Process interactions

Eric Baković & Lev Blumenfeld — for de Lacy & Jardine (eds.), *The Cambridge Handbook of Phonology*, 2nd ed.

1 Introduction

Generative phonological grammars are complex functions that map input representations (≈ underlying forms) to output representations (≈ surface forms). These complex functions are understood to consist of simpler parts; models of generative phonology differ in how they break the entire complex map down into those simpler parts, what those parts are, and how they combine. Sometimes the mechanism by which these simpler parts combine is irrelevant: the mere presence of each of the simpler parts in the overall grammar is sufficient. But sometimes the combination mechanism *does* matter, because different ways of combining some simpler parts can lead to different results. It is these meaningful combinations that are referred to as INTERACTIONS.

In the earliest models of generative phonology (e.g. Chomsky & Halle 1968, henceforth *SPE*), the complex function is broken down into REWRITE RULES that combine with each other via SERIAL ORDERING. In more recent models (e.g. Prince & Smolensky 1993/2004, henceforth OT), the complex function is broken down into individual CONSTRAINTS that combine with each other via RANKING. The *SPE*-like rewrite rule corresponds most directly to what is commonly referred to as a PROCESS, and serial ordering of rewrite rules has dominated discussions of 'process interaction' in the literature. We follow these direct and dominant threads in this chapter, presenting some of the work that has attempted to elucidate how processes can interact in a grammar.

We first set out on some well-worn paths in Section 2, briefly surveying the literature on rule ordering that has been sufficiently influential for its ideas to be considered "classical". In Section 3 we map out this territory in a new way, by offering a formalized treatment that gives the tools to delimit a more precise and complete typology of process interactions. In Section 4 we focus on some consequences of process interaction for phonological grammars and the extent to which phonologically significant generalizations are true of input-output maps. In Section 4.1, we use our formalization of process interaction to circumscribe phonological opacity. In Section 4.2, we turn our attention to a superficially different condition on phonological grammars, output-drivenness, and explore its connection with opacity. Finally, in Section 5 we discuss types of interactions that involve blocking: disjunctive application in Section 5.1, nonderived environment blocking in Section 5.2, and blocking by constraint in Section 5.3. A few common threads tie our discussion together. Methodologically, we hope to show by example that questions about phonological theory are productively addressed with formally precise treatment. Substantively, we hope to show that many of the concepts relating to interaction that have been explored by generative phonologists over the years are amenable to a unified treatment within our formal approach.

In the remainder of this introduction we offer some remarks on the definition of "process" that we use in this chapter. We begin by assuming that phonological representations consist of strings of segments, with

each segment being a matrix (set) of feature-value pairs.[1] Let us assume further that we are given a set of input strings, the output strings that those input strings map to, a correspondence relation between segments in the input strings and segments in the output strings, and information on the morphological identity of input segments. Together, we refer to these givens as a LANGUAGE. A language can be described using a GRAMMAR, which consists of some combination of string-to-string functions that we call PROCESSES.

What a process is, and how processes are combined in the grammar to describe the language, depends on the theoretical frame. Adopting the frame of *SPE*, a process P is a function statable as a rewrite rule with the familiar $A \rightarrow b \ / \ C \_\_ \ D$ schema. A is the FOCUS of P, a feature matrix describing a natural class of segments. The CHANGE made by P is given by b, a set of feature-value pairs to be substituted for potentially conflicting values in A by priority union (Reiss 2022): if α is a member of the natural class described by A, then $\alpha \uplus b$ is a feature matrix identical to α except that the feature-value pairs of b replace α's corresponding feature-value pairs, or are added to α if α does not contain them. C and D represent the (local) CONTEXT of A, with C to the left and D to the right, and are either feature matrices describing natural classes of segments (like A), boundaries (word, syllable, etc.), or null. Lastly, A or b, but not both, may be "∅", or null: if A is null, P inserts a segment with the feature values specified in b; if b is null, P deletes any member of the natural class described by A.[2] We will say that an input string x is SUBJECT TO a process P iff $P(x) = y$ and $y \neq x$ — in other words, if P applies to x non-vacuously.

Within these very broad parameters, any given language can be described by a multitude of grammars. One goal of phonological theorizing is to delimit the space of possible grammars by imposing further conditions on them. Examples of such conditions are listed in (1).

(1) Potential additional conditions on processes in *SPE*-style grammars

a. *Single change condition*. Every process changes the value of at most one feature, deletes a segment, or inserts a segment. Under this condition, if A is a feature matrix then b either specifies only one feature-value pair or is null; if b specifies more than one feature-value pair, then A is null.

b. *Alternation condition*. If a process P maps x to y, then there exists at least one morpheme that surfaces with a y in the context specified by P and that surfaces with a corresponding x when not in that context. (Note: this means that P must apply non-vacuously to at least one input.)

c. *Collapsing condition*. No two processes adjacent in the ordering may be replaced by a single process resulting in the same grammar. (That is, if two adjacent processes can be notationally collapsed into a single process that respects all the other conditions, they must be so collapsed.)

d. *Inventory non-reduction condition*. The segmental inventory of the output of a process cannot be a proper subset of its input segmental inventory. (Note: subsumed by the alternation condition.)

Such conditions impose further limitations on what a "process" is in the analysis of a particular language: a process is not simply any function stateable as a rewrite rule, but also one that meets certain well-formedness conditions such as those in (1). For our own expository purposes in this chapter, a process is an

[1] Of course, phonological representations may consist of more than this: syllabic and other prosodic structure, autosegmental structure, and so forth. The limitation to strings of feature matrices here is purely expository.

[2] The *SPE* rule schema also allows for a number of notational devices with special interpretations that greatly expand the power of rewrite rules: curly braces, parentheses, angled brackets, and so on. We set these aside in the interests of clarity.

$A \rightarrow b \,/\, C \_\_ D$ rewrite rule as defined above, one that makes only one change (respecting the single change condition, (1a)) and that is directly motivated by at least one surface alternation (respecting the alternation condition, (1b)).

2 Background

At the most basic level, an interaction between processes P and Q holds when their order of composition matters for at least some inputs; i.e., if there exists an input x such that $Q(P(x)) \neq P(Q(x))$. Non-interacting processes found in the same grammar require no further comment: the output of the grammar is simply the result of the separate application of those processes, in either order, or simultaneously. If, on the other hand, two interacting processes cooccur in the same grammar, an independent stipulation is needed to specify how their action is combined. Historically in the generative literature, (at least) three ways in which processes *qua* rewrite rules can in principle combine with each other to form a grammar have been put forth (see Kenstowicz & Kisseberth 1979, Ch. 8).[3]

One of these mechanisms is serial ordering, formally equivalent to function composition and the primary mechanism adopted in *SPE*. In this model, two processes P, Q must be ordered with respect to each other — that is, if they interact, their order matters. Whichever order results in the correct outcome for all inputs is then claimed to be the correct order between those two processes in the entire grammar.

The literature has long identified two basic relations between two processes P, Q that make them 'relevant' to each other such that serial ordering between P and Q is claimed to be necessary (Kiparsky 1968, Chafe 1968, Wang 1969, and many more since; see a historical overview in Kenstowicz & Kisseberth 2022). One relation is traditionally termed FEEDING. For distinct strings[4] x, y, z, P feeds Q on input x if $P(x) = y$, $Q(x) = x$, and $Q(y) = z$. That is, x is a form that is subject to P but not to Q, except that once P applies to x, the result y is (newly) subject to Q. The other relation is traditionally termed BLEEDING: P bleeds Q on input x if $P(x) = y$, $Q(x) = z$, and $Q(y) = y$. That is, x is a form that is subject to both P and Q, except that once P applies to x, the result y is no longer subject to Q. Feeding and bleeding as defined are relations between processes, regardless of their actual order of application.[5]

Serial ordering thus produces the 'classical' interaction typology shown in (2), with the feeding vs. bleeding relations along one axis, and one order vs. its opposite along the other axis. Here and in the rest of the chapter we use the character "$\rhd$" to refer to process order, and ordering statements are placed in angled brackets: "$\langle P \rhd Q \rangle$". For better or worse, the terms "feeding" and "bleeding" are also used to mean that P *successfully* affects Q in the relevant way due to the order $\langle P \rhd Q \rangle$, while the "counter" prefix (originally due to Newton 1971) is used to mean that P *fails* to affect Q in the relevant way due to the order $\langle Q \rhd P \rangle$.

[3] In the interests of space, we set aside here what Kenstowicz & Kisseberth (1979:318ff) refer to as the 'multiple application problem' — the question what to do when an input string contains multiple substrings that are each subject to a given rule, or what to do when the application of a rule to an input string results in a string that is newly subject to the same rule.

[4] Two strings are identical if their corresponding segments are pairwise identical; they are distinct otherwise.

[5] Chafe's (1968) more perspicuous (but less memorable) terms for feeding and bleeding relations are 'additive interference' and 'subtractive interference', respectively.

(2) Classical typology of process interaction types

	P feeds Q on x	P bleeds Q on x
$\langle P \rhd Q \rangle$	FEEDING $Q(P(x)) = Q(y) = z$	BLEEDING $Q(P(x)) = Q(y) = y$
$\langle Q \rhd P \rangle$	COUNTERFEEDING $P(Q(x)) = P(x) = y$	COUNTERBLEEDING $P(Q(x)) = P(z) = w$

Kiparsky (1968, 1971, 1976) offers two different cross-classifications of this typology.

In Kiparsky (1968), feeding and counterbleeding are grouped together as 'unmarked' interactions in that the rules are MAXIMALLY APPLIED in the derivation, whereas bleeding and counterfeeding are 'marked' in that one or the other of the rules does not have a chance to apply.

In Kiparsky (1971, 1976), counterfeeding and counterbleeding are grouped together as marked interactions resulting in OPACITY of Q. In the case of counterfeeding, Q is opaque because there is a surface string y; McCarthy (1999) calls this UNDERAPPLICATION OPACITY, because Q has not applied when it appears it should have. In the case of counterbleeding, Q is opaque because P changes the output of Q, z, to another output w that disguises the effect of Q;[6] McCarthy (1999) calls this OVERAPPLICATION OPACITY, because Q has applied when it appears that it *shouldn't* have. By contrast, the unmarked feeding and bleeding interactions are called TRANSPARENT. This substantive classification is discussed further in Section 4.1.

Serial ordering is not the only possible process combination mechanism. Another possibility is FREE REAPPLICATION: processes randomly apply and reapply to each other's outputs until a final output is reached that is not subject to any of the processes (= 'convergence'). This mechanism, implicitly adopted as part of the Natural Generative Phonology framework (Hooper[Bybee] 1976), alters the classical typology in (2) in two distinct ways. First, free reapplication makes it impossible to achieve the effect of counterfeeding: Q can always reapply to the result of $P(x)$, producing z, which is identical to the result achieved by feeding. Second, if P and Q stand in a bleeding relation, free reapplication does not produce a consistent result but instead randomly results in one of two distinct outputs depending on which rule happens to apply first. This is because in the case of the bleeding relation, the outputs of both $Q(P(x))$ and $P(Q(x))$ are not further subject to P or Q. If P applies first, then bleeding is inevitable because Q cannot apply to $P(x)$. If Q applies first, then counterbleeding is inevitable because P *can* apply to $Q(x)$. It is thus incoherent to include bleeding relations in a grammar with free reapplication — unless perhaps the two outcomes are in free variation. Feeding is thus the only interaction type that is able to be reliably modeled with free reapplication, and any cases appearing to require one of the other three interaction types must be analyzed otherwise.

The third process combination mechanism is DIRECT MAPPING, otherwise known as SIMULTANEOUS APPLICATION. In this model, each process applies to the same (initial, underlying) input, if that input is subject to the process — but then no other processes apply. This mechanism also alters the classical typology in two ways. First, it is not possible to get the effect of feeding, because Q cannot 'wait' to apply to the result of $P(x)$. Second, it is not possible to get the effect of bleeding, in this case because Q cannot 'wait' to *not* apply to the result of $P(x)$. Thus counterfeeding and counterbleeding are the only interaction types that can be modeled with direct mapping, and any cases appearing to require feeding or bleeding must be

[6] This presumes that $w \neq z$; otherwise, P and Q are in a (non-opaque) MUTUAL BLEEDING relationship; see Section 3.

analyzed otherwise.[7] These are ways in which direct mapping is less expressive than serial ordering, but there is at least one way where it is more expressive. When two processes stand in the relation of mutual bleeding, serial ordering can only result in one of them applying, but direct mapping causes both of them to apply. A specific example will be provided in footnote 9.

Actual rule-based alternatives to *SPE* often retain some measure of serial ordering. For example, work under the rubric of the "Universal Rule Application Hypothesis" (e.g. Koutsoudas et al. 1974) aims to discover universal principles that determine how a given pair of rules is to apply to all forms in the language, thus making any order between them 'intrinsic' as opposed to 'extrinsic'. Anderson (1969, 1974, 1975) develops a "local ordering" approach, whereby the order of any given pair of rules is in part determined by properties of the input, such that the order between two rules with respect to one input might differ from their order with respect to another.

It is worth noting that the full *SPE* model includes two additional mechanisms affecting interactions between processes, beyond serial ordering. One is *the cycle*, which is sensitive to the morphosyntactic bracketing of the string that is input to the phonological grammar. The entire grammar (or a portion thereof) is first applied to the innermost bracketed substring(s), then those innermost brackets are erased and the grammar is applied to the next innermost bracketed substring(s), and so on until the outermost brackets are reached. Each pass of the grammar in this fashion is referred to as a 'cycle'. This mechanism results in certain well-defined situations where one process P might crucially precede another process Q *within* a cycle but also crucially follow Q *across* cycles.

The other additional mechanism of process interaction in *SPE* is known as DISJUNCTIVE APPLICATION, which in *SPE* was generally attributed to special properties of certain notational devices (such as parentheses and angled brackets; see footnote 2). If two rules P, Q apply disjunctively to an input x, one rule (say, P) is privileged such that if P applies to x, Q is blocked from applying to x and to any other representation arising from x in the overall derivation of the form. A more general principle of disjunctive application was later formulated, first by Anderson (1969) and in more lasting form by Kiparsky (1973), motivated by examples where *SPE*'s notational devices fell short: this is known as the Elsewhere Condition, to be discussed further in Section 5.1.

There is no direct analogue of a 'process' in OT and related constraint-based theories, and so grammars within these theories do not fit into this discussion in an obvious way. Nevertheless, there is a vast literature aiming to clarify the ways in which OT differs from *SPE* in terms of the interaction types (broadly construed) that each predicts, and/or of attempts to modify the most basic architecture of OT or of *SPE* to accommodate attested interaction types that pose a challenge to it. Much of this literature focuses on the fact that opaque interactions are a challenge for OT (*cf*. Baković 2011), and attributes this challenge to the non-serial nature of the theory. Pruitt (2023) demonstrates that the attribution to (non-)serialism is wrong, pointing out that (if anything) it is the transparent interactions that require serial ordering — this is why direct mapping is able to describe only the opaque ones. A better diagnosis is that, under some reasonable assumptions about constraints, input-output maps in OT are OUTPUT-DRIVEN (Tesar 2014:83ff) — a formal property whose empirical consequences are closely aligned with transparency, as we discuss in Section 4.2.

[7] This property of direct mapping thus serves as the basis of a formal diagnostic for opacity, on which see Section 4.1.

3 Interactions

A more precise and complete handle on the universe of process interactions predicted by serial ordering is provided in Baković & Blumenfeld (2023). Here we highlight the main idea of that work, omitting much detail and complexity. One simplification we will make is to consider only strings to which any given process may apply at most once (recall footnote 3). With that simplification, for any process P it is possible to talk about its INPUT SET, $I(P)$: the set of all strings to which P applies non-vacuously, those strings that are "subject to" P as defined previously. It is also possible to define the OUTPUT SET, $O(P)$: the set of strings with a non-vacuous P-input. Work on the classical typology in (2), in all its variety, has only focused on how one rule potentially affects the input set of another rule. The main idea of Baković & Blumenfeld (2023) is that in order to glean the full typology one must also give due to how one rule potentially affects the output set of another rule. This move not only yields a more complete typology of interactions, but also illuminates the notion of opacity.

To appreciate the utility of this approach, consider two apparent cases of feeding. Our examples are hypothetical for consistency, sacrificing some degree of phonetic realism to achieve parallelism across all of the examples.[8] The rules are given both featurally and segmentally, for clarity, assuming the segmental inventory $\{k, c, i, e\}$, with the feature [±back] distinguishing the consonants and [±high] the vowels, which are assumed to be [–back]. This minimal inventory allows us to write rules in a simple way, e.g. "V → [–high]" specifies the mapping $\{i \mapsto e\}$.

In the first case (3), a velar palatalization process P applies before all front vowels; a high vowel lowering process Q then applies after palatals. Clearly P feeds Q: $ki \mapsto ci \mapsto ce$. If the order is $\langle P \rhd Q \rangle$, this is a feeding interaction; under the opposite order $\langle Q \rhd P \rangle$, it is counterfeeding: $ci \mapsto ce$ and $ki \mapsto ci$.

(3) Case 1: velar palatalization (P) interacts with high vowel lowering (Q)

P:	C → [–back] / __ [V, –back]	k → c / __ {i,e}
Q:	V → [–high] / [C, –back] __	i → e / c __

The second case (4) is identical to the first except that palatalization only applies before the high front vowel.

(4) Case 2: velar palatalization (P) interacts with high vowel lowering (Q)

P:	C → [–back] / __ [V, +high, –back]	k → c /__ i
Q:	V → [–high] / [C, –back] __	i → e / c __

[8] We may be criticized for using simple hypothetical examples to illustrate interactions, a habit we maintain below in Section 4.1 when discussing opacity using the same strings, along with an equally simple hypothetical stress rule. One basis for such a criticism might be a general notion that it is good for linguists to "engage with data". That is of course true, but attention to data does not preclude attention to other things. A more substantive reason to ask for "real" examples might be the fear that simple hypothetical examples are of insufficient complexity to illustrate the entire set of possibilities. We believe, on the contrary, that idealization is necessary when the goal is to enumerate an inventory of formal possibilities rather than some "holistic" analysis. Every "real" example is formally indistinguishable from one of our simple hypothetical examples in relevant respects, i.e. in how one process affects the input and output set of another process.

Again there is apparent feeding under the $\langle P \rhd Q \rangle$ order, *ki* ↦ *ci* ↦ *ce*, and apparent counterfeeding under the opposite order $\langle Q \rhd P \rangle$, in the same way as in Case 1: *ci* ↦ *ce*, *ki* ↦ *ci*.

And yet, there is something subtly different about these two examples. In Case 1, the feeding interaction is transparent: the output form *ce* has undergone both palatalization and lowering, and the conditions motivating the application of both rules are present in this surface form. In Case 2, on the other hand, the feeding interaction is opaque: the same output form *ce* shows the application of palatalization, but the added [+high] condition for its application is not present in that output — the effect of palatalization is disguised by the subsequent application of lowering. This is overapplication, much as in the case of counterbleeding.

This feeding with overapplication opacity is puzzling from the classical perspective. The problem stems from the fact that the classical approach asks only if and how P affects the input set of Q. Does P create new members of $I(Q)$? If so, there is feeding. Does P destroy members of $I(Q)$? If so, there is bleeding. But asking questions about input sets alone does not help to distinguish Case 1 from Case 2. Indeed, in both cases $I(Q)$ contains the string *ci*, and in both cases palatalization (P) creates new instances of such strings. Yet, one case is transparent and the other is opaque, and that difference must be due to the broader context of application of palatalization in Case 1, i.e. the fact that Case 1 includes a (potential) mapping *ke* ↦ *ce* while Case 2 does not. To understand this difference, one must also ask if and how P affects the output set of Q. Does P create new members of $O(Q)$, or does it destroy them? This is where our two cases differ. The set $O(Q)$ contains the string *ce*. In Case 1, P creates new instances of such strings, from *ke*. In Case 2, no string *ce* is created by P.

Let us call situations where P creates members of $I(Q)$ INPUT PROVISION — P input-provides Q — and situations where P destroys members of $I(Q)$ INPUT REMOVAL — P input-removes Q. Likewise for the effect of P on members of $O(Q)$, we will refer to OUTPUT PROVISION and OUTPUT REMOVAL. In shorthand notation, we will write "+i" for input provision, "–i" for input removal, and *mutatis mutandis* for "+o", "–o": P+iQ, P–iQ, P+oQ, and P–oQ. More careful definitions are given in (5). We will refer to these basic elements of an interaction as 'atoms', for reasons that will become clearer below.

(5) a. **Input provision.** P+iQ iff $\exists x$ such that $x \notin I(Q)$ and $P(x) \in I(Q)$.
b. **Input removal.** P–iQ iff $\exists x$ such that $x \in I(Q)$ and $P(x) \notin I(Q)$.
c. **Output provision.** P+oQ iff $\exists x$ such that $x \notin O(Q)$ and $P(x) \in O(Q)$.
d. **Output removal.** P–oQ iff $\exists x$ such that $x \in O(Q)$ and $P(x) \notin O(Q)$.

Observe that the feeding example (Case 1 above) involves two of these atoms: P+iQ, P+oQ. Case 2, in contrast, only involves P+iQ but not P+oQ. We will see below that Case 2 involves another atom. For completeness, let us now introduce Case 3, to illustrate bleeding. It is just like Case 1, but with a different environment for Q.

(6) Case 3: velar palatalization (P) interacts with high vowel lowering (Q)

P:	C → [–back] / __ [V, –back]	k → c / __ {i,e}
Q:	V → [–high] / [C, +back] __	i → e / k __

To examine the atoms contained in the interactions exemplified by these three cases, we use arrow diagrams illustrating the configurations of mappings affecting relevant strings (7). In our toy examples, there are four

such strings to be considered: {*ki, ci, ke, ce*}. We will place these strings in the diagrams in such a way that the mappings effected by P in each case are displayed horizontally while those effected by Q in each case are displayed vertically. Cases 1 and 3 instantiate feeding and bleeding, respectively. Following Baković & Blumenfeld (2023), we will refer to the situation in Case 2 as SEEDING.[9]

(7) Case 1: feeding

ki	→	*ci*
		↓
ke	→	*ce*

Case 2: seeding

ki	→	*ci*
		↓
ke		*ce*

Case 3: bleeding

ki	→	*ci*
↓		
ke	→	*ce*

Focusing on Case 3, observe that, as expected, P–iQ: the string *ki* is an input to Q, but P(*ki*) = *ci* is not. But this is not the only atom involved here: there is also P–oQ: *ke* is a possible Q-output, but P(*ke*) = *ce* is not.

More generally, for an atom to be present in such a diagram, there must exist an arrow with no other arrow parallel to it. This is the case with all three vertical Q-arrows in the diagrams in (7). If that Q-arrow lacks a parallel arrow to the left, as in Cases 1 and 2, we are dealing with provision, of inputs or outputs or both. If it lacks a corresponding arrow to the right, as in Case 3, we are dealing with removal, of inputs or outputs or both. Looking at Case 2, seeding, we see that P's arrow also lacks a parallel arrow at the bottom of the diagram. This indicates the presence of another atom besides P+iQ: in this case, Q–oP. Specifically, Q takes the string *ci*, which is in the output set of P, and produces the string *ce* which is not in that set. Summarizing:

(8) a. **Feeding.** P feeds Q iff P+iQ and P+oQ.
b. **Seeding.** P seeds Q iff P+iQ and Q–oP.
c. **Bleeding.** P bleeds Q iff P–iQ and P–oQ.

The atomic metaphor should now become clear. The basic provision (e.g. P+iQ) and removal (e.g. P–oQ) properties are atoms which are found in various combinations in observed phenomena; interactions like feeding, seeding, and bleeding are 'molecules' containing different combinations (specifically, pairs) of atoms.

These three interactions do not exhaust the space of molecular possibilities. There are two more topologically distinct configurations, illustrated diagrammatically below. (A sixth case, with *ki* ↦ *ke* ↦ *ce*, is topologically identical to Case 2; it instantiates seeding, but with the arbitrary roles of P and Q reversed: Q+iP, P–oQ.) In Case 4, MUTUAL BLEEDING, the palatalization rule P is identical to that in Case 2 and the lowering rule Q is identical to that in Case 3.[10] In Case 5, MERGER, P applies only before the mid front vowel *e* and Q is identical to the lowering rule in Case 2.[11]

[9] Seeding is referred to as 'opaque feeding' in Lee (1999, 2007) and as 'self-destructive feeding' in Baković (2007, 2011).

[10] Note that simultaneous application in the case of mutual bleeding would result in the output *ce*, with both rules applying to the same input form *ki*. This result is impossible to achieve when the rules are ordered serially, illustrating one way in which simultaneous application is more expressive than serial ordering (Vago 1977, Wolf 2011).

[11] Merger is not an interaction in the sense that, unlike feeding, (mutual) bleeding, and seeding, for merger the composition of P and Q is commutative, i.e. the order of application does not matter: $P(Q(x)) = Q(P(x))$ for all x. Order of application matters under at least one of two topological conditions: when two arrows originate in one string, as in (mutual) bleeding, or when one arrow ends on a string where another arrow begins, as in feeding, seeding, and bleeding. Merger is the only structure that meets neither of these conditions.

(9) Case 4: mutual bleeding

ki	→	*ci*
↓		
ke		*ce*

Case 5: merger

ki		*ci*
		↓
ke	→	*ce*

The atoms in Case 4, mutual bleeding, are P–iQ and Q–iP. The atoms in Case 5, merger, are P+oQ and Q+oP. There are further interactional possibilities in this more complete typology, but they would require us to expand our hypothetical examples beyond mappings between members of the four strings {*ki*, *ci*, *ke*, *ce*}. We refer the reader to Baković & Blumenfeld (2023) for a complete treatment. The above discussion, however, is sufficient to make clear one significant consequence of the atomic approach: it offers a formal (and complete) handle on opacity. This will be the subject of Section 4.1.

4 Zooming out

The previous section established a basic inventory of abstract atomic elements generating the typology of interactions between two processes — that is, local interactions. In this section we take a step back to consider interactions from a more global, grammar-level perspective.

Any theory that describes phonological systems using at least two levels (input and output) raises a basic question concerning the locus of significant generalizations about phonology. To what extent do surface forms reveal linguistically significant generalizations? The general intuition that phonological mappings are motivated by restrictions on outputs has been called SURFACE ORIENTEDNESS (Tesar 2014:1). While individual processes may be surface-oriented in some way, the issue becomes a complex one when interaction is considered: one process may affect the surface orientation of another process. In this section we examine the consequences of interaction for surface orientedness from two distinct angles: opacity and output-drivenness.

Presence or lack of motivation for a process in surface structures can be made precise in many ways. The approach that has received the most attention in this context is opacity (and its converse, transparency). In the early days of generative grammar, opacity figured in discussions of language change in the generative framework (Kiparsky 1971). More recently, because OT encounters difficulties in accommodating classical opaque generalizations, many have critiqued the theory on these grounds (e.g. Idsardi 2000, Vaux 2008), attempted to dispute the existence of opacity (e.g. Sanders 2003, Green 2007), or proposed modifications of OT to accommodate it (e.g. McCarthy 1999, 2003, 2007). In Section 4.1, we focus on the formal properties of opacity as illuminated by our atomic approach to interactions more generally.

A similar and formally well-understood concept is OUTPUT-DRIVENNESS (Tesar 2014, Magri 2018a,b), a surface-oriented property of the overall grammar taken as an input-output map. While applicable to any theory mapping inputs to outputs, output-drivenness has received attention mostly in the context of constraint-based theories like OT. In Section 4.2 we discuss the connections between output-drivenness, opacity, and our atomic approach to interactions, showing that output-drivenness is a more stringent requirement than transparency.

4.1 Opacity

Opacity has received numerous semi-formal characterizations in the literature. The most basic, originally due to Kiparsky (1971, 1976), defines opacity as two disjoint sets of situations where surface forms do not show the expected effects of the application of a process.[12] (Recall that '$A \uplus b$' notates priority union: a feature matrix identical to A except that the feature-value pairs of b are substituted or added.)

(10) A process P of the form $A \rightarrow b \,/\, C \_\_ D$ is OPAQUE to the extent that

a. there are surface instances of A in the environment $C \_\_ D$ that are subject to P;[13]

b. there are surface instances $A \uplus b$ derived by P in an environment other than $C \_\_ D$.

Clause (10a) describes underapplication, where there are output strings that are subject to P but to which P has failed to apply. Clause (10b) describes overapplication, where there are output strings that have non-vacuously undergone P despite not showing the necessary conditions for the application of P. As noted in Section 2, counterfeeding orders result in underapplication, and counterbleeding orders result in overapplication.

To see how opacity emerges from the nature of the atoms, let us first examine the opacity facts in the hypothetical delimited universe of four strings $\{ki, ci, ke, ce\}$ and the five molecular interactions summarized in Section 3. The table in (11) lists those interactions along with observations about their opacity. Mutual bleeding and merger are transparent no matter what the order between P and Q. 'P feeds Q' and 'P bleeds Q' are opaque under the order $\langle Q \rhd P \rangle$, with underapplication and overapplication of Q, respectively. 'P seeds Q' is opaque in both directions: there is underapplication of Q when $\langle Q \rhd P \rangle$ and there is overapplication of P when $\langle P \rhd Q \rangle$.

(11) Atomic classification of molecular interactions and opacity

Molecules	$\langle P \rhd Q \rangle$	$\langle Q \rhd P \rangle$	Atoms
feeding	transparent	Q: i → e / c __ underapplies in *ki* ↦ *ci*	$\underline{P{+}\text{i}Q}$, $P{+}\text{o}Q$
seeding	P: k → c / __ i overapplies in *ki* ↦ *ci* ↦ *ce*	Q: i → e / c __ underapplies in *ki* ↦ *ci*	$\underline{P{+}\text{i}Q}$, $\underline{Q{-}\text{o}P}$
bleeding	transparent	Q: i → e / k __ overapplies in *ki* ↦ *ke* ↦ *ce*	$P{-}\text{i}Q$, $\underline{P{-}\text{o}Q}$
mutual bleeding	transparent	transparent	$P{-}\text{i}Q$, $Q{-}\text{i}P$
merger	transparent	transparent	$P{+}\text{o}Q$, $Q{+}\text{o}P$

[12] Kiparsky (1976) addresses deficiencies noted by Kaye (1974) in Kiparsky's original 1971 definition of opacity.

[13] We add "that are subject to P" to this first clause because the string CAD matching the structural description of a process of the form $A \rightarrow b \,/\, C \_\_ D$ can just as well be a vacuous input to — and thus not subject to — P. Consider for example final devoicing, typically written [–sonorant] → [–voice] / __ #. Both strings with voiced final obstruents and strings with voiceless final obstruents match the structural description of this process, but only voiced final obstruents are subject to it.

The atoms responsible for each type of opacity are underlined for clarity in each case. The two cases of underapplication of Q are those where P+iQ and $\langle Q \rhd P\rangle$ — that is, where P counterfeeds or counterseeds Q. The atom responsible for overapplication is output-removal. There is one case in each direction: in the case of bleeding, P–oQ and Q overapplies when $\langle Q \rhd P\rangle$ — that is, when P counterbleeds Q — while in the case of seeding, Q–oP and P overapplies when $\langle P \rhd Q\rangle$ — that is, when P seeds Q.

This leads to the following general characterization of opacity under our atomic approach.

(12) Atomic opacity: given two processes X, Y,
   a. Y underapplies if X+iY and $\langle Y \rhd X\rangle$, and
   b. Y overapplies if X–oY and $\langle Y \rhd X\rangle$.

Underapplication obtains when a process X creates inputs for another process Y, but those inputs fail to undergo Y because Y is ordered before X. Overapplication obtains when Y applies to a string, but then X affects that output in such a way that the result is no longer in the output set of Y — to put it another way, there is overapplication when a form has undergone a process but does not ultimately appear to be an output of it.

There are other opaque interactions which are neither underapplication nor overapplication for which this atomic approach provides a formal treatment. One is exemplified by interactions between stress, which may be sensitive to syllable position or weight, and processes disrupting syllable position or weight. The following hypothetical example is due to Rasin (2022), who proposes the term SHIFTING to refer to this type of interaction.[14]

(13) a. P: V → Ø / __ # (apocope)
   b. Q: σ → σ́ / __ σ # (penultimate stress)

The shifting order $\langle P \rhd Q\rangle$ is transparent: *badupi* ↦ *badup* ↦ *bádup*. Opacity arises under the countershifting order, $\langle Q \rhd P\rangle$: *badupi* ↦ *badúpi* ↦ *badúp*. This output unexpectedly exhibits stress on the final syllable, rather than on the penult as Q requires. This example is correctly identified as opaque in our atomic approach. The appropriately stressed input to apocope, *badúpi*, is a member of the output set $O(Q)$ of penultimate stress, but after apocope applies, the form *badúp* is no longer a member of $O(Q)$. In other words, there is output-removal: P–oQ, and because Q is ordered first, it is diagnosed as overapplication opacity according to (12b).

But the phenomenon here is distinct from counterbleeding and seeding, because penultimate stress doesn't exactly "overapply" in the way that a counterbled or seeded process does. The general situation resulting from opaque output removal is that there are surface forms to which a process appears to have applied in an unexpected way. For seeding and counterbleeding, the expected alternative is non-application. In the seeding case in (4), for example, palatalization overapplies in the derivation *ki* ↦ *ci* ↦ *ce* because palatalization has applied even though the expected alternative in the surface environment (__ *e*) would have been for it not to have applied. Similarly in the counterbleeding case in (6): lowering overapplies in the derivation

[14] Baković & Blumenfeld (2023) follow Zwicky (1987), following unpublished work by Don Churma, in using the term *transfusion* for this type of interaction. We use Rasin's term instead here, as it is more amenable to the "counter" prefix.

ki ↦ *ke* ↦ *ce*, because lowering has applied even though the expected alternative in the surface environment (*c* __) would have been for it not to have applied. For countershifting, on the other hand, the expected alternative is application in a different locus. This is the case in (13), with the derivation *badupi* ↦ *badúpi* ↦ *badúp*. We thus refer to countershifting interactions as cases of MISAPPLICATION rather than overapplication. Both of these types of opacity share the fact that a process has applied in an unexpected way, but differ in what the expected alternative would be: non-application in the case of overapplication, and application somewhere else in the case of misapplication. In sum, output removal is the atom responsible for misapplication as well as overapplication. Which of these two phenomena are observed in any given case results from properties of the processes involved that are orthogonal to the atoms we have been discussing.

Joshi & Kiparsky (1979, 2006) and Kiparsky (1982a, 2015) attempt to unify the disjoint definition of opacity into a single diagnostic, inspired by Pāṇini, which we call the Simultaneous Application Condition (SAC). Using "P,$Q(x)$" to denote the effect of applying P and Q simultaneously to x (recall the discussion of direct mapping / simultaneous application in Section 2), the SAC diagnoses a process as opaque as follows.

(14) Simultaneous Application Condition (SAC)

Q is opaque if there exists an input x such that $P(\mathrm{Q}(x)) = P,Q(x) \neq \mathrm{Q}(P(x))$.

As noted in Section 2, the only interactions that are possible under simultaneous application are opaque counterfeeding and counterbleeding because simultaneous application does not permit Q to make use of any changes effected by P. This is in fact the basis of the SAC as stated. The SAC has broader coverage than the original disjoint definition of opacity in (10), in that it covers misapplication-type opacity as well. However, it is still not fully adequate; see Baković & Blumenfeld (2023) for discussion. It suffices to observe here that because it is asymmetrical, the SAC cannot handle a situation where both orders are opaque, as is the case with seeding.

4.2 Output-drivenness

Opacity is a property of an individual process vis-à-vis the grammar of which it is part. A related but distinct concept known as output-drivenness applies at the global level of the grammar (Tesar 2014). It is a property that holds of phonological grammars taken as input-output maps, and is independent of a particular analysis of that map (constraints vs. rules, specific choices of constraints or rules, etc.). Defining this concept requires the following minimal assumptions: a set of representations with a partial order defined on them reflecting some measure of similarity, and a grammar $\mathcal{G}$ that maps input representations to output representations. Tesar (2014:26ff) proposes a precisely formalized process for evaluating similarity between pairs of forms, by comparing sets of DISPARITIES between them. For our purposes we can conceive of similarity as a measure of featural differences between forms. In particular, if the set of featural differences between segments x and y is a subset of the set of featural differences between x and z, then y is more similar to x than z is to x.

Informally, a grammar $\mathcal{G}$ is output-driven if for every input-output mapping /in/ ↦ [out], any form that is more similar to [out] than /in/ is also maps to [out]. "[I]f 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" (Tesar 2014:21). One immediate consequence is that an output-driven grammar $\mathcal{G}$ is IDEMPOTENT: every possible output of

$\mathcal{G}$ should map to itself if used as an input to $\mathcal{G}$, because a form is always more similar to itself than it is to any other form. Another consequence is that $\mathcal{G}$ does not contain SALTATIONS: mappings from input to output that 'skip over' a form that is more similar to the output than the input is.[15] For example, in an output-driven map, the mapping /*a*/ ↦ [*i*] entails the mapping /*e*/ ↦ [*i*], because the featural differences between *e* and *i* (= {[±high]}) are a (proper) subset of the featural differences between *a* and *i* (= {[±high], [±low]}).

Output-drivenness is a distinct notion from process opacity (Tesar 2014:37), but the two are intimately connected. First, we consider the conditions under which *SPE*-style rules define output-driven maps. Then we examine the connection between opacity and output-drivenness in the interaction of processes.

Suppose map M is defined by $A \rightarrow b \;/\; C \_\_ D$, where A, C, D are classes of segments, and b is a set of feature values to be changed in A. There are some sufficient conditions on the output-drivenness of M. First, if b contains a single feature-value pair — that is, if M respects the single change condition in (1a) — then M is output-driven. Indeed, for any mapping $p \mapsto q$, if there is a single feature disparity between p and q, there are by definition no forms that are intermediate in the number of disparities between p and q.

Note that the foregoing presupposes a fully explicit statement of the feature changes effected by M. So, the map $\{a \mapsto i;\ e \mapsto i\}$ cannot be due to a process making a single feature change. A rule stated in the form V → [+high] is either incoherent, because whether it applies to /*a*/ cannot be determined, or it simply does not apply to /*a*/. Instead, the map $\{a \mapsto i;\ e \mapsto i\}$ must be described with two feature changes, V → [+high, –low].[16]

Many reasonable analyses involve processes like this one that change more than one feature at a time. For such processes, there is a more complex sufficient condition on output-drivenness. If b contains more than one feature-value pair, the resulting map M is output-driven if A does not contain a feature-value pair that opposes one of the feature-value pairs contained in b. Suppose that $b = [\alpha_1 F_1, ..., \alpha_n F_n]$. Then a segment p containing $[-\alpha_1 F_1, ..., -\alpha_n F_n]$ would map to segment q containing $[\alpha_1 F_1, ..., \alpha_n F_n]$. Now suppose that A mentions $[-\alpha_i F_i]$. Then M would exclude a segment r identical to p except for $[\alpha_i F_i]$ in place of $[-\alpha_i F_i]$. However, this segment r is more similar to q than p is to q, and thus M must also map it to q if M is output-driven. Thus, if r exists, M is not output-driven. For example, V → [+high, –low] is output-driven, but [V, +low] → [+high, –low] is not: the latter contains the mapping $\{a \mapsto i\}$ but not the mapping $\{e \mapsto i\}$ — that is, it contains a saltation.[17]

While it is possible to ask whether a given process defines an output-driven map, we reiterate that output-drivenness is not a property of processes nor of any particular analysis of a map, but instead that it is a more

[15] The term 'saltation' is due to Hayes & White (2015), in reference to specific cases where one type of segment maps to another, apparently 'jumping over' another type of segment between them, similarity-wise. As we'll see below, the same term can also be fruitfully used to characterize opaque interactions involving output removal.

[16] Note that evidence for a map like $\{a \mapsto i;\ e \mapsto i\}$ does not necessarily entail abandoning the single change condition on processes because the map can be broken down into two processes, each making a single feature change and one feeding the other. There are two ways to accomplish this. One way is for the first process to be V → [–low], changing *a* to *e*, which feeds the second process, [V, –low] → [+high], changing *e* (whether underlying or itself from *a*) to *i*. Another way is for the first process to be V → [+high] and to assume that it is allowed to apply to *a*, temporarily resulting in an articulatorily impossible [+high, +low] vowel; this then feeds a second process that dispenses with the impossible, [V, +high] → [–low].

[17] Note that the condition given in the text is sufficient, but not always necessary. For example, [V, –high] → [+high, –low] is still output-driven, assuming that [+high, +low] segments are impossible.

theory-neutral notion that holds of any map as long as a partial order based on similarity is defined on the strings over which the map operates. It is thus possible to ask not only whether a given process (*qua SPE*-style rewrite rule) defines an output-driven map, but also whether compositions (= serial orderings) of such processes do. In particular, we are interested in how output-drivenness of compositions of processes relates to their opacity. The following table shows the cross-classification of output-drivenness and opacity of the composition $\langle P \rhd Q \rangle$.

(15) Output-drivenness and opacity of compositions of output-driven P and Q

	Transparent	Opaque
Output-driven	yes	no
Non-output-driven	yes	yes

Transparent output-driven compositions are of course possible, trivially, when P and Q do not interact. Otherwise, the relationship between the two concepts is one of stringency: every opaque composition is non-output-driven, but some non-output-driven compositions may be transparent. The following example illustrates the latter point. P raises all [–back] vowels to *i*, and Q backs the mid vowel *e*. If $\langle Q \rhd P \rangle$, the composed map is effectively saltatory: while *a* maps to *i*, the mid vowel *e*, which is more similar to *i* than *a* is, maps to ɤ.

(16) Vowel raising (P) interacts with mid vowel centralization (Q)

P: [V, –back] → [+high, –low] {a ↦ i; e ↦ i}

Q: [–high, –low] → [+back] {e ↦ ɤ}

Still, this interaction is transparent: P and Q mutually bleed each other on the input *e*. Regardless of the order between P and Q, there is no output that is subject to either process (that is, no underapplication) and no output that has mysteriously undergone either process (no overapplication or misapplication).

Conversely, all opaque compositions of two processes are non-output-driven (see Magri 2018a for a more detailed discussion of the topic of this section). Recall that opacity has two atomic sources: input provision and output removal. All cases of input-provision opacity — where P counterfeeds or counterseeds Q — involve underapplication, i.e. a situation where a potential input to Q does not undergo Q. This situation is inevitably non-idempotent: there are forms which, if underlying, undergo P, but the same forms, when derived, map to themselves. Therefore the composed map is not output-driven. In fact, the non-output-drivenness of $\langle Q \rhd P \rangle$ if P+iQ holds regardless of the output-drivenness of P and Q individually.

The case of output-removal opacity — P–oQ and $\langle Q \rhd P \rangle$ — is more complex but it is also true that such compositions are not output-driven. To help manage the complexity, we introduce a new schematic notation to keep track of changes effected by P and Q. Suppose, first, that P and Q each changes the value of a distinct single feature; these features could be on the same segment or on different segments. Let us use "0" and "1" to refer to the feature values, and linear position to refer to the features themselves: P's feature on the left, Q's feature on the right. The schematic forms under consideration are thus "00", "01", "10", "11". This notation makes the similarity distance amongst the forms perspicuous, as can be appreciated by the diagram in (17), where wavy lines connect two forms that differ from each other by only one featural disparity.

(17)
$$\begin{array}{ccc} 00 & \rightsquigarrow & 10 \\ \wr & & \wr \\ 01 & \rightsquigarrow & 11 \end{array}$$

One necessary condition for a map to be output-driven is as follows: if $\{00 \mapsto 11\}$, then $\{10 \mapsto 11\}$ and $\{01 \mapsto 11\}$ as well, because both 01 and 10 are more similar to 11 than 00 is.

Now suppose Q–oP. Then, minimally, the following two mappings must exist: $P = \{00 \mapsto 10\}$, $Q = \{10 \mapsto 11\}$, and furthermore, 11 must not be an output of P. (Whether Q also maps 00 to 01 is immaterial to Q–oP.)

(18)
$$\begin{array}{ccc} & P & \\ 00 & \rightarrow & 10 \\ \wr & & \downarrow Q \\ 01 & \rightsquigarrow & 11 \end{array}$$

The diagram in (18) makes it clear that as long as $\langle P \rhd Q \rangle$, it is the case that $\{00 \mapsto 11\}$. As noted above, for an output-driven map it must also be the case that $\{10 \mapsto 11\}$, which indeed holds here, as well as $\{01 \mapsto 11\}$, which does not, because that mapping is incompatible with Q output-removing P. Therefore, Q–oP entails that the order $\langle P \rhd Q \rangle$ is not output-driven in this scenario, because it contains a saltation: 00 maps to 11, but 01 does not.

This argument does not depend on the difference between "0" and "1" being that of a single feature value. In fact, as long as P and Q do not undo each other's disparities, the set of disparities in the map $\{00 \mapsto 11\}$ where both processes have applied is a strict superset of the sets of disparities in the strings $\{00 \mapsto 10\}$ and $\{00 \mapsto 01\}$ where at most only of the two processes has applied.

To make this argument more concrete we invoke the countershifting example from Section 4.1, (13): the input *badupi* maps to the output *badúp*, with misapplication of penultimate stress due to the subsequent application of apocope. In (19) below, the first feature's 0 value indicates absence of stress on the penultimate vowel *u*, while 1 indicates presence of stress on that vowel. The second feature's value reflects presence (0) or absence (1) of the final vowel *i*. There is indeed saltation here, just as in (18): the form *badup* (= 01) is more similar to the output *badúp* (= 11) than is the input *badupi* (= 00), but *badup* does not also map to *badúp*.[18]

(19)
$$\begin{array}{ccc} & P & \\ \underset{badupi}{00} & \rightarrow & \underset{badúpi}{10} \\ \wr & & \downarrow Q \\ \underset{badup}{01} & \rightsquigarrow & \underset{badúp}{11} \end{array}$$

Assuming that input provision and output removal are the only atomic sources of opacity, this concludes the argument that all opaque compositions of two processes result in non-output-driven maps. Taking stock

[18] Recall that input *badup* in this example maps to a form outside of the square in (19): it maps to *bádup*, with penultimate stress on the *a*. Recall that this is one of the key ways in which countershifting differs from counterbleeding and seeding: in both of those cases, the relevant intermediate form maps to itself.

of this section, it is evident that in a specific sense output-drivenness is a more restrictive notion than opacity: all map interactions that are opaque are non-output-driven, but the converse is not true. The non-output-drivenness of map compositions holds under very weak assumptions about the nature of P and Q.

Furthermore, these observations make it possible to attempt another unified definition of opacity, following Magri (2023:15), this time as a global property of a grammar rather than in the more usual way as a local property of a map embedded within a grammar.

(20) Magri's definition of opacity

> Grammar $\mathcal{G}$ is opaque to the extent that there are maps $x \mapsto z$ such that there exists a phonotactically licit form y that is intermediate between x and z, and different from z.

Essentially, this definition covers the input provision (underapplication) and output removal (overapplication) cases of non-output-drivenness. However, Magri's definition cannot account for the opacity of countershifting — misapplication with output removal, as in (19) above — because there is no licit form (different from *badúp*) that is intermediate between *badupi* and *badúp*. The only form that could potentially fit the bill is *bádup* (from input *badup*), but this form is not intermediate between *badupi* and *badúp*: it differs from the first in terms of stress (*á/a*) and apocope (ø/*i*), and from the second in terms of stress in two different loci (*á/a* and *u/ú*).

5 Blocking interactions

In the preceding sections we have assumed that the only factor relevant to the application of a process P to a string s is set membership: if $s \in I(P)$ and it's P's turn to apply in the course of the derivation, then P applies to s. Conversely, if it's P's turn to apply but it fails to apply to s, it must be because $s \notin I(P)$.

Phonological theory has long recognized that there is yet another way that a process P can be prevented from applying to a string to which P is expected to apply because it is in P's input set: P can be BLOCKED, according to some non-ordering type of intervention, either by another process or by a constraint. Because such cases result in the surface realization of strings in the input set of P, they involve underapplication opacity in the sense defined in (10a) (Baković 2011). In order to accommodate them within the atomic approach in (12), we would have to add conditions beyond process ordering. This would take us too far afield here, so we simply proceed with some examples: two types of blocking by process in Sections 5.1 and 5.2, and blocking by constraint in Section 5.3.

5.1 Disjunctive application

Thus far we have discussed interactions between processes P, Q that are sensitive to their order of application: either $\langle P \rhd Q \rangle$ or $\langle Q \rhd P \rangle$. This type of application is referred to as CONJUNCTIVE in *SPE*: regardless of their order, *both P and Q* get a crack at the derivation, though it may sometimes be too early or too late for one or the other to apply non-vacuously. As noted in Section 2, a different type of application referred to as DISJUNCTIVE was also entertained in *SPE*. In this type of case, *either P or Q* gets a crack at the derivation, but not both. We focus here on the type of example that could not be accommodated by *SPE*'s notational conventions and that motivated the introduction of a broader principle known as the Elsewhere Condition (Kiparsky 1973).

There are two types of example that are relevant to this discussion. One type is exemplified by the interaction between two processes of English, referred to here simply as Shortening and Lengthening.[19] Shortening generally shortens vowels in the heads of branching feet, and is responsible for alternations such as *di*(*vī**n**e*) ~ *di*(*vĭ**n**i*)*ty*, (*nā*)*ture* ~ (*nă**t**u*)*ral*, *ob*(*scē**n**e*) ~ *ob*(*scĕ**n**i*)*ty*, and so on. Lengthening more specifically lengthens *non-high* vowels in the heads of branching feet *when the non-head is high, front, and in hiatus*, and is responsible for alternations such as (*grăd**u*)*al* ~ (*grā**d**i*)*ent* and (*remĕ*)*dy* ~ *re*(*mē**d**i*)*al*. The question at hand is why Shortening, which is applicable in a strictly more general context than Lengthening is, does not apply in these latter cases.

One possible answer is simply that Shortening precedes Lengthening: regardless of whether the relevant vowel is underlyingly long or short, the prior effect of Shortening is undone by the later effect of Lengthening. This is most notable in the derivation of a form like (*grā**d**i*)*ent*, which has an underlying long vowel (cf. *grā**d**e*): after footing, the derivation proceeds as follows: (*grā**d**i*)*ent* ↦ (*grăd**i*)*ent* ↦ (*grā**d**i*)*ent*. This type of derivation, where one process applies only to be undone by another, has been called a DUKE OF YORK DERIVATION (Pullum 1976).

Kiparsky (1973) proposes a different answer: Lengthening *blocks* Shortening, via disjunctive application by the Elsewhere Condition. Stated in our formal terms, the Elsewhere Condition requires that two processes P and Q apply disjunctively, with P blocking Q, iff (a) $I(P)$ is a proper subset of $I(Q)$ and (b) $O(P)$ and $O(Q)$ are disjoint. These two subconditions hold here, with P = Lengthening and Q = Shortening: (a) the input set of Shortening (= the head of a branching foot) properly includes that of Lengthening (= the head of a branching foot meeting several additional conditions), and (b) the output sets of the two processes are disjoint: Shortening results in a short vowel in the head of the branching foot while Lengthening results in a long vowel in that same position.

Given that there is a viable conjunctive analysis of this type of example, the question becomes why disjunctive application is necessary. One answer is that there is something suspicious about Duke of York derivations, either because these derivations involve 'superfluous steps' that are undesirable in principle (Halle & Idsardi 1997, 1998) or because there is (purportedly) no evidence for the fleeting intermediate step (Kiparsky 1973, McCarthy 2003b; see also Norton 2003). Whether or not this suspicion is valid, note that the Elsewhere Condition carves out only a subset of such potential derivations: those where the input set of one process properly includes that of the other. Duke of York derivations are still possible in cases where the output sets of two processes are disjoint but their input sets merely *overlap* as opposed to being in a proper inclusion relationship.[20]

Another answer comes in the form of the second type of example that motivates the Elsewhere Condition, exemplified by the interaction of two processes of Diola Fogny (Sapir 1965). One process place-assimilates nasals to following stops (Assimilation), and the other process deletes pre-consonantal consonants (Deletion). As in the English case, these processes meet the two subconditions of the Elsewhere Condition, with

[19] The most immediately relevant discussions of these rules and their disjunctive application are found in Kenstowicz (1994), Halle (1995), Prince (1997), and Baković (2013). The examples cited are purely orthographic, with alternating vowels in boldface type and the long/short distinction indicated with a macron (e.g. ***ā***) or a breve (e.g. ***ă***), respectively.

[20] This can be appreciated with an example from Nuuchahnulth, previously known as Nootka (Sapir & Swadesh 1939, Pullum 1976, McCarthy 1999, Kim 2003, Stonham & Kim 2008). Dorsals labialize after round vowels (/*nu:**k***+*i:s*/ → [*nu:**k**ʷi:s*] 'canoe song') and delabialize pre-consonantally (/*ʦapχ**ʷ***+*sa:p*/ → [*ʦapχsa:p*] 'to boil'); in overlapping contexts, delabialization wins out: /*mamu:**k***+*ʃitɬ*/ → *mamu:**k**ʷʃitɬ* → [*mamu:**k**ʃitɬ*] 'work (perfective)'.

Assimilation as the more specific process P and Deletion as the more general process Q: (a) the input set of Deletion properly includes that of Assimilation, and (b) the output sets of the two processes are disjoint. Assimilation thus blocks Deletion: *le**tk**udʒaw* ↦ *le**k**udʒaw* 'they won't go', but *niga**mg**am* ↦ *niga**ŋg**am*, **niga**g**am* 'I judge'.

Because Deletion is technically applicable both to the input and to the output of Assimilation, a conjunctive alternative to this analysis must somehow ensure that Deletion only applies when Assimilation does not. One way to accomplish this is to have Deletion apply to (a) pre-consonantal *non*-nasals and (b) pre-*continuant* nasals (Kiparsky 1973:97; Baković 2013:27). Another, more promising alternative is to make Assimilation somehow bleed Deletion, a path pursued by Itô (1986): coda consonants are initially 'unlicensed', Assimilation 'licenses' coda nasals by place-linking them to following stops, and Deletion targets all and only unlicensed consonants.

Note that there is an atomic difference between the two types of example discussed here. In the English case, Lengthening and Shortening both input-provide and output-remove each other — a type of 'mutual seeding' interaction more complex than any of the ones we've encountered thus far, with four atoms instead of just two. In the Diola Fogny case, Deletion both input-removes and output-removes Assimilation, which is like bleeding except that both orders result in Deletion deleting pre-consonantal nasals whether or not they have assimilated. The former type of case is tailor-made for a conjunctive Duke of York analysis, while the latter requires some modification of (the interpretation of) the processes to reverse the bleeding relationship between them.

5.2 Nonderived environment blocking

Kiparsky (1976) identifies another type of case in which a process is blocked from applying to a string in its input set. In Finnish, a process (Assibilation) changes *t* to *s* before *i*, but only if that *i* is introduced in the course of the derivation, either morphologically (e.g., a suffix beginning with *i* is attached to a stem ending in *t*; /*halut+i*/ → [*halusi*] 'wanted') or phonologically, the case of most immediate interest to us here.

Another process raising word-final *e* to *i* successfully feeds Assibilation (/*vete*/ → *veti* → [*vesi*] 'water'), but an otherwise comparable form in which the *ti* sequence is underlying (and tautomorphemic) fails to assibilate: /*äiti*/ → [*äiti*] 'mother'. There is some kind of interaction between Raising and Assibilation here, but it isn't due to serial ordering: we can say that Raising (precedes and) feeds Assibilation, but in that case Assibilation should also be applicable to an underlying *ti* sequence, not only when such a sequence results from Raising.

This situation is often referred to as NONDERIVED ENVIRONMENT BLOCKING: Assibilation is only permitted to apply if its context of application has been derived by another process, or conversely is blocked from applying when its context has *not* been derived.[21] Many proposals have been made to address the problem of nonderived environment blocking, too many to do justice to here, so again we refer our readers to another modest selection: Kiparsky (1976, 1982b, 1993), Mascaró (1976), Burzio (2000), Inkelas (2000), Łubowicz

[21] Note that this situation is roughly the inverse of counterfeeding. Here, Assibilation only applies to derived forms, not to underlying forms, whereas a counterfed process only applies to underlying forms, not to derived forms. The main difference is that counterfeeding is assumed to be determined on a case-by-case basis (P may be fed by Q but counterfed by R) whereas nonderived environment blocking is assumed to be all-or-nothing (P is blocked in *all* nonderived environments).

(2002), McCarthy (2003a), van Oostendorp (2007), Kula (2008), Wolf (2008), Rasin (2016, 2023), and Chandlee (2021).

5.3 Blocking by constraint

Processes can also be blocked from applying to strings in their input set by the intervention of CONSTRAINTS, which typically proscribe particular output structures or configurations.[22] Kisseberth (1970) is the *locus classicus* of the argument that constraints curtailing the application of processes should be a part of phonological theory. In that work, and in much subsequent work in the primarily rule-based literature, a process P is blocked from applying to a string x by a constraint C if $P(x) = y$, $y \neq x$, and y but not x violates the proscription of C.

One area in which blocking by constraint has flourished as an explanatory device is in the analysis of vowel harmony; an influential albeit unpublished paper on this topic is Kiparsky (1981). Vowels that do not undergo harmony (so-called 'neutral' vowels) are ones whose expected harmonic counterparts are proscribed by a constraint that is also held to be responsible for their absence from the underlying segmental inventory. The principle that processes do not output segments absent from the underlying inventory, known as STRUCTURE PRESERVATION, in this case acts as a constraint on the outputs of harmony.[23]

The increasing invocation of constraints as descriptive and explanatory devices throughout the 1980s led to the development of more purely constraint-based theories, most significantly OT (Prince & Smolensky 1993/2004). In this theory, an inclusive set of candidate outputs for a given input are compared in parallel across a set of constraints, presumed universal; some of these are MARKEDNESS constraints, proscribing output structures or configurations like their predecessors, while others are FAITHFULNESS constraints, proscribing disparities between input and output. The constraints are violable and interact with each other via language-particular ranking: an output candidate x is OPTIMAL — i.e., the *actual* output — if, for every comparison with an alternative candidate y, x performs better than y on the highest-ranked constraint that distinguishes them.[24]

Among the advantages of this type of constraint interaction is that various types of blocking as well as triggering by constraint can be straightforwardly and insightfully analyzed. Consider again the case of neutral vowels in vowel harmony, for example, following Baković (2000). Harmony for (say) the feature [±ATR] is there proposed to be driven by a markedness constraint AGREE[ATR], proscribing adjacent vowels with different values of [±ATR], ranked above a faithfulness constraint IDENT[ATR], proscribing changes in [±ATR] from input to output. A common neutral vowel in languages with ATR harmony is the [–ATR, +low] vowel /*a*/, the expected harmonic counterpart of which would be [+ATR, +low]. If another markedness constraint C proscribing this feature combination is itself higher-ranked than AGREE[ATR], then harmony obtains *except when* C is at stake. Note that the position of C in this constraint hierarchy also guarantees the absence of the proscribed vowel from the segmental inventory: given a hypothetical [+ATR, +low]

[22] A long list of antecedent works on constraints in phonological theory is cited in Prince & Smolensky (1993/2004); see also the more recent historical discussion in Calabrese (2022).

[23] An alternative is to initially allow these vowels to undergo harmony and then to undo the non-structure-preserving result with an 'absolute neutralization' rule in Duke-of-York fashion and re-establishing structure preservation (Vago 1976). In this case, the absolute neutralization rule can be understood to be *triggered* by structure-preserving constraints.

[24] This remarkably concise and accurate definition of optimality is originally due to Grimshaw (1997).

vowel in an input, the ranking of C above IDENT[ATR] — obtained by transitivity via AGREE[ATR] — provokes a change from [+ATR] to [–ATR]. Another possibility is that hypothetical [+ATR, +low] vowels in the input change from [+low] to [–low], which is predicted if IDENT[ATR] is itself higher-ranked than IDENT[low]. Together with the previous ranking, this now predicts that harmony will always obtain but that /*a*/ will alternate with a [+ATR, –low] vowel.[25]

Both of these predicted patterns are consistent with structure preservation and avoid what has been referred to as the 'duplication problem' (Kenstowicz & Kisseberth 1977; cf. Paster 2013). This type of analysis extends naturally to more complex patterns in which a single constraint is effectively responsible for blocking and/or triggering more than one of what would otherwise be characterized as separate processes, providing an account of the 'conspiracies' originally adduced by Kisseberth (1970); see e.g. Casali (1996, 1997) and Pater (1999).

As alluded to in the final paragraph of Section 2, the lack of a direct analogue of a 'process' in OT makes it difficult to assess what the theory does and does not predict in terms of 'process interaction' as we have circumscribed this term in this chapter, and more importantly *how* it does and doesn't predict it.[26] The work cited in that paragraph takes that challenge on, though focusing mainly on the classical typology in (2) and in particular on the distinction between opaque and transparent interactions. One productive approach to understanding what OT predicts about process interaction is thus via the connection between opacity and output-drivenness (recall Section 4.2): since OT grammars, under some assumptions about constraints and the generation of output candidates, can only describe output-driven languages, opacity is out of their reach.

6 Concluding remarks

Our goals in this chapter have been to document how phonological processes of the form $A \rightarrow b \,/\, C \_\_ D$ have been proposed to interact with each other in generative phonology and to explore different ways of classifying different types of interactions, both from a local pairwise perspective and from the more global perspective of the phonological grammar as a whole. We also introduced our own formal approach to classifying process interactions, as 'molecules' composed of 'atoms' that precisely define how one process potentially affects the input or the output of another. In addition to properly distinguishing different types of interaction that had been classically lumped together, a promising result of this atom-based approach is its formalization of opacity in (12). Given that opaque phenomena of various types are perennially invoked in phonological theory comparisons, it is especially useful to have a formal definition — albeit one framed in terms of a particular model, involving serially-ordered processes like *SPE* — against which to weigh those comparisons.

[25] This prediction is borne out in the harmony patterns of Maasai and Turkana (Eastern Nilotic; see Baković 2000, Ch. 4).

[26] There are two significant models that add some kind of serial ordering to OT in limited and principled ways: Harmonic Serialism (McCarthy 2000, 2010) and Stratal (or 'Derivational') OT (Bermúdez-Otero 1999, 2010/2022; Kiparsky 2000, 2015, in press; Rubach 2000, 2020). Other notable efforts to build serialism and/or more process-like machinery into OT include sympathy (McCarthy 1999), targeted constraints (Wilson 2000, 2001), comparative markedness (McCarthy 2003), procedural constraints (Blumenfeld 2006), and OT with candidate chains (McCarthy 2007; Wolf 2008, 2011).

References

Anderson, Stephen. R. 1969. *West Scandinavian Vowel Systems and the Ordering of Phonological Rules*. Doctoral dissertation, MIT. http://dspace.mit.edu/handle/1721.1/12964

Anderson, Stephen R. 1974. *The Organization of Phonology*. Academic Press.

Anderson, Stephen R. 1975. On the interaction of phonological rules of various types. *Journal of Linguistics* 11, 39–62. doi:10.1017/S0022226700004266

Baković, Eric. 2000. *Harmony, Dominance, and Control*. Doctoral dissertation, Rutgers University. doi:10.7282/T3TQ60BJ

Baković, Eric. 2007. A revised typology of opaque generalizations. *Phonology* 24. 217–259. doi:10.1017/S0952675707001194

Baković, Eric. 2011. Opacity and ordering. In John A. Goldsmith, Jason Riggle & Alan C. L. Yu (eds.), *The Handbook of Phonological Theory* (2nd ed.), 40–67. Malden, MA: Wiley Blackwell. doi:doi:10.1002/9781444343069.ch2

Baković, Eric. 2013. *Blocking and Complementarity in Phonological Theory*. London: Equinox.

Baković, Eric & Lev Blumenfeld. 2023. A formal typology of process interactions. *Phonological Data and Analysis* (to appear).

Bermúdez-Otero, Ricardo. 1999. *Constraint Interaction in Language Change: Quantity in English and Germanic* [*Opacity and Globality in Phonological Change*]. Doctoral dissertation, University of Manchester & Universidad de Santiago de Compostela.

Bermúdez-Otero, Ricardo. 2010/2022. Stratal Optimality Theory: an overview. http://www.bermudez-otero.com/Stratal_Optimality_Theory.htm, accessed January 2024.

Blumenfeld, Lev. 2006. *Constraints on Phonological Interactions*. Doctoral dissertation, Stanford University.

Burzio, Luigi. 2000. Cycles, non-derived-environment blocking, and correspondence. In Joost Dekkers, Frank van der Leeuw, & Jeroen van de Weijer (eds.), *Optimality Theory: Phonology, Syntax, and Acquisition*, 47–87. Oxford: Oxford University Press. doi:10.1093/oso/9780198238430.003.0002

Calabrese, Andrea. 2022. Historical notes on constraint-and-repair approaches. In B. Elan Dresher & Harry van der Hulst (eds.), *The Oxford History of Phonology*, 530–550. Oxford University Press. doi:10.1093/oso/9780198796800.003.0025

Casali, Roderic F. 1996. *Resolving Hiatus*. Doctoral dissertation, UCLA. https://linguistics.ucla.edu/images/stories/casali.1996.pdf

Casali, Roderic F. 1997. Vowel elision in hiatus contexts: Which vowel goes? *Language* 73.3, 493–533. doi:10.2307/415882

Chafe, Wallace. 1968. The ordering of phonological rules. *International Journal of American Linguistics* 34, 115–136. https://www.jstor.org/stable/1263517

Chandlee, Jane. 2021. Nonderived environment blocking and input-oriented computation. *Evolutionary Linguistic Theory* 3.2, 129–153. doi:10.1075/elt.00031.cha

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

Green, Anthony D. 2007. *Phonology Limited*. Potsdam: Universitätsverlag.

Grimshaw, Jane. 1997. Projection, heads, and optimality. *Linguistic Inquiry* 28.3, 373–422. https://www.jstor.org/stable/4178985

Halle, Morris. 1995. A letter from Morris Halle: Comments on Luigi Burzio's "The rise of Optimality Theory". *GLOT International* 1.9/10, 27–28.

Halle, Morris & William J. Idsardi. 1998. A response to Alan Prince's letter. *GLOT International* 3, 1 & 22.

Hayes, Bruce & James White. 2015. Saltation and the P-map. *Phonology* 32, 267–302. doi:10.1017/S0952675715000159

Hooper[Bybee], Joan. 1976. *An Introduction to Natural Generative Phonology*. Academic Press.

Idsardi, William. 2000. Clarifying opacity. *The Linguistic Review* 17, 337–350. doi:10.1515/tlir.2000.17.2-4.337

Inkelas, Sharon. 2000. Phonotactic blocking through structural immunity. In Barbara Stiebels & Dieter Wunderlich (eds.), *Lexicon in Focus*, 7–40. Berlin: Akademie Verlag.

Joshi, S.D. & Paul Kiparsky. 1979. Siddha and asiddha in Pāṇinian phonology. In Dan Dinnsen (ed.), *Current Approaches to Phonological Theory*, 223–250. Bloomington, IN: Indiana University Press. https://publish.iupress.indiana.edu/read/current-approaches-to-phonogical-theory/section/c5d8e269-ed37-446f-b6ce-791e4d746895

Joshi, S.D. & Paul Kiparsky. 2005. The extended *siddha*-principle. *Annals of the Bhandarkar Oriental Research Institute* 86, 1–26. https://www.jstor.org/stable/41692378

Kaye, Jonathan. 1974. Opacity and recoverability in phonology. *Canadian Journal of Linguistics* 19, 134–149. doi:10.1017/S0008413100007726

Kenstowicz, Michael. 1994. *Phonology in Generative Grammar*. Oxford: Blackwell.

Kenstowicz, Michael & Charles W. Kissebert. 2022. Phonological derivation in early generative phonology. In B. Elan Dresher & Harry van der Hulst (eds.), *The Oxford History of Phonology*, 419–493. Oxford University Press. doi:10.1093/oso/9780198796800.003.0020

Kenstowicz, Michael & Charles W. Kisseberth. 1977. *Topics in Phonological Theory*. Academic Press.

Kenstowicz, Michael J. & Charles W. Kisseberth. 1979. *Generative Phonology: Description and Theory*. San Diego, CA: Academic Press.

Kim, Eun-Sook. 2003. *Theoretical issues in Nuu-chah-nulth phonology and morphology*. Doctoral dissertation, University of British Columbia. doi:10.14288/1.0091365

Kiparsky, Paul. 1968. Linguistic universals and linguistic change. In Emmon Bach & Robert T. Harms (eds.), *Universals in Linguistic Theory*, 170–202. New York: Holt, Reinhart, and Winston. doi:10.1515/9783111666242.13 [Reprinted in *Explanation in Phonology*, 13–55. Dordrecht: Foris, 1982.]

Kiparsky, Paul. 1971. Historical linguistics. In William O. Dingwall (ed.), *A Survey of Linguistic Science*, 576–642. College Park: University of Maryland Linguistics Program. [Reprinted in *Explanation in Phonology*, 57–80. Dordrecht: Foris, 1982.]

Kiparsky, Paul. 1973. "Elsewhere" in phonology. In Stephen R. Anderson & Paul Kiparsky (eds.), *A Festschrift for Morris Halle*, 93–106. New York: Holt, Reinhart, and Winston.

Kiparsky, Paul. 1976. Abstractness, opacity, and global rules. In Andreas Koutsoudas (ed.), *The Application and Ordering of Grammatical Rules*, 160-186. The Hague: Mouton.

Kiparsky, Paul. 1981. Vowel harmony. Unpublished ms., MIT.

Kiparsky, Paul. 1982a. Lexical morphology and phonology. In Linguistic Society of Korea (ed.), *Linguistics in the Morning Calm: Selected Papers from SICOL-1981*, 3–91. Seoul: Hanshin Publishing Company.

Kiparsky, Paul. 1982b. Some theoretical problems in Pāṇini's grammar. Poona: Bhandarkar Oriental Research Institute.

Kiparsky, Paul. 1993. Blocking in nonderived environments. In Sharon Hargus & Ellen Kaisse (eds.), *Studies in Lexical Phonology*, 277–314. San Diego: Academic Press. doi:10.1016/B978-0-12-325071-1.50016-9

Kiparsky, Paul. 2000. Opacity and cyclicity. *The Linguistic Review* 17, 351-365. doi:10.1515/tlir.2000.17.2-4.351

Kiparsky, Paul. 2015. Stratal OT: A Synopsis and FAQ. In Yuchau E. Hsiao & Lian-Hee Wee (eds.), *Capturing Phonological Shades Within and Across Languages*, 2–44. Newcastle upon Tyne: Cambridge Scholars Library.

Kiparsky, Paul. In press. *Paradigms and Opacity*. Stanford: CSLI Publications.

Kisseberth, Charles. 1970. On the functional unity of phonological rules. *Linguistic Inquiry* 1, 291–306. https://www.jstor.org/stable/4177568

Koutsoudas, Andreas, Gerald Sanders & Craig Noll. 1974. The application of phonological rules. *Language* 50.1, 1–28. doi:10.2307/412007

Kula, Nancy C. 2008. Derived environment effects: A representational approach. *Lingua* 118, 1328–1343. doi:10.1016/j.lingua.2007.09.011

Lee, Minkyung. 1999. A case of sympathy in Javanese affixation. In Karen Baertsch & Daniel A. Dinnsen (eds.), *Optimal Green Ideas in Phonology*, 31–36. Bloomington, IN: IULC Publications.

Lee, Minkyung. 2007. OT-CC and feeding opacity in Javanese. *Studies in Phonetics, Phonology and Morphology* 13.2, 333–350.

Łubowicz, Anna. 2002. Derived environment effects in Optimality Theory. *Lingua* 112, 243–280. doi:10.1016/S0024-3841(01)00043-2

Magri, Giorgio. 2018a. Idempotency, output-drivenness and the faithfulness triangle inequality: Some consequences of McCarthy's (2003) categoricity generalization. *Journal of Logic, Language, and Information* 27, 1–60. doi:10.1007/s10849-017-9256-0

Magri, Giorgio. 2018b. Output-drivenness and partial phonological features. *Linguistic Inquiry* 49.3, 577–598. doi:10.1162/ling_a_00283

Magri, Giorgio. 2023. Extensional opacity à la Tesar. Handout, LSA Summer Institute, UMass Amherst.

Mascaró, Joan. 1976. *Catalan Phonology and the Phonological Cycle*. Doctoral dissertation, MIT. http://dspace.mit.edu/handle/1721.1/7582

McCarthy, John J. 1999. Sympathy and phonological opacity. *Phonology* 16.3, 331–399. doi:10.1017/S0952675799003784

McCarthy, John J. 2003. Comparative markedness. *Theoretical Linguistics* 29, 1–51. doi:10.1515/thli.29.1-2.1

McCarthy, John J. 2007. *Hidden Generalizations: Phonological Opacity in Optimality Theory*. London: Equinox.

Norton, Russell James. 2003. *Derivational Phonology and Optimality Phonology: Formal Comparison and Synthesis*. Doctoral dissertation, University of Essex. ROA- 613, Rutgers Optimality Archive, http://roa.rutgers.edu/.

Oostendorp, Marc van. 2007. Derived environment effects and consistency of exponence. In Sylvia Blaho, Patrik Bye & Martin Krämer (eds.), *Freedom of Analysis?*, 123–148. Berlin: Mouton de Gruyter. doi:10.1515/9783110198591.123

Paster, Mary. 2013. Rethinking the 'duplication problem'. *Lingua* 126, 78–91. doi:10.1016/j.lingua.2012.11.015

Pater, Joe. 1999. Austronesian nasal substitution and other NÇ effects. In René Kager, Harry van der Hulst & Wim Zonneveld (eds.), *The Prosody-Morphology Interface*, 310–343. Cambridge: Cambridge University Press.

Prince, Alan. 1997. Elsewhere & otherwise. *GLOT International* 2.1, 23–24. Expanded and reformatted as ROA-217, Rutgers Optimality Archive, http://roa.rutgers.edu.

Prince, Alan & Paul Smolensky. 2004[1993]. *Optimality Theory: Constraint Interaction in Generative Grammar*. Malden: Blackwell. ROA-537, Rutgers Optimality Archive, http://roa.rutgers.edu.

Pruitt, Kathryn. 2023. Serialism and opacity in phonological theory. *Annual Review of Linguistics* 9, 497–517. doi:10.1146/annurev-linguistics-031220-120748

Pullum, Geoffrey K. 1976. The Duke of York gambit. *Journal of Linguistics* 12, 83–102. https://www.jstor.org/stable/4175335

Rasin, Ezer. 2015. A rule-ordering theory of blocking in nonderived environments. *Proceedings of AMP 2014*. doi:10.3765/amp.v3i0.3689

Rasin, Ezer. 2022. Shifting interactions and countershifting opacity: A note on opacity in harmonic serialism. *Linguistic Inquiry* 53.4, 836–851. doi:10.1162/ling a 00430

Rasin, Ezer. 2023. Morpheme structure constraints solve three puzzles for theories of blocking in nonderived environments. *Linguistic Inquiry*. doi:10.1162/ling_a_00514

Reiss, Charles. 2022. Priority union and feature logic in phonology. *Linguistic Inquiry* 53.1, 199–209. doi:10.1162/ling_a_00400

Rubach, Jerzy. 2000. Glide and glottal stop insertion in Slavic languages: A DOT analysis. *Linguistic Inquiry* 31, 271-317. doi:10.1162/002438900554361

Rubach, Jerzy. 2020. Chain effects in Kurpian. *Journal of Linguistics* 56, 663-690. doi:10.1017/S002222671900015X

Sanders, Nathan. 2003. *Opacity and Sound Change in the Polish Lexicon*. Doctoral dissertation, UC Santa Cruz. doi:10.7282/T3319TPP

Sapir, Edward & Morris Swadesh. 1939. *Nootka texts, tales and ethnological narratives, with grammatical notes and lexical materials*. Philadelphia & Baltimore, MD: Linguistic Society of America.

Stonham, John & Eun-Suk Kim. 2008. Labialization in Nuuchahnulth. *Journal of the International Phonetics Association* 38.1, 25-50. doi:10.1017/S0025100308003253

Tesar, Bruce. 2014. *Output-Driven Phonology: Theory and Learning*. Cambridge: Cambridge University Press.

Vago, Robert M. 1976. Theoretical implications of Hungarian vowel harmony. *Linguistic Inquiry* 7.2, 243–263. https://www.jstor.org/stable/4177921

Vago, Robert M. 1977. In support of extrinsic ordering. *Journal of Linguistics* 13, 25–41. doi:10.1017/S0022226700005181

Vaux, Bert. 2008. Why the phonological component must be serial and rule-based. In Bert Vaux & Andrew Nevins (eds.), *Rules, Constraints, and Phonological Phenomena*. Oxford: Oxford University Press.

Wang, William S-Y. 1969. Competing changes as a cause of residue. *Language* 45.1, 9–25. doi:10.2307/411748

Wilson, Colin. 2000. *Targeted Constraints: An Approach to Contextual Neutralization in Optimality Theory*. Doctoral dissertation, Johns Hopkins University.

Wilson, Colin. 2001. Consonant cluster neutralization and targeted constraints. *Phonology* 18.1, 147-197. doi:10.1017/S0952675701004043

Wolf, Matthew. 2008. *Optimal Interleaving: Serial Phonology-Morphology Interaction in a Constraint-based Model*. Doctoral dissertation, UMass Amherst. https://scholarworks.umass.edu/dissertations/AAI3336987

Wolf, Matthew. 2011. Limits on global rules in Optimality Theory with Candidate Chains. *Phonology* 28, 87–128. doi:10.1017/S0952675711000042

Zwicky, Arnold. 1987. Rule interactions: Another gloss on K&K. *Innovations in Linguistics Education* 5, 91–111. https://web.stanford.edu/~zwicky/rule-interactions.pdf