A DISCOVERY PROCEDURE FOR CERTAIN PHONOLOGICAL RULES

Mark Johnson
Linguistics, UCSD.

ABSTRACT

Acquisition of phonological systems can be insightfully studied in terms of discovery procedures. This paper describes a discovery procedure, implemented in Lisp, capable of determining a set of ordered phonological rules, which may be in opaque contexts, from a set of surface forms arranged in paradigms.

1. INTRODUCTION

For generative grammarians, such as Chomsky (1965), a primary problem of linguistics is to explain how the language learner can acquire the grammar of his or her language on the basis of the limited evidence available to him or her. Chomsky introduced the idealization of *instantaneous acquisition*, which I adopt here, in order to model the *language acquisition device* as a function from *primary linguistic data* to possible grammars, rather than as a process.

Assuming that the set of possible human languages is small, rather than large, appears to make acquisition easier, since there are fewer possible grammars to choose from, and less data should be required to choose between them. Accordingly, generative linguists are interested in delimiting the class of possible human languages. This is done by looking for properties common to all human languages, or *universals*. Together, these universals form *universal grammar*, a set of principles that all human languages obey. Assuming that universal grammar is innate, the language learner can use it to restrict the number of possible grammars he or she must consider when learning a language.

As part of universal grammar, the language learner is supposed to innately possess an *evaluation metric*, which is used to "decide" between two grammars when both are consistent with other principles of universal grammar and the available language data.

2. DISCOVERY PROCEDURES

This approach deals with acquisition without reference to a specific discovery procedure, and so in some sense the results of such research are general, in that in principle they apply to all discovery procedures. Still, I think that there is some utility in considering the problem of acquisition in terms of actual discovery procedures.

Firstly, we can identify the parts of a grammar that are underspecified with respect to the available data. Parts of a grammar or a rule are *strongly data determined* if they are fixed or uniquely determined by the data, given the requirement that overall grammar be empirically correct.

By contrast, a part of a grammar or of a rule is *weakly data determined* if there is a large class of grammar or rule parts that are all consistent with the available data. For example, if there are two possible analyses that equally well account for the available data, then the choice of which of these analyses should be incorporated in the final grammar is weakly data determined. Strong or weak data determination is therefore a property of the grammar formalism and the data combined, and independent of the choice of discovery procedure.

Secondly, a discovery procedure may partition a phonological system in an interesting way. For instance, in the discovery procedure described here the evaluation metric is not called upon to compare one grammar with another, but rather to make smaller, more local, comparisons. This leads to a factoring of the evaluation metric that may prove useful for its further investigation.

Thirdly, focussing on discovery procedures forces us to identify what the surface indications of the various constructions in the grammar are. Of course, this does not mean one should look for a one-to-one correspondence between individual grammar constructions and the surface data; but rather complexes of grammar constructions that interact to yield particular patterns on the surface. One is then investigating the logical implications of the existence of a particular constructions in the data.

Following from the last point, I think a discovery procedure should have a *deductive* rather than *enumerative* structure. In particular, procedures that work essentially by enumerating all possible (sub)grammars and seeing which ones work are not only in general very inefficient, but also not very insightful. These discovery by enumeration procedures simply give us a list of all rule systems that are empirically adequate as a result, but they give us no idea as to what properties of these systems were crucial in their being empirically adequate. This is because the structure imposed on the problem by a simple recursive enumeration procedure is in general not related to the intrinsic structure of the rule discovery problem.

3. A PHONOLOGICAL RULE DISCOVERY PROCEDURE

Below and in Appendix A I outline a discovery procedure, which I have fully implemented in Franz Lisp on a VAX 11/750 computer, for a restricted class of phonological rules, namely rules of the type shown in (1).

(1) $a \rightarrow b \ / \ C$

Rule (1) means that any segment a that appears in context C in the input to the rule appears as a b in the rule's output. Context C is a feature matrix, and to say that a appears in context C means that C is a subset of the feature matrix

formed by the segments around a[1]. A phonological system consists of an ordered[2] set of such rules, where the rules are considered to apply in a cascaded fashion, that is, the output of one rule is the input to the next.

The problem the discovery procedure must solve is, given some data, to determine the set of rules. As an idealization, I assume that the input to the discovery procedure is a set of surface paradigms, a two dimensional array of words with all words in the same row possessing the same stem and all words in the same column the same affix. Moreover, I assume the root and suffix morphemes are already identified, although I admit this task may be non-trivial.

4. DETERMINING THE CONTEXT THAT CONDITIONS AN ALTERNATION

Consider the simplest phonological system: one in which only one phonological rule is operative. In this system the alternating segements a and b can be determined by inspection, since a and b will be the only alternating segments in the data (although there will be a systematic ambiguity as to which is a and which is b). Thus a and b are strongly data determined.

Given a and b, we can write a set of equations that the rule context C that conditions this alternation must obey. Our rule must apply in all contexts C_b where a b appears that alternates with an a, since by hypothesis b was produced by this rule. We can represent this by equation (2).

(2) $\forall C_b$, C matches C_b

The second condition that our rule must obey is that it doesn't apply in any context C_a where an a appears. If it did, of course, we would expect a b, not an a, in this position on the surface. We can write this condition by equation (3).

(3) $\forall C_a$, C does not match C_a

These two equations define the rule context C. Note that in general these equations do not yield a unique value for C; depending apon the data there may be no C that simultaneously satisfies (2) and (3), or there may be several different C that simultaneously satisfies (2) and (3). We cannot appeal further to the data to decide which C to use, since they all are equally consistent with the data.

Let us call the set of C that simultaneously satisfies (2) and (3) S_C. Then S_C is strongly data determined; in fact, there is an efficient algorithm for computing S_C from the C_as and C_bs that does not involve enumerating and testing all imaginable C (the algorithm is described in Appendix A).

However, if S_C contains more than one C, the choice of which C from S_C to actually use as the rule's context is weakly data determined. Moreover, the choice of which C from S_C to use does not affect any other decisions that the discovery procedure has to make - that is, nothing else in the complete grammar must change if we decide to use one C instead of another.

Plausibly, the evaluation metric and universal principles decide which C to use in this situation. For example, if the alternation involves nasalization of a vowel, something that usually only occurs in the context of a nasal, and one of the contexts in S_C involves the feature *nasal* but the other C in S_C do not, a reasonable requirement is that the discovery procedure should select the context involving the feature *nasal* as the appropriate context C for the rule.

Another possibility is that S_C's containing more than one member indicates to the discovery procedure that it simply has too little data to determine the grammar, and it defers making a decision on which C to use until it has the relevant data. The decision as to which of these possibilities is correct is is not unimportant, and may have interesting empirical consequences regarding language acquisition.

McCarthy (1981) gives some data on a related issue. Spanish does not tolerate word initial *s*C clusters, a fact which might be accounted for in two ways; either with a rule that inserts *e* before word initial *s*C clusters, or by a constraint on well-formed underlying structures (a redundancy rule) barring word initial *s*C. McCarthy reports that either constraint is adequate to account for Spanish morphophonemics, and there is no particular language internal evidence to prefer one over the other.

The two accounts make differing predictions regarding the treatment of loan words. The *e* insertion rule predicts that loan words beginning with *s*C should receive an initial *e* (as they do: *esnob, esmoking, esprey*), while the well-formedness constraint makes no such prediction.

McCarthy's evidence from Spanish therefore suggests that the human acquisition procedure can adopt one potential analysis and rejects an other without empirical evidence to distinguish between them. However, in the Spanish case, the two potential analyses differ as to which components of the grammar they involve (active phonological processes versus lexical redundancy rules) which affects the overall structure of the adopted grammar to a much greater degree than the choice of one C from S_C over another.

5. RULE ORDERING

In the last section I showed that a single phonological rule can be determined from the surface data. In practice, very few, if any, phonological systems involve only one rule. Systems involving more than one rule show complexity that single rule systems do not. In particular, a rules may be ordered in such a fashion that one rule affects segments that are part of the context that conditions the operation of another rule. If a rule's context is visible on the surface (ie. has not been destroyed by the operation of another rule) it is said to be *transparent*, while if a rule's context is no longer visible on the surface it is *opaque*. On the face of it, opaque contexts could pose problems for discovery procedures.

---

[1] What is crucial for what follows is that saying context C matches a portion of a word W is equivalent to saying that C is a subset of W. Since both rule contexts and words can be written as sets of features, I use "contexts" to refer both to rule contexts and to words.

[2] I make this assumption as a first approximation. In fact, in real phonological systems phonological rules may be unordered with respect to each other.

Ordering of rules has been a topic substantial research in phonology. My main objective in this section is to show that extrinsically ordered rules in principle pose no problem for a discovery procedure, even if later rules obscure the context of earlier ones. I don't make any claim that the procedure presented here is optimal - in fact I can think of at least two ways to make it perform its job more efficiently. The output of this discovery procedure is the set of all possible ordered rule systems[3] and their corresponding underlying forms that can produce the given surface forms.

As before, I assume that the data is in the form of sets of paradigms. I also assume that for every rule changing an *a* to a *b*, an alternation between *a* and *b* appears in the data; thus we know by listing the alternations in the data just what the possible *a*s and *b*s of the rule are[4].

From the assumption that rules are extrinsically ordered it follows that one of the rules must have applied last; that is, there is a unique "most surfacy" rule. The context of this rule will necessarily be transparent (visible in the surface forms), as there is no later rule to make its context opaque.

Of course, the discovery procedure has no *a priori* way of telling which alternation corresponds to the most surfacy rule. Thus although the identity of the segments involved in the most surfacy rule may be strictly data determined, at this stage this information is not available to the discovery procedure.

So at this point, the discovery procedure proposed here systematically investigates all of the surface alternations: for each alternation it makes the hypothesis that it is the the alternation of the most surfacy rule, checks that a context can be found that conditions this alternation (this must be so if the hypothesis is correct) using the single rule algorithm presented earlier, and then investigates if it is possible to construct an empirically correct set of rules based on this hypothesis.

Given that we have found a potential "most surfacy" rule, all of the surface alternates are replaced by the putative underlying segment to form a set of intermediate forms, in which the rule just discovered has been undone. We can undo this rule because we previously identified the alternating segments. Importantly, undoing this rule means that all other rules whose contexts had been made opaque in the surface data by the operation of the most surfacy rule will now be transparent.

The hypothesis tester proceeds to look for another alternation, this time in the *intermediate* forms, rather than in the surface forms, and so on until all alternations have been accounted for.

If at any stage the hypothesis tester fails to find a rule to describe the alternation it is currently working with, that is, the single-rule algorithm determines that no rule context exists that can capture this alternation, the hypothesis tester discards the current hypothesis, and tries another.

The hypothesis tester is responsible for proposing different rule orderings, which are tested by applying the rules in reverse to arrive at progressively more removed representations, with the single-rule algorithm being applied at each step to determine if a rule exists that relates one level of intermediate representation with the next. We can regard the hypothesis tester as systematically searching through the space of different rule orderings, seeking rule orderings that successfully accounts for the observed data.

The output of this procedure is therefore a list of all possible rule orderings. As I mentioned before, I think that the enumerative approach adopted here is basically flawed. So although this procedure is relatively efficient in situations where rule ordering is strictly data determined (that is, where only one rule ordering is consistent with the data), in situations where the rules are unordered (any rule ordering will do), the procedure will generate all possible *n!* orderings of the *n* rules.

This was most striking while working with some Japanese data, with 6 distinct alternations, 4 of which were unordered with respect to each other. The discovery procedure, as presented above, required approximately 1 hour of CPU time to completely analyse this data; it found 4 different underlying forms and 512 different rule systems that generate the Japanese data, differing primarily in the ordering of the rules.

This demonstrates that a discovery procedure that simply enumerates all possible rule ordering is failing to capture some important insight regarding rule ordering, since unordered rules are much more difficult for this type of procedure to handle, yet unordered rules are the most common situation in natural language phonology.

This problem may be traced back to the assumption made above that a phonological system consists of an *ordered* set of rules. The Japanese example shows that in many real phonological systems, the ordering of particular rules is simply not strongly data determined. What we need is some way of partitioning different rule orderings into equivalence classes, as was done with this the different rule contexts in the single rule algorithm, and then compute with these equivalence classes rather than individual rule systems; that is, seek to localize the weak data determinacy.

Looking at the problem in another way, we asked the discovery procedure to find all sets of ordered rules that generate the surface data, which it did. However, it seems that this simply was not right question, since the answer to this question, a set of 512 different systems, is virtually

---

[3] Thus if the *n* rules in the system are unordered, this procedure returns *n!* solutions corresponding to the *n* ways of ordering these rules.

[4] The reason why the class of phonological rules considered in this paper was restricted to those mapping segments into segments was so that all alternations could be identified by simply comparing surface forms segment by segment. Thus in this discovery procedure the algorithm for identifying possible alternates can be of a particularly simple form. If we are willing to complicate the machinery that determines the possible alternations in some data, we can relax the restriction prohibiting epenthesis and deletion rules, and the requirement that all alternations are visible on the surface. That is, if the approach here is correct, the problem of identifying which segments alternate is a different problem to discovering the context that conditions this alternation.

uninterpretable by human beings. Part of the problem is that phonologists in general have not yet agreed what exactly the principles of rule ordering are[5].

Still, the present discovery procedure, whatever its deficiencies, does demonstrate that rule ordering in phonology does not pose any principled insurmountable problems for discovery procedures (although the procedure presented here is certainly practically lacking in certain situations), even if a later rule is allowed to disturb the context of an earlier rule, so that the rule's context is no longer "surface true". None the less, it is an empirical question as to whether phonology is best described in terms of ordered interacting rules; all that I have shown is that such systems are not in principle unlearnable.

6. CONCLUSION

In this paper I have presented the details of a discovery procedure that can determine a limited class of phonological rules with arbitrary rule ordering. The procedure has the interesting property that it can be separated into two separate phases, the first phase being superificial data analysis, that is, collecting the sets C_a and C_b of equations (2) and (3), and the second phase being the application of the procedure proper, which need never reference the data directly, but can do all of its calculations using C_a and C_b[6]. This property is interesting because it is likely that C_a and C_b have limiting values, as the number of forms in the surface data increases. That is, presumably the language only has a fixed number of alternations, and each of these only occurs in some fixed contexts, and as soon as we have enough data to see all of these contexts we will have determined C_a and C_b, and extra data will not make these sets larger. Thus the computational complexity of the second phase of the discovery procedure is more or less independent of the size the lexicon, making the entire procedure require linear time with respect to the size of the data. I think this is a desirable result, since there is something counterintuitive to a situation in which the difficulty of discovering a grammar increases rapidly with the size of the lexicon.

7. APPENDIX A: DETERMINING A RULE'S CONTEXT

In this appendix I describe an algorithm for calculating the set of rule contexts $S_C = \{ C \}$ that satisify equations (2) and (3) repeated below in set notation as (4) and (5). Recall that C_b are the contexts in which the alternation did take place, and C_a are the contexts in which the alternations did not take place. We want to find (the set of) contexts that simultaneously match all the C_b, while not matching any C_a.

(4) $\forall C_b, C \subseteq C_b$

(5) $\forall C_a, C \not\subseteq C_a$

We can manipulate these into computationally more tractable forms. Starting with (4), we have

$\forall C_b, C \subseteq C_b \quad (= (4))$
$\forall C_b, \forall f \in C, f \in C_b$
$\forall f \in C, f \in \bigcap C_b \; C \subseteq \bigcap C_b$

Put $C_1 = \bigcap C_b$. Then $C \subseteq C_1$.

Now consider equation (5).

$\forall C_a, C \not\subseteq C_a$
$\forall C_a, \exists f \in C, f \notin C_a$
$\forall C_a, \exists f \in (C - C_a)$

But since $C \subseteq C_1$, if $f \in (C - C_a)$. then $f \in (C_1 - C_a) \bigcap C$. Then

$\forall C_a, \exists f \in (C_1 - C_a), f \in C$

This last equation says that every context that fulfills the conditions above contains at least one feature that distinguishes it from each C_a, and that this feature must be in the intersection of all the C_b. If for any C_a, $C_1 - C_a = \emptyset$ (the null set of features), then there are *no* contexts C that simultaneously match all the C_b and none of the C_a, implying that no rule exists that accounts for the observed alternation.

We can construct the set S_C using this last formula by first calculating C_1, the intersection of all the C_b, and then for each C_a, calculating $C_f = (C_1 - C_a)$, a member of which must be in every C. The idea is to keep a set of the minimal C needed to account for the C_a so far; if C contains a member of C_f we don't need to modify it; if C does not contain a member of C_f then we have to add a member of C_f to it in order for it to satisfy the equations above. The algorithm below acomplishes this.

set $C_1 = \bigcap C_b$
set $S_C = \{\emptyset\}$

foreach C_a
  set $C_f = C_1 - C_a$
  if $C_f \equiv \emptyset$
    return "*No rule contexts*"

  foreach C **in** S_C

    if $C \bigcap C_f \equiv \emptyset$
      remove C **from** S_C

      foreach f **in** C_f
        add $C \bigcup \{ f \}$ **to** S_C

return S_C

where the subroutine "add" adds a set to S_C only if it or its subset is not already present.

After this algorithm has applied, S_C will contain all the minimal different C that satisfy equations (4) and (5) above.

---

[5] In this paper I adopted strict ordering of all rules because it is one of the more stringent rule ordering hypotheses available.

[6] In fact, the sets C_a and C_b as defined above do not contain quite enough information alone. We must also indicate which segments in these contexts alternate, and what they alternate to. This may form the basis of a very different rule order discovery procedure.