In *Word Stress: Theoretical and Typological Issues*. Harry van der Hulst, ed. Cambridge University Press. 2014.

8 Culminativity times harmony equals unbounded stress

Jeffrey Heinz

8.1 Introduction

Well-studied computational representations of stress patterns are desirable in phonological analysis for several reasons.* Perhaps the most important one is that they reveal insights which are (i) relevant to *any* particular theory of phonology and (ii) otherwise difficult to divine. This is because these representations highlight the importance of *what* is being computed and are less concerned with *how* it is being computed. This chapter demonstrates one example of this by showing concretely how once culminativity is factored out, simple unbounded stress patterns over syllables are of the same formal character as simple harmony systems over consonants and vowels. This analysis thus shows that long-distance phenomena in two seemingly different domains are actually of the same kind.

This chapter defines culminativity as 'exactly one stress per word', which is one way in which the term has been understood in the past (see, for example, Hayes 1995). In other words, culminativity here encompasses *both* the 'obligatory' (at least one stress per word) and 'culminative' (at most one stress per word) properties discussed by Hyman (this volume).

This chapter begins by reviewing what is known about unbounded stress patterns, and then explains the advantages of such computational representations. Next, this chapter reviews simple segmental harmony systems and their computational representations. Finally, it is shown how factoring culminativity out of the unbounded stress patterns yields patterns of the same formal character as simple segmental harmony systems. An 'Appendix' is included which precisely defines the formalisms used in this chapter.

* I thank James Rogers for valuable discussions and inspiration, William Idsardi for important feedback, and Harry van der Hulst and two anonymous reviewers for their comments on, and very careful reviewing of, earlier versions of this paper. This research has been supported by NSF grant no. 1123692.

8.2 Stress patterns

Cross-linguistic investigations of the dominant stress patterns in languages have typically focused on predicting the location of stresses given the syllabic make-up of words (Hyman 1977; Halle and Vergnaud 1987; Idsardi 1992; Hayes 1995; Goedemans *et al.* 1996; Tesar 1998; Gordon 2002; van der Hulst *et al.* 2010). To illustrate, let L and H denote *light* and *heavy* syllables, respectively. Consider a word with the syllable profile L H H. Latin predictably assigns stress to such words to the penultimate syllable (Jensen 1974; Odden 1994), whereas Yapese predictably assigns stress to the final syllable (Jensen 1977).[1]

Both Latin and Yapese may have some words which are unexpected exceptions to the rule. Nonetheless, phonologists have identified the generalization "If the penult is a heavy syllable then it gets stress" as capturing part of the stress rule in Latin. Similarly, the rule "If the final syllable is heavy then it gets stress" is a crucial part of the generalization which describes the dominant stress pattern of Yapese. Lexical exceptions are usually noted in typological analyses, even if the analyses themselves do not address them directly. Following these earlier studies, this chapter likewise abstracts away from lexical exceptions and other such variation in order to focus exclusively on the nature of the generalizations.

8.3 Unbounded stress patterns and computational analysis

Unbounded stress patterns can be defined as those where the generalization describing primary stress placement within some domain does not require that stress fall within a certain fixed distance of either edge of the domain. Unbounded analyses of the stress pattern of several languages have been presented, including Amele, Kwakwala, Chuvash, and Golin (Hayes 1995). As an example, consider the unbounded stress pattern in Kwakwala (Walker 2000).

(1) Primary stress falls on the leftmost heavy syllable in a word, and if there are no heavy syllables, on the final syllable.

Following Hayes (1995), I will refer to this pattern as the 'Leftmost Heavy Otherwise Rightmost' (LHOR) pattern. According to (1), in words with syllable profiles LLH, LLHL, and LLHLH, the primary stress will always fall on the third syllable because that is the leftmost heavy syllable in each word. Table 8.1 shows all words up to four syllables in length which exemplify this pattern. This pattern is unbounded because the primary stress could fall arbitrarily far from

[1] In Latin, syllables with codas and long vowels are considered heavy, whereas in Yapese only syllables with long vowels are heavy. Although which syllables are heavy and light is determined on a language-specific basis, such determinations are not wholly arbitrary. Readers are referred to Gordon (2006) for factors which determine syllable weight.

Table 8.1 *All LHOR words up to four syllables in length*

H́	Ĺ	H́ L	H́ H	L H́
L Ĺ	H́ L L	H́ L H	H́ H L	H́ H H
L H́ L	L H́ H	L L Ĺ	L L H́	L H́ L L
L H́ L H	H́ L L L	H́ L L H	H́ H L L	H́ H L H
L H́ H L	L H́ H H	H́ L H L	H́ L H H	H́ H H L
H́ H H H	L L H́ L	L L H́ H	L L L Ĺ	L L L H́

either word edge. For example, in words with the syllable profile LLLHLLH, stress is predicted to fall on the fourth syllable.

It is important to realize that the generalization in (1) applies equally well to longer words, like the one just given, even if no such words of that length exist in the lexicon (or are constructible by word-formation rules). For the same reason that a rule of word-final obstruent devoicing applies equally well to words hundreds of syllables long as it does to words one syllable in length, there are infinitely many logically possible words which obey the LHOR pattern. The words in Table 8.1 are just among the shortest words drawn from this set. In other words, in every case, the generalizations that phonologists make when describing the competence of native speakers with respect to the dominant stress patterns in languages can be thought of as infinite sets. Our interest in the nature of these phonological generalizations leads us to examine the nature of these mathematical objects – the infinite sets with which these generalizations are identified.

Theoretical computer science provides mathematically rigorous ways to classify sets of infinite size (Harrison 1978). For example, the Chomsky Hierarchy classifies all logically possible sets into the following nested regions.

(2) finite $\subset$ regular $\subset$ context-free $\subset$ context-sensitive $\subset$ recursively enumerable

Each of these regions has multiple, independently motivated, converging characterizations which makes these boundaries useful in pattern classification (Harrison 1978; Hopcroft *et al.* 2001).

Interestingly, there are no attested stress patterns which are non-regular. All known stress patterns belong to the *regular* region (Heinz 2009) and thus 'being regular' is a *universal* property of known stress patterns. 'Being regular' means the infinite sets of words which exemplify these patterns can be represented exactly with certain kinds of grammar.

I will use different grammatical formalisms in this chapter, but I will make special use of *finite-state acceptors* (FSAs). Any regular set can be represented

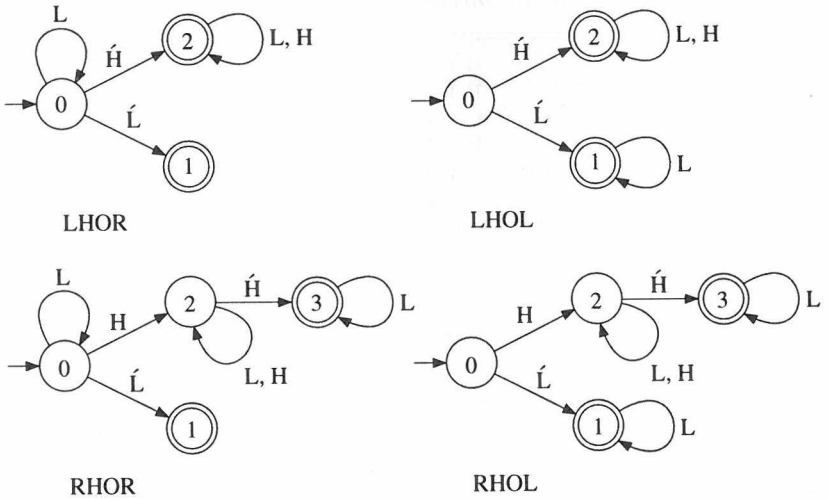


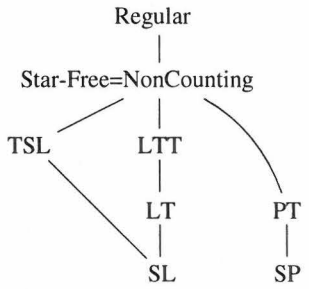
Figure 8.1 Finite-state acceptors for the four basic unbounded stress patterns

exactly with a finite-state acceptor[2] and they can be manipulated in well-understood ways (Hopcroft *et al.* 2001). One such manipulation, the product operation, plays a key role and will be discussed in section 8.6. (The 'Appendix' defines formal languages, finite-state acceptors, and the product operation.)

Representing the phonological generalizations with finite-state acceptors emphasizes that any grammar only ever need distinguish finitely many states even though the generalizations describe infinite sets. The states in these acceptors can be thought of as a kind of memory. Each state identifies some kind of information that is important to keep track of. To illustrate, Figure 8.1 shows finite-state acceptors for each of the four basic types of unbounded stress pattern.

It is important to see how the finite-state representation relates to the words exemplifying each pattern. Consider the acceptor for LHOR shown in Figure 8.1 and imagine a pebble placed on state 0. This state has an incoming arrow from the left, which indicates this is the *initial* state. Now one can move the pebble along any transition emanating from state 0. As one moves the pebble along a particular transition, the label of that transition is written out. (Some transitions in the acceptors above have more than one label separated by commas, in which case one may choose which label gets spelled out.) When the pebble is in a state with two double circles the process may end. Such states are called *final* states. For example, in acceptor LHOR in Figure 8.1, states 1 and 2 are

[2] In fact, this can be taken to be their defining property.



Key	
TSL	Tier-based Strictly Local
LTT	Locally Threshold Testable
LT	Locally Testable
SL	Strictly Local
PT	Piecewise Testable
SP	Strictly Piecewise

Figure 8.2 Proper inclusion relationships among subregular language classes (indicated from top to bottom)

final states. Once the pebble is finished moving, one can see which word has been spelled out. The acceptor is said to *accept*, or equivalently *generate* or *recognize*, every word that can be spelled out by a pebble which begins at the initial state and ends at any final state.

For example, beginning in state 0, let us move the pebble along the arc labeled 'L', which brings the pebble back to state 0. Next, we move the pebble along the arc labeled 'H' to bring it to state 2. Lastly we move it along the arc labeled 'H' to bring it to state 2 again. Here we decide to stop since state 2 is a final state. The word that has been spelled out is 'LḢH' which exemplifies the LHOR pattern. A little experimentation will show that every word which exemplifies the LHOR pattern can be generated by the LHOR acceptor and every word which does not cannot be generated by this acceptor.

The above procedure inadvertently suggests that the FSAs are modeling language processing as opposed to speakers' competence. While the above FSAs could be used as a processing model, that is not how they are being used here. Here, they are being used solely to represent an infinite set that faithfully captures the relevant phonological generalization.

The hypothesis that all humanly possible stress patterns are regular is thus well supported, since all known stress patterns can be described with finite-state acceptors of the above type. This hypothesis should not be misunderstood to mean that any regular pattern is a possible stress pattern. On the contrary, the claim is that 'being regular' is a *necessary* property, but not a *sufficient* one (Heinz 2011a, 2011b).

Much stronger hypotheses can be formulated. There are several *subregular* regions which have been defined according to established language-theoretic, logical, and cognitive criteria (McNaughton and Papert 1971; Simon 1975; Rogers *et al.* 2010; Rogers and Pullum 2011). The most important of these, and their relationships to each other, are shown in Figure 8.2.

It is possible to classify stress patterns according to these regions, which provides a measure of pattern complexity. In typological analysis, such measures are invaluable: for example, the number of phonemes in a language provides a measure of how complex its inventory is (Maddieson 1984, 1992). Since stress patterns describe infinitely sized sets, established complexity measures like the ones provided by the subregular hierarchies in Figure 8.2 are invaluable.

To illustrate, consider the Strictly Local languages. Informally, these languages are those which can be described in terms of a finite set of *permissible* sequences of symbols of length k (in which case we call the language Strictly k-Local). Consider the simple stress pattern which places primary stress on the first syllable and places no secondary stress. This pattern, for example, can be described by a set of permissible sequences of length two. These are $\{\#\acute{\sigma}, \acute{\sigma}\sigma, \sigma\sigma, \sigma\#, \acute{\sigma}\#\}$ where # indicates a word boundary. Every sequence of length two in every word which obeys the "stress initial syllable" rule is drawn from this set. The 2-length sequences of the word $\acute{\sigma}\sigma$, for example, are $\{\#\acute{\sigma}, \acute{\sigma}\sigma, \sigma\#\}$, which are all permissible. Therefore it can be said that this pattern is Strictly 2-Local. (Strictly Local languages are defined in the 'Appendix'.)

As an exercise, the reader is encouraged to verify that the stress patterns which place primary stress on the peninitial syllable cannot be described with a set of permissible sequences of length two, but can be described with a set of permissible sequences of length three. Therefore, the peninitial stress pattern is Strictly 3-Local. Also, the unbounded patterns above cannot be described with any finite set of permissible sequences of length k for any k. Readers are referred to Rogers and Pullum (2011) for an accessible introduction to these subregular classes and to McNaughton and Papert (1971), Rogers *et al.* (2010), and Heinz *et al.* (2011) for more detailed treatments of these and other classes in the subregular hierarchies.

Some work has already begun to classify stress patterns with respect to these hierarchies. Edlefsen *et al.* (2008) examines the stress patterns in the stress typology in Heinz (2009) with respect to the Strictly Local languages. They report that 72 percent are SL_k for $k \leq 6$ (the other 28 percent are not SL for any k) and 49 percent are SL_3.

Graf (2010) provides a formal analysis of the stress patterns of Creek and Cairene Arabic, as they have been characterized in the literature. According to Graf's analysis, these stress patterns are not star-free, under the assumption that there is no secondary stress in these languages, which is a contentious issue.

The language classes in Figure 8.2 are not the only subregular classes. For example, Heinz (2009) finds that nearly all stress patterns are neighborhood-distinct, which is a class of formal languages which cross-cuts the subregular hierarchies in Figure 8.2.

Apart from providing established, universal measures of pattern classification, and from allowing strong hypotheses regarding universal properties of stress patterns to be stated, there are other advantages to computational representations of stress patterns.

First, both *SPE*-style and OT-style grammars can be converted into finite-state transducers (Johnson 1972; Kaplan and Kay 1994; Eisner 1997; Frank and Satta 1998; Karttunen 1998; Riggle 2004). Transducers describe input/output mappings as opposed to sets of well-formed words.[3] To illustrate, consider an OT analysis of the LHOR pattern. This analysis describes an input/output mapping, and the set of outputs is *exactly the same* as the one generated by the LHOR acceptor in Figure 8.1.

Consequently, finite-state grammars are a *lingua franca* in which different phonological analyses developed in different grammatical frameworks can be compared. Readers are referred to Heinz (2011a, 2011b) for an overview of this literature, to Beesley and Kartunnen (2003) for a more detailed and technical but still accessible introduction, and to Kaplan and Kay (1994) and Riggle (2004) for technical details.

Second, finite-state representations and algorithms are easy, efficient, and sufficiently powerful to automatically compute the predicted locations of stress given the syllabic make-up of a word, and to generate lists of words exemplifying the pattern. This can be used to check predictions of a theory against existing data. These features are implemented on the online stress pattern database maintained by the author at http://phonology.cogsci.udel.edu/dbs/stress/, and are planned to be included in the next version of StressTyp (Goedemans *et al.* 1996; van der Hulst *et al.* 2010).

Also, these representations are standardly used in many distinct fields. This allows stress patterns to be immediately accessible to researchers in other communities, such as computer science, computational linguistics, and computational biology.

Perhaps most important, however, is that these representations can lead to new insights about the nature of phonological generalizations. This chapter presents an example of this by revealing exactly how simple unbounded stress patterns and simple segmental harmony patterns are the same and how they are different. In fact, the only difference between the two kinds of pattern is that stress patterns are *culminative*, whereas segmental harmony patterns are not. Once culminativity is factored out, the two kinds of long-distance pattern are formally identical.

[3] Finite-state transducers describe *regular* relations – sets of input/output pairs. The set of pairs is regular like the sets of words considered in this chapter because they share the crucial computational property that they can be described with machines which possess only finitely many states (acceptors and transducers both being a kind of machine).

8.4 Long-distance sound patterns

Much research in phonology is devoted to studying the nature of long-distance sound patterns in the world's languages, in particular consonantal harmony (Hansson 2001; Rose and Walker 2004) and vowel harmony (van der Hulst and van de Weijer 1995; Archangeli and Pulleyblank 2007). In this chapter, we are concerned only with simple harmony systems, and set aside harmony systems with neutral elements, such as vowel harmony patterns with transparent and opaque vowels.

As an example, (3) and (4) provide examples of simple vowel harmony and consonantal harmony patterns, respectively. (3) illustrates how in Degema there are no words with both advanced ([i u e o ə]) and retracted ([ɪ ʊ ɛ ɔ a]) vowels. Every vowel in every word agrees in the feature ATR.

(3) ATR harmony in Degema (Archangeli and Pulleyblank 2007, citing Elugbe 1984, Kari 1995, 1997)

Advanced		Retracted	
u-bí-ə̄	'state of being black'	á-kī	'pot'
u-pú-ə̄m	'closing'	ʊ-fʊ́-ā	'state of being white'
u-dér-ə̄m	'cooking'	ɔ-ɗɛ́ɗɛ̄	'chief'
i-sór-ə̄	'passing liquid faeces'	ʊ-bɔ̄m-ām	'beating'
o-gədəgə́	'mighty'	ɔ-kpakɪraká	'tough'

(4) illustrates a similar phenomenon in Samala Chumash. Due to a rule of regressive harmony, there are no words on the surface that include both alveolar ([s sˀ sʰ t͡s t͡sˀ t͡sʰ]) and post-alveolar ([ʃ ʃˀ ʃʰ t͡ʃ t͡ʃˀ t͡ʃʰ]) sibilants and affricates.

(4) Samala Chumash regressive sibilant harmony (Applegate 1972)

Alveolar		
/s-api-t͡ʃʰo-us/	sapit͡sholus	'he has a stroke of good luck'
/s-iʃ-tiʃi-jep-us/	sistisijepus	'they (2) show him'
Post-alveolar		
/s-api-t͡ʃʰo-us-waʃ/	ʃapit͡ʃʰoluʃwaʃ	'he had a stroke of good luck'
/ha-s-xintila-waʃ/	haʃxintilawaʃ	'his former Indian name'

Because harmony patterns permit agreement between potentially unboundedly many segments (cf. Samala Chumash *ʃtoyonowonowaʃ* 'it stood upright'; Applegate 1972: 72), they are similar to unbounded stress patterns in that the distance between certain linguistic units in featural agreement appears unbounded.

Heinz (2007, 2010a) shows that consonantal harmony patterns belong to the Strictly Piecewise (SP) languages (see Figure 8.2). Languages in this class

make distinctions on the basis of (potentially discontiguous) *subsequences* of length k. A string is a subsequence of another string only if its symbols occur in order in the other string. For example, both *ʃʃ* and *twoʃ* are subsequences of *ʃtoyonowonowaʃ*. Informally, Strictly 2-Piecewise languages are those which can be described by grammars which are sets of forbidden subsequences of length two. (Strictly Piecewise languages are defined formally in the 'Appendix' in terms of *permissible* subsequences.) Here, I limit discussion to the case when $k = 2$; i.e. to the Strictly 2-Piecewise languages.[4]

To illustrate a Strictly 2-Piecewise language, consider the Samala Chumash example in (4). What formal language describes the surface sibilant harmony pattern? Heinz (2010a) shows that it is exactly that language whose words do not contain disagreeing sibilants as a subsequence of length two. In other words, in Samala Chumash, the subsequences [sʃ] and [ʃs] are *forbidden*. More generally, all strings of length two which match the feature bundles [+strident,α anterior][+strident,−α anterior] are forbidden as subsequences in words in Samala Chumash. To facilitate notation, I will use *s* as an abbreviation for [+strident,+anterior], *ʃ* for [+strident,−anterior], *T* for [+consonantal,−strident], and *V* for [+syllabic].

The language this grammar generates is simply all the logically possible words which do not contain any forbidden subsequence. Every sequence of length two which is not forbidden is called *permissible*. Thus, in Samala Chumash the forbidden subsequences are {sʃ, ʃs} and the permissible subsequences are everything else; that is, {sT, sV, ʃT, ʃV, Ts, Tʃ, TV, TT, Vs, Vʃ, VT, VV}. Equivalently, a Strictly 2-Piecewise language can be described as all and only those words which contain only its permissible subsequences.

Similarly, ATR harmony in Degema forbids subsequences of length two with exactly one advanced and exactly one retracted vowel; i.e. every pair that matches [+syllabic,α ATR][+syllabic,−α ATR]. Every other subsequence of length two is permissible; i.e. every pair of phones that matches either [α ATR][α ATR], [+syllabic][+consonantal], or [+consonantal][+syllabic].[5]

Rogers *et al.* (2010) show how to construct a finite-state acceptor which generates exactly the same infinite set as the one generated by a SP grammar, given as a list of forbidden subsequences. I will not discuss this process here, but the reader can verify that the acceptor in Figure 8.3 shows the acceptor

[4] Heinz (2007, 2010a) originally called this class of languages the 'Precedence languages'. The term 'Strictly Piecewise' comes from formal language theory and emphasizes the similarities this class has to the Strictly Local languages on the one hand, and the Piecewise Testable languages on the other (Rogers *et al.* 2010; Heinz and Rogers 2010).

[5] The notion of subsequence at first blush appears similar to Vergnaud's (1977) notion of projection, which, for example, was used by Hayes and Wilson (2008) for learning long-distance phonotactic patterns. However, there are significant differences; see Heinz (2010a) for discussion and Heinz *et al.* (2011) for mathematical details.

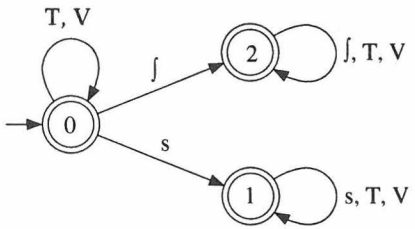


Figure 8.3 An acceptor which recognizes the sibilant harmony pattern in Samala Chumash

which accepts all and only those words which do not contain the forbidden subsequences {sʃ, sʃ}.

Heinz (2010a) is clear that the Strictly 2-Piecewise language describing the sibilant harmony pattern is not intended to accept only possible words of Samala Chumash. It is intended only to capture the long-distance dependency between sibilants in Samala Chumash. This is why the acceptor in Figure 8.3 accepts impossible Samala Chumash words like [ʃkkllkkʃ]. This logically possible word obeys the sibilant harmony generalization. Other aspects of Samala Chumash phonotactics, such as its syllable structure, are captured by other generalizations – formal languages / infinite sets – which can also be represented by other finite-state acceptors. The word [ʃkkllkkʃ] does not obey these generalizations and would not be included in the infinite set of words that do.

In other words, the possible Chumash words are those which obey every phonotactic generalization. This set of words would be exactly those formed by the intersection of the infinite sets describing each individual generalization. Intersection of regular sets can be computed via the acceptor product operation, discussed later in this chapter.

Now we have reached a critical juncture and can pose a very interesting question. Can the unbounded stress patterns also be described with grammars that forbid certain subsequences? To determine whether this is possible, it is necessary to identify the permissible and forbidden subsequences of length two for each unbounded stress pattern. For example, in the LHOR pattern, an unstressed heavy syllable may be followed by another unstressed heavy syllable, but not by a heavy syllable bearing primary stress. In other words, the subsequence HH is permissible and HH́ is forbidden.

Table 8.2 *The permissible and forbidden subsequences for each of the unbounded stress patterns*

LHOR		LHOL	
Permissible	Forbidden	Permissible	Forbidden
L L	Ĺ L	L L	L Ĺ
L Ĺ	Ĺ Ĺ	Ĺ L	Ĺ Ĺ
L H	Ĺ H	L H	Ĺ H
L H́	Ĺ H́	L H́	Ĺ H́
H H	H H́	H H	H H́
H L	H Ĺ	H L	H Ĺ
H́ H	H́ H́	H́ H	H́ H́
H́ L	H́ Ĺ	H́ L	H́ Ĺ
RHOR		**RHOL**	
Permissible	Forbidden	Permissible	Forbidden
L L	Ĺ L	L L	L Ĺ
L Ĺ	Ĺ Ĺ	Ĺ L	Ĺ Ĺ
L H	Ĺ H	L H	Ĺ H
L H́	Ĺ H́	L H́	Ĺ H́
H H	H́ H	H H	H́ H
H L	H Ĺ	H L	H Ĺ
H H́	H́ H́	H H́	H́ H́
H́ L	H́ Ĺ	H́ L	H́ Ĺ

In fact, the Strictly 2-Piecewise languages permit the expression of the generalization "No more than one primary stress exists within a word". For example, by forbidding subsequences such as {ĹĹ, ĹH́, H́Ĺ, H́H́}, it is possible to ensure that at most one primary stress occurs in a word.

Table 8.2 shows the permissible and forbidden subsequences for each unbounded stress pattern. As discussed above, the sets of permissible and forbidden factors can be thought of as grammars that accept all and only those logically possible words which contain only permissible subsequences (or equivalently, which do not contain any of the forbidden subsequences).

Using the algorithm given in Rogers *et al.* (2010), I computed the finite-state acceptors which recognizes exactly the same language as these grammars in Table 8.2. These are shown in Figure 8.4.

These acceptors – which I will call the SP versions of the unbounded patterns – are not the same as the ones in Figure 8.1. For example, each of the acceptors accepts the string LLLLL, which carries no stress at all. Why is this? The reason is straightforward. For every one of the four unbounded stress

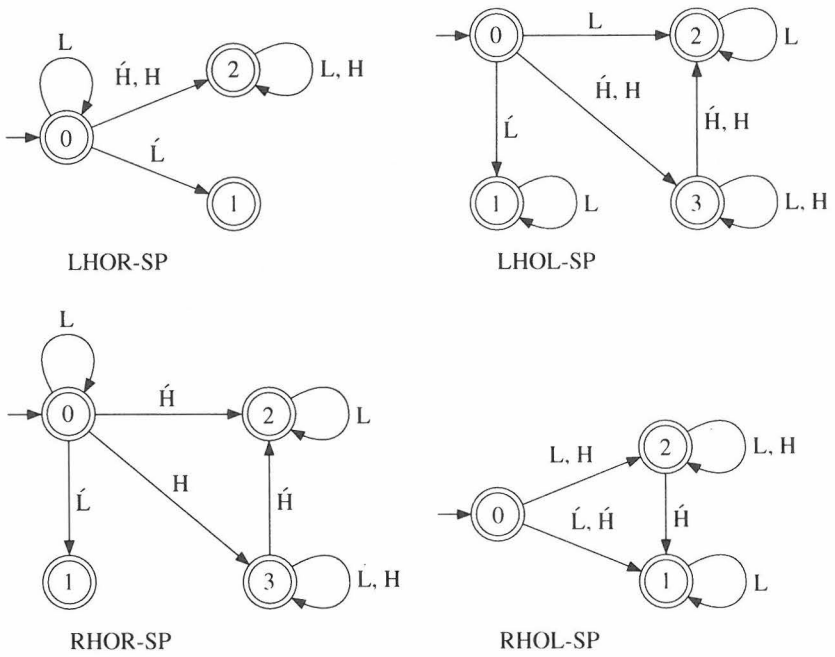


Figure 8.4 Finite-state acceptors accepting the Strictly Piecewise languages given by the permissible and forbidden factors in Table 8.2

patterns, the subsequence LL is permissible. This is because this subsequence occurs in words exemplifying the pattern; e.g. ĹLLL exemplifies the LHOL pattern. But this means that strings of light syllables without *any* stress belong to this formal language! This is because a sequence like LLLLL contains no forbidden subsequence and is therefore, by definition of SP grammars, a perfectly acceptable word. More generally, the Strictly 2-Piecewise languages are unable to express the generalization "At least one primary stress must exist within a word".

Has this enterprise ended in failure? I think not. There is another factor, independently motivated, which is relevant to stress and not to segmental patterns. This factor is *culminativity*, which I take to be the principle that stress patterns have exactly one primary stress in a word.[6] In the remainder of this chapter, I show that once culminativity is factored in, the resulting patterns are exactly the ones given in Figure 8.1. Equivalently, once culminativity is factored out

[6] As mentioned at the outset, some phonologists reserve the term 'culminativity' to mean the principle that every stress pattern 'has at most one primary stress' and use the term 'obligatoriness' to mean the principle that every stress pattern 'has at least one primary stress' (Hyman, 2009).

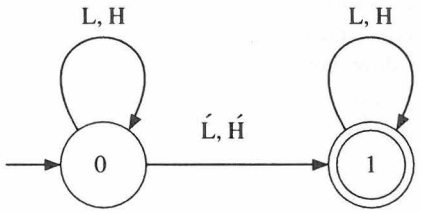


Figure 8.5 A finite-state acceptor describing Culminativity

of the unbounded stress patterns, what is left are exactly languages describable by SP grammars, which therefore are of the same formal character as simple segmental harmony patterns.

8.5 Culminativity

Culminativity is the principle that each word has exactly one prosodic peak. It has long been recognized as a central principle in virtually every theory of stress (Hyman 1977; Halle and Vergnaud 1987; Idsardi 1992; Hayes 1995; van der Hulst *et al.* 2010).

From the perspective of formal language theory, culminativity is the formal language made up of all logically possible words with exactly one primary stress. Figure 8.5 shows the finite-state representation of culminativity. It is not difficult to verify that this acceptor recognizes words like LLLĹ, LLĹL, LĹLL, ĹLLL, HHHĤ, HHĤH, HĤHH, ĤHHH, and so on. Culminativity is simply that constraint which requires words to have exactly one primary stress; it says nothing about where in the word the primary stress falls.

8.6 Factoring unbounded stress patterns

At this point, all the ingredients necessary for the analysis are present except for one. In section 8.3, finite-state representations were provided which exactly describe the unbounded stress patterns. In section 8.4, the permissible and forbidden subsequences of length two (which are sufficient for describing simple harmony patterns) were given for the unbounded stress patterns. Also in this section it was shown that the languages generated by such grammars do not capture the unbounded stress patterns exactly because they permit words with no primary stress at all. Section 8.4 also provided finite-state representations for the SP versions of the unbounded stress patterns. Finally, section 8.5 provided the finite-state representation of culminativity.

What still needs to be described is some operation that can combine culminativity with the SP versions of each of the unbounded stress patterns to yield the exact infinite sets described by the acceptors in Figure 8.1. This is

because one goal of this chapter is to show that the unbounded stress patterns can be *factored* into Culminativity and the SP versions. In order to complete the argument, it is necessary to show how these two factors together can yield exactly the unbounded stress patterns.

Thankfully, the operation required is simple and was mentioned earlier: set intersection. The words belonging to each of the unbounded stress patterns are exactly those words which are common to *both* the Culminativity pattern *and* the SP versions of the unbounded patterns. In other words, (4) can be written equivalently as (5), where L(G) can be read as 'the language generated by grammar G'.

(5) Specific claims in this chapter (language-theoretic version).
   a. $L(\text{LHOR-SP}) \cap L(\text{Culminativity}) = L(\text{LHOR})$
   b. $L(\text{LHOL-SP}) \cap L(\text{Culminativity}) = L(\text{LHOL})$
   c. $L(\text{RHOL-SP}) \cap L(\text{Culminativity}) = L(\text{RHOL})$
   d. $L(\text{RHOR-SP}) \cap L(\text{Culminativity}) = L(\text{RHOR})$

Unlike the case with the intersection of finite sets, it is not immediately obvious how to compute the intersection of infinite sets. However, for *regular* sets, there is a a well-studied operation known as the *acceptor product*. This operation is defined precisely in the 'Appendix'. The product of two acceptors A and B, written $A \times B$, is a third acceptor C. It is known that the language recognized by acceptor C is exactly the intersection of the languages recognized by acceptors A and B. In other words, the claims above can be translated as in (6).

(6) Specific claims in this chapter (acceptor-theoretic version).
   a. LHOR-SP $\times$ Culminativity = LHOR
   b. LHOL-SP $\times$ Culminativity = LHOL
   c. RHOL-SP $\times$ Culminativity = RHOL
   d. RHOR-SP $\times$ Culminativity = RHOR

It is straightforward to verify that each of these claims is correct.[7] For example, the words without any primary stress that are generated by the acceptors in Figure 8.4 are not generated by Culminativity and so they do not survive the intersection process. Likewise, all the words generated by Culminativity which do not exemplify the stress pattern under consideration are also eliminated. The only words which survive the product operation are those which bear exactly one primary stress and which exemplify the stress pattern under consideration.

The main result of this chapter has now been established. Once culminativity is factored out, the unbounded stress patterns are Strictly Piecewise.

[7] Readers can verify this for themselves by using any finite-state manipulation software, such as *xfst* (Beesley and Kartunnen, 2003), or the open-source *foma* (Hulden 2009).

8.7 Discussion

One implication of this result is clear. This analysis unifies long-distance phenomena in both unbounded stress systems and simple segmental harmony systems: they are both Strictly Piecewise modulo Culminativity. This result is important because it suggests the hypothesis that all long-distance phenomena in phonology have an SP characterization as a common denominator. This makes it all the more imperative to examine a broader array of stress patterns and harmony patterns in the context of the subregular hierarchies (Figure 8.2) to see the extent to which this hypothesis is viable.

Another implication of this result is that this analysis reduces the overall complexity of these simple unbounded stress patterns in a concrete way. Recall the classes in Figure 8.2, which classify the complexity of regular sets. By examining the computational complexity of each factor, in addition to the complexity of the whole pattern, one can determine whether the factorization lowers the overall computational complexity.

To illustrate, consider the above analysis. As demonstrated, the SP versions of the simple unbounded stress patterns are Strictly Piecewise. Although not discussed in any detail here, Culminativity belongs to the Locally Threshold Testable region and the simple unbounded stress patterns themselves belong to the NonCounting class.[8] Thus, this factorization lowers the overall complexity of these simple unbounded stress patterns. Factoring Culminativity into 'at most one stress' and 'at least one stress' would lower the complexity even more.[9]

On a technical note, a careful reader may ascertain that it is not necessary to forbid the subsequences {ĹĹ, ĹĤ, ĤĹ, ĤĤ} in the SP versions of the unbounded stress patterns (Table 8.2). This is because words with more than one primary stress will also be eliminated by the intersection with Culminativity. Let us call this version of the unbounded stress patterns the SP' version.

Nothing in the discussion so far serves to distinguish which of these possibilities, the SP or SP' version, is 'correct'. However, if we consider the problem of learning from surface forms, then an argument can be made in favor of the SP version.

As described in Heinz (2010a, 2010b), Strictly 2-Piecewise languages can be learned. The learner simply observes the subsequences of length two that are present in words it observes and adds them to its list of permissible subsequences (which begins empty).[10] The learner will never observe subsequences like ĤĤ and so it will never add those to its list of permissible subsequences; hence, they will always be considered forbidden.

[8] Readers are referred to Rogers and Pullum (2011) for details regarding these classes.
[9] Thanks to Jim Rogers for making this clear to me.
[10] Heinz and Rogers (2010) extend this result stochastically.

Another important implication of the above result is that it simplifies the problem of learning simple unbounded stress patterns. This is because the SP version of the unbounded stress patterns can be learned and because the principle of Culminativity may not need to be since it is arguably a constant principle of Universal Grammar (UG). If it is, then it is always present and available to learners. If it is not, then the problem of learning unbounded stress patterns has been reduced to the problem of learning Culminativity, since the problem of learning SP patterns is solved.

8.8 Conclusion

This chapter has argued for the utility of computational representations in phonological analysis. One reason is that they are at the computational level in Marr's (1982) sense and are therefore relevant to any particular theory of phonology (see also Barton *et al.* 1987: 96–7).

Another reason is that such representations can be used to classify the complexity of known stress patterns according to mathematically principled criteria. The computational classes in Figure 8.2 allow one to identify the nature of phonological patterns – again, independently of any particular grammatical formalism. In fact, the methodology presented in this chapter can be employed to address a much broader question regarding stress patterns: Can the stress patterns be factored in such a way as to reduce their overall complexity?

This chapter focused on a third reason: the fact that insights can be obtained with these representations that would otherwise be difficult to discern. To this end, this chapter showed that unbounded stress patterns can be factored into two parts, each recognizable to phonologists. One part is culminativity, and the other part, like simple segmental harmony systems, can be described exactly in terms of forbidden subsequences of length two. This unification of long-distance phenomena in different phonological domains was made possible by a computational analysis which emphasizes *what* is being computed as opposed to *how* it is computed. It will be interesting to see how far this result can be pushed when more complicated unbounded stress patterns and segmental harmony patterns are considered.

Appendix

This appendix introduces the formalities necessary to establish the results in this chapter.

A8.1 Formal language theory

There is an *alphabet* A, which is fixed and unchanging. In this chapter, the alphabet can be taken to be $A = \{\text{L, H, Ĺ, Ĥ}\}$. The notation A^* is the set of all logically possible finite sequences of length zero or more constructible from this alphabet. Such sequences are often called *strings*. The concatenation of two strings x and y is written xy. The unique string of length 0 is called the *empty string* and is denoted λ. The length of a string w is denoted $|w|$.

A *formal language* L is a subset of A^*. The concatenation of two languages $L_1L_2 = \{xy : x \in L_1 \text{ and } y \in L_2\}$. Please note that for some language $L \subseteq A^*$ and $a \in A$, sometimes I write La instead of the more technically correct, but cumbersome, $L\{a\}$.

Since many formal languages are infinitely sized sets, they can only be defined in terms of *grammars*. The next three parts of this appendix describe some such grammars.

A8.2 Finite-state acceptors

A finite-state acceptor is a machine with five parts: an alphabet, its states, its initial state, its final states, and its transition function. We write $\mathcal{M} = (A, Q, q_0, F, T)$, where A is the alphabet, Q is the state of states, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is the set of final states. The transition function is a map $T : Q \times A \rightarrow Q$.

Every transition function can be extended to a map $T : Q \times A^* \rightarrow Q$ recursively. For all states $q \in Q$ let $T(q, \lambda) = q$. Then for all $x, y \in \Sigma^*$ and $q \in Q$, let $T(q, xy) = T(T(q, x), y)$.

The language generated by a machine $\mathcal{M}$ is defined to be

$$L(\mathcal{M}) = \{w \ : T(q_0, \ w) \in F\}$$

In other words, it is all and only those words which the extended transition function maps to final states from the initial state. Any language which can be generated by a finite-state acceptor is said to be *regular*.

The *acceptor product* is defined below. Recall that, for any sets A and B, the *Cartesian product* $A \times B$ is the set of all pairs (a, b) where a is an element of A and b is an element of B. Recall also that, for any sets A and B, the *intersection* $A \cap B$ is the set containing only those elements common to both A and B.

Consider two acceptors with the same alphabet

$$\mathcal{M}_1 = (A, Q_1, q_{01}, F_1, T_1) \text{ and } \mathcal{M}_2 = (A, Q_2, q_{02}, F_2, T_2)$$

Then

$$\mathcal{M}_1 \times \mathcal{M}_2 = (A, Q, q_0, F, T)$$

where

- $Q = Q_1 \times Q_2$;
- $q_0 = (q_{01}, q_{02})$;
- $F = F_1 \times F_2$;

- and for all $a \in A$, $q_1, q'_1 \in Q_1$ and $q_2, q'_2 \in Q_2$, it is the case that $T((q_1, q_2), a) = (q'_1, q'_2)$ if and only if $T_1(q_1, a) = q'_1$ and $T_2(q_2, a) = q'_2$.

An old result is given in the following theorem. A proof can be found in Hopcroft *et al.* (2001).

Theorem 1 $L(\mathcal{M}_1 \times \mathcal{M}_2) = L(\mathcal{M}_1) \cap L(\mathcal{M}_2)$.

This means that a finite-state acceptor generating exactly the intersection of two regular sets can be obtained by finding the acceptor product of the finite-state acceptor for each of those sets.

A8.3 Strictly Local languages

A string u is a *factor* of string w if and only if there exist $x, y \in A^*$ such that $w = xuy$. In this case we write $u \sqsubseteq_f w$. The following function picks out all factors of length k in some string.

$$F_k(w) = \{u : u \sqsubseteq_f w \text{ and } |u| = k\}$$

This function can be applied to sets by extending the function in the usual way:

$$F_k(S) = \bigcup_{w \in S} F_k(w)$$

A Strictly k-Local grammar G is a subset of $F_k(\{\#\}A^*\{\#\})$, where # indicates a word boundary. G is the set of permissible factors. By definition, G is always finite. The language generated by G is:

$$L(G) = \{w : F_k(\#w\#) \subseteq G\}$$

A language is Strictly k-Local if and only if there is some finite grammar G as just defined which generates it exactly. A language is said to be Strictly Local if and only if there exists some k such that it is Strictly k-Local.

A8.4 Strictly Piecewise languages

Informally, a string $u = a_1 a_2 \ldots a_n$ is a subsequence of string w if the letters $a_1, a_2, \ldots a_n$ occur within w in that order. Formally, u is a *subsequence* of w if and only if

$$w \in A^* a_1 A^* a_2 \ldots A^* a_n A^*$$

In this case we write $u \sqsubseteq_s w$. The following function picks out all subsequences up to length k in some string.

$$P_{\leq k}(w) = \{u : u \sqsubseteq_s w \text{ and } |u| \leq k\}$$

This function can be applied to sets by extending it in the usual way.

A Strictly k-Piecewise grammar G is a subset of $P_{\leq k}(A^*)$. G is the set of permissible subsequences. The language generated by the G is

$$L(G) = \{w : P_{\leq k}(w) \subseteq G\}.$$

Equivalently, Strictly k-Piecewise languages can be defined in terms of a finite set of forbidden subsequences; see Rogers *et al.* (2010) for details.

A language is Strictly k-Piecewise if and only if there is some finite grammar G as just defined which generates it exactly. A language is said to be Strictly Piecewise if and only if there exists some k such that it is Strictly k-Piecewise.

References

Applegate, R. B. 1972. Ineseño Chumash grammar. Unpublished doctoral dissertation, University of California, Berkeley.

Archangeli, Diana, and Douglas Pulleyblank 2007. Harmony. In Paul de Lacy (ed.), *The Cambridge handbook of phonology*, 353–78. Cambridge University Press.

Barton, G. Edward, Robert Berwick, and Eric Ristad 1987. *Computational complexity and natural language*. Cambridge, MA: MIT Press.

Beesley, Kenneth, and Lauri Kartunnen 2003. *Finite state morphology*. Stanford, CA: CSLI Publications.

Edlefsen, Matt, Dylan Leeman, Nathan Myers, Nathaniel Smith, Molly Visscher, and David Wellcome 2008. Deciding strictly local (SL) languages. In Jon Breitenbucher (ed.), *Proceedings of the Midstates Conference for Undergraduate Research in Computer Science and Mathematic*, 66–73. College of Wooster, Ohio.

Eisner, Jason 1997. Efficient generation in primitive Optimality Theory. In *Proceedings of the 35th Annual ACL and 8th EACL*, 313–20. Madrid: Association for Computational Linguistics. Available at: www.aclweb.org/anthology/.

Elugbe, Ben O. 1984. Morphology of the gerund in Degema and its reconstruction in Proto-edoid. *Studies in African Linguistics* 15:77–89.

Frank, Robert, and Giorgo Satta 1998. Optimality Theory and the generative complexity of constraint violability. *Computational Linguistics* 24: 307–15.

Goedemans, R.W.N., H. G. van der Hulst, and E. A. M. Visch 1996. *Stress patterns of the world part 1: background* (HIL Publications II). The Hague: Holland Academic Graphics.

Gordon, Matthew 2002. A phonetically-driven account of syllable weight. *Language* 78: 51–80 [also in UCLA Working Papers in Phonology 2].

2006. *Syllable weight: phonetics, phonology, typology*. London: Routledge.

Graf, Thomas 2010. Comparing incomparable frameworks: a model theoretic approach to phonology. *University of Pennsylvania Working Papers in Linguistics* 16: Article 10. Available at: http://repository.upenn.edu/pwpl/vol16/iss1/10.
Halle, Morris, and Jean-Roger Vergnaud 1987. *An essay on stress*. Cambridge, MA: MIT Press.
Hansson, Gunnar 2001. Theoretical and typological issues in consonant harmony. Unpublished doctoral dissertation, University of California, Berkeley.
Harrison, Michael A. 1978. *Introduction to formal language theory*. Boston, MA: Addison-Wesley Publishing Company.
Hayes, Bruce 1995. *Metrical Stress Theory*. Chicago University Press.
Hayes, Bruce, and Colin Wilson 2008. A maximum entropy model of phonotactics and phonotactic learning. *Linguistic Inquiry* 39: 379–440.
Heinz, Jeffrey 2007. The inductive learning of phonotactic patterns. Unpublished doctoral dissertation, University of California, Los Angeles.
2009. On the role of locality in learning stress patterns. *Phonology* 26: 303–51.
2010a. Learning long-distance phonotactics. *Linguistic Inquiry* 41: 623–61.
2010b. String extension learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 897–906. Uppsala: Association for Computational Linguistics. Available at: www.aclweb.org/anthology/.
2011a. Computational phonology part I: foundations. *Language and Linguistics Compass* 5: 140–52.
2011b. Computational phonology part II: grammars, learning, and the future. *Language and Linguistics Compass* 5: 153–68.
Heinz, Jeffrey, Chetan Rawal, and Herbert Tanner 2011. Tier-based strictly local constraints for phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 58–64. Portland, OR: Association for Computational Linguistics. Available at: www.aclweb.org/anthology/.
Heinz, Jeffrey, and James Rogers 2010. Estimating strictly piecewise distributions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 886–96. Uppsala: Association for Computational Linguistics. Available at: www.aclweb.org/anthology/.
Hopcroft, John, Rajeev Motwani, and Jeffrey Ullman 2001. *Introduction to automata theory, languages, and computation*. Boston, MA: Addison-Wesley.
Hulden, Mans 2009. Finite-state machine construction methods and algorithms for phonology and morphology. Unpublished doctoral dissertation, University of Arizona.
Hulst, H. G. van der, R. Goedemans, and E. van Zanten (eds.) 2010. *A survey of word accentual patterns in the languages of the world*. Berlin: Mouton de Gruyter.
Hulst, Harry van der, and Jeroen van de Weijer 1995. Vowel harmony. In John A. Goldsmith (ed.), *The handbook of phonological theory*, 495–534. Cambridge, MA/Oxford: Blackwell.
Hyman, Larry M. 1977. On the nature of linguistic stress. In Larry Hyman (ed.), *Studies in stress and accent* (Southern California Occasional Papers in Linguistics 4), 37–82. Department of Linguistics, University of Southern California.
2009. How (not) to do phonological typology: the case of pitch accent. *Language Sciences* (Data and Theory: Papers in Phonology in Celebration of Charles W. Kisseberth), 31: 213–38.

Idsardi, William 1992. The computation of prosody. Unpublished doctoral dissertation, MIT.
Jensen, John 1974. Variables in phonology. *Language* 50: 675–86.
1977. *Yapese reference grammar*. Honolulu: University Press of Hawaii.
Johnson, C. Douglas 1972. *Formal aspects of phonological description*. The Hague: Mouton.
Kaplan, Ronald, and Martin Kay 1994. Regular models of phonological rule systems. *Computational Linguistics* 20: 331–78.
Kari, Ethelbert E. 1995. The structure of the Degema verb. Unpublished master's thesis, University of Port Harcourt.
1997. *Degema*. München-Newcastle: Lincom Europa.
Karttunen, Lauri 1998. The proper treatment of optimality in computational phonology. In Lauri Karttunen and Kemal Oflazer (eds.), *FSMNLP'98*, 1–12. International Workshop on Finite-State Methods in Natural Language Processing. Bilkent University, Ankara.
Maddieson, Ian 1984. *Patterns of sounds*. Cambridge University Press.
1992. UCLA phonological segment inventory database. Los Angeles: UCLA.
Marr, David 1982. *Vision*. New York: W. H. Freeman and Company.
McNaughton, Robert, and Seymour Papert 1971. *Counter-free automata*. Cambridge, MA: MIT Press.
Odden, David 1994. Adjacency parameters in phonology. *Language* 70: 289–330.
Riggle, Jason 2004. Generation, recognition, and learning in finite state Optimality Theory. Unpublished doctoral dissertation, University of California, Los Angeles.
Rogers, James, Jeffrey Heinz, Gil Bailey, Matt Edlefsen, Molly Visscher, David Wellcome, and Sean Wibel 2010. On languages piecewise testable in the strict sense. In Christian Ebert, Gerhard Jäger, and Jens Michaelis (eds.), *The mathematics of language* (Lecture Notes in Artifical Intelligence, 6149), 255–65. Berlin: Springer.
Rogers, James, and Geoffrey Pullum 2011. Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20: 329–42.
Rose, Sharon, and Rachel Walker 2004. A typology of consonant agreement as correspondence. *Language* 80: 475–531.
Simon, Imre 1975. Piecewise testable events. In H. Brakhage (ed.), *Automata Theory and Formal Languages*, 214–222. London: Springer-Verlag.
Tesar, Bruce 1998. An interactive strategy for language learning. *Lingua* 104: 131–45.
Vergnaud, Jean-Roger 1977. Formal properties of phonological rules. In Robert E. Butts and Jaakko Hintikka (eds.), *Basic problems in methodology and linguistics*, 299–318. Dordrecht: Reidel.
Walker, Rachel 2000. Mongolian stress, licensing, and factorial typology. ROA-172, Rutgers Optimality Archive. Available at: http://roa.rutgers.edu/.