities, due to different values for feature F in s_a and s_b: *akx* has disparity $s_a(\alpha)$:$s_x(\gamma)$, while *bmx* has disparity $s_b(\beta)$:$s_x(\gamma)$.

In the first row of Table 12, there are no disparities: both s_a and s_b have the same value for feature F as their output correspondent s_x. In the second row, s_a and s_b have the same value for feature F, and *akx* and *bmx* have (identical) corresponding feature identity disparities.

In the fourth row, *akx* has a disparity, $s_a(\alpha){:}s_x(\beta)$, with no correspondent in *bmx*. In the present case, s_b has an output correspondent in *bmx*, but not in *bpy*. Because s_b lacks an output correspondent in *bpy*, by the definition of *aoy*, s_a has no output correspondent in *aoy*. Therefore, the disparity $s_a(\alpha){:}s_x(\beta)$ in *akx* has no analogous feature identity disparity in *aoy*.

(100) Each feature identity disparity $s_a(\alpha){:}s_x(\beta)$ in *akx* without a corresponding disparity in *bmx*, where s_a has input-input correspondent s_b and s_b has no output correspondent in *bpy*, has no analogous disparity in *aoy*.

s_b has an output correspondent in *bpy*, but not in *bmx*.

In this case, the possible values for feature F that could be assigned to corresponding segments s_a, s_b, and s_y are considered. Since s_a and s_b are input-input correspondents, and s_b has no output correspondent in *bmx*, it must be the case (by the definition of greater internal similarity) that s_a has no output correspondent in *akx*, and further that s_a=s_b. Thus, none of the combinations in which s_a and s_b have differing values for F are possible; these are the three shaded rows at the bottom of Table 13.

Table 13 Feature identity disparities when s_b has an output correspondent in *bpy*, but not in *bmx*

F(s_a)	F(s_b)	F(s_y)	*aoy*	*bpy*	Observations
α	α	α	α→α	α→α	no disparities
α	α	δ	α→δ	α→δ	identical corresponding disparities *aoy*:*bpy*
α	β	α	α→α	β→α	NOT POSSIBLE, by def. of greater internal similarity
α	β	β	α→β	β→β	NOT POSSIBLE, by def. of greater internal similarity
α	β	δ	α→δ	β→δ	NOT POSSIBLE, by def. of greater internal similarity

In the first row of Table 13, there are no disparities: both s_a and s_b have the same value for feature F as their output correspondent s_y. In the second row, s_a and s_b have the same value for feature F, and *aoy* and *bpy* have identical corresponding feature identity disparities. Because s_a has no output correspondent in *akx*, the disparity $s_a(\alpha){:}s_y(\delta)$ in *aoy* has no analogous disparity in *akx*.

(101) Each feature identity disparity $s_a(\alpha){:}s_y(\delta)$ in *aoy*, where s_a has input-input correspondent s_b and s_b has no output correspondent in *bmx*, has an identical corresponding disparity in *bpy*, and has no analogous disparity in *akx*.

s_b has an output correspondent in both *bmx* and *bpy*.

In this case, the possible values for feature F that could be assigned to segments s_a, s_b, s_x, and s_y are considered, where s_a and s_b are input-input correspondents with output correspondents s_x in both *akx* and *bmx*, and s_b has output correspondent s_y in *bpy*. It follows from the definition of *aoy* that s_a has output correspondent s_y in *aoy*.

Table 14 Feature identity disparities when s_b has an output correspondent in both *bmx* and *bpy*

F(s_a)	F(s_b)	F(s_x)	F(s_y)	*akx*	*bmx*	*bpy*	*aoy*	Observations
α	α	α	α	α→α	α→α	α→α	α→α	no disparities
α	α	α	δ	α→α	α→α	α→δ	α→δ	identical corresp. disparities *aoy*:*bpy*
α	α	γ	α	α→γ	α→γ	α→α	α→α	identical corresp. disparities *akx*:*bmx*
α	α	γ	γ	α→γ	α→γ	α→γ	α→γ	identical corresp. disparities *akx*:*bmx*, *aoy*:*bpy*
α	α	γ	δ	α→γ	α→γ	α→δ	α→δ	identical corresp. disparities *akx*:*bmx*, *aoy*:*bpy*
α	β	β	α	α→β	β→β	β→α	α→α	*akx* has no corresp. disparity in *bmx* *bpy* has no corresp. disparity in *aoy*
α	β	β	β	α→β	β→β	β→β	α→β	*aoy* has no corresp. disparity in *bpy* identical analogous disparities *aoy*:*akx* *akx* has no corresp. disparity in *bmx*
α	β	β	δ	α→β	β→β	β→δ	α→δ	**non-identical** corresp. disparity *aoy*:*bpy* **non-identical** analogous disparity *aoy*:*akx* *akx* has no corresp. disparity in *bmx*
α	β	γ	*	α→γ	β→γ	*	*	NOT POSSIBLE by def. of greater internal similarity

In the last row of Table 14, which is shaded, both *akx* and *bmx* have disparities on feature F for the corresponding segments, but the disparities are not identical, and therefore cannot be corresponding disparities, due to different values for feature F in s_a and s_b: *akx* has disparity s_a(α):s_x(γ), while *bmx* has disparity s_b(β):s_x(γ). In that row, the value F(s_y) is immaterial.

In the first row of Table 14, there are no disparities: all four relevant segments have the same value for feature F. In rows 2-5, disparities have identical corresponding disparities, between *akx* and *bmx*, between *aoy* and *bpy*, or both. In row 4, the disparity s_a(α):s_y(γ) of *aoy* has an identical corresponding disparity in *bpy*, and also an identical analogous disparity in *akx*; the analogous disparity in *akx* furthermore has an identical corresponding disparity in *bmx*. In row 5, the disparity s_a(α):s_y(δ) of *aoy* has an identical corresponding disparity in *bpy*, and also a **non-identical** analogous disparity s_a(α):s_x(γ) in *akx*; the disparity s_a(α):s_x(γ) in *akx* furthermore has an identical corresponding disparity in *bmx*.

In row 6, *aoy* and *bmx* have no disparity on the relevant segments. *akx* has a feature identity disparity, which thus has no corresponding disparity in *bmx* and no analogous disparity in *aoy*. *bpy* also has a feature disparity, with no corresponding disparity in *aoy*.

In row 7, *aoy* has a feature identity disparity with no corresponding disparity in *bpy*. That disparity has an identical analogous disparity in *akx*; the analogous disparity in *akx* further has no corresponding disparity in *bmx*.

In row 8, the feature identity disparity s_a(α):s_y(δ) of *aoy* has a **non-identical** corresponding disparity s_b(β):s_y(δ) in *bpy*; the disparities have different input segment values for F (the output segment is shared). That disparity of *aoy* also has a **non-identical** analogous disparity s_a(α):s_x(β) in *akx*; the disparities have different output segment values for F (the input segment is shared). The disparity $s_a(\alpha){:}s_x(\beta)$ in *akx* has no corresponding disparity in *bmx*.

From (97), (101), and Table 14, we can infer that the only feature identity disparities of *aoy* that have no corresponding disparities in *bpy* are those described in row 7 of Table 14. It follows that all feature identity disparities of *aoy* with no corresponding disparities in *bpy* have identical analogous disparities in *akx* with no corresponding disparity in *bmx*.

(102) Each feature identity disparity $s_a(\alpha){:}s_y(\beta)$ in *aoy* without a corresponding disparity in *bpy* has an identical analogous disparity in *akx* with no corresponding disparity in *bmx*.

From (97), (101), and Table 14, we can infer that the only feature identity disparities of *aoy* that have non-identical corresponding disparities in *bpy* are those described in row 8 of Table 14. Each such feature identity disparity $s_a(\alpha){:}s_y(\delta)$ of *aoy* has a non-identical corresponding disparity $s_b(\beta){:}s_y(\delta)$ in *bpy*, and also a non-identical analogous disparity $s_a(\alpha){:}s_x(\beta)$ in *akx* with no corresponding disparity in *bmx*.

(103) Each feature identity disparity $s_a(\alpha){:}s_y(\delta)$ in *aoy* with a non-identical corresponding disparity $s_b(\beta){:}s_y(\delta)$ in *bpy* has a non-identical analogous disparity $s_a(\alpha){:}s_x(\beta)$ in *akx* with no corresponding disparity in *bmx*.

10.1.4 Comments/Discussion

In a sense, the insertion disparities of *akx* with no corresponding/analogous disparities "go with" the deletion disparities of *bpy* with no corresponding disparities. Both involve input segment s_b with output correspondent s_x in *bmx*, where s_b has no input-input correspondent, and s_b has no output correspondent in *bpy*.

10.2 MAX

MAX-IO (hereafter MAX) evaluates the input-output correspondence relation of a candidate, and assesses a violation for every segment of the input that has no output correspondent (McCarthy and Prince 1999). In other words, MAX assesses a one violation for each deletion disparity in a candidate. MAX(*akx*) denotes the number of violations of MAX incurred by *akx*, and therefore is also the number of segments of the input that lack an output correspondent in the candidate.

10.2.1 Partition of the input segments

For each candidate, each deletion disparity of that candidate either does or does not have a corresponding deletion disparity in the other candidate with the same output (*akx* with *bmx*, *aoy* with *bpy*). Therefore, for each candidate, the total set of MAX violations can be partitioned into those assessed to disparities with corresponding disparities and those assessed to disparities lacking corresponding disparities.

(104) MAX(*akx*) = MAX(*akx* : corr) + MAX(*akx* : no-corr)

(105) MAX(*bmx*) = MAX(*bmx* : corr) + MAX(*bmx* : no-corr)

(106) MAX(*aoy*) = MAX(*aoy* : corr) + MAX(*aoy* : no-corr)

(107) MAX(*bpy*) = MAX(*bpy* : corr) + MAX(*bpy* : no-corr)

10.2.2 Corresponding Deletion Disparities for *aoy* and *bpy*

By (107),

MAX(*bpy*) = MAX(*bpy* : corr) + MAX(*bpy* : no-corr)

Violation counts must be non-negative, so 0 <= MAX(*bpy* : no-corr). Therefore, the total number of violations of MAX assessed to *bpy* must be at least the number of violations assessed to deletion disparities without corresponding disparities in *aoy*.

MAX(*bpy* : corr) ≤ MAX(*bpy*)

By (82), only deletion disparities in *aoy* that involve input segments with input-input correspondents can have corresponding disparities in *bpy*. By (87) and (88), every deletion disparity in *aoy* that involves an input segment with an input-input correspondent has a corresponding disparity in *bpy*. Although some of the deletion disparities in *aoy* may be non-identical to their correspondents in *bpy*, specifically the ones described in (88), MAX assesses a separate violation to each deletion disparity regardless of segment identity. The number of violations of MAX assessed to deletion disparities in *aoy* with corresponding disparities in *bpy* is equal the number of violations of MAX assessed to deletion disparities in *bpy* with corresponding disparities in *aoy*.

MAX(*aoy* : corr) = MAX(*bpy* : corr)

Substituting into the previous result, we reach the conclusion that

(108) MAX(*aoy* : corr) ≤ MAX(*bpy*)

10.2.3 Corresponding Deletion Disparities for *akx* and *bmx*

By hypothesis, *bmx* has greater internal similarity than *akx*. Therefore, every disparity of *bmx* has an identical corresponding disparity in *akx*.

(109) MAX(*akx* : corr) = MAX(*bmx*)

10.2.4 Non-Corresponding Deletion Disparities for *aoy* and *akx*

From (87) and (88), we may conclude that deletion disparities of *aoy* lack corresponding disparities in *bpy* only when they involve input segments that lack input-input correspondents. From (82) we may conclude that all such deletion disparities in *aoy* have identical analogous disparities in *akx* that lack corresponding disparities in *bmx*.

Although some of the deletion disparities in *aoy* may be non-identical to their analogs in *akx*, MAX assesses a separate violation to each deletion disparity, so the number of violations of MAX assessed to deletion disparities in *aoy* lacking corresponding disparities in *bpy* is at most the number of violations of MAX assessed to deletion disparities in *akx* lacking corresponding disparities in *bmx*.

(110) MAX(*aoy* : no-corr) ≤ MAX(*akx* : no-corr)

10.2.5 MAX(*bpy*) < MAX(*bmx*) entails MAX(*aoy*) < MAX(*akx*)

By (106),

MAX(*aoy*) = MAX(*aoy* : corr) + MAX(*aoy* : no-corr)

By (108), MAX(*aoy* : corr) ≤ MAX(*bpy*).

MAX(*aoy*) ≤ MAX(*bpy*) + MAX(*aoy* : no-corr)

By hypothesis, MAX(*bpy*) < MAX(*bmx*).

MAX(*aoy*) < MAX(*bmx*) + MAX(*aoy* : no-corr)

By (109), MAX(*akx* : corr) = MAX(*bmx*).

MAX(*aoy*) < MAX(*akx* : corr) + MAX(*aoy* : no-corr)

By (110), MAX(*aoy* : no-corr) ≤ MAX(*akx* : no-corr).

$$\text{MAX}(aoy) < \text{MAX}(akx : \text{corr}) + \text{MAX}(akx : \text{no-corr})$$

By (104), MAX(*akx*) = MAX(*akx* : corr) + MAX(*akx* : no-corr).

$$\text{MAX}(aoy) < \text{MAX}(akx)$$

End of Proof

10.2.6 MAX(*bpy*) = MAX(*bmx*) entails MAX(*aoy*) ≤ MAX(*akx*)

By (106),

$$\text{MAX}(aoy) = \text{MAX}(aoy : \text{corr}) + \text{MAX}(aoy : \text{no-corr})$$

By (108), MAX(*aoy* : corr) ≤ MAX(*bpy*).

$$\text{MAX}(aoy) \leq \text{MAX}(bpy) + \text{MAX}(aoy : \text{no-corr})$$

By hypothesis, MAX(*bpy*) = MAX(*bmx*).

$$\text{MAX}(aoy) \leq \text{MAX}(bmx) + \text{MAX}(aoy : \text{no-corr})$$

By (109), MAX(*akx* : corr) = MAX(*bmx*).

$$\text{MAX}(aoy) \leq \text{MAX}(akx : \text{corr}) + \text{MAX}(aoy : \text{no-corr})$$

By (110), MAX(*aoy* : no-corr) ≤ MAX(*akx* : no-corr).

$$\text{MAX}(aoy) \leq \text{MAX}(akx : \text{corr}) + \text{MAX}(akx : \text{no-corr})$$

By (104), MAX(*akx*) = MAX(*akx* : corr) + MAX(*akx* : no-corr).

$$\text{MAX}(aoy) \leq \text{MAX}(akx)$$

End of Proof

10.3 DEP

DEP-IO (hereafter DEP) evaluates the input-output correspondence relation of a candidate, and assesses a violation for every element of the output that has no input correspondent (McCarthy and Prince 1999). In other words, DEP assesses one violation for each insertion disparity in a candidate. DEP(*akx*) denotes the number of violations of DEP incurred by *akx*, and therefore is also the number of segments of the output that lack an input correspondent in the candidate.

10.3.1 Partition of Disparities

For each candidate, each insertion disparity of that candidate either does or does not have a corresponding insertion disparity in the other candidate with the same output (*akx* with *bmx*, *aoy* with *bpy*). Therefore, for each candidate, the total set of DEP violations can be partitioned into those assessed to disparities with corresponding disparities and those assessed to disparities lacking corresponding disparities.

(111) DEP(*akx*) = DEP(*akx* : corr) + DEP(*akx* : no-corr)

(112) DEP(*bmx*) = DEP(*bmx* : corr) + DEP(*bmx* : no-corr)

(113) DEP(*aoy*) = DEP(*aoy* : corr) + DEP(*aoy* : no-corr)

(114) DEP(*bpy*) = DEP(*bpy* : corr) + DEP(*bpy* : no-corr)

10.3.2 Corresponding Insertion Disparities for *aoy* and *bpy*

By (114),

DEP(*bpy*) = DEP(*bpy* : corr) + DEP(*bpy* : no-corr)

Violation counts must be non-negative, so 0 <= DEP(*bpy* : no-corr). Therefore, the total number of violations of DEP assessed to *bpy* must be at least the number of violations assessed to insertion disparities without corresponding disparities in *aoy*.

DEP(*bpy* : corr) ≤ DEP(*bpy*)

By (92), every insertion disparity in *aoy* with a corresponding disparity in *bpy* is identical to its corresponding disparity. Every such disparity is a single separate violation of DEP.

DEP(*aoy* : corr) = DEP(*bpy* : corr)

Substituting into the previous result, we reach the conclusion that

(115) DEP(*aoy* : corr) ≤ DEP(*bpy*)

10.3.3 Corresponding Insertion Disparities for *akx* and *bmx*

By (91), every insertion disparity of *bmx* has an identical corresponding disparity in *akx*. Every such insertion disparity is a single separate violation of DEP.

(116) DEP(*akx* : corr) = DEP(*bmx*)

10.3.4 Non-corresponding Insertion Disparities for *aoy* and *akx*

From (94) and (95), we may conclude that every insertion disparity in *aoy* without a corresponding disparity in *bpy* has an analogous disparity in *akx* that has no corresponding disparity in *bmx*. Although some of the insertion disparities may be non-identical to their analogs, DEP assesses a violation to every insertion disparity, so the number of violations of DEP assessed to insertion disparities in *aoy* lacking corresponding disparities in *bpy* is at most the number of violations of DEP assessed to insertion disparities in *akx* lacking corresponding disparities in *bmx*.

(117) DEP(*aoy* : no-corr) ≤ DEP(*akx* : no-corr)

10.3.5 Proof: DEP(*bpy*) < DEP(*bmx*) entails DEP(*aoy*) < DEP(*akx*)

By (113),

DEP(*aoy*) = DEP(*aoy* : corr) + DEP(*aoy* : no-corr)

By (115), DEP(*aoy* : corr) ≤ DEP(*bpy*).

DEP(*aoy*) ≤ DEP(*bpy*) + DEP(*aoy* : no-corr)

By hypothesis, DEP(*bpy*) < DEP(*bmx*).

DEP(*aoy*) < DEP(*bmx*) + DEP(*aoy* : no-corr)

By (116), DEP(*akx* : corr) = DEP(*bmx*).

DEP(*aoy*) < DEP(*akx* : corr) + DEP(*aoy* : no-corr)

By (117), DEP(*aoy* : no-corr) ≤ DEP(*akx* : no-corr).

DEP(*aoy*) < DEP(*akx* : corr) + DEP(*akx* : no-corr)

By (111), DEP(*akx*) = DEP(*akx* : corr) + DEP(*akx* : no-corr).

DEP(*aoy*) < DEP(*akx*)

End of Proof

10.3.6 Proof: DEP(*bpy*) = DEP(*bmx*) entails DEP(*aoy*) ≤ DEP(*akx*)

By (113),

DEP(*aoy*) = DEP(*aoy* : corr) + DEP(*aoy* : no-corr)

By (115), DEP(*aoy* : corr) ≤ DEP(*bpy*).

DEP(*aoy*) ≤ DEP(*bpy*) + DEP(*aoy* : no-corr)

By hypothesis, DEP(*bpy*) = DEP(*bmx*).

DEP(*aoy*) ≤ DEP(*bmx*) + DEP(*aoy* : no-corr)

By (116), DEP(*akx* : corr) = DEP(*bmx*).

DEP(*aoy*) ≤ DEP(*akx* : corr) + DEP(*aoy* : no-corr)

By (117), DEP(*aoy* : no-corr) ≤ DEP(*akx* : no-corr).

DEP(*aoy*) ≤ DEP(*akx* : corr) + DEP(*akx* : no-corr)

By (111), DEP(*akx*) = DEP(*akx* : corr) + DEP(*akx* : no-corr).

DEP(*aoy*) ≤ DEP(*akx*)

End of Proof

10.4 IDENT[F_{in}∈V]

IDENT-IO[F] (hereafter IDENT[F]) evaluates the input-output correspondence relation of a candidate, and assesses a violation for every corresponding pair of segments that do not have identical values of the feature F (McCarthy and Prince 1999).

IDENT-IO[F_{in}∈V] (hereafter IDENT[F_{in}∈V]) evaluates the input-output correspondence relation of a candidate, and assesses a violation for every corresponding pair of segments for which the input segment has a value v for feature F that is a member if the set V, but the output correspondent does not have the value v for feature F. This constraint is like IDENT[F] but is restricted to evaluate only corresponding segment pairs where the input segment has a specific value of the feature being evaluated for identity.

Pater (1999) first proposed constraints that were like IDENT, but restricted to correspondences in which the input segment had a particular value.[31] de Lacy (2002) proposed IDENT constraints restricted to correspondences in which the input segment had a value belonging to a subset of the possible values for a feature, in the context of markedness scales and scale category conflation. Here, I analyze the general class of IDENT constraints with input value restrictions. The constraints proposed by Pater are equivalent to having the restriction set of feature values V contain only a single value. If V contains all of the possible values for feature F, then IDENT[F_{in}∈V] becomes equivalent to IDENT[F]. Thus, the result proven here for IDENT[F_{in}∈V] is a generalization that includes IDENT[F].

F(s_a) represents the value of feature F in the segment s_a.

[31] Pater used the notation IDENT I→O [F] for such constraints.

10.4.1 Partition of Identity Disparities

For each candidate, each identity disparity of that candidate either does or does not have a corresponding identity disparity in the other candidate with the same output (*akx* with *bmx*, *aoy* with *bpy*). Therefore, for each candidate, the total set of IDENT[F_{in}∈V] violations can be partitioned into those assessed to disparities with corresponding disparities and those assessed to disparities lacking corresponding disparities.

A further distinction is made here with respect to candidates *aoy* and *bpy*. Because some of the disparities of *aoy* with corresponding disparities in *bpy* are possibly non-identical to their corresponding disparities, we further partition the violations assessed to *aoy* into those assessed to disparities with identical corresponding disparities in *bpy*, and those assessed to disparities with non-identical corresponding disparities in *bpy*. The disparities of *bpy* are similarly partitioned.

(118) IDENT[F_{in}∈V](*akx*) = IDENT[F_{in}∈V](*akx* : corr) + IDENT[F_{in}∈V](*akx* : no-corr)

(119) IDENT[F_{in}∈V](*bmx*) = IDENT[F_{in}∈V](*bmx* : corr) + IDENT[F_{in}∈V](*bmx* : no-corr)

(120) IDENT[F_{in}∈V](*aoy*) = IDENT[F_{in}∈V](*aoy* : id corr) + IDENT[F_{in}∈V](*aoy* : non-id corr) + IDENT[F_{in}∈V](*aoy* : no-corr)

(121) IDENT[F_{in}∈V](*bpy*) = IDENT[F_{in}∈V](*bpy* : id corr) + IDENT[F_{in}∈V](*bpy* : non-id corr) + IDENT[F_{in}∈V](*bpy* : no-corr)

10.4.2 Identical Corresponding Identity Disparities for *aoy* and *bpy*

By (121),

IDENT[F_{in}∈V](*bpy*) = IDENT[F_{in}∈V](*bpy* : id corr) + IDENT[F_{in}∈V](*bpy* : non-id corr) + IDENT[F_{in}∈V](*bpy* : no-corr)

Violation counts must be non-negative, so

0 ≤ IDENT[F_{in}∈V](*bpy* : no-corr)

0 ≤ IDENT[F_{in}∈V](*bpy* : non-id corr)

Therefore, the total number of violations assessed to *bpy* must be at least the number of violations assessed to identity disparities with identical corresponding disparities in *aoy*.

IDENT[F_{in}∈V](*bpy* : id corr) ≤ IDENT[F_{in}∈V](*bpy*)

Each pair of identical corresponding identity disparities between *aoy* and *bpy* have the same feature values for their input segments: that value is either in V, in which case both disparities violate the constraint, or the value is not in V, in which case neither disparity violates the constraint. The number of violations assessed to disparities with identical correspondents is thus the same for *aoy* and *bpy*.

IDENT[F_{in}∈V](*aoy* : id corr) = IDENT[F_{in}∈V](*bpy* : id corr)

Substituting into the previous result yields

(122) IDENT[F_{in}∈V](*aoy* : id corr) ≤ IDENT[F_{in}∈V](*bpy*)

10.4.3 Corresponding Identity Disparities for *akx* and *bmx*

By hypothesis, *bmx* has greater internal similarity than *akx*. Therefore, every disparity of *bmx* has an identical corresponding disparity in *akx*.

(123) IDENT[F_{in}∈V](*akx* : corr) = IDENT[F_{in}∈V](*bmx*)

10.4.4 Non-Corresponding and Non-Identical Corresponding Identity Disparities for *aoy* and *akx*

By (102), each feature identity disparity $s_a(\alpha){:}s_y(\beta)$ in *aoy* without a corresponding disparity in *bpy* has an identical analogous disparity in *akx* with no corresponding disparity in *bmx*. Because the analogous disparities share the same input segment, s_a, either $F(s_a) \in V$, in which case both disparities constitute violations, or $F(s_a) \notin V$, in which case neither disparity constitutes a violation.

By (103), each feature identity disparity $s_a(\alpha){:}s_y(\delta)$ in *aoy* with a non-identical corresponding disparity $s_b(\beta){:}s_y(\delta)$ in *bpy* has a non-identical analogous disparity $s_a(\alpha){:}s_x(\beta)$ in *akx* with no corresponding disparity in *bmx*. The analogous identity disparities for F involving s_a in *aoy* and *akx* might not have the same values for F in their respective output segments. However, because these disparities share the same input segment, s_a, either $F(s_a) \in V$, in which case both disparities constitute violations, or $F(s_a) \notin V$, in which case neither disparity constitutes a violation.

Every identity disparity in *aoy* lacking an identical corresponding disparity in *bpy*, be it a disparity with no corresponding disparity in *bpy* or a disparity with a non-identical corresponding disparity in *bpy*, has an analogous disparity in *akx* that lacks a corresponding disparity in *bmx*. Each pair of such analogous disparities share the same input segment, and thus either both or neither violate IDENT[$F_{in} \in V$]. Each such analog disparity in *akx* has no corresponding disparity in *bmx*.

(124) IDENT[$F_{in} \in V$](*aoy* : no-corr) + IDENT[$F_{in} \in V$](*aoy* : non-id corr) ≤ IDENT[$F_{in} \in V$](*akx* : no-corr)

10.4.5 Proof: IDENT[$F_{in} \in V$](*bpy*) < IDENT[$F_{in} \in V$](*bmx*) entails IDENT[$F_{in} \in V$](*aoy*) < IDENT[$F_{in} \in V$](*akx*)

By (120),

IDENT[$F_{in} \in V$](*aoy*) = IDENT[$F_{in} \in V$](*aoy* : id corr) + IDENT[$F_{in} \in V$](*aoy* : non-id corr) + IDENT[$F_{in} \in V$](*aoy* : no-corr)

By (122), IDENT[$F_{in} \in V$](*aoy* : id corr) ≤ IDENT[$F_{in} \in V$](*bpy*).

IDENT[$F_{in} \in V$](*aoy*) ≤ IDENT[$F_{in} \in V$](*bpy*) + IDENT[$F_{in} \in V$](*aoy* : non-id corr) + IDENT[$F_{in} \in V$](*aoy* : no-corr)

By hypothesis, IDENT[$F_{in} \in V$](*bpy*) < IDENT[$F_{in} \in V$](*bmx*).

IDENT[$F_{in} \in V$](*aoy*) < IDENT[$F_{in} \in V$](*bmx*) + IDENT[$F_{in} \in V$](*aoy* : non-id corr) + IDENT[$F_{in} \in V$](*aoy* : no-corr)

By (123), IDENT[$F_{in} \in V$](*akx* : corr) = IDENT[$F_{in} \in V$](*bmx*).

IDENT[$F_{in} \in V$](*aoy*) < IDENT[$F_{in} \in V$](*akx* : corr) + IDENT[$F_{in} \in V$](*aoy* : non-id corr) + IDENT[$F_{in} \in V$](*aoy* : no-corr)

By (124), IDENT[$F_{in} \in V$](*aoy* : no-corr) + IDENT[$F_{in} \in V$](*aoy* : non-id corr) ≤ IDENT[$F_{in} \in V$](*akx* : no-corr).

IDENT[$F_{in} \in V$](*aoy*) < IDENT[$F_{in} \in V$](*akx* : corr) + IDENT[$F_{in} \in V$](*akx* : no-corr)

By (118), IDENT[$F_{in} \in V$](*akx*) = IDENT[$F_{in} \in V$](*akx* : corr) + IDENT[$F_{in} \in V$](*akx* : no-corr).

IDENT[$F_{in} \in V$](*aoy*) < IDENT[$F_{in} \in V$](*akx*)

End of Proof

10.4.6 Proof: IDENT[F_{in}∈V](*bpy*) = IDENT[F_{in}∈V](*bmx*) entails IDENT[F_{in}∈V](*aoy*) ≤ IDENT[F_{in}∈V](*akx*)

By (120),

$$\text{IDENT}[F_{in}\in V](aoy) = \text{IDENT}[F_{in}\in V](aoy : \text{id corr}) + \text{IDENT}[F_{in}\in V](aoy : \text{non-id corr}) + \text{IDENT}[F_{in}\in V](aoy : \text{no-corr})$$

By (122), IDENT[F_{in}∈V](*aoy* : id corr) ≤ IDENT[F_{in}∈V](*bpy*).

$$\text{IDENT}[F_{in}\in V](aoy) \le \text{IDENT}[F_{in}\in V](bpy) + \text{IDENT}[F_{in}\in V](aoy : \text{non-id corr}) + \text{IDENT}[F_{in}\in V](aoy : \text{no-corr})$$

By hypothesis, IDENT[F_{in}∈V](*bpy*) = IDENT[F_{in}∈V](*bmx*).

$$\text{IDENT}[F_{in}\in V](aoy) \le \text{IDENT}[F_{in}\in V](bmx) + \text{IDENT}[F_{in}\in V](aoy : \text{non-id corr}) + \text{IDENT}[F_{in}\in V](aoy : \text{no-corr})$$

By (123), IDENT[F_{in}∈V](*akx* : corr) = IDENT[F_{in}∈V](*bmx*).

$$\text{IDENT}[F_{in}\in V](aoy) \le \text{IDENT}[F_{in}\in V](akx : \text{corr}) + \text{IDENT}[F_{in}\in V](aoy : \text{non-id corr}) + \text{IDENT}[F_{in}\in V](aoy : \text{no-corr})$$

By (124), IDENT[F_{in}∈V](*aoy* : no-corr) + IDENT[F_{in}∈V](*aoy* : non-id corr) ≤ IDENT[F_{in}∈V](*akx* : no-corr).

$$\text{IDENT}[F_{in}\in V](aoy) \le \text{IDENT}[F_{in}\in V](akx : \text{corr}) + \text{IDENT}[F_{in}\in V](akx : \text{no-corr})$$

By (118), IDENT[F_{in}∈V](*akx*) = IDENT[F_{in}∈V](*akx* : corr) + IDENT[F_{in}∈V](*akx* : no-corr).

$$\text{IDENT}[F_{in}\in V](aoy) \le \text{IDENT}[F_{in}\in V](akx)$$

End of Proof

10.5 IDENT[F_{out}∈V]

IDENT-IO[F] (hereafter IDENT[F]) evaluates the input-output correspondence relation of a candidate, and assesses a violation for every corresponding pair of segments that do not have identical values of the feature F (McCarthy and Prince 1999).

IDENT-IO[F_{out}∈V] (hereafter IDENT[F_{out}∈V]) evaluates the input-output correspondence relation of a candidate, and assesses a violation for every corresponding pair of segments for which the output segment has a value v for feature F that is a member if the set V, but the input correspondent does not have the value v for feature F. This constraint is like IDENT[F] but is restricted to evaluate only corresponding segment pairs where the output segment has a specific value of the feature being evaluated for identity.

Pater (1999) first proposed constraints that were like IDENT, but restricted to correspondences in which the output segment had a particular value.[32] Here, I analyze the general class of IDENT constraints with output value restrictions. The constraints proposed by Pater are equivalent to having the restriction set of feature values V contain only a single value. If V contains all of the possible values for feature F, then IDENT[F_{out}∈V] becomes equivalent to IDENT[F]. Thus, the result proven here for IDENT[F_{out}∈V] is a generalization that includes IDENT[F].

F(s_a) represents the value of feature F in the segment s_a.

[32] Pater used the notation IDENT O→I [F] for such constraints.

10.5.1 Partition of Identity Disparities

For each candidate, each identity disparity of that candidate either does or does not have a corresponding identity disparity in the other candidate with the same output (*akx* with *bmx*, *aoy* with *bpy*). Therefore, for each candidate, the total set of IDENT[F_{out}∈V] violations can be partitioned into those assessed to disparities with corresponding disparities and those assessed to disparities lacking corresponding disparities.

(125) IDENT[F_{out}∈V](*akx*) = IDENT[F_{out}∈V](*akx* : corr) + IDENT[F_{out}∈V](*akx* : no-corr)

(126) IDENT[F_{out}∈V](*bmx*) = IDENT[F_{out}∈V](*bmx* : corr) + IDENT[F_{out}∈V](*bmx* : no-corr)

(127) IDENT[F_{out}∈V](*aoy*) = IDENT[F_{out}∈V](*aoy* : corr) + IDENT[F_{out}∈V](*aoy* : no-corr)

(128) IDENT[F_{out}∈V](*bpy*) = IDENT[F_{out}∈V](*bpy* : corr) + IDENT[F_{out}∈V](*bpy* : no-corr)

10.5.2 Corresponding Identity Disparities for *aoy* and *bpy*

By (128),

IDENT[F_{out}∈V](*bpy*) = IDENT[F_{out}∈V](*bpy* : corr) + IDENT[F_{out}∈V](*bpy* : no-corr)

Violation counts must be non-negative, so

0 ≤ IDENT[F_{out}∈V](*bpy* : no-corr)

Therefore, the total number of violations assessed to identity disparities in *bpy* must be at least the number of violations assessed to identity disparities with corresponding disparities in *aoy*.

IDENT[F_{out}∈V](*bpy* : corr) ≤ IDENT[F_{out}∈V](*bpy*)

Each identity disparity in *aoy* with a corresponding identity disparity in *bpy* has the same output feature value as its corresponding disparity. This is because, by the definition of disparity correspondence, the two disparities share the same output segment s_y. That output feature value is either in V, in which case both disparities violate the constraint, or the value is not in V, in which case neither disparity violates the constraint. The number of violations assessed to disparities with correspondents is thus the same for *aoy* and *bpy*.

IDENT[F_{out}∈V](*aoy* : corr) = IDENT[F_{out}∈V](*bpy* : corr)

Substituting into the previous result yields

(129) IDENT[F_{out}∈V](*aoy* : corr) ≤ IDENT[F_{out}∈V](*bpy*)

10.5.3 Corresponding Identity Disparities for *akx* and *bmx*

By hypothesis, *bmx* has greater internal similarity than *akx*. Therefore, every disparity of *bmx* has an identical corresponding disparity in *akx*.

(130) IDENT[F_{out}∈V](*akx* : corr) = IDENT[F_{out}∈V](*bmx*)

10.5.4 Non-Corresponding Identity Disparities for *aoy* and *akx*

By (102), each feature identity disparity s_a(α):s_y(β) in *aoy* without a corresponding disparity in *bpy* has an identical analogous disparity s_a(α):s_x(β) in *akx* with no corresponding disparity in *bmx*. Because the analogous disparities are identical, the value of feature F must be the same in the two output correspondents, s_y and s_x: F(s_y) = F(s_x) = β. Either β∈V, in which case both disparities constitute violations, or β∉V, in which case neither disparity constitutes a violation.

(131) IDENT[F_{out}∈V](*aoy* : no-corr) ≤ IDENT[F_{out}∈V](*akx* : no-corr)

10.5.5 Proof: $\text{IDENT}[F_{out}\in V](bpy) < \text{IDENT}[F_{out}\in V](bmx)$ entails $\text{IDENT}[F_{out}\in V](aoy) < \text{IDENT}[F_{out}\in V](akx)$

By (127),

$$\text{IDENT}[F_{out}\in V](aoy) = \text{IDENT}[F_{out}\in V](aoy : \text{corr}) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By (129), $\text{IDENT}[F_{out}\in V](aoy : \text{corr}) \le \text{IDENT}[F_{out}\in V](bpy)$.

$$\text{IDENT}[F_{out}\in V](aoy) \le \text{IDENT}[F_{out}\in V](bpy) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By hypothesis, $\text{IDENT}[F_{out}\in V](bpy) < \text{IDENT}[F_{out}\in V](bmx)$.

$$\text{IDENT}[F_{out}\in V](aoy) < \text{IDENT}[F_{out}\in V](bmx) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By (130), $\text{IDENT}[F_{out}\in V](akx : \text{corr}) = \text{IDENT}[F_{out}\in V](bmx)$.

$$\text{IDENT}[F_{out}\in V](aoy) < \text{IDENT}[F_{out}\in V](akx : \text{corr}) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By (131), $\text{IDENT}[F_{out}\in V](aoy : \text{no-corr}) \le \text{IDENT}[F_{out}\in V](akx : \text{no-corr})$.

$$\text{IDENT}[F_{out}\in V](aoy) < \text{IDENT}[F_{out}\in V](akx : \text{corr}) + \text{IDENT}[F_{out}\in V](akx : \text{no-corr})$$

By (125), $\text{IDENT}[F_{out}\in V](akx) = \text{IDENT}[F_{out}\in V](akx : \text{corr}) + \text{IDENT}[F_{out}\in V](akx : \text{no-corr})$.

$$\text{IDENT}[F_{out}\in V](aoy) < \text{IDENT}[F_{out}\in V](akx)$$

End of Proof

10.5.6 Proof: $\text{IDENT}[F_{out}\in V](bpy) = \text{IDENT}[F_{out}\in V](bmx)$ entails $\text{IDENT}[F_{out}\in V](aoy) \le \text{IDENT}[F_{out}\in V](akx)$

By (127),

$$\text{IDENT}[F_{out}\in V](aoy) = \text{IDENT}[F_{out}\in V](aoy : \text{corr}) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By (129), $\text{IDENT}[F_{out}\in V](aoy : \text{corr}) \le \text{IDENT}[F_{out}\in V](bpy)$.

$$\text{IDENT}[F_{out}\in V](aoy) \le \text{IDENT}[F_{out}\in V](bpy) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By hypothesis, $\text{IDENT}[F_{out}\in V](bpy) = \text{IDENT}[F_{out}\in V](bmx)$.

$$\text{IDENT}[F_{out}\in V](aoy) \le \text{IDENT}[F_{out}\in V](bmx) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By (130), $\text{IDENT}[F_{out}\in V](akx : \text{corr}) = \text{IDENT}[F_{out}\in V](bmx)$.

$$\text{IDENT}[F_{out}\in V](aoy) \le \text{IDENT}[F_{out}\in V](akx : \text{corr}) + \text{IDENT}[F_{out}\in V](aoy : \text{no-corr})$$

By (131), $\text{IDENT}[F_{out}\in V](aoy : \text{no-corr}) \le \text{IDENT}[F_{out}\in V](akx : \text{no-corr})$.

$$\text{IDENT}[F_{out}\in V](aoy) \le \text{IDENT}[F_{out}\in V](akx : \text{corr}) + \text{IDENT}[F_{out}\in V](akx : \text{no-corr})$$

By (125), $\text{IDENT}[F_{out}\in V](akx) = \text{IDENT}[F_{out}\in V](akx : \text{corr}) + \text{IDENT}[F_{out}\in V](akx : \text{no-corr})$.

$$\text{IDENT}[F_{out}\in V](aoy) \le \text{IDENT}[F_{out}\in V](akx)$$

End of Proof

11 References

Alderete, John. 1999a. Morphologically Governed Accent in Optimality Theory. PhD. dissertation, University of Massachusetts, Amherst.

Alderete, John. 1999b. Head dependence in stress-epenthesis interaction. In *The Derivational Residue in Phonological Optimality Theory*, ed. by Ben Hermans and Marc van Oostendorp, 29-50. Amsterdam: John Benjamins. ROA-453.

Alderete, John, and Tesar, Bruce. 2002. Learning covert phonological interaction: An analysis of the problem posed by the interaction of stress and epenthesis. Technical Report RuCCS-TR-72, Rutgers Center for Cognitive Science, Rutgers University. ROA-543.

Ambrazas, Vytautas ed. 1997. *Lithuanian Grammar*. Vilnius Baltos: Lankos.

Archangeli, Diana, Moll, Laura, and Ohno, Kazutoshi. 1998. Why not *NC. In *Proceedings of the Thirty-Fourth Meeting of the Chicago Linguistics Society, Part 1: Main Session*, ed. by M. Catherine Guber, Derrick Higgins, Kenneth S. Olson, and Tamara Wysocki, 1-26. Chicago: CLS.

Bakovic, Eric. 2005. Antigemination, assimilation and the determination of identity. *Phonology* 22:279-315.

Bakovic, Eric. 2007. A revised typology of opaque generalizations. Ms., University of California San Diego. ROA-850.

Balari, Sergio, Marín, Rafael, and Vallverdú, Teresa. 2000. Implicational constraints,defaults, and markedness. GGT Report de Recerca GGT-00-8, Universitat Autònoma de Barcelona. ROA-396.

Beckman, Jill N. 1999. *Positional Faithfulness: An Optimality Theoretic Treatment of Phonological Asymmetries*: Outstanding Dissertations in Linguistics: Garland Publishing.

Benua, Laura. 1997. Transderivational Identity: Phonological Relations between Words. PhD. dissertation, University of Massachusetts, Amherst.

Bermudez-Otero, Ricardo. 2003. The acquisition of phonological opacity. In *Variation within Optimality Theory: Proceedings of the Stockholm workshop on 'Variation within Optimality Theory'*, ed. by Jennifer Spenader, Anders Eriksson, and Osten Dahl, 25-36. Stockholm: Dept. of Linguistics, Stockholm University. ROA-593 (expanded version).

Chomsky, Noam. 1959. On Certain Formal Properties of Grammars. *Information and Control* 2:137-167.

Chomsky, Noam. 1964. *Current Issues in Linguistic Theory*. The Hague: Mouton.

Chomsky, Noam, and Halle, Morris. 1968. *The Sound Pattern of English*. New York City: Harper & Row.

Crowhurst, Megan, and Hewitt, Mark. 1997. Boolean operations and constraint interactions in Optimality Theory. Ms., University of North Carolina, Chapel Hill, and Brandeis University, Waltham, MA. ROA-229.

Dambriunas, Leonardas, Klimas, Antanas, and Schmalstieg, William. 1966. *Introduction to Modern Lithuanian*. Brooklyn: Franciscan Fathers Press.

de Lacy, Paul. 2002. The formal expression of markedness. PhD. dissertation, University of Massachusetts, Amherst.

Demuth, Katherine. 1995. Markedness and the development of prosodic structure. In *Proceedings of the North East Linguistics Society 25*, ed. by Jill Beckman, 13-25. Amherst, MA: GLSA, University of Massachusetts. ROA-50.

Gnanadesikan, Amalia. 1995. Markedness and faithfulness constraints in child phonology. Ms., University of Massachusetts, Amherst, MA. ROA-67.

Gnanadesikan, Amalia. 1997. Phonology with ternary scales. Doctoral dissertation, University of Massachusetts, Amherst.

Hayes, Bruce. 2004. Phonological acquisition in Optimality Theory: The early stages. In *Constraints in Phonological Acquisition*, ed. by René Kager, Joe Pater, and Wim Zonneveld, 158-203. Cambridge: Cambridge University Press.

Hewitt, Mark S., and Crowhurst, Megan J. 1996. Conjunctive constraints and templates in Optimality Theory. In *Proceedings of the North East Linguistic Society 26*, ed. by Kiyomi Kusumoto, 101-116. Amherst, MA: GLSA.

Idsardi, William J. 2000. Clarifying opacity. *The Linguistic Review* 17:337-350.

Itô, Junko, and Mester, Armin. 2001. Structure Preservation and Stratal Opacity in German. In *Segmental Phonology in Optimality Theory: Constraints and Representations*, ed. by Linda Lombardi, 261-295. Cambridge University Press.

Itô, Junko, and Mester, Armin. 2003. On the Sources of Opacity in OT: Coda Processes in German. In *The Syllable in Optimality Theory*, ed. by Caroline Féry and Ruben van de Vijver, 271-303. Cambridge University Press.

Jarosz, Gaja. 2006. Rich Lexicons and Restrictive Grammars - Maximum Likelihood Learning in Optimality Theory. PhD. dissertation, The Johns Hopkins University, Baltimore, MD.

Kager, Rene. 1999. *Optimality Theory*. Cambridge: Cambridge University Press.

Kenstowicz, Michael, and Kisseberth, Charles. 1971. Unmarked bleeding orders. *Studies in the Linguistic Sciences* 1:8-28.

Kenstowicz, Michael. 1972. Lithuanian phonology. *Studies in the Linguistic Sciences* 2:1-85.

Kiparsky, Paul. 1971. Historical linguistics. In *A survey of linguistic science*, ed. by W. O. Dingwall, 576-649. College Park: University of Maryland Linguistics Program.

Kiparsky, Paul. 1973. Abstractness, opacity and global rules (Part 2 of "Phonological representations"). In *Three Dimensions of Linguistic Theory*, ed. by O. Fujimura, 57-86. Tokyo: TEC.

Kiparsky, Paul. 1982. Lexical phonology and morphology. In *Linguistics in the Morning Calm*, ed. by I. S. Yang, 3-91. Seoul: Hanshin.

Kiparsky, Paul. 1985. Some consequences of lexical phonology. *Phonology* 2:85-138.

Kiparsky, Paul. 2003. Syllables and Moras in Arabic. In *The Syllable in Optimality Theory*, ed. by Caroline Féry and Ruben van de Vijver, 147-182. Cambridge University Press.

Kiparsky, Paul. to appear. *Paradigm effects and opacity*. Stanford: CSLI Publications.

Kirchner, Robert. 1995. Going the distance: Synchronic chain shifts in Optimality Theory. Ms., UCLA. ROA-66.

Kisseberth, Charles. 1970. On the functional unity of phonological rules. *Linguistic Inquiry* 1:291-306.

Lombardi, Linda. 1995. Why place and voice are different: Constraint interactions and feature faithfulness in Optimality Theory. Ms., University of Maryland, College Park. ROA-105.

Lombardi, Linda. 2001. Why Place and Voice are Different: Constraint-Specific Alternations in Optimality Theory. In *Segmental Phonology in Optimality Theory: Constraints and Representations*, ed. by Linda Lombardi, 13-45. Cambridge University Press.

Mathiassen, Terje. 1996. *A short grammar of Lithuanian*. Columbus: Slavica.

McCarthy, John J., and Prince, Alan. 1993. Prosodic Morphology I: Constraint Interaction and Satisfaction.

McCarthy, John J., and Prince, Alan. 1995. Faithfulness and Reduplicative Identity. In *University of Massachusetts Occasional Papers 18: Papers in Optimality Theory*, ed. by Jill Beckman, Laura Walsh Dickey, and Suzanne Urbancyzk, 249-384. Amherst, MA: GLSA, University of Massachusetts.

McCarthy, John J. 1999. Sympathy and phonological opacity. *Phonology* 16:331-399.

McCarthy, John J., and Prince, Alan. 1999. Faithfulness and identity in Prosodic Morphology. In *The Prosody-Morphology Interface*, ed. by René Kager, Harry van der Hulst, and Wim Zonneveld, 218-309. New York City: Cambridge University Press. ROA-216.

McCarthy, John J. 2006. Restraint of analysis. In *Wondering at the Natural Fecundity of Things: Essays in Honor of Alan Prince*, ed. by Eric Bakovic, Junko Ito, and John J. McCarthy. ROA-844.8.

McCarthy, John J. 2007a. Derivations and levels of representation. In *The Cambridge Handbook of Phonology*, ed. by Paul de Lacy, 99-117. Cambridge: Cambridge University Press.

McCarthy, John J. 2007b. *Hidden Generalizations: Phonological Opacity in Optimality Theory*: Advances in Optimality Theory. Oakville, CT: Equinox.

Moreton, Elliott. 2004. Non-computable Functions in Optimality Theory. In *Optimality Theory in Phonology: a reader*, ed. by John J. McCarthy, 141-163. Blackwell.

Odden, David. 2005. *Introducing Phonology*. Cambridge: Cambridge University Press.

Padgett, Jaye. 1995. Partial class behavior and nasal place assimilation. In *Proceedings of the Arizona Phonology Conference: Workshop on features in Optimality Theory*. Tucson, AZ: Coyote Working Papers, University of Arizona, Tucson.

Pater, Joe. 1999. Austronesian nasal substitution and other NC effects. In *The Prosody-Morphology Interface*, ed. by Rene Kager, Harry van der Hulst, and Wim Zonneveld, 310-343. Cambridge: Cambridge University Press.

Prince, Alan, and Smolensky, Paul. 1993/2004. *Optimality Theory: Constraint interaction in generative grammar*. Malden, MA: Blackwell.
Prince, Alan. 2002. Entailed Ranking Arguments. Ms., Rutgers University, New Brunswick. ROA-500.
Prince, Alan. 2003. Arguing Optimality. In *UMOP 26: Papers in Optimality Theory II*, ed. by Angela Carpenter, Andries Coetzee, and Paul de Lacy, 269-304. GLSA. ROA-562.
Prince, Alan, and Smolensky, Paul. 2004. *Optimality Theory: Constraint Interaction in Generative Grammar*: Blackwell.
Prince, Alan, and Tesar, Bruce. 2004. Learning phonotactic distributions. In *Constraints in Phonological Acquisition*, ed. by René Kager, Joe Pater, and Wim Zonneveld, 245-291. Cambridge: Cambridge University Press.
Shaw, Patricia. 1976. Theoretical issues in Dakota phonology and morphology. PhD. dissertation, University of Toronto.
Shaw, Patricia. 1985. Modularism and substantive constraints in Dakota lexical phonology. *Phonology Yearbook* 2:173-202.
Smolensky, Paul. 1995. On the internal structure of the constraint component Con of UG. Ms.
Smolensky, Paul. 1996. The initial state and "richness of the base" in Optimality Theory. Technical Report JHU-CogSci-96-4, The Johns Hopkins University, Baltimore, MD. ROA-154.
Tesar, Bruce. 2006. Faithful contrastive features in learning. *Cognitive Science* 30:863-903.
Tesar, Bruce, and Prince, Alan. to appear. Using phonotactics to learn phonological alternations. In *Proceedings of the Thirty-Ninth Conference of the Chicago Linguistics Society, vol. II: The Panels*, ed. by Johnathon E. Cihlar, Amy L. Franklin, David W. Kaiser, and Irene Kimbara. ROA-620.
Wilson, Colin. 2001. Consonant Cluster Neutralization and Targeted Constraints. *Phonology* 18:147-197.
Wolf, Matthew. 2006. A note on the construction of complex OT constraints by material implication. Ms., University of Massachusetts, Amherst. ROA-886.
Zoll, Cheryl. 1998. *Parsing below the segment in a Constraint-based Framework*: Dissertation in Linguistics. Stanford: CSLI.