Computational structure of generative phonology and its relation to language comprehension.

Eric Sven Ristad*
MIT Artificial Intelligence Lab
545 Technology Square
Cambridge, MA 02139

Abstract

We analyze the computational complexity of phonological models as they have developed over the past twenty years. The major results are that generation and recognition are undecidable for segmental models, and that recognition is NP-hard for that portion of segmental phonology subsumed by modern autosegmental models. Formal restrictions are evaluated.

1 Introduction

Generative linguistic theory and human language comprehension may both be thought of as computations. The goal of language comprehension is to construct structural descriptions of linguistic sensations, while the goal of generative theory is to enumerate all and only the possible (grammatical) structural descriptions. These computations are only indirectly related. For one, the input to the two computations is not the same. As we shall see below, the most we might say is that generative theory provides an extensional characterisation of language comprehension, which is a function from surface forms to complete representations, including underlying forms. The goal of this article is to reveal exactly what generative linguistic theory says about language comprehension in the domain of phonology.

The article is organized as follows. In the next section, we provide a brief overview of the computational structure of generative phonology. In section 3, we introduce the segmental model of phonology, discuss its computational complexity, and prove that even restricted segmental models are extremely powerful (undecidable). Subsequently, we consider various proposed and plausible restrictions on the model, and conclude that even the maximally restricted segmental model is likely to be intractable. The fourth section introduces the modern autosegmental (nonlinear) model and discusses its computational complexity. We prove that the natural problem of constructing an autosegmental representation of an underspecified surface form is NP-hard. The article concludes by arguing that the complexity proofs are unnatural despite being true of the phonological models, because the formalism of generative phonology is itself unnatural.

The central contributions of this article are: (i) to explicate the relation between generative theory and language processing, and argue that generative theories are not models of language users primarily because they do not consider the inputs naturally available to language users; and (ii) to analyze the computational complexity of generative phonological theory, as it has developed over the past twenty years, including segmental and autosegmental models.

2 Computational structure of generative phonology

The structure of a computation may be described at many levels of abstraction, principally including: (i) the goal of the computation; (ii) its input/output specification (the problem statement), (iii) the algorithm and representation for achieving that specification, and (iv) the primitive operations in which terms the algorithm is implemented (the machine architecture).

Using this framework, the computational structure of generative phonology may be described as follows:

- The computational goal of generative phonology (as distinct from it's research goals) is to enumerate the phonological dictionaries of all and only the possible human languages.
- The problem statement is to enumerate the observed phonological dictionary of a particular language from some underlying dictionary of morphemes (roots and affixes) and phonological processes that apply to combinations of underlying morphemes.
- The algorithm by which this is accomplished is a derivational process g ('the grammar') from underlying forms x to surface forms $y = g(x)$. Underlying forms are constructed

*The author is supported by a IBM graduate fellowship and eternally indebted to Morris Halle and Michael Kenstowicz for teaching him phonology. Thanks to Noam Chomsky, Sandiway Fong, and Michael Kashket for their comments and assistance.

by combining (typically, with concatenation or substitution) the forms stored in the underlying dictionary of morphemes. Linguistic relations are represented both in the structural descriptions and the derivational process.

The structural descriptions of phonology are representations of perceivable distinctions between linguistic sounds, such as stress levels, syllable structure, tone, and articulatory gestures. The underlying and surface forms are both drawn from the same class of structural descriptions, which consist of both segmental strings and autosegmental relations. A segmental string is a string of segments with some representation of constituent structur. In the SPE theory of Chomsky and Halle (1968) concrete boundary symbols are used; in Lexical Phonology, abstract brackets are used. Each segment is a set of phonological features, which are abstract as compared with phonetic representations, although both are given in terms of phonetic features. Suprasegmental relations are relations among segments, rather than properties of individual segments. For example, a syllable is a hierarchical relation between a sequence of segments (the nucleus of the syllable) and the less sonorous segments that immediately preceed and follow it (the onset and coda, respectively). Syllables must satisfy certain universal constraints, such as the sonority sequencing constraint, as well as language particular ones.

- The derivational process is implemented by an ordered sequence of unrestricted rewriting rules that are applied to the current derivation string to obtain surface forms.

According to generative phonology, comprehension consists of finding a structural description for a given surface form. In effect, the logical problem of language comprehension is reduced to the problem of searching for the underlying form that generates a given surface form. When the surface form does not transparently identify its corresponding underlying form, when the space of possible underlying forms is large, or when the grammar g is computationally complex, the logical problem of language comprehension can quickly become very difficult.

In fact, the language comprehension problem is intractable for all segmental theories. For example, in the formal system of *The Sound Pattern of English* (SPE) the comprehension problem is undecidable. Even if we replace the segmental representation of cyclic boundaries with the abstract constituents of Lexical Phonology, and prohibit derivational rules from readjusting constituent boundaries, comprehension remains PSPACE-complete. Let us now turn to the technical details.

3 Segmental Phonology

The essential components of the segmental model may be briefly described as follows. The set of features includes both phonological features and diacritics and the distinguished feature **segment** that marks boundaries. (An example diacritic is **ablaut**, a feature that marks stems that must undergo a change vowel quality, such as tense-conditioned ablaut in the English *sing, sang, sung* alternation.) As noted in SPE, "technically speaking, the number of diacritic features should be at least as large as the number of rules in the phonology. Hence, unless there is a bound on the length of a phonology, the set [of features] should be unlimited." (fn.1, p.390) Features may be specified + or - or by an integral value $1, 2, \ldots, N$ where N is the maximal degree of differentiation permitted for any linguistic feature. Note that N may vary from language to language, because languages admit different degrees of differentiation in such features as vowel height, stress, and tone. A set of feature specifications is called a *unit* or sometimes a *segment*. A string of units is called a *matrix* or a *segmental string*.

A *elementary rule* is of the form $ZXAYW \rightarrow ZXBYW$ where A and B may be ϕ or any unit, $A \neq B$; X and Y may be matrices (strings of units), and Z and W may be thought of a brackets labelled with syntactic categories such as 'S' or 'N' and so forth. A *complex rule* is a finite schema for generating a (potentially infinite) set of elementary rules.[1] The rules are organized into

[1]Following Johnson (1972), we may define schema as follows. The empty string and each unit is a schema; schema may be combined by the operations of union, intersection, negation, kleene star, and exponentiation over the set of units. Johnson also introduces variables and Boolean conditions into the schema. This "schema language" is a extremely powerful characterisation of the class of regular languages over the alphabet of units; it is not used by practicing phonologists. Because a given complex rule can represent an infinite set of elementary rules, Johnson shows how the iterated, exhaustive application of one complex rule to a given segmental string can "effect virtually any computable mapping," (p.10) ie., can simulate any TM computation. Next, he proposes a more restricted "simultaneous" mode of application for a complex rule, which is only capable of performing a finite-state mapping in any application. This article considers the independent question of what computations can be performed by a set of elementary rules, and hence provides loose lower bounds for Johnson's model. We note in passing, however, that the problem of simply determining whether a given rule is subsumed by one of Johnson's schema is itself intractable, requiring at least exponen-

linear sequence $R_1, R_2, \ldots R_n$, and they are applied in order to an underlying matrix to obtain a surface matrix.

Ignoring a great many issues that are important for linguistic reasons but irrelevant for our purposes, we may think of the derivational process as follows. The input to the derivation, or "underlying form," is a bracketed string of morphemes, the output of the syntax. The output of the derivation is the "surface form," a string of phonetic units. The derivation consists of a series of cycles. On each cycle, the ordered sequence of rules is applied to every maximal string of units containing no internal brackets, where each R_{i+1} applies (or doesn't apply) to the result of applying the immediately preceding rule R_i, and so forth. Each rule applies simultaneously to all units in the current derivational string. For example, if we apply the rule $A \rightarrow B$ to the string AA, the result is the string BB. At the end of the cycle, the last rule R_n erases the innermost brackets, and then the next cycle begins with the rule R_1. The derivation terminates when all the brackets are erased.

Some phonological processes, such as the assimilation of voicing across morpheme boundaries, are very common across the world's languages. Other processes, such as the arbitrary insertion of consonants or the substitution of one unit for another entirely distinct unit, are extremely rare or entirely unattested. For this reason, all adequate phonological theories must include an explicit measure of the naturalness of a phonological process. A phonological theory must also define a criterion to decide what constitutes two independent phonological processes and what constitutes a legitimate phonological generalization. Two central hypotheses of segmental phonology are (i) that the most natural grammars contain the fewest symbols and (ii) a set of rules represent independent phonological processes when they cannot be combined into a single rule schema according to the intricate notational system first described in SPE. (Chapter 9 of Kenstowicz and Kisseberth (1979) contains a less technical summary of the SPE system and a discussion of subsequent modifications and emendations to it.)

3.1 Complexity of segmental recognition and generation.

Let us say a dictionary D is a finite set of the underlying phonological forms (matrices) of morphemes. These morphemes may be combined by concatenation and simple substitution (a syntactic category is replaced by a morpheme of that category) to form a possibly infinite set of underlying forms. Then we may characterize the two central computations of phonology as follows.

tial space.

The *phonological generation problem* (PGP) is: Given a completely specified phonological matrix x and a segmental grammar g, compute the surface form $y = g(x)$ of x.

The *phonological recognition problem* (PRP) is: Given a (partially specified) surface form y, a dictionary D of underlying forms, and a segmental grammar g, decide if the surface form $y = g(x)$ can be derived from some underlying form x according to the grammar g, where x constructed from the forms in D.

Lemma 3.1 *The segmental model can directly simulate the computation of any deterministic Turing machine M on any input w, using only elementary rules.*

Proof. We sketch the simulation. The underlying form x will represent the TM input w, while the surface form y will represent the halted state of M on w. The immediate description of the machine (tape contents, head position, state symbol) is represented in the string of units. Each unit represents the contents of a tape square. The unit representing the currently scanned tape square will also be specified for two additional features, to represent the state symbol of the machine and the direction in which the head will move. Therefore, three features are needed, with a number of specifications determined by the finite control of the machine M. Each transition of M is simulated by a phonological rule. A few rules are also needed to move the head position around, and to erase the entire derivation string when the simulated machine halts.

There are only two key observations, which do not appear to have been noticed before. The first is that contrary to popular misstatement, phonological rules are not context-sensitive. Rather, they are unrestricted rewriting rules because they can perform deletions as well as insertions. (This is essential to the reduction, because it allows the derivation string to become arbitarily long.) The second observation is that segmental rules can freely manipulate (insert and delete) boundary symbols, and thus it is possible to prolong the derivation indefinitely: we need only employ a rule R_{n-1} at the end of the cycle that adds an extra boundary symbol to each end of the derivation string, unless the simulated machine has halted. The remaining details are omitted, but may be found in Ristad (1990). □

The immediate consequences are:

Theorem 1 *PGP is undecidable.*

Proof. By reduction to the undecidable problem $w \in L(M)$? of deciding whether a given TM M accepts an input w. The input to the generation problem consists of an underlying form x that represents w and a segmental grammar

g that simulates the computations of M according to lemma 3.1. The output is a surface form $y = g(x)$ that represents the halted configuration of the TM, with all but the accepting unit erased. □

Theorem 2 *PRP is undecidable.*

Proof. By reduction to the undecidable problem $L(M) =?\phi$ of deciding whether a given TM M accepts any inputs. The input to the recognition problem consists of a surface form y that represents the halted accepting state of the TM, a trivial dictionary capable of generating Σ^*, and a segmental grammar g that simulates the computations of the TM according to lemma 3.1. The output is an underlying form x that represents the input that M accepts. The only trick is to construct a (trivial) dictionary capable of generating all possible underlying forms Σ^*. □

An important corollary to lemma 3.1 is that we can encode a universal Turing machine in a segmental grammar. If we use the four-symbol seven-state "smallest UTM" of Minsky (1969), then the resulting segmental model contains no more than three features, eight specifications, and 36 very simple rules (exact details in Ristad, 1990). As mentioned above, a central component of the segmental theory is an evaluation metric that favors simpler (ie., shorter) grammars. This segmental grammar of universal computation appears to contain significantly fewer symbols than a segmental grammar for any natural language. Therefore, this corollary presents severe conceptual and empirical problems for the segmental theory.

Let us now turn to consider the range of plausible restrictions on the segmental model. At first glance, it may seem that the single most important computational restriction is to prevent rules from inserting boundaries. Rules that manipulate boundaries are called readjustment rules. They are needed for two reasons. The first is to reduce the number of cycles in a given derivation by deleting boundaries and flattening syntactic structure, for example to prevent the phonology from assigning too many degrees of stress to a highly-embedded sentence. The second is to rearrange the boundaries given by the syntax when the intonational phrasing of an utterance does not correspond to its syntactic phrasing (so-called "bracketing paradoxes"). In this case, boundaries are merely moved around, while preserving the total number of bounndaries in the string. The only way to accomplish this kind of bracket readjustment in the segmental model is with rules that delete brackets and rules that insert brackets. Therefore, if we wish to exclude rules that insert boundaries, we must provide an alternate mechanism for boundary readjustment. For the sake of argument—and because it is not too hard to construct such a boundary readjustment mechanism—let us henceforth adopt this restriction. Now how powerful is the segmental model?

Although the generation problem is now certainly decidable, the recognition problem remains undecidable, because the dictionary and syntax are both potentially infinite sources of boundaries: the underlying form x needed to generate any given surface form according to the grammar g could be arbitrarily long and contain an arbitrary number of boundaries. Therefore, the complexity of the recognition problem is unaffected by the proposed restriction on boundary readjustments. The obvious restriction then is to additionally limit the depth of embeddings by some fixed constant. (Chomsky and Halle flirt with this restriction for the linguistic reasons mentioned above, but view it as a performance limitation, and hence choose not to adopt it in their theory of linguistic competence.)

Lemma 3.2 *Each derivational cycle can directly simulate any polynomial time alternating Turing machine (ATM) M computation.*

Proof. By reduction from a polynomial-depth ATM computation. The input to the reduction is an ATM M on input w. The output is a segmental grammar g and underlying form x s.t. the surface form $y = g(x)$ represents a halted accepting computation iff M accepts w in polynomial time. The major change from lemma 3.1 is to encode the entire instantaneous description of the ATM state (ie., tape contents, machine state, head position) in the features of a single unit. To do this requires a polynomial number of features, one for each possible tape square, plus one feature for the machine state and another for the head position. Now each derivation string represents a level of the ATM computation tree. The transitions of the ATM computation are encoded in a block B as follows. An AND-transition is simulated by a triple of rules, one to insert a copy of the current state, and two to implement the two transitions. An OR-transition is simulated by a pair of disjunctively-ordered rules, one for each of the possible successor states. The complete rule sequence consists of a polynomial number of copies of the block B. The last rules in the cycle delete halting states, so that the surface form is the empty string (or reasonably-sized string of 'accepting' units) when the ATM computation halts and accepts. If, on the other hand, the surface form contains any nonhalting or nonaccepting units, then the ATM does not accept its input w in polynomial time. The reduction may clearly be performed in time polynomial in the size of the ATM and its input. □

Because we have restricted the number of embeddings in an underlying form to be no more than

a fixed language-universal constant, no derivation can consist of more than a constant number of cycles. Therefore, lemma 3.2 establishes the following theorems:

Theorem 3 *PGP with bounded embeddings is PSPACE-hard.*

Proof. The proof is an immediate consequence of lemma 3.2 and a corollary to the Chandra-Kozen-Stockmeyer theorem (1981) that equates polynomial time ATM computations and PSPACE DTM computations. □

Theorem 4 *PRP with bounded embeddings is PSPACE-hard.*

Proof. The proof follows from lemma 3.2 and the Chandra-Kozen-Stockmeyer result. The dictionary consists of the lone unit that encodes the ATM starting configuration (ie., input w, start state, head on leftmost square). The surface string is either the empty string or a unit that represents the halted accepting ATM configuration. □

There is some evidence that this is the most we can do, at least for the PGP. The requirement that the reduction be polynomial time limits us to specifying a polynomial number of features and a polynomial number of rules. Since each feature corresponds to a tape square, ie., the ATM space resource, we are limited to PSPACE ATM computations. Since each phonological rule corresponds to a next-move relation, ie., one time step of the ATM, we are thereby limited to specifying PTIME ATM computations.

For the PRP, the dictionary (or syntax-interface) provides the additional ability to nondeterministically guess an arbitrarily long, boundary-free underlying form x with which to generate a given surface form $g(x)$. This ability remains unused in the preceeding proof, and it is not too hard to see how it might lead to undecidability.

We conclude this section by summarizing the range of linguistically plausible formal restrictions on the derivational process:

Feature system. As Chomsky and Halle noted, the SPE formal system is most naturally seen as having a variable (unbounded) set of features and specifications. This is because languages differ in the diacritics they employ, as well as differing in the degrees of vowel height, tone, and stress they allow. Therefore, the set of features must be allowed to vary from language to language, and in principle is limited only by the number of rules in the phonology; the set of specifications must likewise be allowed to vary from language to language.

It is possible, however, to postulate the existence of a large, fixed, language-universal set of phonological features and a fixed upper limit to the number N of perceivable distinctions any one feature is capable of supporting. If we take these upper limits seriously, then the class of reductions described in lemma 3.2 would no longer be allowed. (It will be possible to simulate any $\mathcal{NP}$ computation in a single cycle, however.)

Rule format. Rules that delete, change, exchange, or insert segments—as well as rules that manipulate boundaries—are crucial to phonological theorizing, and therefore cannot be crudely constrained. More subtle and indirect restrictions are needed. One approach is to formulate language-universal constraints on phonological representations, and to allow a segment to be altered only when it violates some constraint.

McCarthy (1981:405) proposes a morpheme rule constraint (MRC) that requires all morphological rules to be of the form $A \rightarrow B/X$ where A is a unit or ϕ, and B and X are (possibly null) strings of units. (X is the immediate context of A, to the right or left.) It should be obvious that the MRC does not constrain the computational complexity of segmental phonology.

4 Autosegmental Phonology

In the past decade, generative phonology has seen a revolution in the linguistic treatment of suprasegmental phenomena such as tone, harmony, infixation, and stress assignment. Although these autosegmental models have yet to be formalised, they may be briefly described as follows. Rather than one-dimensional strings of segments, representations may be thought of as "a three-dimensional object that for concreteness one might picture as a spiral-bound notebook," whose spine is the segmental string and whose pages contain simple constituent structures that are indendent of the spine (Halle 1985). One page represents the sequence of tones associated with a given articulation. By decoupling the representation of tonal sequences from the articulation sequence, it is possible for segmental sequences of different lengths to nonetheless be associated to the same tone sequence. For example, the tonal sequence Low-High-High, which is used by English speakers to express surprise when answering a question, might be associated to a word containing any number of syllables, from two (*Brazil*) to twelve (*floccinaucinihilipilification*) and beyond. Other pages (called "planes") represent morphemes, syllable structure, vowels and consonants, and the tree of articulatory (ie., phonetic) features.

4.1 Complexity of autosegmental recognition.

In this section, we prove that the PRP for autosegmental models is NP-hard, a significant reduction in complexity from the undecidable and PSPACE-hard computations of segmental theories. (Note however that autosegmental representations have augmented—but not replaced—portions of the segmental model, and therefore, unless something can be done to simplify segmental derivations, modern phonology inherits the intractability of purely segmental approaches.)

Let us begin by thinking of the NP-complete 3-Satisfiability problem (3SAT) as a set of interacting constraints. In particular, every satisfiable Boolean formula in 3-CNF is a string of clauses $C_1, C_2, \ldots, C_p$ in the variables $x_1, x_2, \ldots, x_n$ that satisfies the following three constraints: (i) negation: a variable x_j and its negation $\overline{x_j}$ have opposite truth values; (ii) clausal satisfaction: every clause $C_i = (a_i \vee b_i \vee c_i)$ contains a true literal (a literal is a variable or its negation); (iii) consistency of truth assignments: every unnegated literal of a given variable is assigned the same truth value, either 1 or 0.

Lemma 4.1 *Autosegmental representations can enforce the 3SAT constraints.*

Proof. The idea of the proof is to encode negation and the truth values of variables in features; to enforce clausal satisfication with a local autosegmental process, such as syllable structure; and to ensure consistency of truth assignments with a nonlocal autosegmental process, such as a nonconcatenative morphology or long-distance assimilation (harmony). To implement these ideas we must examine morphology, harmony, and syllable structure.

Morphology. In the more familiar languages of the world, such as Romance languages, morphemes are concatenated to form words. In other languages, such as Semitic languages, a morpheme may appear more that once *inside* another morpheme (this is called infixation). For example, the Arabic word *katab*, meaning 'he wrote', is formed from the active perfective morpheme *a* doubly infixed to the *ktb* morpheme. In the autosegmental model, each morpheme is assigned its own plane. We can use this system of representation to ensure consistency of truth assigments. Each Boolean variable x_i is represented by a separate morpheme μ_i, and every literal of x_i in the string of formula literals is associated to the one underlying morpheme μ_i.

Harmony. Assimilation is the common phonological process whereby some segment comes to share properties of an adjacent segment. In English, consonant nasality assimilates to immediately preceding vowels; assimilation also occurs across morpheme boundaries, as the varied surface forms of the prefix *in-* demonstrate: *in+logical* → *illogical* and *in+probable* → *improbable*. In other languages, assimilation is unbounded and can affect nonadjacent segments: these assimilation processes are called harmony systems. In the Turkic languages all suffix vowels assimilate the backnesss feature of the last stem vowel; in Capanahua, vowels and glides that precede a word-final deleted nasal (an underlying nasal segment absent from the surface form) are all nasalized. In the autosegmental model, each harmonic feature is assigned its own plane. As with morpheme-infixation, we can represent each Boolean variable by a harmonic feature, and thereby ensure consistency of truth assignments.

Syllable structure. Words are partitioned into syllables. Each syllable contains one or more vowels V (its nucleus) that may be preceded or followed by consonants C. For example, the Arabic word *ka.tab* consists of two syllables, the two-segment syllable CV and the three-segment closed syllable CVC. Every segment is assigned a sonority value, which (intuitively) is proportional to the openness of the vocal cavity. For example, vowels are the most sonorous segments, while stops such as *p* or *b* are the least sonorous. Syllables obey a language-universal sonority sequencing constraint (SSC), which states that the nucleus is the sonority peak of a syllable, and that the sonority of adjacent segments swiftly and monotonically decreases. We can use the SSC to ensure that every clause C_i contains a true literal as follows. The central idea is to make literal truth correspond to the stricture feature, so that a true literal (represented as a vowel) is more sonorous than a false literal (represented as a consonant). Each clause $C_i = (a_i \vee b_i \vee c_i)$ is encoded as a segmental string $C - x_a - x_b - x_c$, where C is a consonant of sonority 1. Segment x_a has sonority 10 when literal a_i is true, 2 otherwise; segment x_b has sonority 9 when literal b_i is true, 5 otherwise; and segment x_c has sonority 8 when literal c_i is true, 2 otherwise. Of the eight possible truth values of the three literals and the corresponding syllabifications, only the syllabification corresponding to three false literals is excluded by the SSC. In that case, the corresponding string of four consonants C-C-C-C has the sonority sequence 1-2-5-2. No immediately preceeding or following segment of any sonority can result in a syllabification that obeys the SSC. Therefore, all Boolean clauses must contain a true literal. (Complete proof in Ristad, 1990) □

The direct consequence of this lemma 4.1 is:

Theorem 5 *PRP for the autosegmental model is NP-hard.*

Proof. By reduction to 3SAT. The idea is to construct a surface form that completely identi-

fies the variables and their negation or lack of it, but does not specify the truth values of those variables. The dictionary will generate all possible underlying forms (infixed morphemes or harmonic strings), one for each possible truth assignment, and the autosegmental representation of lemma 4.1 will ensure that generated formulas are in fact satisfiable. □

5 Conclusion.

In my opinion, the preceding proofs are unnatural, despite being true of the phonological models, because the phonological models themselves are unnatural. Regarding segmental models, the undecidability results tell us that the empirical content of the SPE theory is primarily in the particular rules postulated for English, and not in the extremely powerful and opaque formal system. We have also seen that symbol-minimization is a poor metric for naturalness, and that the complex notational system of SPE (not discussed here) is an inadequate characterization of the notion of "appropriate phonological generalization."[2]

Because not every segmental grammar g generates a natural set of sound patterns, why should we have any faith or interest in the formal system? The only justification for these formal systems then is that they are good programming languages for phonological processes, that clearly capture our intuitions about human phonology. But segmental theories are not such good programming languages. They are notationally-constrained and highly-articulated, which limits their expressive power; they obscurely represent phonological relations in rules and in the derivation process itself, and hide the dependency relations and interactions among phonological processes in rule ordering, disjunctive ordering, blocks, and cyclicity.[3]

Yet, despite all these opaque notational constraints, it is possible to write a segmental grammar for any decidable set.

A third unnatural feature is that the goal of enumerating structural descriptions has an indirect and computationally costly connection to the goal of language comprehension, which is to construct a structural description of a given utterance. When information is missing from the surface form, the generative model obligates itself to enumerate all possible underlying forms that might generate the surface form. When the generative process is lengthy, capable of deletions, or capable of enforcing complex interactions between nonlocal and local relations, then the logical problem of language comprehension will be intractable.

Natural phonological processes seem to avoid complexity and simplify interactions. It is hard to find an phonological constraint that is absolute and inviolable. There are always exceptions, exceptions to the exceptions, and so forth. Deletion processes like apocope, syncopy, cluster simplication and stray erasure, as well as insertions, seem to be motivated by the necessity of modifying a representation to satisfy a phonological constraint, not to exclude representations or to generate complex sets, as we have used them here.

Finally, the goal of enumerating structural descriptions might not be appropriate for phonology and morphology, because the set of phonological words is only finite and phrase-level phonology is computationally simple. There is no need or rational for employing such a powerful derivational system when all we are trying to do is capture the relatively little systematicity in a finite set of representations.

6 References.

Chandra, A., D. Kozen, and L. Stockmeyer, 1981. Alternation. *J. ACM* 28(1):114–133.

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

Halle, Morris. 1985. "Speculations about the representation of words in memory." In *Phonetic Linguistics, Essays in Honor of Peter Ladefoged*, V. Fromkin, ed. Academic Press.

Johnson, C. Douglas. 1972. *Formal Aspects of Phonological Description*. The Hague: Mouton.

Kenstowicz, Michael and Charles Kisseberth. 1979. *Generative Phonology*. New York:

[2] The explication of what constitutes a "natural rule" is significantly more elusive than the symbol-minimization metric suggests. Explicit symbol-counting is rarely performed by practicing phonologists, and when it is, it results in unnatural rules. Moreover, the goal of constructing the smallest grammar for a given (infinite) set is not attainable in principle, because it requires us to solve the undecidable TM equivalence problem. Nor does the symbol-counting metric constrain the generative or computational power of the formalism. Worst of all, the UTM simulation suggested above shows that symbol count does not correspond to "naturalness." In fact, two of the simplest grammars generate ϕ and Σ^*, both of which are extremely unnatural.

[3] A further difficulty for autosegmental models (not brought out by the proof) is that the interactions among planes is obscured by the current practice of imposing an absolute order on the construction of planes in the derivation process. For example, in English phonology, syllable structure is constructed before stress is assigned, and then recomputed on the basis of the resulting stress assignment. A more natural approach would be to let stress and syllable structure computations intermingle in a nondirectional process.

Academic Press.

McCarthy, John. 1981. "A prosodic theory of nonconcatenative morphology." *Linguistic Inquiry* 12, 373–418.

Minsky, Marvin. 1969. *Computation: finite and infinite machines.* Englewood Cliffs: Prentice Hall.

Ristad, Eric S. 1990. *Computational structure of human language.* Ph.D dissertation, MIT Department of Electrical Engineering and Computer Science.