Chapter 1

Intensional and Extensional Descriptions of Phonological Generalizations

Jeffrey Heinz

1.1 Generative Phonology

Within languages, the pronunciation of a morpheme often differs depending on the word in which it occurs. Examples like English *go/went* may indicate that these different pronunciations have almost nothing in common, it is much more typical that the pronunciations of the same morpheme in different words are in fact similar, as with common English plural *cat[s]/dog[z]*. The main empirical conclusion linguists have drawn is that the variation in the pronunciation of morphemes is *systematic*. It is no accident that the plural form of *tip* uses [s] just like *cat[s]* and that the plural form of *dud* is [z] just like *dog[z]*. Explaining this systematic variation is thus an important goal of linguistic theory.

The central hypothesis of Generative Phonology (GP) holds is presented below.

† The observed systematic variation in the pronunciation of morphemes is best explained if people hold a single mental representation of the pronunciation of each morpheme (the underlying representation, UR) which is *lawfully transformed* into its pronounced variants (the surface

representation, SR).

This book assumes this hypothesis is correct, and does not review any arguments for it.[1] Readers interested in arguments for this position are directed to Odden (2014, chapter 4) and Kenstowicz and Kisseberth (1979, chapter 6).

If this hypothesis is correct, then there are three questions every theory of generative phonology must address:

1. What is the nature of the underlying representations?
2. What is the nature of the surface representations?
3. What is the nature of the transformations between these representations?

These questions are certainly not exhaustive but they are centrally important. For instance, another important question "How different can the underlying representations be from the surface representations?" (the question of abstraction) has been raised and studied (Kenstowicz and Kisseberth, 1977).

This book provides a general framework which addresses these questions from a *computational* perspective. The computational perspective addresses both the nature of the representations and the nature of the transformations. It is flexible in the sense that different representational schemes can be studied and compared. This is accomplished through *model-theoretic* representations of words and phrases. It is also flexible in the sense that different types of computational power can be studied and compared. This is accomplished by studying what can be accomplished with logical expressions of different types. As will be explained, model theory and logic provide a mathematical foundation for theory construction, theory comparison, and even descriptive linguistics.

The study of phonology from the computational perspective allows one to construct theories of phonology which provide answers to the above questions. Representational choices and choices of logical power essentially determine the theory and its empirical predictions. Theories of phonology developed

[1] The words *transformed* and *transformation* are used here in their original meaning simply to signify that the URs become SRs, and that the SR derived from some UR may not be identical to this UR. If a UR is related to a SR via the transformative component of a phonological grammar, it is also often said the UR is mapped to the SR. These words are deliberately neutral with respect to the specific type of grammar being employed.

under this framework are examples of Computational Generative Phonology (CGP).

To begin motivating CGP, I would like to give some examples of how phonological theories aim to answer these questions. It is not possible to comprehensively survey here the range of answers that have been offered. Therefore, I only highlight some answers (and only in very broad strokes).

Rule-based theories, as exemplified by Chomsky and Halle (1968a), for example, have argued that the abstract underlying representations are subject to language-specific morpheme structure constraints (MSCs). The transformation from underlying forms to surface forms are due to language-specific rules, which are applied in a language-specific order. Constraints on surface representations were, generally speaking, not part of the ontology of these theories, and therefore were not posited to have any psychological reality. Such generalizations—the phonotactic generalizations—were derivable from the interaction of the MSCs and the rules.

On the other hand, in classic Optimality Theory (Prince and Smolensky, 1993, 2004), there are no constraints on underlying representations (richness of the base), but there are psychologically real, universal constraints on surface forms (markedness constraints). The transformation from underlying forms to surface forms is formulated as an *optimization* over these markedness constraints, in addition to constraints which penalize differences between surface and underlying forms (faithfulness constraints). While both the markedness and faithfulness constraints are universal, their relative importance is language-specific. So in every language the surface pronunciation of an underlying representation is predicted to be the optimal form (the one that violates the most important constraints the least). Of course what is optimal varies across languages because the relative importance of the constraints may vary across languages.

These two theories are radically different in what they take to be psychologically real. The ontologies of the theories are very different. Perhaps this is most clear with respect to the concept of phonemes (Dresher, 2011). Phonemes exist as a consequence of the ontology of rule-based theories, but they do not as a consequence of the ontology of OT. This is simply because phonemes are a kind of MSC; underlying representations of morphemes must be constructed out of them, and nothing else. In OT, there are no MSCs and hence there are no phonemes. The principle of Lexicon Optimization guarantees that the URs of *pit* and *spit* are [pʰɪt] and [spɪt], respectively (Kager, 1999). The underlying, mental representation of the voiceless labial stops in

both words are not the same. Consequently, the complementary distribution of speech sounds are explained in a very different manner in the two theories, and these theories promote different views of the notion of *contrast*. Despite these differences however, there is an important point of agreement: In both theories, complementary distribution of speech sounds in surface forms is the outcome of a transformation of underlying forms to surface forms.

This is the point I wish to emphasize: neither theory abandons the fundamental insight stated in 1.1.[2] The theories offer radical different answers to the questions in 1.1, but *they agree on the questions being asked*.

In the remainder of this chapter, I motivate a computational approach to phonology. I first make an important distinction between extensional and intensional descriptions of linguistic generalizations and argue that the former is important for understanding the latter. I then argue that neither rule-based or constraint-based formalisms as practiced provide adequate intensional descriptions of phonological generalizations.

This is then contrasted with automata and logical descriptions of language. The chapter concludes that logical descriptions of linguistic generalizations are preferable to automata-theoretic descriptions for several reasons. This is not to say automata are not useful (they are!) but that logic offers more in the short term to linguists interested in writing and analyzing grammars. So when we consider the ways in which we spend our time, logic is a good place to start.

1.2 Extensional and Intensional Descriptions

McCarthy (2008, pp. 33–34) emphasizes the importance of descriptive generalizations in preparing analyses. "Good descriptive generalizations," he writes "are accurate characterizations of the systematic patterns that can be observed in the data." They are, as he explains, "the essential intermediate step between data and analysis." This is because descriptive generalizations go beyond the data; they make predictions about things not yet observed.

Descriptive generalizations are important for computational phonology too. They are typically stated in prose. For example, consider the phonological generalizations below.

[2]It is true that periodically some work is published in that direction, for example the work on output-to-output correspondence (Benua, 1997, and others).

1. Word final vowels are prohibited.

2. Consonant clusters are prohibited word-finally.

These generalizations are good ones because they allow the analyst to recognize that potentially unobserved forms like *tapaka* is ill-formed but *tanak* is well-formed with respect to 1. Similarly, we recognize that 2 distinguishes between forms like *tapakt* and *tanakta*.

The generalizations above divide every word of every length cleanly into two sets: those that obey the description and those that do not. This is illustrated in the figure below. The set of words that is well-formed according

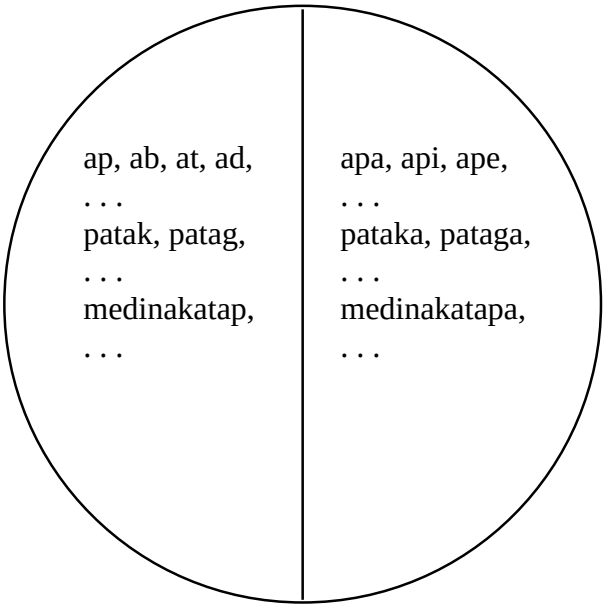

Figure 1.1: Generalizations about well-formedness partition the set of all possible forms.

to 1 is called the *extension* of 1.

Importantly, this set—the extension—is infinite in size. For instance, it is not possible to write down every word that obeys 1. If a set of words formed from a finite alphabet is infinite then there is no upper bound on the length of words. Likewise, if there is no upper bound on the length of words, then the set of words formed from a finite alphabet is also infinite. Thus whether the size of a set of words is infinite or not is intertwined with whether or

not there is an upper bound on the length of words. These issues are so important to get clear that they are discussed in further detail below.

Extensional descriptions contrast with *intensional descriptions* of generalizations. For now, intensional descriptions can be thought of as grammars that denote the extension. The prose in 1 and 2 are examples of intensional descriptions. Rule-based grammars and OT grammars are also examples of intensional descriptions. A good intensional description is one where the the extension can be rigorously and precisely defined from the intensional description. Generally, English prose does not make for good intensional descriptions. Further below, I will argue that in their current forms and practice, rule-based grammars and OT grammars are more like English prose than good intensional descriptions.

Let us now return to the infinitely-sized extensions. Is it reasonable for descriptive generalizations 1 to denote an infinite set of words? Yes, it is. One reason is that these generalizations make no reference to length at all. If the length of words mattered, it ought to be part of the generalization. Another way of thinking about this is that if there were a principled upper bound on the length of words, then that would be a generalization *distinct* from 1 above, and hence ought not be included within it. Finally, even if for some reason 1 ultimately denoted a finite set, there are reasons to treat its extension as infinite anyway. Savitch (1993) argues that large finite sets of strings are often best understood if they are factored into two parts: an infinite set of strings and a separate finite-length condition. They are, in his words, "essentially infinite." The basis of the argument is a demonstration that intensional descriptions of infinite sets can be smaller in size than the intensional descriptions of finite sets.

These infinite-sized extensions do not exist in the same way that your fingernails, your bed, or your brain exists. Instead they exist mathematically. Each generalization is an infinite object like a circle, a set of infinitely many points each exactly the same distance from a center. But we can never see the mathematical object in its entirety in the real world. It is a fact that circles as infinite objects do not exist. The situation with linguistic generalizations is similar. The extension is there mathematically, but we cannot write down every element of the extension in a list for the same reason all points of a circle cannot be written down in a list since there are infinitely many. But we can write down a grammar which can be understood as generating the infinite set, in the same way that a perfect circle can be generated by specifying a center point and a distance, the radius.

The same circle can be described in other ways as well. If we employ the Cartesian plane, we could generate a circle with an equation of the form $(x-a)^2+(y-b)^2=r^2$ where the r is the radius of the circle and (a,b) is its center. The equation is interpreted as follows: all and only points (x,y) which satisfy the equation belong to the circle. The equation is an intensional description and the set of points, the circle, is its extension.

We can also describe a circle on a plane with polar coordinates instead of Cartesian ones. Recall that polar coordinates are of the form (r,θ) where r is the radius and θ is an angle. The equation $r=2a\cos(\theta)+2b\sin(\theta)$ provides the general form of the circle with the radius given by $\sqrt{a^2+b^2}$ and the center by (a,b) (in Cartesian coordinates). The polar equation is interpreted like the Cartesian one: all and only points (r,θ) which satisfy the equation belong to the circle.

There are some interesting differences between these two coordinate systems. Each point in the Cartesian system has a unique representation, but each point in the polar system has infinitely many representations (since the same angle can be described in infinitely many ways, e.g. $0^\circ = 360^\circ = 720^\circ = \ldots$). If the center of the circle is the origin, the polar equation simplifies to $r=a$ whereas the Cartesian equation remains more complicated $x^2+y^2=r^2$. Thus, the polar equation $r=4$ and the Cartesian equation $x^2+y^2=16$ are different equations with different interpretations, but they describe the same unique circle: one of radius four centered around the origin. The two equations differ intensionally, but their extension is the same.

It seems strange to ask which of these two descriptions is the 'right' description of a circle. They are different descriptions of the same thing. Some descriptions might be more useful than others for some purposes. It also interesting to ask what properties the circles have irrespective of a particular description. For instance the length of the perimeter and the area of a circle are certainly relatable to these descriptions, but they are also in a sense independent of the particulars. The perimeter and area depend on the radius but not the center, though both the radius and the center appear in the equations. This suggests that the radius is a more fundamental structure to a circle than its center, though both certainly matter.

The analogy I wish to draw is that rule-based and OT-theoretic formalisms are like the Cartesian and polar systems. The analogy is far from perfect, but it is instructive. Both rule-based and OT analyses provide descriptions of platonic, infinitely sized objects. In many cases, but not all, the two formalisms describe the same object, insofar as the empirical evidence

allows.

What is this object? The transformations from underlying forms to surface forms can be thought of as a *function*, in the mathematical sense of the word. Another word for function becoming prevalent in the phonological literature is *map* (Tesar, 2014). For example, consider the descriptive generalizations below.

1. Word final vowels delete.

2. Word final vowels delete except when preceded by a consonant cluster.

These generalizations also have infinite-sized extensions, but the extensions are better understood as functions.

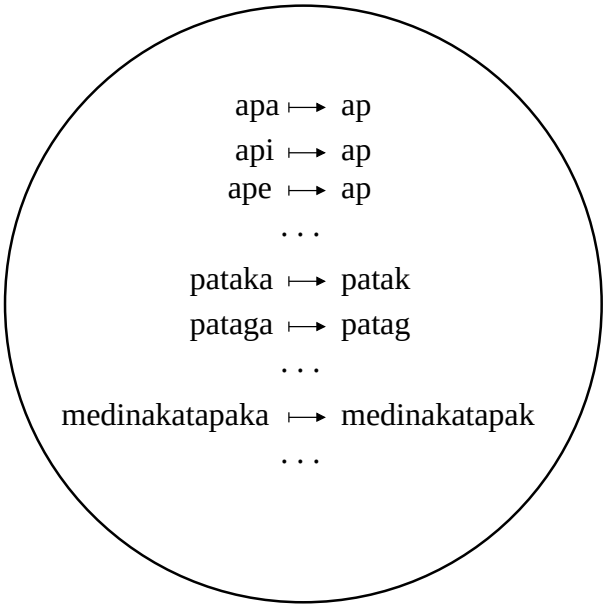

Figure 1.2: Generalizations about transformations are functions.

There are three parts to a function. One, there is its domain, which is the set of objects the function applies to. Two, there is its co-domain, which is the set of objects to which the elements of the domain are mapped. Three, there is the map itself, which says which domain elements are transformed to which co-domain elements. Thus to specify a function, one needs to provide a description of its domain, its co-domain, and a description of which

domain elements become which co-domain elements. Following traditional phonological terminology, I use the term *constraint* to refer to intensional descriptions of either the domain or co-domain.

This lines up nearly perfectly with the fundamental questions of phonological theory. The underlying representations correspond to the domain. The surface representations are the co-domain. And the transformation from underlying to surface forms is the map from domain elements to co-domain elements. From this perspective, describing the phonology of a language requires describing aspects of this function.

Further, in linguistic typology we are actually interested in the *class* of such functions that correspond to *possible* human phonologies. If the phonologies of languages are circles we would be interested in the universal properties of circles and the extent of their variation. Circles are pretty simple, so the answers are straightforward. All circles have a center and a radius, but their centers can be different points and their radii can have different lengths. What universal properties do phonological functions share? What kind of variation does the human animal permit across these functions?

The point is that when we develop a linguistic generalization, it is important to know what its extension is. Ultimately, the intensional (grammatical) description we provide must generate this extension. The emphasis placed here on the extensional description as an infinite object should not be taken to mean intensional descriptions do not matter. Of course they matter – theories of these intensional descriptions ought to make predictions about what is psychologically real, predictions that in principle are testable with the right kinds of psycholinguistic and neurolinguistic experimentation. They also make predictions about linguistic typology: the available intensional descriptions limit the extensions accordingly. In addition to making correct predictions, phonologists expect that intensional descriptions express the 'right' generalizations.

Extensional descriptions are an essential, intermediate step between the prose descriptive generalizations and the formal intensional descriptions (the grammatical analysis).

It is critically important that it is well-understood how the intensional descriptions relate to the extensional ones. We want to be able to answer questions like the following:

1. Given a word w and an intensional description of a constraint C, does w violate C? (We may also be interested in the number of violations

and their locations in the word.)

2. Given a word w in the domain of a transformation f what words in the co-domain of f does f map w to, if any?

3. Given a word v in the co-domain of a transformation f what words in the domain of f map to v, if any?

Question 1 is often called the membership problem. Question 2 is often called the generation problem. Question 3 is often called the recognition or parsing problem. Good intensional descriptions allow answers to these questions to be computed effectively. In the next section, I argue that rule-based intensional descriptions and OT grammars are not good intensional descriptions in this narrow sense.

1.3 Issues with Familiar Grammars

Chomsky and Halle (1968b) present a formalization based on rewrite rules. The basic rewrite rule is of the form A $\longrightarrow$ B / C $\_\_$ D . This notation is intended to mean that if an input string contains CAD then the output string will output CBD (so A is rewritten as B in the context C $\_\_$ D). To understand the extension of a rule, we need to know how to apply it. Originally, Chomsky and Halle (1968a, p. 344) intended for the rules to apply simultaneously to all the relevant targets in an input string. They wrote, "To apply a rule, the entire string is first scanned for segments that satisfy the environmental constraints of the rule. After all such segments have been identified in the string, the changes required by the rule are applied simultaneously." For many phonological rules, this explanation appears sufficient to denote the extension. For instance the rules corresponding to the descriptive generalizations (1) is V $\longrightarrow \varnothing$ / $\_\_$ # . Humans have no difficulty using this rule to answer the generation and parsing problems above given this intensional description. However, it is much less clear what the extension of *any* rule would be.[3]

The phonological literature after SPE addressed the question of rule application (Anderson, 1974), and other types of rule application were identified such as left-to-right or right-to-left. It was clear that the mode of application determined the extension of the rule. For example, for the input string

[3]Of course this depends in part on what A, B, C and D themselves are able to denote.

iana and rule V $\longrightarrow$ [+nasal] / $\_\_$ [+nasal] simultaneous application yields output *iãna* but right-to-left application yields output *ĩãna*. While linguistically-chosen examples served to distinguish one mode of application from another, general solutions to the generation and recognition questions by Johnson (1972) and Kaplan and Kay (1994) were for the most part ignored by generative phonologists.

It is my contention that rule application is still not well-understood by most students of phonology, despite the careful computational analyses by Johnson (1972); Kaplan and Kay (1994) and Mohri and Sproat (1996). In informal surveys of phonologists in-training, many have difficulty of applying the rule aa $\longrightarrow$ b simultaneously to the input *aaa*. People wonder whether the right output is *ab*, *ba*, or *aa*. According to Kaplan and Kay's analysis, there are two outputs for this input when the rule aa $\longrightarrow$ b is applied simultaneously. They are *ab* and *ba*. Their analysis translates rewrite rules into finite-state automata, which are grammars whose extensions are very well defined and understood. These will be explained in a bit more detail in the next section.

Interestingly, Kaplan and Kay's analyses of rule application, which has been implemented in software programs like `xfst` (Beesley and Kartunnen, 2003) and `foma` (Hulden, 2009a,b), do not exhaust the possible natural interpretations of the rewrite rule A $\longrightarrow$ B / C $\_\_$ D . Like Johnson and Kaplan and Kay's analyses, Chandlee's (2014) analysis also uses finite-state automata to determine an extension of a rule A $\longrightarrow$ B / C $\_\_$ D , provided that CAD is a finite set of strings. Unlike Kaplan and Kay, her interpretation of the extension of the rule aa $\longrightarrow$ b maps input *aaa* to *bb*. This result is arguably what Chomsky and Halle in mind when they described simultaneous application because each *aa* sequence satisfies "the environmental constraints of the rule."

The point of the foregoing discussion is simply this: a rule A $\longrightarrow$ B / C $\_\_$ D underdetermines its extension. The extensions are a critical part of any rule-based theory and there is more than one way such rules determine extensions. This point is not news nor is it controversial. It is a well-known chapter in the history of phonological theory. Chandlee (2014) shows a good understanding of the extensions of SPE-style rules is not a closed chapter in a phonological theory based on rewrite rules. Bale and Reiss (2018) may be the first textbook on phonology that provides an adequate interpretation of the application of rewrite rules.

Optimality Theory is an improvement in some sense. Given an OT gram-

mar, and an input form there is a well-defined solution to the generation problem. This solution follows from the architecture of the OT grammar. The GEN component generates the set of possible candidates and the EVAL component uses the grammar of ranked constraints to select the optimal candidates.

Nonetheless in actual phonological analyses the generation problem faces two difficulties, each acknowledged in the literature. The first one is ensuring that all the possible candidates are actually considered by EVAL. The absence of an overlooked candidate can sink an analysis. The proposed optimal candidate turns out to be less harmonic than some other candidate that the analysts failed to consider. How can analysts ensure that every candidate has been considered?

The second is ensuring that all the relevant constraints are present in the analysis. The absence of a relevant constraint can also sink an analysis. (Prince, 2002, p. 276) makes this abundantly clear. He explains that if a constraint is ignored that must be dominated by some other constraint then the analysis is "dangerously incomplete." Similarly, if a constraint is omitted that may dominate some other constraint then the analysis is "too strong and may be literally false."

As a result, any phonological analysis of a language which does not incorporate the entire set of constraints is not guaranteed to be correct. This makes studying some aspect of the phonology of the language difficult. The constraints deemed irrelevant to the fragment of the phonology under investigation (and which are therefore excluded) actually need to be shown to be irrelevant for analysts to establish the validity of their analyses.

Both these problems in OT can be overcome. The solution again comes from the theory of computation, in particular from the theories of finite-state automata and so-called regular languages (defined and discussed in the next section). The primary result is that even if the constraints and GEN can be defined in these terms, the maps OT produces are not guaranteed to be definable in these terms — unless the constraints have a finite bound on the maximum number of violations they can assign (Frank and Satta, 1998). Karttunen (1998) uses this fact to provide a solution and software for the generation and recognition problems (see also (Gerdemann and Hulden, 2012)), and he assumes each constraint has some maximum number of violations. While some theoretical phonologists have argued for this position (McCarthy, 2003), most do not adopt it. Riggle (2004) provides a different solution which does not require bounding the number of violations

constraints assign. His solution is guaranteed to be correct provided the map the OT grammar is in fact representable as a finite-state relation (not all of them are). Another solution is present in Albro's (2005) dissertation, which provides a comprehensive OT analysis of the phonology of Malagasy.

Each of these authors make use of finite-state automata to guarantee the correctness of their solutions. However, none of these approaches have yet to make its way into the more commonly used software for conducting OT analyses such as OTSoft (Hayes *et al.*, 2013), OT-Help (Staubs *et al.*, 2010), and OTWorkplace (Prince *et al.*, 2016). A particular weakness of this software, unlike Karttunen's, Riggle's, and Albro's is that they can only work with finite candidate sets, despite the fact that GEN is typically understood as generating an infinite candidate set. Consequently, the commonly used software amounts to nothing more than pen-and-paper approaches with lots of paper and lots of pens, and so the aforementioned issues remain (Karttunen, 2006).

McCarthy (2008, p. 76) argues the aforementioned computational approaches are only possible in "narrowly circumscribed phenomenon," which ignores Albro's detailed, thorough analysis of the whole phonology of Malagasy. He also argues the methods are only as good as the algorithm that generates the candidates. That may be true, but the alternatives are manual, heuristic methods.[4] People may differ on which is better, but I will place my bets on the algorithm which is guaranteed not to leave out candidates that GEN produces. McCarthy's dismissal of the value of computational approaches is unfortunate, but it is representative of attitudes in the field.

Regardless of the extent the which different researchers appreciate the computational treatments of phonological theories, it is noteworthy and no accident that every attempt to guarantee a solution of the recognition and generation problems (and the membership problem when constraints are involved) makes use of finite-state automata and the theory of regular languages. Even OTWorkplace employs the finite-state calculus (with regular expressions) to automatically assign candidates constraint violations. What are these devices and what makes them so good for denoting extensions of generalizations?

[4]It is true that the GEN function in the Albro's, Karttunen's, and Riggle's methods is not exactly the same as the one assumed in Correspondence Theory (McCarthy and Prince, 1995), but it is instructive to understand why.

1.4 Computational Theory of Language

Automata are a cornerstone of the computational theory of language. Automata are machines that process specific types of data structures like strings or trees. They form a fundamental chapter of computer science. There are many kinds of automata. The Turing machine is just one example. Pushdown automata are another. Readers are referred to texts such as Sipser (1997) and Hopcroft *et al.* (2001) for overviews of the theory of computation.

There are also deep connections between automata and logic. In this section, I will briefly review finite-state automata for string processing. Then I will informally introduce logic as another way of providing an intensional description of phonological generalizations. Their extensions are also well-defined; and in fact in many cases there are algorithms which convert a logical description into an automaton that describes exactly the same extension.

We begin with a simple automaton, the finite-state acceptor. It is an intensional description with a well-defined extension. As a matter of fact, it is a precise finite description of a potentially infinite set of strings.

A finite-state acceptor contains a finite set of states. We give the states names so we can talk about them; for instance they are often indexed with numbers. Some states are designated 'start' states. Some states are designated 'accepting' states. (Some states can be both 'start' and 'accepting' states.) Transitions lead from one state to another; they are labeled with letters from some alphabet. That's it. So a finite-state acceptor is of finite size. What is its extension? Well the extension is defined as follows. A word w is accepted/generated/recognized by a finite-state acceptor A if there is a path along the transitions of A which begins in a start state of A, which ends in a final state of A, and which spells out w exactly.

As an example, consider Figure 1.3, which shows the finite-state acceptor for the generalization that word-final vowels are prohibited. Per convention, the start state is designated by the unanchored incoming arrow and final states are marked with a double perimeter. The word *nok* is generated by this machine since there is a path beginning in a start state and ending in a final state which spells it out. This path is shown below.

Input:		n		o		k	
States:	0	$\rightarrow$	1	$\rightarrow$	0	$\rightarrow$	1

A minute of inspection reveals that every path for every word which ends in a vowel ends in state 0, which is not an accepting state. But every path for

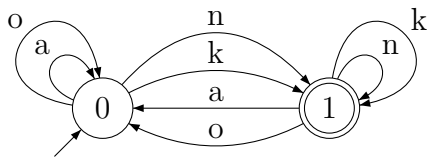

Figure 1.3: A finite state acceptor for the generalization "Word final vowels are prohibited." A simple alphabet {n,k,a,o} is assumed.

every word which does not end in a vowel ends in state 1, which is accepting. Algorithms which solve membership problem for finite-state acceptors are well understood (Hopcroft *et al.*, 2001).

Finite-state automata are not limited to acceptors. String-to-string functions can be described with automata that are called transducers. These are acceptors whose labels have been augmented with an additional coordinate. Labels are now pairs instead of a single point. Figure 1.4, which shows the finite-state acceptor for the generalization that word-final vowels delete. As before, valid paths through this machine (those that begin in start states and end in accepting states) spell out input words and the output word they map to. In the figure, the colon separates the left coordinate (input) from the right coordinate (output). The symbol λ denotes the empty string. To

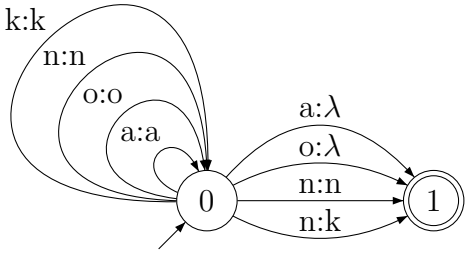

Figure 1.4: A finite state acceptor for the generalization "Word final vowels delete." A simple alphabet {n,k,a,o} is assumed.

illustrate, consider the path which shows that the output of *nako* is *nak*.

Input:		n		a		k		o	
States:	0	→	0	→	0	→	0	→	1
Output:		n		a		k		λ	

As with the membership problem and finite-state acceptors, there are algorithms which solve the generation and recognition problems for finite-state transducers.

There are some interesting things to observe about the finite-state transducer. The first is that it is non-deterministic. This means for a given input, there is more than one path. For instance, the input *nok* maps to *nok*, and there are two paths that spell it out. But only one is valid: the one that reads and writes *k* and moves from state 0 to state 1.[5]

Another point is that the transducer in Figure 1.4 maps the input word *nakao* to *naka*. Thus, this machine provides the extension of the rule V $\longrightarrow \varnothing$ / $\_\_$ # applying simultaneously. In OT, if Final-C outranks Max, then the output would be *nak* with the last two vowels deleting. With rules, this could be accomplished by applying the former rule right-to-left. The finite-state transducer shown in Figure 1.5

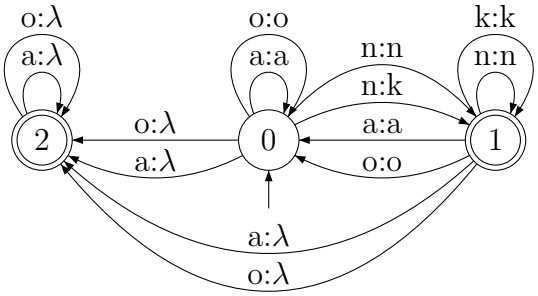

Figure 1.5: A finite state acceptor for the generalization "Strings of vowels word-finally delete." A simple alphabet {n,k,a,o} is assumed.

Transducers can also map strings to numbers. The simple one shown in Figure 1.6 counts the number of *o*s in a word. The idea here is that instead of combining the output side of valid paths with concatenation as for strings, they are combined with addition. Below is an example of the only valid path for the word *naoko* which would be mapped to 2.

Input:		n		a		o		k		o	
States:	A	→	A	→	A	→	A	→	A	→	A
Output:		0		0		1		0		1	

This is exactly the approach used by Riggle (2004) to define markedness and faithfulness constraints in OT. Again, the extension of the transducer in Fig-

[5]Non-determinism is one way optionality can be handled with finite-state transducers. If state A was also an accepting state then there would be two valid paths for the input *noko*: one would write the output *nako* and the other the output *nak*.

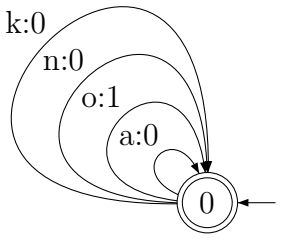

Figure 1.6: A finite state transducer which counts the number of *o*s in words. A simple alphabet {n,k,a,o} is assumed.

ure 1.5 is precisely defined and the corresponding generation and recognition problems solvable.

There are many generalizations of this kind available to transducers made possible by the study of semirings (Droste and Kuich, 2009; Goodman, 1999). Semirings are discussed in more detail in Chapter 2.

What of the recognition problem? Another important advantage of finite-state automata is that they are *invertible*. Consequently, a solution to the generation problem entails a solution to the recognition problem. Given a string *nak*, the transducer can tell you that it is the output of the any of the following inputs: *nak, naka, nako*.

Nonetheless, despite the advantage of a well-defined extensions, there are some shortcomings to using finite-state automata for phonological analyses. One is that letter of the alphabet are treated atomically. For instance, there is no sense in which the symbols [p,t,k] share any properties. It remains unclear how to incorporate phonological features and natural classes in a natural way into these machines. The most common way seems to just group the letters together that behave together as I have done int he examples above. While this is certainly sufficiently expressive, it is not satisfying. We want our intensional descriptions to somehow speak directly to the descriptive ones. In the case of "Word final vowels are prohibited" we want to be able to express this directly.

Another drawback is that as the generalizations become more complex, so do the finite-state automata. They become spaghetti-like and difficult to read. This drawback is mitigated, however, in a couple of ways. The first is that it is very well understood how to combine different finite-state automata to produce new ones. This allows the generalizations instantiated by the 'primitive' ones to persist to some degree in the complex ones. For

instance, it is straightforward to construct a finite-state acceptor that generates exactly the intersection of two infinite sets of strings which are generated by two acceptors. Similarly, it is straightforward to construct a finite-state transducer that generates the composition of two functions which are generated by finite-state transducers. In this way, more complex finite-state automata can be constructed from simpler parts, much in the same way more complex phonological grammars are built up from identifying generalizations that interact in some manner.

A third problem is that even simple machines are not easy to write in text. They are often pictured as diagrams, and in the same way it can be tiring to read them, it can be tiring to draw them as well. This problem is mitigated in a couple of ways. Some researchers use tables or matrix notation, others use regular expressions (Beesley and Kartunnen, 2003; Hulden, 2009b), and still others use logic.

In this book, we are going to use logic and not automata to represent linguistic generalizations. There are several reasons for this. Most importantly, like automata, the extensions of logical formula are precisely defined. Another key reason is that the representations are *flexible*. We can represent words exactly as any phonologist would want to. As this book will show, phonological features, syllable structure, autosegmental representations, phonetic information, and a host of as-of-yet unconsidered possibilities are available and directly representable with logic. Thirdly, as this book will show, the combination of logical power and representation provides a natural way to entertain distinct theories of phonology and compare them. Additionally, there is a literature showing how logical formula can be translated into automata which are equivalent in the sense that they solve the same membership, generation, and recognition problems. While this literature does not address every phonological representation proposed, the basic analytical methods which show how this can be done for strings and trees are there.

Finally, logic is not going anywhere. This is very important. If a linguist describes a generalization with logic and the representations they want, they can be guaranteed that people in will be able to read their description and understand it *hundreds of years later*.

In short, logical formula have all of the advantages, and none of the disadvantages, of automata.

1.5 Doing Computational Phonology

How does one do computational generative phonology? This book provides an answer.

In the first part, logical foundations and model theory are presented in the context of strings. It is explained how model theory allows one to precisely describe different representations of words and phrases. It is explained how the primitive elements in these representations would have ontological status in the theory. It is also explained how logical expressions can be used to define constraints to delimit possible representations in words and phrases and transformations to show how one representation is mapped to another. It is explained how weighted logical expressions allow ones to express a variety of linguistic generalizations, including gradient ones. These definitions and techniques are illustrated with examples drawn from phonology, as well as examples showing the terrific expressivity of the framework. The first part of this books opens a large umbrella of techniques and possibilties.

In the second part, these techniques are applied to the kinds of phonology problems one finds in standard textbooks on phonology. The focus here is descriptive in the following sense. The linguist marshalls arguments from a collection of linguistic forms before her in favor of particular linguistic generalizations. These arguments are presented and then the linguistic generalizations are formalized in terms of model-theoretic representations and logic. The chapters are short, each dealing with a relatively small and straightforward phonological problems. These examples serve as models for how analysis of other small and straightforward can be analyzed within CGP.

In the third part, the chapters address a variety of theoretical issues addressing both aspects of representation and computional power. Sebastian shows how to incorporate insights from phonetically-based phonology into CGP representationally. Hwangbo shows how representing vowel height in terms of degrees of aperture leads to straightforward analysis of vowel lowering. Strother-Garcia analyzes syllable structure and the sonority sequencing principle. Rogers and company show how the stress patterns in the world's languages can be understood as particular the combination of primitive constraints, characterizes their complexity and identifies sources of complexity. Lindell and Chandlee provide a logical characterization of Input Strictly Local functions, which Chandlee showed earlier to well-characterize an important natural class of phonological transformations. Deovletian shows that the Raimy-style linearization is computationally very complex. Once the

source of the complexity is identified, he suggests way to mitigate it. Payne shows the comptuational complexity of GEN is also very complex. Vu shows how transformations can also be expressed as constraint on correspondence structures. These chapters are but a small sample of the kinds of research questions and investigations that can be addressed with the tools introduced in part one.

Computational generative phonology is not hard. We believe theories of generative phonology developed in this tradition will lead to advances in our understanding of the nature of phonological grammars and the minds which know them.

