To appear in *Linguistic Inquiry*

Priority union and feature logic in phonology

Charles Reiss
Concordia University, Montréal
charles.reiss@concordia.ca

Abstract

Rules built from a single (partial) operation, priority union, can model both feature-filling and feature-changing processes. The distinction is handled by specifying which argument of priority union is defeasible. Priority union should be considered as a candidate for inclusion in phonological Universal Grammar.
Keywords: *phonology, feature logic, rules, Universal Grammar*

1 Introduction

Consider a traditional phonological rule like (1):

(1) $a \rightarrow b$ / c ___ d

This rule can be thought of as a function mapping strings of segments to strings of segments, with the arrow '$\rightarrow$' denoting the *operation* of replacing the segment a with the segment b whenever it occurs between the segments c and d, so that the sequence cad in inputs is mapped to the sequence cbd in outputs. Let's maintain this distinction between the whole rule (1) and the operation (or operator) contained in the rule. Bale and Reiss 2018 provides a semantics for such rules built from the $\rightarrow$ operator, and extend it to deal with rules that insert and delete segments. The significant differences needed among the semantics of rules like (1) that replace one segment with another, rules that insert segments, and rules that delete segments demonstrate that the arrow does not correspond to a single, coherent operation on strings.

In *The Sound Pattern of English* (Chomsky and Halle 1968) and most work in generative phonology, the arrow is maintained even as the representations that constitute the input and output of mappings become more sophisticated, for example with the decomposition of segments into feature matrices and the characterization of rule targets and environments in terms of feature-based natural classes of segments. Even considering only rules that affect the internal feature structure of individual segments (so, no metathesis or full segment insertion or deletion) it has become apparent that the arrow operator is not used in a consistent manner, since it sometimes inserts features into segments (feature-filling), and sometimes replaces one valued feature with another (feature-changing). One solution is to 'deconstruct' the arrow into (at least) two

[0]Thanks to David Ta-Chun Shen, Hilton Alers-Valentín, Veno Volenec, Alan Bale, Dana Isac, Marjorie Leduc, Ash Asudeh, Ida Toivonen, Björn Köhnlein and the *LI* squib editors and reviewers.

distinct operations, with each rule built with just one operator (see Bale, Papillon, and Reiss 2014; Bale and Reiss 2018; Bale, Reiss, and Shen To appear). In other words, allowing a, b, c, d to represent entities more complex than just segments, we proposed, in essence, that any rule conforming to (2) is licit, where ω must be a member of Ω, the *set* of operators available to phonological Universal Grammar:

(2) $a\ \omega\ b\ /\ c\ \_\_\ d$

One task for phonologists is to discover the content of Ω, the set of operators that can instantiate ω, in order to answer the question "What is the set of operations that can express the structural changes effected by phonological rules?" Once we start positing operations, we can ask questions about their logico-mathematical properties, like how many arguments do they take, whether they are commutative, and whether they are partial or total operations. Answering such questions is part of what it means to understand phonological computation.[1] Having explicit answers to these questions also allows us to compare different proposals concerning the "correct" set of basic operations for human phonology.

Given the difference between feature-filling and feature-changing operations, one could just posit the existence of two corresponding distinct operators that can instantiate ω. However, in recent work, we took a different approach. We followed a tradition dating at least to Harris 1984 and adopted by many other scholars (such as Poser 1993, 2004; Wiese 2000; Samuels 2011; McCarthy 2007) of treating feature-*changing* processes, say /d/ to [t], as a sequence of (a) deletion of the underlying $+$VOICE specification, yielding underspecified /D/, followed by (b) insertion of $-$VOICE. A benefit of breaking down feature-changing processes is that it allows a feature-*filling* process mapping an underlying /D/ to surface [t] to be modeled with just (b), the insertion rule. Our implementation of feature deletion used *set subtraction*, a binary, non-commutative, operation, and for feature insertion we used *unification* (see below), a binary, commutative partial operation. Thus, a set subtraction operator '$-$' and a unification operator '$\sqcup$' were posited as possible values for ω. In our model, the phonology of a language is a complex function of composed rules, but the basic operations do not compose with each other inside of individual rules.

In this squib, I also consider only feature-filling and feature-changing processes, but I show that a single operation, *priority union* can be used to model both. The two approaches to modeling the domain of ω are thus extensionally equivalent within the narrow scope of phenomena considered here. This is interesting from a purely formal perspective, but I will also suggest how further empirical work might allow us to choose between the two.

Any feature logic argument exists within a framework of basic assumptions, and these will be sketched, but cannot be justified here. I follow the assumptions of my recent work with Bale and others, except with regard to the use of the mechanisms for feature insertion and deletion.

[1] Of course, there are many other questions, like "Should we be using privative features?" and "Should we be using ranked constraints instead of rules?", but we can't do everything at once.

2 Priority Union

Abstracting away from issues such as the representation of contours (e.g., affricates), I treat segments as sets of valued binary features with the condition that a segment must be *consistent*:

(3) A set of valued features ρ is consistent if and only if there is no feature F such that $+\text{F} \in \rho$ and $-\text{F} \in \rho$.

Valued features like $+$F and $-$F are called *opposites*, so more colloquially, a consistent set of features is one that does not contain opposites.

Given the treatment of segments as sets, it is tempting to model feature insertion using simple set *union*.[2] However, applying the union of a segment-set and another set of valued features potentially leads to an output that is not consistent:

(4) Union can yield output that is not consistent:
$\{+\text{COR}\ -\text{SON}, +\text{VOICE}\} \cup \{-\text{VOICE}\} = \{+\text{COR}, -\text{SON}, +\text{VOICE}, -\text{VOICE}\}$

Because the output in (4) contains both $+$VOICE and $-$VOICE as members, it is not consistent. In order to generate output segments that are always consistent we need a different operation from simple union for phonological rules.

We used *unification*, a partial operation instead of normal *union* in previous work (quod vide).[3] I explore here the use of yet another operation, priority union, adapting a definition from Kaplan 1987. Priority union has been applied to complex feature structures in syntactic, semantic and discourse representations (*e.g.* Kaplan 1987; Dalrymple 2001; Dahl 2002), but for our phonological purposes, we restrict discussion to sets of valued features like $\{+\text{ROUND}, -\text{HIGH}\}$:

(5) The **priority union** of two sets of valued features, A and B, is the union of the set of valued features that are in A, and the set of valued features in B whose opposites are not in A.

Paraphrasing Dahl 2002, priority union (also called default unification) takes two feature structures, one of which is identified as *strict*, and the other as *defeasible*, and combines the information in both such that the information in the strict structure takes priority over that in the defeasible structure if the two contain opposites.[4] For reasons that become clear below, I denote priority union by the normal union symbol

[2] The union of two sets $X \cup Y$ is the set whose members are in X or in Y (or in both X and Y).

[3] For any two sets, X and Y, the unification $X \sqcup Y$ is defined iff $X \cup Y$ is consistent. When defined, $X \sqcup Y = X \cup Y$. Because $X \sqcup Y$ may be undefined, unification is called a *partial* operation—it does not always yield an output, even when the arguments are chosen from the right domain. The *rules* built from the unification operation *are* total, because when unification is undefined, the rule provides an identity mapping, a form of vacuous application. Normal union of two sets is always defined, so union is a *total* operation. It is important that phonological *rules* be total functions so that they can compose—each rule takes as its input the output of the previous rule (or the UR).

[4] An alternative version of priority union would make certain *values of features* strict and other values defeasible. For example, $+$F might always take priority over $-$F. This is *not* the version I use here, although one could imagine that it could be useful for modeling, say, putative dominant-recessive harmony systems.

with an arrow underneath. The arrow's direction points from the strict argument to the defeasible argument:

(6) Priority union examples

a. $\{+\text{F}\} \underset{\rightarrow}{\cup} \{+\text{G}\} = \{+\text{F}, +\text{G}\}$

b. $\{+\text{F}, -\text{G}\} \underset{\rightarrow}{\cup} \{+\text{G}\} = \{+\text{F}, -\text{G}\}$

c. $\{+\text{F}, -\text{G}\} \underset{\rightarrow}{\cup} \{+\text{G}, +\text{H}\} = \{+\text{F}, -\text{G}, +\text{H}\}$

d. $\{+\text{F}, -\text{G}\} \underset{\leftarrow}{\cup} \{+\text{G}, +\text{H}\} = \{+\text{F}, +\text{G}, +\text{H}\}$

In (a) priority union gives the same result as normal union. In (b) the valued feature $-$G of the strict argument takes priority over the $+$G of the defeasible argument. The same situation holds in (c), except that the defeasible argument contributes $+$H, since there is no conflict with the strict argument. In (d), the arrow has been reversed, so that $+$G is a member of the strict argument and $-$G is in the defeasible argument. Comparing (c) and (d) we see that, unlike simple set union, priority union is not commutative—the priority union of A with B is not necessarily the same as the priority union of B with A. Like normal union, but unlike unification, priority union is a total operation.

3 Priority Union for Feature-Filling

On the basis of data like that in (7), Inkelas (1995) analyses Turkish as having a three-way underlying consonant contrast that we can use to illustrate how priority union can capture feature-filling phonological rules.[5]

(7) Turkish voicing alternations

a. Non-alternating voiceless: [-voiced] /t/
sanat 'art' *sanatlar* 'art-plural' *sanatım* 'art-1sg.poss'

b. Non-alternating voiced: [+voiced] /d/
etüd 'etude' *etüdler* 'etude-plural' *etüdüm* 'etude-1sg.poss'

c. Alternating: [∅voiced] (unmarked for [voiced]) /D/
kanat 'wing' *kanatlar* 'wing-plural' *kanadım* 'wing-1sg.poss'

The root-final segment of (a) is /t/, is a voiceless coronal stop; the root-final segment of (b) is /d/, a voiced coronal stop; and the alternating root-final segment of (c) must be neither /d/ nor /t/, so it can be analyzed as /D/, a coronal stop with no specification for voicing. The fully specified stops do not alternate in the relevant environments, but /D/ surfaces as [d] in onsets and as [t] in coda.

[5]The exact analysis of Turkish is controversial, but there exist other phenomena (e.g. well-known vowel harmony systems) that share the same structure as our presentation here, so we can abstract away from phonetic details—we are just concerned with three underlying segments a, b, c, such that b is the intersection of a and c. In other words, b is unspecified for a feature with respect to which a and c contain opposites. This is not a rare situation—it makes sense to posit such underlying forms whenever there is a three-way behavior manifested by just two surface segments.

Let's look at the rule based on priority union that turns /D/ to [t] in codas:[6]

(8) $\begin{bmatrix} +\text{Cor} \\ -\text{Cont} \\ -\text{Son} \end{bmatrix} \underset{\rightarrow}{\cup} \{-\text{Voice}\}$ / in Coda position

The target natural class contains the three segments /t, d, D/. Of course, we need to make sure that the rule only affects /D/, but there is no way to refer to underspecified /D/ without referring to /t/ and /d/ as well.[7]

(9) Class w/ underspecified member:$[+\text{Cor}, -\text{Cont}, -\text{Son}, \dots]$[8]

a. /t/ = $\{-\text{Voice}, +\text{Cor}, -\text{Cont}, -\text{Son}, \dots\}$

b. /d/ = $\{+\text{Voice}, +\text{Cor}, -\text{Cont}, -\text{Son}, \dots\}$

c. /D/ = $\{+\text{Cor}, -\text{Cont}, -\text{Son}, \dots\}$ (no Voice feature)

The following schematic strings show the workings of priority union for feature-filling in codas, with a dot '.' marking the boundary between syllables—a consonant before a dot is in a coda:

(10) Feature-filling priority union with $\{-\text{Voice}\}$ in coda

a. Input /mat.na/: /t/ $\underset{\rightarrow}{\cup} \{-\text{Voice}\}$ is [t] so output is vacuously [matna]

b. Input /mad.na/: /d/ $\underset{\rightarrow}{\cup} \{-\text{Voice}\}$ is [d] so output is vacuously [madna]

c. Input /maD.na/: /D/ $\underset{\rightarrow}{\cup} \{-\text{Voice}\}$ is [t] so output is [matna]

In (10a), priority union is vacuous, because /t/ is $-$Voice. Therefore the output is [t]. In (10b), priority union yields [d] because the $+$Voice target /d/ is the strict argument, and the structural change $\{-\text{Voice}\}$ is the defeasible argument. Priority union is vacuous here, too. In (10c), priority union adds the valued feature $-$Voice to /D/, yielding [t]. Priority union thus allows us to make a feature-filling rule that changes underlying /D/, but not /d/, to [t]. The logic for the voicing of /D/ in onsets is identical—the only non-vacuous application would be /D/ to [d].

[6]The rule maps between representations—strings or strings with syllable structure. The structural change is the set of *features* $\{-\text{Voice}\}$ in normal set brackets; the target structural description refers to a set of *segments*, so we use the square brackets (and assume operator overloading) in order to differentiate the set-theoretic types: the target is the set of *segments* that are supersets of the set of *features* $\{+\text{Cor}, -\text{Cont}, -\text{Son}\}$. This distinction of brackets to match set-theoretic types is a notational innovation we first recommended in Bale et al. (2014). Most of the literature uses square brackets for both sets of features and sets of segments. Note that using '$\rightarrow$' in this rule would attribute to that symbol an operation *inserting* a valued feature into a segment.

[7]Natural classes are defined by (generalized) intersection (see Bale and Reiss, 2018, Unit 45 for details), and the intersection of /d/ and /t/ yields the set of features that define /D/. We assume that rules cannot make reference to *obligatory* absence of a feature specification, e.g. '0Voice'. Allowing such use of a third overt value '0' leads to dubious natural classes and a four-way contrast: $+$, $-$, 0 and 'absent' (and thus to a potential infinite regress). This problem of referring to unmarked elements is ubiquitous in linguistics—see Bonet (1995, fn. 28) for an instance in morphology. Perhaps further research will show that priority union can address the difficulty that Bonet identifies, but does not solve.

[8]The ellipsis '…' denotes whatever other features may be relevant to characterizing the segments in question, in this case, anything except Voice.

4 Priority Union for Feature-Changing

In order to use priority union to account for feature-changing rules, we need to dissociate two aspects of rule notation. In a traditional rule like (1) '$a \rightarrow b$ / c ___ d' the first element a is called the *target*. The element b, after the $\rightarrow$ operator is called the *structural change* of the rule. We need to maintain this convention for expressing rules, but distinguish it from the issue of which argument of priority union is the strict one and which is the defeasible one. This information is expressed by the direction of the arrow in our priority union symbol. Consider these examples of rules using priority union:

(11) Rule target vs. strict argument

i. $a \underset{\rightarrow}{\cup} b$ / c ___ d
   a is the target; a is the strict argument

ii. $a \underset{\leftarrow}{\cup} b$ / c ___ d
   a is the target; b is the strict argument

In (i), a is both the target of the rule and the strict argument of priority union. In (ii), a is again the target of the rule, but now b is the strict argument of priority union. In both rules, the environment is determined by a preceding c and a following d. To reiterate: the syntax of rules places a characterization of a target natural class first, an operator symbol second and the characterization of a change third; this is independent from the direction of the priority union arrow which points from the strict argument to the defeasible one.

To illustrate feature-changing rules, we don't need an underlyingly underspecified segment like /D/ so let's consider a language with just /t/ and /d/.[9] Hungarian has a pattern of reciprocal neutralization whereby /t/ becomes [d] before a voiced obstruent and /d/ becomes [t] before a voiceless one. The process generalizes to all pairs of voiced and voiceless obstruents, like /p/ and /b/ (with some irrelevant complications). For example, the forms in (12) show stem-final /p,t/ mapping to [b,d], respectively, before a suffix that starts with voiced /b/; and stem-final /b,d/ mapping to [p,t], respectively, before a suffix that starts with voiceless /t/:[10]

(12) Hungarian feature-changing voicing assimilation

Lexical	Bare Noun	in Noun /-ban/	from Noun /-to:l/	to Noun /-nak/	gloss
/ku:t/	ku:t	ku:dban	ku:tto:l	ku:tnak	'well'
/ka:d/	ka:d	ka:dban	ka:tto:l	ka:dnak	'tub'
/kalap/	kalap	kalabban	kalapto:l	kalapnak	'hat'
/rab/	rab	rabban	rapto:l	rabnak	'prisoner'

[9] If there were a /D/, it would undergo feature-filling whenever one of the others underwent feature-changing. For this reason, we can't distinguish with the tools here, say, an underlying /t,d/ inventory from a /t,D/ inventory.

[10] Using '$\rightarrow$' here would ascribe to that symbol an operation *changing* a valued feature from $+$F to $-$F or vice versa, as opposed to the *filling* operation mentioned in fn. 6.

We can capture this process with a single feature-changing rule that uses priority union and α notation:

(13) Hungarian voicing assimilation

Examples:

i. p,t $\rightarrow$ b,d / ___b

ii. b,d $\rightarrow$ p,t / ___t

Rule: $[-\text{SON}] \underset{\leftarrow}{\cup} \{\alpha\text{VOICE}\}$ / ___ $\begin{bmatrix} \alpha\text{VOICE} \\ -\text{SON} \end{bmatrix}$

The rule target is the natural class of segments that are specified $-$SON, so this specification is listed leftmost in the rule. Next comes the priority union symbol, but with the arrow pointing from the right to the left, showing that the rule target's values are defeasible. This means that the values specified in the structural change will always 'win': /tb/ will surface as [db] and /bt/ will surface as [pt]. This makes the rule feature-*changing*.

Again, we illustrate with schematic examples:

(14) Feature-changing priority union with following $\{\alpha$VOICE$\}$

a. Input /mat.ba/: /t/ $\underset{\leftarrow}{\cup} \{+\text{VOICE}\}$ is [d] so output is [madba]

b. Input /mat.ta/: /t/ $\underset{\leftarrow}{\cup} \{-\text{VOICE}\}$ is [t] so output is vacuously [matta]

c. Input /mad.ba/: /d/ $\underset{\leftarrow}{\cup} \{+\text{VOICE}\}$ is [d] so output is vacuously [madba]

d. Input /mad.ta/: /d/ $\underset{\leftarrow}{\cup} \{-\text{VOICE}\}$ is [t] so output is [matta]

Because of the reciprocal neutralization, modeled with α notation, the feature of the structural change in Hungarian 'comes from' a segment that occurs in the rule environment, the trigger, but this does not mean that *all* the features of the trigger take priority over the those of the target, otherwise there would be total assimilation. A form like (14a) shows assimilation of VOICE, but not of place features.

The Hungarian data is useful to show that priority union is compatible with α notation, which results in agreement of feature values between target and triggering environment.[11] However, no segmental trigger is necessary—a language could have, say, feature-changing devoicing in coda position, as in (15):

(15) Feature-changing coda devoicing

$[-\text{SON}] \underset{\leftarrow}{\cup} \{-\text{VOICE}\}$ / in Coda position

Compare this rule in (15) to the Turkish feature-filling mapping of /D/ to [t] in (8), where the arrow of priority union points from the rule target towards the structural change. The rule in (15) is feature-changing, because the voicing values on the targets are defeasible.[12]

[11] The careful reader will appreciate that there are non-trivial issues to work out concerning the semantics of rules with α notation, especially with regard to how natural classes are defined.

[12] It is trivial to replace the two versions of priority union with a single operator symbol, as long as rules are given in some format that specifies which argument is strict and which defeasible.

5 Conclusions

Chomsky (1957:5) points out that "a formalized theory may automatically provide solutions for many problems other than those for which it was explicitly designed." It follows that it is also necessary to consider alternatives. Two formalisms A and B for modeling a body of grammatical phenomena may appear to be extensionally equivalent for several reasons. Trivially, the two may be notational variants, or differ in arbitrary and uninteresting ways: for example, A uses BACK and B uses FRONT to name a binary feature.

For a more interesting case, consider that fully equivalent logical systems may be expressible with different-sized inventories of basic operations. For example, a propositional logic that uses symbols for AND, OR, NOT, IF–THEN and IF AND ONLY IF can be replaced with one using just the 'Sheffer's stroke' (Nicod 1917). In the realm of pure logic it makes no sense to ask whether a particular computation 'really' involves NOT(NOT p AND NOT q) or else a combination of Sheffer's strokes, but actual physically instantiated information processing systems use some specific inventory of operations. The operator corresponding to the Sheffer's stroke is known as NAND in computer science, and it is widely used in computer processor engineering. So, there is a matter of truth in the question of whether a particular computer, or the software it runs, is making use of NAND gates, and the answer depends on how the system was designed. Linguists, unlike engineers can't look at a manual or take the machine apart to see what's inside—but there are ways to make progress.

A third way in which two formal systems may differ is that, although they appear to be equivalent for modeling the range of, say, grammatical patterns under consideration, one might imagine that there could be other patterns that have not been considered, or perhaps even attested, for which A works, but B does not. In other words, the two systems are not extensionally equivalent over a larger domain than previously considered, and they lead to different predictions about the actual scope of possible human language data. Recognizing these differences between A and B can lead to further research to expand the range of empirical phenomena that the systems need to model, and help choose between them (cf. Chomsky 1986:38).

A first step in figuring out the 'correct' model for a module of grammar is to describe some alternatives. This squib presents a different system for the computation of feature-filling and feature-changing processes from that based on set subtraction and unification. I have focussed here on formalism, but it is apparent that there are different predictions to be made by the 'subtraction and unification' model and the 'priority union' model. For example, the former allows for the existence of *derived surface underspecification*, but the latter does not. Using the symbols introduced for Turkish, this would be a situation in which, an underlying /d/ could *surface* as [D] by set subtraction of $\{+\text{VOICED}\}$. We know that surface underspecification exists, since the work of Keating (1988), but it is not yet known if surface underspecified segments are sometimes derived from fully specified ones. Without subtraction, priority union alone could not model such processes.

In contrast, purely phonological *exchange* processes (Fitzpatrick, Nevins, and Vaux 2004), which replace a valued feature with its opposite, are trivial to express with

priority union: $[\alpha\text{F}] \underset{\leftarrow}{\cup} \{-\alpha\text{F}\}$. However, exchange processes cannot be modeled straightforwardly by subtraction and unification, because subtraction of both $+$F and $-$F in a given context will neutralize an underlying distinction (say, *t* and *d* to *D*) before unification can supply the opposite value. So, the open question of whether phonological exchange rules exist is relevant to the choice of possible phonological operators. These are examples of how "sterile" formal work can drive empirical research, as it always should (Halle 1975).

If rule based phonology is to be of any value, it is necessary to develop theories of what constitutes a possible rule. This demonstration of the power of a simple operation like priority union is a step in this direction. The alternative proposals in (16) and (17) for the content of Ω, and thus for the shape of possible rules, are explicit (assuming that a and b are of the correct logical type):

(16) Possible rules if Ω contains subtraction and unification

i. $a - b \,/\, c \_\_ d$

ii. $a \sqcup b \,/\, c \_\_ d$

(17) Possible rules if Ω contains priority union

i. $a \underset{\rightarrow}{\cup} b \,/\, c \_\_ d$

ii. $a \underset{\leftarrow}{\cup} b \,/\, c \_\_ d$

Comparing the two proposals for phonological Universal Grammar in terms of elegance and empirical coverage remains a challenge for future research.

References

Bale, Alan, Maxime Papillon, and Charles Reiss. 2014. Targeting underspecified segments: A formal analysis of feature changing and feature filling rules. *Lingua* 148:240–253.

Bale, Alan, and Charles Reiss. 2018. *Phonology: A formal introduction*. Cambridge, MA: MIT Press.

Bale, Alan, Charles Reiss, and David Ta-Chun Shen. To appear. Sets, rules and natural classes: { } *vs.* [ ]. *Loquens* .

Bonet, Eulália. 1995. Feature structure of Romance clitics. *Natural Language and Linguistic Theory* 13:607–647.

Chomsky, Noam. 1986. *Knowledge of language: Its nature, origin, and use*. Westport, CT: Praeger.

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

Dahl, Veronica. 2002. On implicit meanings. In *Computational logic: Logic programming and beyond* 506–525. Springer.

Dalrymple, Mary. 2001. *Lexical functional grammar*. Leiden: Brill.

Fitzpatrick, Justin, Andrew Nevins, and Bert Vaux. 2004. Exchange rules and feature-value variables. Presented at the Third North American Phonology Conference (*NAPhC3*), Concordia University, Montréal, Québec.

Halle, Morris. 1975. Confessio grammatici. *Language* 51:525–535.

Harris, James W. 1984. Autosegmental phonology, lexical phonology and Spanish nasals. In *Language sound structure*, ed. Mark Aronoff and Richard Oehrle 67–82. Cambridge, MA: MIT Press.

Inkelas, Sharon. 1995. The consequences of optimization for underspecification. In *Proceedings of the North East Linguistic Society 25*, ed. Jill Beckman 287–302. University of Pennsylvania: Graduate Linguistic Student Association.

Kaplan, Ronald M. 1987. Three seductions of computational psycholinguistics. In *Linguistic theory and computer applications*, ed. P. Whitelock, M. M. Wood, H. L. Somers, R. Johnson, and P. Bennett 149–188. London: Academic Press.

Keating, Patricia. 1988. Underspecification in phonetics. *Phonology* 5:275–292.

McCarthy, John J. 2007. Slouching toward optimality: Coda reduction in OT-CC. *Phonological Studies (Journal of the Phonological Society of Japan)* 7:89–104.

Nicod, Jean. 1917. A reduction in the number of primitive propositions of logic. In *Proceedings of the Cambridge Philosophical Society* volume 19 32–41.

Poser, William J. 1993. Are strict cycle effects derivable? In *Studies in lexical phonology*, ed. Sharon Hargus and Ellen M Kaisse volume 4 315–321. San Diego, CA: Academic Press.

Poser, William J. 2004. On the status of Chumash sibilant harmony. Ms., University of Pennsylvania. Philadelphia.

Samuels, Bridget D. 2011. *Phonological architecture: a biolinguistic perspective*. Oxford: Oxford University Press.

Wiese, Richard. 2000. *The phonology of German*. Oxford: Oxford University Press.