Phonological Rule Induction: An Architectural Solution

David S. Touretzky[1], Gillette Elvgren III[1], and Deirdre W. Wheeler[2]

[1]School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

[2]Department of Linguistics
University of Pittsburgh
Pittsburgh, PA 15260

Abstract: Acquiring phonological rules is hard, especially when they do not describe generalizations that hold for all surface forms. We believe it can be made easier by adopting a more cognitively natural architecture for phonological processing. We briefly review the structure of M^3P, our connectionist Many Maps Model of Phonology, in which extrinsic rule ordering is virtually eliminated, and "iterative" processes are handled by a parallel clustering mechanism. We then describe a program for inducing phonological rules from examples. Our examples, drawn from Yawelmani, involve several complex rule interactions. The parallel nature of M^3P rule application greatly simplifies the description of these phenomena, and makes a computational model of rule acquisition feasible.

1. Introduction

In previous publications [10-14] we described our work on a connectionist approach to phonology inspired initially by the ideas of John Goldsmith [2] and George Lakoff [6,7]. Our "Many Maps Model of Phonology," M^3P, is an attempt to provide a computational account for both the regularities and peculiarities of human phonological behavior. The parallel rule formalism we developed is constrained by properties of an underlying connectionist sequence manipulation architecture. We suggested that rule acquisition should be easier in this parallel formalism than in the classical generative formalism, because rule ordering constraints are virtually eliminated, and iterative rule application, such as required for vowel harmony, is replaced by a single application of a clustering rule.

In this paper we present some experimental evidence to support our claim that rule acquisition is more tractable in the M^3P formalism. We describe a rule-learning program that examines a set of underlying and surface forms and induces rules to account for the differences. Our program is language-independent, but for reasons of space we will confine our discussion to examples from a single language. We choose the Yawelmani dialect of Yokuts (an American Indian language from California) because it combines several interesting processes in a small number of examples: epenthesis, lowering, shortening, and vowel harmony. Our Yawelmani data come from Kenstowicz & Kisseberth [4], who draw on data from Newman [9] and the analysis of Kuroda [5]. Lakoff offers an alternative to the Kenstowicz & Kisseberth analysis in [7].

Of course, humans are not supplied with underlying forms when they learn their language, so in this respect at least, our program is not a realistic model of human language acquisition. However, we view it as an important first step toward more ambitious, human-like models. We are aware of no other phonological rule learning program, connectionist or otherwise, that can deal with complex rule interactions of the sort found in Yawelmani and many other languages. Phonological rule acquisition is simply a hard problem. Our purpose here is to show that it can be made less hard by adopting a

more cognitively natural architecture. We include a detailed discussion of Yawelmani phonology below to underscore the complexity of the task. Non-linguists can safely skim the following section.

2. The Facts of Yawelmani

In the following discussion we will focus on four rules from Yawelmani, all involving vowels. The (underlying) phonemic and (surface) phonetic vowel systems for Yawelmani are given below. Colons denote long vowels. Note that there are no long high vowels (/i:/ or /u:/) at the surface, and that /e/ and /e:/ do not exist phonemically.

Phonemic (underlying)		Phonetic (surface)	
i/i:	u/u:	i	u
	o/o:	e/e:	o/o:
	a/a:		a/a:

For typographical convenience, following Kenstowicz & Kisseberth, the short and long mid vowels are represented here as 'e' and 'o', although phonetically they are actually [ɛ] and [ɔ]. Given this phonemic inventory, we have evidence for the following distinctive feature analysis of the vowels of Yawelmani:

	i	u	o	a	i:	u:	o:	a:
high	+	+	–	–	+	+	–	–
round	–	+	+	–	–	+	+	–
long	–	–	–	–	+	+	+	+

Suffixes in Yawelmani exhibit alternations between non-round and round vowels. The basic pattern is that a vowel is pronounced round (and back) if immediately preceded by a round vowel of the same height, that is, u-i becomes u-u, and o-a becomes o-o. Note the alternation of -hin/-hun and -al/-ol below:

xat-hin	'eats'	xat-al	'might eat'
bok'-hin	'finds'	bok'-ol	'might find'
xil-hin	'tangles'	xil-al	'might tangle'
dub-hun	'leads by the hand'	dub-al	'might lead by the hand'

Rounding applies iteratively, throughout an entire word, as illustrated in the following examples:

Underlying	Surface	Gloss
/dub-wis-mi/	[dub-wus-mu]	'having led each other by the hand'
/t'ul-sit-hin/	[t'ul-sut-hun]	'burns for'

Returning now to the vowel inventories of Yawelmani, it is worth noting again that there are no underlying segments /e/ and /e:/. The surface segments [e] and [e:] are derived from /i:/ through the application of two rules: Lowering and Shortening. All long vowels lower, and vowels in closed syllables shorten. These rules are shown below; the $ denotes a syllable boundary.

Yawelmani Lowering

$$\begin{matrix} V \\ [+long] \end{matrix} \rightarrow [-high]$$

Yawelmani Shortening

$$V \rightarrow [-long] \,/\, \_\, C\ \$$$

Evidence for shortening comes from paradigms like the following, which show an alternation in vowel length (such as o:/o or a:/a) in the root. Notice that the short variants are followed by *two* consonants; their syllable structure is CVC$CV(C), with the first vowel satisfying the environment for shortening.

Nonfuture	Imperative	Dubitative	Future	
dos-hin	dos-k'o	do:s-ol	do:s-en	'report'
lan-hin	lan-k'a	la:n-al	la:n-en	'hear'

We now argue, in agreement with previous researchers, that not only does [e] derive from a high underlying vowel, but some occurrences of [o] do as well. In particular, the underlying form of the root [c'om-] "destroy" should be /c'u:m-/. One source of evidence for this is harmony. If the underlying form were /c'o:m-/, the non-future form would be *[c'om-hin], but it is actually [c'om-hun]. Lowering and shortening apply only after harmony has had its effect. This also explains why we get [c'om-al] (no rounding of the suffix vowel), not *[c'om-ol]. Compare this with the root meaning 'report,' which is underlyingly /do:s-/, and triggers harmony as expected, giving [do:s-ol], not *[do:s-al].

c'o:m-al	'might destroy'	do:s-ol	'might report'
c'om-hun	'destroys'	dos-hin	'reports'

The interaction of harmony, lowering, and shortening is illustrated below:

	/c'u:m-al/	/c'u:m-hin/
harmony	—	u
lowering	o:	o:
shortening	—	o
	[c'o:mal]	[c'o m h u n]

There is another set of apparent counterexamples to harmony, in that a sequence of vowels that agree in height at the surface do not agree in rounding (o-e), and vowels that disagree in height do agree in rounding (u-o):

bok'-en	'will find'	xat-en	'will eat'
dub-on	'will lead by the hand'	giy'-en	'will touch'

Notice that in [dub-on] the surface *non*-high vowel of the suffix appears rounded after a *high* vowel in the root. Here again, the regularity in the system reemerges if we assume that the future suffix is underlyingly long and high, i.e., /-i:n/. It surfaces as [-en] through the combined effects of lowering and shortening, or as [-on] if harmony has applied first.

There is one additional rule involving vowels that interacts with vowel harmony, namely epenthesis. There is a class of roots of the form CVCC- which have alternants of the form CVCiC- or CVCuC- depending on the preceding vowel and the effects of harmony.

?ugn-al	'might drink'	?ugun-hun	'drinks'
logw-ol	'might pulverize'	logiw-hin	'pulverizes'

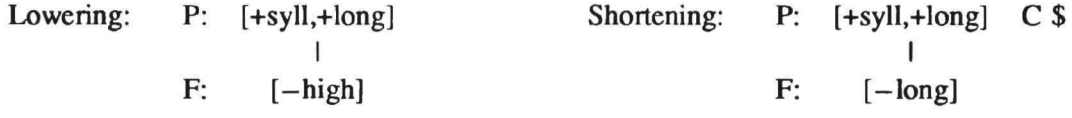
Clearly, epenthesis precedes vowel harmony. This is further supported by examples like [logiw-xa] 'let's pulverize', derived from /logw-xa/, in which the appearance of the epenthetic vowel blocks rounding from applying to the final vowel.

To summarize, the phonological system of Yawelmani offers an example of a very complex set of interactions between independently-motivated rules. The classical generative analysis of this data depends heavily on ordered application [4,5], but learning these rules would be easier in an architecture where they could apply simultaneously. Another factor complicating learning is that the environment of vowel harmony is only regular at the abstract underlying level; the alternations attributable to harmony are not completely systematic based solely on surface forms.

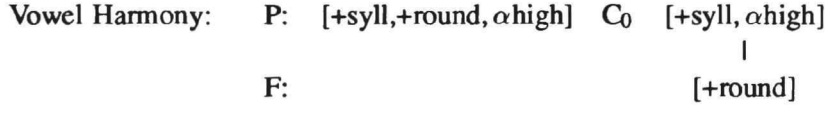
3. The "Many Maps" Architecture

As an alternative to standard generative analyses involving long lists of ordered rules, Lakoff [6,7] develops a theory of phonology which is essentially non-derivational in character. "Rules" are replaced by "constructions," which state well-formedness constraints within levels and correlations between levels. He recognizes three levels: a morphophonemic level (M), a phonemic level (P), and a phonetic level (F). Our attempts to actually implement Lakoff's work in a connectionist framework led to several revisions in the analysis of Yawelmani [11,14]. In this section we will describe our analysis, then turn to the question of learning this complex set of phonological rules in the next section.

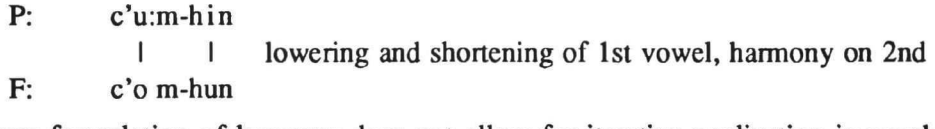
In a generative analysis, shortening comes *after* lowering because only long vowels can lower. In our model, both these processes are P-F constructions that apply simultaneously. Both have their environments satisfied at P-level and their changes realized at F-level. The rules are presented below using Lakoff's parallel mapping notation:



Vowel harmony is traditionally extrinsically ordered before both lowering and shortening, because of the restriction that vowels undergoing harmony must agree in height. If harmony followed lowering, there would be no rounding in the final vowel of examples like [c'omhun] 'destroys' (underlying form /c'u:m-hin/). But in our model, vowel harmony is another P-F construction. It applies simultaneously with the other two P-F rules; all three rules therefore have their environment at P. Ignoring iteration for the moment, the vowel harmony rule might be stated as:



An example of the interaction of these three constructions is given below.



The above formulation of harmony does not allow for iterative application in words like [t'ul-sut-hun] (from /t'ul-sit-hin/). In our model, harmony is actually stated as a P-F "cluster rule." Cluster rules use special circuitry described in [11,14] to process groups of segments undergoing the same change. The cluster rule for harmony first identifies the cluster "trigger" (a round vowel), and then marks as cluster "elements" all subsequent vowels in sequence that agree with it in height. The change sanctioned by the rule is applied to all these elements simultaneously. Our cluster rules are stated in a tabular format:

P-F Cluster Rule for Vowel Harmony:

Cluster type:	[+syllabic]
Trigger:	[+round, αhigh]
Element:	[αhigh]
Change:	[+round]

Our model also includes a syllabifier component that is part of the mapping from M-level to P-level [13]. Epenthesis and deletion processes in most languages are necessitated by the requirement that strings be

fully syllabified. In Yawelmani, epenthesis serves to break up consonant clusters which would otherwise be unsyllabifiable, because its syllable types are restricted to CV and CVC [5]. Since epenthesis is an M-P rule and harmony processes are always P-F, we correctly predict that the epenthetic vowel [i] will be visible at P-level, and thus undergo or block harmony, depending on the height of the preceding vowel.

The following examples illustrate the interactions of the four rules in our parallel formalism:

M:	du:ll-al		M:	du:l l-hin	
				|	epenthesis
P:	du:ll-al		P:	du:lil-hin	
	|	shortening		| | |	lowering, harmony
F:	doll-al	'might climb'	F:	do:lul-hun	'climbs'

With all three mutation rules, lowering, shortening, and harmony, stated as P-F rules, the problem of rule induction is considerably simplified, since there is no need to stipulate constraints on rule application. These are determined by the architecture of the model. We're now in a position to consider the problem of rule induction.

4. The Rule Learning Program

Our rule learning program takes as input a set of of M-level forms and their corresponding F-level realizations. Its output is a set of rules that collectively account for the differences between the two levels. We will discuss the process of learning the shortening rule to illustrate how mutation rules in general are acquired. Each rule accounts for a single change to a single segment. Rule environments are induced using Mitchell's version space technique [8]. Rules have two environments, a most specific environment (SPEC) and a most general (GEN). The SPEC is the intersection of all environments in which the rule has been seen to apply. Initially the SPEC is the string consisting of the changed segment of the first example seen, and up to three segments of context on either side. The GEN states the minimal conditions required for the rule to apply. Initially it contains just the major class features of the first input, i.e., sequences of [+cons] and [+syll], represented below as Cs and Vs. When the SPEC and GEN match, the rule has been refined completely.

Assume the first pair of inputs the learner examines is /do:shin/ and [doshin]. A comparison of the two strings reveals a difference in length of the initial vowel, so a prototype shortening rule is created. The shortening rule's initial SPEC is <d>o<s$hi>, where segments in angle brackets denote context, and the $ indicates a syllable boundary that was detected by the syllabifier. The /o:/ is not marked [+long] by convention in the SPEC, because this feature is the one changed by the rule. The rule's initial GEN is <C>V<CCV>. When shortening is seen to operate in other environments, the GEN will contract to the minimal common substring that characterizes all examples, namely <C>V<C>.

Suppose the second input happens to be /c'u:mhin/ and [c'omhun]. There are three differences in this string: the initial vowel lowers and shortens, and the final vowel rounds. The learner will hypothesize a separate rule for each change. Rules are indexed by the changes they cause, so the previously-formulated rule for shortening will be expected to account for this case as well. Further examples of shortening help to relax the SPEC by eliminating overly-restrictive features. After incorporating the /c'u:mhin/ example, the proto-rule's SPEC is <C>[V,+round]<C$hi>. And after processing the next pair, /bok'i:n/ and [bok'en], the SPEC becomes <C>V<C$>, and the GEN contracts to <C>V<C>.

Examples where shortening doesn't apply are also important, because they force the learner to make the GEN more specific. The underlying form /do:sal/ satisfies the GEN for shortening of the first vowel, but the surface form [do:sol] shows that shortening did not in fact happen. This causes the learner to look for some feature of the SPEC that is absent from the GEN and uniquely accounts for the rule's failure to apply. In this case, the only such feature is the coda marking of the final consonant. The GEN is updated to <C>V<C$>, which matches the SPEC, and the rule is now complete. The lowering rule is learned in a similar fashion.

When a sequence of vowels is observed to undergo the same change, a cluster rule is postulated. On the basis of examples such as /t'ul-sit-hin/, [t'ul-sut-hun], where the second and third high vowels undergo rounding, the learner creates a cluster rule for rounding. It generates single-segment GENs and SPECs for the trigger and element portions of the rule, and refines them as it processes additional examples, such as /do:s-al/, [do:s-ol]. The trigger GEN eventually settles on [+syll,+round], and the element GEN remains [+syll]. Unfortunately, these specifications do not explain certain cases where vowel harmony fails to apply, such as /do:s-hin/, which does not become *[dos-hun], but rather surfaces as [dos-hin].

The learner collects correlation statistics on trigger and element features in order to detect agreement relationships that would be expressed with α variables in conventional rules. The correlation matrix indicates that in all cases where harmony applies, triggers and elements agree in height, and in cases where harmony fails to apply, they disagree. Therefore the learner abandons the original harmony rule and generates a pair of rules. In one, the trigger and element specifications are both [+high], while in the other they are [−high]. We find this solution preferable to introducing variable binding for a feature [αhigh] into the clustering machinery. That would increase the complexity of the connectionist architecture that must implement these rules.

Although we are covering epenthesis last in this section, the learner is actually designed to detect and account for epenthesis and deletion phenomena first. They can generally be explained by syllabification, an M-P process. At this stage in the development of the model, the syllabifier relies on language-specific parameters which we supply for Yawelmani; it does not yet acquire these parameter values on its own. Learning the M-P epenthesis process in cases such as /ʔugn-hin/, [ʔugunhun] is simple and straightforward given the phonotactic constraints of Yawelmani. The syllabifier predicts an epenthetic vowel will appear in this example; all the rule learner needs to do is specify the quality of the vowel. The interaction of epenthesis with other rules follows from the architecture of the model. Notice that this automatically predicts that epenthesis (M-P) will feed harmony (P-F), as it does in the [ʔugunhun] example.

After epenthesis applies to the M-level form /ʔugn-hin/, its P-level representation is /ʔugIn-hin/. The /I/ is a [+high,−long] segment left unspecified for roundness, because the epenthetic vowel surfaces as either [i] or [u] depending on context. The P-level sequence then forms part of the data for learning the harmony rule. Thus we see that when learning P-F rules, the actual input strings must be obtained by applying previously-acquired M-P processes to the M-level strings, rather than looking at the M-level strings themselves. This also lets the model naturally account for the failure to round the final vowel in /logw-xa/. If the harmony rule looked at the underlying representation we would expect it to apply, but the epenthetic high vowel /I/ appearing at P-level in /logIw-xa/ blocks it, since it does not agree in height. This correctly predicts the surface form [logiw-xa], not *[logiw-xo].

As this section has shown, our rule learner has several components. One component learns SPECs and GENs for ordinary rules. Another learns the components of cluster rules: the cluster type, SPECs and GENs for the trigger and element, and the desired change. A third component recognizes insertion and

deletion phenomena that can be accounted for by epenthesis, and so do not need separate rules. Each component corresponds to a different piece of the connectionist architecture that implements these rules.

5. Discussion

Our treatment of vowel harmony is notably different from the proposals of Gasser & Lee [1] and Hare [3], both of which are based on properties of sequential recurrent networks. Their models are trained directly on surface sequences, and do not derive underlying representations to express regularities, although they do develop internal "hidden" representations. They focus on just harmony, and do not consider possible interactions of this process with other phonological rules. Hare's model suggests that a vowel will undergo harmony to become more like the trigger only if it is already sufficiently similar to the trigger. In Yawelmani the crucial similarity governing harmony is height, but the [αhigh] constraint applies only at the abstract underlying level. It is violated on the surface, because long high vowels serving as triggers or elements subsequently undergo lowering.

Insertions and deletions are always handled at M-P in our model (usually by the syllabifier), and mutations at P-F. This works fine for the examples we've tried from Yawelmani and other languages, and is suggestive of a more general pattern across languages. However, it remains an open question whether all languages fall into this pattern. If necessary, we can introduce additional mechanisms to allow rules to migrate between M-P and P-F to establish the correct feeding and blocking relations. We suspect such mechanisms will ultimately be needed.

Our model would be more impressive if it developed its own underlying representations from exposure to annotated surface forms alone. We see no reason why this cannot be done in principle, once a lexical component is added to provide the necessary information (syntactic or semantic) for recognizing allomorphs. We are exploring various possibilities.

The main result of our work to date is that rule learning can be made tractable by adopting a more cognitively natural (less sequential) formalism. Our M^3P architecture has another advantage: it is compatible with a connectionist implementation, and therefore does not violate any fundamental computational constraints associated with neural processing.

Acknowledgements

This research was supported by a contract from Hughes Research Laboratories, by the Office of Naval Research under contract number N00014-86-K-0678, and by National Science Foundation grant EET-8716324. We thank David Evans for helpful comments on an earlier draft of this paper.

References

[1] Gasser, M., and Lee, C.-D. (1989) Networks that learn phonology. Indiana University Computer Science Technical Report #300.

[2] Goldsmith, J. (to appear) Phonology as an intelligent system. To appear in a festschrift for Leila Gleitman, edited by D. Napoli and J. Kegl.

[3] Hare, M. (1989) The role of similarity in Hungarian vowel harmony: a connectionist account. Technical report, University of California at San Diego.

[4] Kenstowicz, M., and Kisseberth, C. (1979) *Generative Phonology: Description and Theory.* San Diego, CA: Academic Press.

[5] Kuroda, S.-Y. (1967) Yawelmani Phonology. Cambridge, MA: The MIT Press.

[6] Lakoff, G. (1988) A suggestion for a linguistics with connectionist foundations. In D. S. Touretzky, G. E. Hinton, and T. J. Sejnowski (eds.), Proceedings of the 1988 Connectionist Models Summer School, pp. 301-314. San Mateo, California: Morgan Kaufmann.

[7] Lakoff, G. (1989) Cognitive phonology. Draft of paper presented at the UC-Berkeley Workshop on Constraints vs. Rules, May 1989.

[8] Mitchell, T. M. (1978) Version Spaces. Doctoral dissertation, Stanford University. Available as technical report STAN-CS-78-711.

[9] Newman, S. (1944) Yokuts Language of California. New York: Viking Fund Publications in Anthropology.

[10] Touretzky, D. S. (1989) Toward a connectionist phonology: the "many maps" approach to sequence manipulation. Proceedings of the Eleventh Annual Conference of the Cognitive Science Society, pp. 188-195. Hillsdale, NJ: Erlbaum.

[11] Touretzky, D. S., and Wheeler, D. W. (1990) A computational basis for phonology. In D. S. Touretzky (ed.), Advances in Neural Information Processing Systems 2. San Mateo, California: Morgan Kaufmann.

[12] Touretzky, D. S., and Wheeler, D. W. (1990) Rationale for a "many maps" phonology machine. Proceedings of EMCSR-90: the Tenth European Meeting on Cybernetics and Systems Research. Vienna, Austria, April 1990.

[13] Touretzky, D. S., and Wheeler, D. W. (unpublished) Two derivations suffice: the role of syllabification in cognitive phonology. Manuscript draft.

[14] Wheeler, D. W., and Touretzky, D. S. (1989) A connectionist implementation of cognitive phonology. Technical Report CMU-CS-89-144, Carnegie Mellon University School of Computer Science. To appear in J. Goldsmith (ed.), Proceedings of the UC-Berkeley Workshop on Constraints vs, Rules in Phonology, University of Chicago Press.