PHONY: A Heuristic Phonological Analyzer*

Lee A. Becker

Indiana University

DOMAIN AND TASK

PHONY is a program to do phonological analysis. Within the generative model of grammar the function of the phonological component is to assign a phonetic representation to an utterance by modifying the underlying representations (URs) of its constituent morphemes. Morphemes are the minimal meaning units of language, i.e. the smallest units in the expression system which can be correlated with any part of the content system, e.g. un+tir+ing+ly. URs are abstract entities which contain the idiosyncratic information about pronounciations of morphemes.

(1)

$$\begin{tikzcd}
\text{Underlying Representations (URs)} \arrow[r, "\text{PHONOLOGICAL COMPONENT}", "\text{(rules)}"'] & \text{Phonetic Representations}
\end{tikzcd}$$

Phonological analysis attempts to determine the nature of the URs and to discover the general principles or rules that relate them to the phonetic representations.

(2)

$$\begin{tikzcd}
 & & \text{URs} \\
\text{Pronounciations} \arrow[r, dash, "\text{PHONY}", "\text{(phonological analysis)}"'] & {} \arrow[ur] \arrow[dr] & \\
 & & \text{Rules}
\end{tikzcd}$$

The input to PHONY are pronounciations of words and phrases upon which a preliminary morphological analysis has been completed. They have been divided into morphemes, and different instances of the same morpheme have been associated. These are represented as strings of phonetic symbols including morpheme- and word-boundaries. Indices are used to associate various instances of the same morpheme.

(3)
```
# 1 s a r a p #   # 1 s a r a b + 2 d a #
# 1 s a r a v + 3 u #   # 1 s a r a v + 4 e #
# 5 a d + 6 a #   # 5 a t #  •••
```

The output of PHONY is a set of phonological rules or regularities in the data, as well as a set of 'underlying representations' for the morphemes. The phonological rules generate the various pronounciations of the morphemes from their underlying representations.

*This research was supported in part by National Science Foundation grant number MCS 81-02291.

REPRESENTATION

In Generative Phonology sounds are represented as matrices of feature specifications, the phonetic symbols being a shorthand for these matrices.

(4)

$$\begin{array}{c}
\text{d} \\
\begin{bmatrix}
-\ \text{syllabic} \\
+\ \text{consonantal} \\
-\ \text{continuant} \\
+\ \text{voice} \\
-\ \text{nasal} \\
+\ \text{anterior} \\
+\ \text{coronal} \\
\cdot \\
\cdot \\
\cdot
\end{bmatrix}
\end{array}$$

The set of 'distinctive features' proposed by Chomsky and Halle [2] were claimed to be sufficient to distinguish the sounds in any language. Further these features were all claimed to have two values; the feature was either present or absent. There has been a fair amount of dispute about the specific features, and several additional ones have been proposed, e.g. gravity, advanced tongue root. There has also been considerable dispute about whether the features are all binary. Nevertheless most phonologists use the original binary features, often with a few additional ones. Phonological rules are operations upon sets of these feature matrices by which feature specifications are assigned to the matrix when it appears in a certain context. The rule expressed (in shorthand) normally as

(6)
s -> s /_i (read s becomes s in position immediately before i)

would be expressed as follows using feature matrices.

(7)

$$\begin{bmatrix} +\ \text{continuant} \\ +\ \text{coronal} \\ +\ \text{anterior} \\ +\ \text{strident} \end{bmatrix} \rightarrow \begin{bmatrix} -\ \text{anterior} \\ +\ \text{high} \end{bmatrix} / \_\_ \begin{bmatrix} +\ \text{syllabic} \\ +\ \text{high} \\ -\ \text{back} \end{bmatrix}$$

The representation provides a language in which to express hypotheses. The task is to find statements in this language to express the data. Thus the representation implicitly defines the search space. The search space is restricted by the following constraint on the 'distance' between a UR and its pronounciations. Every feature specification in the UR must be present in a 'corresponding' segment in at least one of the phonetic forms. Consider, for example, morpheme 1 from (3) above; it has three pronounciations [sarap], [sarab], [sarav].

This constraint restricts its possible URs to /sarap/, /sarab/, /sarav/, /saraf/. Even [f] does not appear in any of the pronouciations of this morpheme, its +continuant specification occurs in [v] and its -voice specification occurs in [p]; its other feature specifications are common to [p], [b], [v]. This constraint is weaker than the "strong alternation condition" (cf. [4]), which would restrict the final UR segment to be /p/, /b/, or /v/. The term "alternation" will be important of the discussion below; here [p] vs. [b] vs. [v] is an alternation.

THE PROBLEM OF MULTIPLE SOLUTIONS

It should be pointed out that most often several sets of combinations of underlying representations and phonological rules can be used to derive the same pronounciations. This could happen in several ways. It could be unclear what the UR is, and different URs together with different rules could derive that same pronounciatons, i.e. the directionality of the rule could be unclear. Consider morpheme 5 from (3) above:

(8)
Pronounciations: #ad+a# #at#

Solution 1: UR /ad/ & Rule d -> t / _ #

Solution 2: UR /at/ & Rule t -> d / a _ a

The symbol # represents a word boundary, and the symbol + represents a morpheme boundary. The difference in the pronounciation of the last segment of this morpheme, d vs. t, is called an alternation. Given this alternation, one could make two hypotheses. One could hypothesize that the UR is /ad/ and that there is a rule which changes d to t when it occurs at the end of a word, or one could hypothesize that the UR is /at/ and that there is a rule which changes t to d between a's. Also some phenomena could be explained by a single more general rule or by several more specific rules.

Generally, there are two approaches that could be taken to deal with the problem of multiple possible solutions. One could attempt to impose restrictions on what could constitute a valid solution, or one could use an evaluation procedure to decide in cases of multiple possible solutions. One could also use both of these approaches; in which case the more restriction, the less evaluation is necessary. An original single evaluation criterion - 'simplicity', as manifested in the number of feature specifications used - has not proved workable. Also no particular proposed restrictions have been embraced by the vast majority of phonologists.

Individual phonologists are generally guided in their evaluations of solutions, i.e. sets of rules and URs, by various criteria. The weighting of these criteria is left open. In this connection the 'codifying function' of the development of expert systems is particulary relevant, i.e. in order to be put into a program the criteria must be formalized and weighted.[5] Although it has sometimes been claimed that no set of discovery procedures can be sufficient to produce phonological analyses, this program is intended to demonstrate the feasibility of a procedural definition of the theory.

The three most widely used criteria and the manner in which they are embedded in PHONY will now be discussed.

Phonological Predictability

This involves the preference of solutions based phonological environment rather than to those in which reference is made to morphological or lexical categories or involving the division of the lexicon into arbitrary classes. In other words, in doing phonological analysis the categories or meanings of morphemes will not be considered, unless no solution can be found based on just the sounds or sound sequences involved. This criterion is embodied in PHONY, since no information about morphological or syntactic categories is available to PHONY. If PHONY cannot handle an alternation by reference to phonological environment, it will return that this is an 'interesting case'. The ability to identify the 'interesting cases' is a most valuable one, since these are often the cases that lead to theory modification. It should be mentioned that PHONY could readily be extended (Extension 1) to handle a certain range of syntactically or morphologically triggered phonological rules. This would involve including in the input information about syntactic category, and, where relevant, morphological category of the constituent morphemes. This informaton would be ignored unless PHONY was unable to produce a solution, i.e. would have returned "interesting cases". It would then search for generalizations based on these categories.

Naturalness

This involves the use of knoweldge about which processes are 'natural' to decide between alternate solutions, i.e. solutions involving natural processes are preferred. A process found in many languages is judged to be 'natural'. Although natural processes are often phonetically plausible, this is not always the case. It should be mentioned that not only is 'naturalness' an arbiter in case of several possible solutions, but it is also a heuristic to lead the investigator to plausible hypotheses which he can pursue. PHONY contains a catalogue of natural processes. When an alternation looks as if it might be the result of one of these processes, the entire input corpus of strings is tested to see if this hypothesis is valid.

Simplicity

'Simplicity' was mentioned above, while it is no longer the only criterion, it is still a primary one. It is reflected in PHONY in a series of attempts to make rules more general, i.e. combine several hypothesized rules into a single hypothesized rule. The more general rules require fewer feature specifications. Also the smaller number of rules can lead to a reduced number of feature specifications.

The various proposed constraints on what can be valid solutions generally would correlate with the differences in the testing process of PHONY. Most of these involve differences in allowable orderings of rules (e.g. 'unrestricted extrinsic ordering', 'free reapplication', 'direct mapping'; cf. [3]). At present PHONY's testing process involves checking if hypothesized rules hold, i.e. do not have counterexamples, in the phonetic representations (such a criterion disallows opacity of type 1; cf. [4]). PHONY could be extended (Extension 2) to allow the user to choose from several of the proposed constraints. This would involve using different testing functions. This extension would allow analyses of the same data under different constraints to easily be compared. Additionally, new constraints could be added and tested.

STRUCTURE OF PHONY

PHONY can be divided into three major parts: ALTFINDER, NATMATCH, and RULERED.

ALTFINDER

ALTFINDER takes the input sting of phonetic symbols and indices indicating instances of the same morpheme, as in (3), and returns for each morpheme in turn a representation including the non-alternating segments and list of alternations with the contexts in which each alternant occurs, for example, for morpheme 1, as in (9).

(9)

```
sara   p      ~          b       ~         v

  # sarap #    # sarab + da #    # sarav + u #
                                 # sarav + e #
```

This process involves comparing in turn each instance of a given key morpheme with the current hypothesized underlying representation for that morpheme, and for each case of alternation storing in N groups the different context strings in which the N alternants occur. The comparison is complicated by the common processes of epenthesis (insertion of a segment) and elision (deletion of a segment), and occasionally by the much more rarely occurring methathesis (interchange in the positions of two segments). These processes are illustrated in (10).

(10)

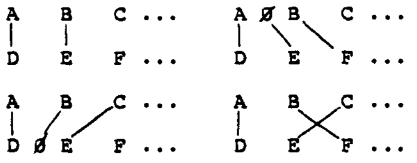
```
Given UR  / t a r i s k /,
[tarisak] would involve Epenthesis Ø -> a
[trisk]         "       Elision a -> Ø
[tariks]        "       Methathesis sk -> ks
```

Therefore in cases where the segments being compared are not identical it is necessary to ascertain whether they are variants of a single underlying segment or one of these processes has applied. The possibilities are illustrated in (11).

(11)
Given two pronounciations of the same morpheme

```
[ A B C . . . ] where A is associated with D
[ D E F . . . ] and B is not identical to E,
```

There are four possible relationships:



The criteria used to decide between these relationships are (a) degree of similarity in each of the conceivable associations, and (b) a measure of the similarity of the rest of the strings for each of the conceivable associations.

ALTFINDER yields a list of alternations based on segments, as in (9). This is then converted into a list of alternations based on features.

(12)

```
       p                    b                 v
  p-contexts           b-contexts        v-contexts
                           ⇓
  VOICE           +                      -
   b-contexts & v-contexts          p-contexts

  CONTINUANT  +                          -
        v-contexts       b-contexts & p-contexts
```

Since every one of the alternations in the former must differ by at least one feature, the new list must contain as many alternations and normally contains more alternations. Where previously for each alternation in a segment there was a list of strings where each alternant occurred, now for each alternation in a feature there are two lists - one with the strings where a positive value for that feature occurred and the other where a negative value occurred.

It should be noted that the elements of these lists, i.e. strings, together with the feature alternating, its value, and an indication of which segment in the string contains the feature, are all potentially rules. They bear the same information as standard phonological rules. Compare the representations in (13); these are for the alternations in morpheme 5 in (3).

(13)

```
#   a   d   +   a   #           #   a   t   #

1                   1           1           1
0           1       0           0           0
0           0       0           0           0
0           0       0           0           0
0           0       0           0           0
0   1       0   1   0           0   1       0
0   0   1   0   0   0           0   0   1   0
0   1   0   0   1   0           0   1   0   0
0   0   0   0   0   0           0   0   0   0
0   1   0   0   1   0           0   1   0   0
0   1   0   0   1   0           0   1   0   0
0   0   1   0   0   0           0   0   1   0
0   0   1   0   0   0           0   0   1   0
0   1   1   0   1   0 VOICE     0   1   0   0
0   1   0   0   1   0           0   1   0   0
0   0   0   0   0   0           0   0   0   0
0   0   0   0   0   0           0   0   0   0
0   0   0   0   0   0           0   0   0   0
0   0   0   0   0   0           0   0   0   0
```

to the rules t -> d / # a + a # d -> t / # a # , i.e. respectively, one can't pronounce t in the environment # a + a # but rather must pronounce d, and one can't pronounce d in the environment # a # but rather must pronounce t. The latter rule and the second representation (both without the initial two segments - in the interests of space) in (13) are juxtaposed in (14).

(14)

```
      1000011000000 1000000000000000

D -> T /   ____              #
```

It is often the case that one or both of these potential 'rules' will be valid, i.e. would be generalizations that would hold over the pronounciations represented in the input. These 'rules' would, however, be much less general than those which are found in phonological analyses. It is assumed that speaker/hearer/language learners can and do generalize from these specific cases to form more general rules. If this were not the case how could speakers correctly pronounce morphemes in new environments. Within the theory the criterion of simplicity is sensitive to these generalizations in that such generalizations reduce the number of feature specifications. Within PHONY the preference for more general rules is manifested by continually trying to generate and test more general rules resulting from the coalescing or combining of two or more specific rules.

Recall that the representation of the segments involved a feature matrix with positive or negative specifications for each feature. In order to generate more general rules this representation is modified to two matrices for each segment - one representing those features which must be positive in the environment and the other for those features which must be negative. The generalization process involves taking the 'greatest common denominator' (GCD) of the positive and negative values of the segments of the environments of two separate 'rules'. In the interests of space an abbreviated example of the GCD operation is given in (15).

(15)

```
        +  -   +  -   +  -      +  -   +  -
 SYLL   1  0   0  1   1  0      0  1   1  0
VOICE   1  0   1  0   1  0      0  1   1  0
 HIGH   0  1   1  0   1  0      1  0   1  0

                       GCD

        +  -   +  -                             ┌+SYLL ┐
 SYLL   0  1   1  0                             │+VOICE│
VOICE   0  0   1  0 ≋ [-SYLL]->[+HIGH]/         └+HIGH ┘
 HIGH   1  0   1  0
```

The GCD operation has generated a more general rule. If the original two rules are a manifestation of a more general rule, the generalized rule must not involve or make reference to the the initial segment of the former rule. Notice also that in the GCD the VOICE feature does not have to be positive or negative; if the two original rules are a manifestation of a single rule the specification of the VOICE feature in the alternating segment must not be relevant.

NATMATCH

After the alternations in terms of segments that were output by ALTFINDER have been changed into alternations in terms of features (12) and after these have been transformed from single matrices into double matrices, the resulting "rules" are sent to NATMATCH. NATMATCH compares these "rules" with the data base of common phonological processes. This involves pattern matching. If a match occurs the entire input corpus is tested to find out if it can be established whether this rule or constraint is valid for this language. If Extension 2 were implemented, this testing process would differ for the different versions of the theory. If the validity can be established, the underlying representations for the morpheme is adjusted and the rule is added to the list of established rules. Common processes in the data base are organized by the feature which is alternating, and among those processes involving the alternation of a given feature the most common process is listed and thus tested first. If it can be shown to be valid, it is added to a list of established rules. It should be mentioned that ALTFINDER makes use of this list, and if an alternation that it discovers can be handled by an established rule, the tentative underlying representation is so adjusted and the alternation need not be passed on to the rest of the program. If within NATMATCH no matches are found in the data base or if the validity of the matches cannot be established, the alternation is added to the list of those as yet not accounted for.

RULERED

RULERED takes the generated "rules" that have not been established. It establishes which of these are valid and takes GCDs to generalize these as much as possible. This is done by going through all the rules involving a certain feature and generating the minimal number of equivalence classes of "rules" and combined (GCDed) "rules" which are valid. The resulting generalized rules have the largest matrices, i.e. the largest set of feature specifications, which all the forms undergoing these rules have in common. However, the elimination of some of these features specification might still result in valid rules. The rules with minimal matrices, i.e. minimal number of feature specifications (recall the "simplicity" criterion), might be termed lowest commmon denominators (LCDs). These are produced by attempting in turn to eliminate each segment in GCDed rule; the new rule is generated and tested, and if valid the segment is out, otherwise it remains. Then an attempt is made to eliminate in turn each feature specification in the remaining segments, again generate and test. Finally, all the established rules are combined, where possible, according to the many abbreviatory conventions of Generative Phonology (cf. [2]). This is done on the basis of the formal properties of the rules. For example, if two generated rules are identical except that one has an additional segment not present in the other, these can be into a single rule; parentheses allow the inclusion of optional segments in the environment of a rule. In addition, all the rules generated above involve a change of only a single feature specification. If there are several rules which are identical except that a different feature specification is changed, i.e. the two changes occur in the same environment, they can be combined into a single rule: in this particular environment both specifications change.

DISCUSSION

PHONY is a learning program. It is discovering the general principles or rules governing pronounciation in a language. As such it can be said to be learning some aspect of a language. PHONY can be thought of either independently or as a part of a larger system designed to learn a language. In the latter context PHONY could help in deciding between ambiguous morphological divisions. In addition, PHONY could be used in adjusting, fine-tuning heuristics for a morphological analyzer. PHONY would act as a "critic" in such a system (cf. [1]). Two sets of heuristics might lead to different morphological analyses, which might each be input to PHONY; if one input lead to analysis that had no "interesting cases", i.e. problems, while the other did, the set of heuristics leading to the former analysis would be supported.

Independently PHONY is an expert system. It provides a procedural definition of phonological theory. Because of this, it could be useful to someone desiring to learn phonological theory. It could also be of use to working phonologists. In addition to producing the analyses, it also isolates the 'interesting cases', e.g. morphologically triggered rules. With Extension 1 it could also be used to compare various versions of the theory and to test the the effects of new modifications of the theory.

It should be emphasized that at present PHONY is a bare program. It is hoped that it is sufficient to demonstrate the feasability and worth of the endeavor. It presents a basic approach: contexts in with alternating segments are transformed into hypothesized "rules", these can be combined via the GCD operation, further simplified to LCDs, and then again combined according to the abbreviatory conventions. There is a "grinding" quality to this process. Phonologists only resort to a similar grind, when all their heuristics have led to deadends. The only heuristic presently incorporated in PHONY is the comparison to a list of natural processes; this allows a tremendous shortcut in the search More heuristics obviously could be added to PHONY.

It would also be possible for a METAPHONY to find heuristics to be to be used by PHONY. (Possible decision criteria to be used in evaluating differing sets of heuristics could be the number of tests of the input corpusand the number of "interesting cases".) These heuristics could improve efficiency of PHONY by obviating much of the "grinding" process. At the same time METAPHONY could also be making discoveries about phonologies of natural languages in general. For example, in the process of generating LCDs instead of going segment by segment and feature by feature, METAPHONY could acquire and incorporate in PHONY knowledge about what aspects of pronounciation are not/rarely pertinent to rules affecting a certain feature.

REFERENCES

1. Buchanan, B.G., T.M. Mitchell, R.G. Smitch, C.R. Johnson, Jr. 1979. Models of learning systems. Encyclopedia of Computer Science and Technology. J. Belzer, A. Holtzman, A. Kent (Eds.). New York: Marcel Dekker, Inc. Vol 3, pp 24-51.

2. Chomsky, N. and M. Halle. 1968. The Sound Pattern of English. New York: Harper and Row.

3. Kenstowicz, M. and C. Kisseberth. 1977. Topics in Phonological Theory. New York: Academic Press.

4. Kiparsky, P. 1968. How abstract is phonology? In O. Fujimura (Ed.), Three Dimensions in Linguistic Theory. 1973. Tokyo: TEC.

5. Michie. D. 1980. Knowledge-based systems. UIUCDCSR-80-1001 and UILU-Eng 80-1704 (University of Illinois).