Using the Mutual Inconsistency of Structural Descriptions to Overcome Ambiguity in Language Learning

Bruce Tesar
Rutgers University

Using the Mutual Inconsistency of Structural Descriptions to Overcome Ambiguity in Language Learning

Bruce Tesar

Rutgers University

0. Introduction

One source of difficulty in language learning is the *hidden structure* presumed to be present in full structural descriptions but not directly perceivable in the overt signal available to the learner (Dresher (1996) calls this problem the 'Epistemological Problem'). The hidden structure problem can be illustrated by foot structure in metrical stress theory. Consider a trisyllabic word with stress on the middle syllable, [0 1 0]. While the stress levels of the syllables are overtly apparent, the foot structure is not. The overt information, termed the *overt form*, is ambiguous between two full structural descriptions: [0 (1 0)], a right-aligned trochaic foot, and [(0 1) 0], a left-aligned iambic foot. A structural description with an overt portion that matches an overt form is called an *interpretation* of that overt form. The hidden structure in this case is the footing of the syllables. A fully competent speaker of a language, possessing the correct grammar for the language, is able to use that grammar to correctly analyze overt forms, assigning them the correct structural description in the process of interpreting them. The difficulty occurs in language learning; the learner needs to infer, on the basis of the overt information, the very grammar that the competent speaker makes use of.

Hidden structure need not be a problem for learning if it is easily reconstructed from the overt information. Hidden structure is more of a problem when overt forms are ambiguous. An overt form is ambiguous when it supports more than one interpretation, as illustrated above with the overt form [0 1 0]. Such ambiguity frequently cannot be resolved on the basis of the overt form alone[1]; other information, presumably from other overt forms,

[1]The elimination of an interpretation independent of other overt forms is possible if the interpretation can be shown to be suboptimal under *any* ranking of the constraints.

Pius N. Tamanji and Kiyomi Kusumoto (eds.), NELS 28, 469-483

is necessary to determine which interpretation is correct for the language being learned. One approach is to simply insist on only learning from unambiguous overt forms. Within the principles and parameters framework (Chomsky 1981), this kind of approach has been proposed, relativized to individual parameters, in the form of an independence principle on the overt effects of parameter settings (Wexler & Manzini 1987), and later in the requirement of the existence of forms, called triggers, that indicate the value of a single parameter (Gibson & Wexler 1993). However, it has proven extremely difficult to find plausible analyses with such unambiguous forms, and the desirability of such analyses is questionable with respect to linguistic theory. A related approach, also within the principles and parameters framework, is to analyze a parametric system and attempt to uncover an ordered series of cues (Dresher & Kaye 1990, Dresher 1996). This approach relaxes the independence requirement somewhat: the first parameter should be unambiguously represented overtly, but then the cue for the second parameter may rely on the setting of the first parameter, the cue for the third on the setting of the second, and so forth. This locates the principles of learning squarely within the full substantive detail of a particular analysis, with each domain requiring its own ordered set of cues. Further, any change in the parametric system can potentially invalidate the system of cues.

Tesar & Smolensky (1996) have proposed that the learning problem of going from overt forms to a grammar be decomposed into two subproblems: (1) the assignment of full structural descriptions to overt forms; and (2) the determination of the grammar from full structural descriptions. They pursued this approach within the framework of Optimality Theory (Prince & Smolensky 1993), and developed an algorithm which solves the second subproblem, Recursive Constraint Demotion (Tesar & Smolensky, 1995).

In this paper, I will present a learning algorithm that makes use of Recursive Constraint Demotion (RCD) as a solution to the second part of Tesar & Smolensky's decomposition. In comparison to the other approaches just discussed, the algorithm presented here makes no attempt to identify in advance necessarily unambiguous configurations within any single overt form. This is not surprising given that the algorithm is couched within Optimality Theory, where the learned part of a grammar is the ranking imposed upon the universal constraints; it is unclear what a cue would even be a cue for, since there is nothing to be learned about any constraint in isolation. In fact, the algorithm will correctly learn a constraint ranking even when every single overt form in the language is ambiguous, as is the case with the languages used in the simulations described in Section 4.

This paper makes two proposals concerning language learning. The first proposal is an algorithm called multi-recursive constraint demotion (MRCD). This algorithm uses a data structure, separate from the hypothesized constraint ranking, to record information obtained from observed data. Specifically, the learner constructs a list of mark-data pairs; the form and role of mark-data pairs is explained in Section 1. MRCD, described in Section 2, is able to obtain a constraint ranking from this list, but by keeping the list itself information is retained that would otherwise be lost if only the generated constraint ranking were retained. The retained information makes it easier for the learner to take account of multiple overt forms simultaneously.

The second proposal, explained in Section 3, is an algorithm for making use of the retained information to contend with ambiguity. When confronted with an ambiguous overt form, this algorithm generates the possible interpretations of the overt form, and then attempts to reconcile each one with the information stored in the mark-data pairs. If an interpretation is inconsistent with one or more of the structural descriptions used in constructing the mark-data pair list, the offending interpretation can be eliminated; if enough information has already been gathered, all interpretations but the correct one will be eliminated due to inconsistency, and the ambiguity will be completely overcome. This algorithm, which makes use of MRCD, is guaranteed to find a correct ranking on the basis of only overt forms, even if every single overt form is ambiguous. Further, the necessary size of the mark-data pair list is quite reasonable; the learner is not required to store an unreasonable amount of data as part of any ranking hypothesis.

Section 4 presents the results of some simulations run to test the efficiency of this algorithm. The simulations apply the algorithm to an optimality theoretic system of grammars for metrical stress, which has 12 constraints, and a respectable degree of ambiguity among the overt forms of the possible languages (every overt form is ambiguous between at least 2 interpretations, and some overt forms support as many as 21 distinct interpretations). The simulations show that the learning algorithm can be extremely efficient, even in the face of a very large hypothesis space and significantly ambiguous forms.

1. Learning in Optimality Theory

Optimality Theory is inherently comparative. When presented with a grammatical structural description, the learner must determine what properties must hold of the constraint ranking such that the grammatical description is more harmonic than any of its competitors. The base unit of data is the pairing of the constraint violations of a grammatical structural description with the constraint violations of a competitor. Such a pairing is called a *mark-data pair* (Tesar & Smolensky 1995, 1998). The grammatical description is termed the *winner*, and the competitor is termed the *loser*, alluding to the learner's goal of finding a constraint ranking such that the winner beats out (is more harmonic than) the loser (mark-data pairs are sometimes called *loser/winner pairs*).

Consider the pair of descriptions shown in the tableau in (1).

		MAIN-RIGHT	ALL-FEET-RIGHT	MAIN-LEFT	ALL-FEET-LEFT	TROCHAIC	IAMBIC
winner	[0 (1 0)]			*	*		*
loser	[(1 0) 0]	*	*				*

(1) Tableau of the winner and loser of a mark-data pair.

The loser and winner have identical violations of IAMBIC, so those marks will cancel. The information contained in this mark-data pair is summarized in (2):

(2) (MAIN-RIGHT **or** ALL-FEET-RIGHT) » (MAIN-LEFT **and** ALL-FEET-LEFT)

At least one of the constraints violated more by the loser must dominate all of the constraints violated more by the winner. This pair alone will not indicate which, if not both, of the constraints violated more by the loser must so dominate. RCD is an algorithm for efficiently finding a constraint ranking consistent with a set of such mark-data pairs.

RCD will be illustrated with the following list of mark-data pairs.

Loser Marks	**Winner Marks**
ALL-FEET-RIGHT MAIN-RIGHT	ALL-FEET-LEFT MAIN-LEFT
TROCHAIC	IAMBIC
PARSE	ALL-FEET-RIGHT IAMBIC
ALL-FEET-RIGHT	ALL-FEET-LEFT

(3) The starting list of mark-data pairs.

The learner begins with all of the constraints unranked. First, the constraints that can possibly be ranked highest are identified, by determining which constraints do not appear in the **Winner Marks** column. Those constraints are placed in the first stratum. The mark-data pairs with one of the now-ranked constraints in the **Loser Marks** column are now removed from the list, as they cannot provide any further information. The result of this first pass is shown in (4).

Stratum 1
PARSE
MAIN-RIGHT
TROCHAIC

Loser Marks	**Winner Marks**
ALL-FEET-RIGHT	ALL-FEET-LEFT

(4) Ranking and remaining mark-data pairs after the first pass.

The second pass performs the same procedure, but now upon the remaining, still unranked constraints and the remaining mark-data pairs. Of the four remaining constraints, only ALL-FEET-LEFT appears in the **Winner Marks** column; the other three may be placed into the second stratum. That includes ALL-FEET-RIGHT, so the remaining mark-data pair can be eliminated. Thus, on the third pass, no mark-data pairs remain, so the remaining constraint may be placed in the third and final stratum. The result is the constraint ranking shown in (5).

Stratum 1		Stratum 2		Stratum 3
PARSE	»	ALL-FEET-RIGHT	»	ALL-FEET-LEFT
MAIN-RIGHT		MAIN-LEFT		
TROCHAIC		IAMBIC		

(5) The final constraint hierarchy.

Note that the ranking in (5) is not total; the three constraints in the first stratum are not ranked relative to each other. This is common, frequently occurring when some constraints do not crucially interact with each other in a language. In general, the learning algorithms discussed here use such rankings as hypotheses, which are sometimes referred to as *constraint hierarchies*. When a correct constraint hierarchy has been obtained for a language, it is the case that any ordering of the constraints in a stratum relative to each other will yield the same outcome (the resulting ranking will generate the same language). The typological predictions of Optimality Theory remain intact; only languages consistent with at least one total ranking of the constraints are learnable by this learner.

RCD has the ability to detect inconsistencies in the data it is given. This property, illustrated here, will be of particular significance in the solution to overt form ambiguity presented later on. The data are inconsistent when there does not exist any ranking of the constraints which can simultaneously satisfy all of the mark-data pairs in the list. When such a list is given as input to RCD, the algorithm does not continue processing endlessly, nor does it return an incorrect constraint hierarchy; it instead returns a code indicating that no ranking exists which is consistent with all of the mark-data pairs in the list.

To see how, consider the list of mark-data pairs in (6).

Loser Marks	**Winner Marks**
MAIN-RIGHT TROCHAIC	MAIN-LEFT IAMBIC
MAIN-LEFT IAMBIC	MAIN-RIGHT TROCHAIC

(6) An inconsistent list of mark-data pairs.

Focusing on just the four constraints depicted, RCD starts by checking for constraints that do not appear in the **Winner Marks** column. Unfortunately, there are none; all four constraints appear there. This means that each of the four constraints is required to be dominated by one of the other constraints, an impossible state of affairs for a grammar generated by a strict dominance ranking. At this point, RCD stops processing mark-data pairs and returns an indication that an inconsistency has been detected. The property of inconsistency is a property of the set of mark-data pairs; in general, there is no way to attribute the inconsistency to any one mark-data pair in the set. Fortunately, no such attribution is necessary, and the learning algorithm presented here succeeds by detecting the inconsistency of full sets.

2. Multi-Recursive Constraint Demotion

Multi-recursive constraint demotion (MRCD) is an error-driven learning algorithm based upon RCD. It takes as input a full structural description and a grammar hypothesis, and returns a grammar hypothesis which is altered (if necessary) to accommodate the structural description. For MRCD, a grammar hypothesis consists of a constraint hierarchy and an associated list of mark-data pairs. The two are directly related: the constraint hierarchy is the one which results from applying RCD to the list of mark-data pairs. The list consists of the informative mark-data pairs used during learning.

At any given time, the learner has a grammar hypothesis; initially, the mark-data pair list is empty, so the associated hierarchy has all constraints in a single stratum. When presented with a new grammatical structural description, the learner extracts the underlying form of that description, and then parses that underlying form according to the constraint hierarchy of its current grammar hypothesis. If the currently optimal structural description matches the observed grammatical one, the learner simply keeps its current grammar hypothesis. If, however, the currently optimal structural description does not match, then an error has occurred. The learner then constructs a new mark-data pair, using the description which is optimal according to the current ranking hypothesis as the loser, and using the grammatical structural description as the winner. This new mark-data pair is then added to the list of mark-data pairs in the grammar hypothesis. RCD is then applied to this new list, and the resulting hierarchy, along with the new list of mark-data pairs, is adopted as the learner's new grammar hypothesis. The process is then repeated with the same structural description and the new grammar hypothesis, continuing until either a grammar hypothesis is reached which makes optimal a description consistent with the overt form, or an inconsistency is detected.

MRCD takes as input a ranking hypothesis and a new interpretation (a full structural description of some overt form). MRCD returns as output either a new ranking hypothesis (which may or may not be identical to the old one), or an indication that the new interpretation is inconsistent with the given ranking hypothesis. MRCD is formally given in (7).

MRCD is an on-line algorithm, accepting a single structural description (the interpretation) and (possibly) performing learning in response, before the next structural description is processed. This is contrasted with batch algorithms, which require all the input data to be collected in advance, before any learning takes place. MRCD is an on-line algorithm which can still take advantage of RCD's ability to detect data inconsistencies.

Start: the mark-data pair list L is initially empty (before any interpretations are processed)
Given: hierarchy H with mark-data pair list L, and interpretation I with underlying form U
begin
    compute the optimal description D assigned by H to U
    while (D≠I) and (L is consistent)
        create a new mark-data pair with D the loser and I the winner
        add the new mark-data pair to L
        apply RCD to L, getting a new hierarchy H or an inconsistency code
        if (L is still consistent)
            compute the optimal description D assigned by H to U
        end-if
    end-while
    if (L is inconsistent)
        return (the inconsistency code)
    else
        return (H, L)
    end-if
end

(7) The Multi-Recursive Constraint Demotion Algorithm

3. Using Inconsistency Detection to Overcome Ambiguity

Now we turn to the problem of reconstructing the correct structural descriptions for overt forms. The overt form [0 1 0], a trisyllabic word with stress on the middle syllable, has several possible interpretations, two of them being [(0 1) 0] and [0 (1 0)]. These interpretations are full structural descriptions (the footform is fully specified), and their overt portions match the overt form [0 1 0]. This distinguishes the interpretations of [0 1 0] from other candidate structural descriptions of a trisyllabic word, such as [(1 0) 0] or [0 (0 1)], whose overt portions do not match [0 1 0]. The difficulty arises when an overt form is ambiguous. The overt form itself provides no basis for choosing among the possible interpretations. The learner must combine the information provided by other overt forms in order to determine the correct interpretation.

First, it is worth considering the nature of the set of possible interpretations. Among the information included in an interpretation but not necessarily included in the overt form is the underlying form. The present work abstracts away from the learning of underlying forms, and we will assume that the correct underlying form is apparent from the surface form (for example, the system will not account for syllable lengthening and shortening processes). This is not an unreasonable strategy for basic metrical stress: the overt form is a string of syllables and their stress levels, while the underlying form is simply the same string of syllables with the stress levels removed. The set of interpretations of an overt form (for example, [0 1 0]) consists of those structural descriptions, out of the set of structural descriptions generated by GEN for the underlying form (for example, three light syllables), whose overt portion matches that overt form. In the case of the system discussed in Section 4, the overt form [0 1 0] has three possible interpretations: [(0 1) 0], [0 (1 0)], and [0 (1) 0].

How can the information from different overt forms be used to mutually constrain their interpretations? The correct interpretations for a set of overt forms must have the property that there exists a constraint ranking holding all of the correct interpretations simultaneously optimal. Suppose, for example, that a language has three overt forms, Overt-X, Overt-Y, and Overt-Z. Each of these forms has two interpretations: the interpretations of Overt-X are Interp-X1 and Interp-X2, the interpretations of Overt-Y are Interp-Y1 and Interp-Y2, and similarly Interp-Z1 and Interp-Z2 for Overt-Z. There are 2 * 2 * 2 = 8 possible combinations of the interpretations, as shown in (8).

Interp-X1	Interp-Y1	Interp-Z1
Interp-X1	Interp-Y1	Interp-Z2
Interp-X1	Interp-Y2	Interp-Z1
Interp-X1	Interp-Y2	Interp-Z2
Interp-X2	Interp-Y1	Interp-Z1
Interp-X2	Interp-Y1	Interp-Z2
Interp-X2	Interp-Y2	Interp-Z1
Interp-X2	Interp-Y2	Interp-Z2

(8) The eight possible combinations of interpretations for Overt-X, -Y, and -Z.

The learner's task can now be characterized as the task of finding, for a given set of overt forms, which combinations of interpretations are consistent, and which are inconsistent. If the three forms Overt-X, Overt-Y, and Overt-Z collectively have enough information to determine the constraint ranking, then only one of the 8 possible combinations listed in (8) should be consistent (assuming that two distinct interpretations of the same overt form do not have identical constraint violations).

This is where RCD's capacity to detect inconsistencies becomes important. The learner can test out different combinations of interpretations for a set of overt forms. Testing out a combination of interpretations means trying to find a constraint ranking which holds all of them optimal. This can be done by applying MRCD to each of the interpretations of the combination in succession, building up the list of mark-data pairs along the way. If the interpretations are inconsistent, MRCD will find that out and report the inconsistency (ruling out that combination), while if the interpretations are consistent, MRCD will return a constraint hierarchy holding all of the interpretations optimal.

The most obvious way to apply MRCD would be to take a set of overt forms, and generate all possible combinations of interpretations of the overt forms, applying MRCD to each one to determine which combination was consistent. There is a problem for this approach, however, due to the rapid growth of the number of possible combinations of interpretations. The number of possible combinations will be the product of the numbers of possible interpretations of all of the overt forms. For languages with even a very modest

degree of ambiguity in overt forms, the set of possible combinations of interpretations will be far too large to search exhaustively.

Fortunately, there is a better way to apply MRCD. The idea is to consider all possible interpretations of overt forms not across the entire language at once, but on a form by form basis. When a grammatical overt form is received, the underlying form for the overt form is extracted, and parsed using the constraint ranking of the learner's current grammar hypothesis. If the overt portion of the currently optimal structural description matches the grammatical overt form, then no modification is made to the grammar hypothesis. In this way, the algorithm is error-driven, an error being diagnosed by a mismatch between the grammatical overt form and the overt portion of the currently optimal structural description.

When an error is detected, the grammar hypothesis needs to be modified. To use MRCD to modify the hypothesis requires a full structural description, not just an overt form. It is here that the learner generates all possible interpretations *for the current overt form*. For each possible interpretation, the learner separately applies MRCD to that interpretation along with their current grammar hypothesis. If MRCD determines that a particular interpretation is inconsistent with the list of mark-data pairs determining learner's current grammar hypothesis, then that interpretation is discarded. Determining that an interpretation is consistent will result in one or more additional mark-data pairs being added to the list; this new list then determines the new, modified grammar hypothesis.

This procedure can leave the learner with more than one tenable grammar hypothesis after the processing of an overt form. It may be that the learner does not yet have enough information to rule out all but one interpretation of that overt form. In this case, the learner keeps all of the tenable grammar hypotheses. On the next overt form, the learner separately checks for an error between the new overt form and each of its grammar hypotheses. Each hypothesis detecting an error on the new overt form is processed along with each possible interpretation of the new overt form by applying MRCD as described above.

The amount of computational effort required by such a learner is dependent on the interrestrictiveness of the overt forms of the language. If the consistent combinations of interpretations are relatively few, then for each overt form causing an error, most of its interpretations will be eliminated due to inconsistency on the spot, and thus will not generate new grammar hypotheses to be carried along.

```
Start: the list of grammar hypotheses G is initially an empty list
Given: a list of grammar hypotheses G and an overt form V with underlying form U
begin
    for (each hypothesis (H,L) in G, consisting of hierarchy H and mark-data list L)
        remove (H,L) from G
        compute the optimal description D assigned by H to U
        if (the overt portion of D does not match V)
            compute the set I-SET of possible interpretations of V
            for each interpretation I in I-SET
                apply MRCD to I and L, getting (H-NEW, L-NEW)
                if (MRCD did not return an inconsistency code)
                    insert (H-NEW, L-NEW) into G-NEW
                end-if
            end-for
        else
            insert (H,L) into G-NEW
        end-if
    end-for
    return (G-NEW)
end
```

(9) The "Try All Interpretations" Procedure for Using MRCD

4. Simulation Results

By design, the "try all interpretations" algorithm for applying MRCD as described in the previous section is guaranteed to find a correct constraint ranking for a language realizable by an optimality theoretic system, provided that (a) the learner is provided with a representative sample of overt forms from the language, (b) each overt form has only a finite number of possible interpretations, and (c) each underlying form is apparent from its corresponding overt form. The interesting question for this algorithm concerns the amount of work required to obtain a correct ranking. The efficiency of the algorithm was tested empirically on a optimality theoretic system for metrical stress.

Metrical stress theory has been a domain of focus for several learning investigations (Daelemans, Gillis & Durieux 1994, Gupta & Touretzky 1994, Dresher & Kaye 1990). Metrical stress is an appealing domain because a lot is known about it, and because it can be treated somewhat in isolation from other aspects of phonology. It was selected for the current investigation because it permits the issue of input/output faithfulness to be set aside. In the present analysis, underlying forms are strings of syllables, and structural descriptions assign stresses to the syllables without any deletion or insertion of syllables. For discussion of the learning of underlying forms see (Tesar & Smolensky 1998, Smolensky 1996, Hale & Reiss 1996) and works cited therein.

The optimality theoretic analysis of stress described here includes ideas from several sources (McCarthy & Prince 1993, Prince & Smolensky 1993, Prince 1990, Kager 1994, Hammond 1990, Hayes 1995, Hayes 1980). The system has as possible inputs words

consisting of strings of syllables, each labeled for weight (light or heavy). A candidate structural description for an input is a grouping of the syllables of the input into feet, under the following conditions: (a) a foot contains either one or two syllables; (b) each foot assigns stress to exactly one of its syllables; (c) each candidate has exactly one head foot assigning main stress, with any other feet assigning secondary stress. The system has 12 constraints, listed in (10).

PARSE	a syllable must be footed
MAIN-RIGHT	the head-foot must be aligned with the word, on the right edge
MAIN-LEFT	the head-foot must be aligned with the word, on the left edge
ALL-FEET-RIGHT	a foot must be aligned with the word, on the right edge
ALL-FEET-LEFT	a foot must be aligned with the word, on the left edge
IAMBIC	a head syllable must be aligned with its foot, on the right edge
FOOT-NON-FINAL	a head syllable must not be rightmost in its foot
WORD-FOOT-LEFT	the word must be aligned with some foot, on the left edge
WORD-FOOT-RIGHT	the word must be aligned with some foot, on the right edge
NON-FINAL	the right-most syllable must not be footed
FOOT-BINARITY	a foot must have two moras or two syllables
WSP	a heavy syllable must be stressed

(10) The constraints of the Metrical Stress System

Because the primary interest is in how well the algorithm contends with the number of combinations of interpretations, the measure of effort used here is the number of times, during the course of learning, that a learner adds a mark-data pair to a mark-data pair list and applies RCD to the list, referred to as *the number of applications of RCD*. This includes all of the additions made to lists that are ultimately discarded; an important part of the measure of work is the amount required to test and eliminate inconsistent combinations.

For each tested language, 62 forms were used: all combinations of light and heavy syllables for words of length 2 to 5 syllables, and words of 6 and 7 light syllables. 124 of the possible languages in the system were tested, meaning that for each tested language, the optimal descriptions for all 62 underlying forms were generated, and the overt forms were extracted; a test presented the 62 overt forms of a language to the learner, starting with the shortest (bisyllabic words) and proceeding in order of increasing size to the longest.

To appreciate the results presented below, it is worth examining the magnitude of the problem. With 12 constraints, a simple exhaustive search of all possible rankings of the constraints would have to examine 12! = 479,001,600 rankings. As large and implausible as that is, it pales in comparison to an enumeration of all possible combinations of interpretations of the 62 overt forms for a language. Different overt forms have differing degrees of ambiguity, but every possible overt form in the system has at least 2 possible interpretations. Many have more: the overt form [1 0 2 0 0] has 5 interpretations, and [0 1 0 2 0 2 0] has 21 distinct interpretations. Even if every form were limited to just two interpretations, the number of combinations would still be $2^{62} = 5 \times 10^{18}$. For most of the tested languages, the number of possible combinations is much, much larger than that. If the learner cannot completely escape the combinatorial growth of the number of combinations of interpretations,

then this approach is hopeless.

Fortunately, the simulations show that the algorithm is quite successful in overcoming the combinatorical challenge posed.

Number of Languages	Number of RCD Applications		
	Median	Minimum	Maximum
124	50	8	160

(11) Simulation Results.

The longest case took only 160 applications of RCD to arrive at a correct constraint hierarchy, and the median was 50 applications.

5. Discussion

Multi-recursive constraint demotion, the use of lists of mark-data pairs to represent grammatical hypotheses, allows a learner to retain information that is not recoverable from a constraint hierarchy alone. MRCD can in principle be used in conjunction with a variety of strategies for hypothesizing interpretations of overt forms. The strategy presented in this paper is to consider all possible interpretations for an overt form, whenever learning is necessitated by an error. This approach is guaranteed to learn correctly for any optimality theoretic system for which each overt form has a finite number of interpretations, and permits the direct inference of its underlying form. The direct inference of underlying forms from overt forms is an idealization that must eventually be removed for an approach to overall language learning.

Assuming each overt form to have only a finite number of interpretations, while an idealization, is not likely to be problematic. In work on algorithms for production-directed parsing (Tesar, 1995), it has been shown that in optimality theoretic systems where each underlying form has an infinite number of candidate structural descriptions, the optimal candidate can be efficiently calculated. Part of the reason for this is that the infinity of candidates comes from the possibility of the insertion (epenthesis) of arbitrary amounts of material. The amount of insertion possible in an *optimal* candidate is necessarily limited as a consequence of the faithfulness constraints, so the computation never need consider candidates with enough insertion to ensure suboptimality. The same principle applies to robust interpretive parsing: if an overt form has an infinite number of possible interpretations, only some finite subset will be possibly optimal. Thus, the learner will be able to restrict itself on principled grounds to only the finite set of 'plausible' interpretations. As is the case for work on parsing algorithms, the primary challenge for learning with ambiguous overt forms lies not with the infinity of the space of candidates provided by the system's formal definitions, but with the structure of the candidate space.

The performance of the algorithm is a consequence of several factors. The use of error-driven constraint demotion limits the length of the lists of mark-data pairs underlying

the ranking hypotheses (Tesar 1995, Tesar & Smolensky 1998). Any list of mark-data pairs assembled with error-driven constraint demotion will either fully determine the correct grammar or reach an inconsistency by the time the list reaches N(N-1)/2 pairs, where N is the number of constraints in the system. This number is an upper bound, and in practice it is generally a large overestimate. It does guarantee that the length of the lists for individual hypotheses will be reasonable, relative to the size of the grammar being learned.

The use of error-driven learning helps in a second way as well. When a new overt form is paired with a ranking hypothesis, learning takes place only if the grammar's optimal description does not match the overt form. If one of the possible interpretations of the overt form is optimal with respect to that ranking hypothesis, the learner does not bother to try other interpretations of that overt form in combination with that ranking hypothesis.

Sources of complexity include the degree of ambiguity of the overt forms and the degree of interdependence of the structural descriptions across different forms. Both are clearly dependent on the particular optimality theoretic system involved. With respect to the degree of ambiguity, the simulations reported here benefitted from a strategy that likely has general application. The algorithm worked through the overt forms in order of increasing length, examining the forms of length two syllables, then those of length three, and so forth. Not surprisingly, shorter forms tend to have a lower degree of ambiguity. Working with forms of low ambiguity early on permits the learner to obtain a significant amount of information without having to consider large numbers of possible interpretations. When the forms of greater ambiguity are reached, and an error occurs, the learner often will have enough information to eliminate the majority of possible interpretations on the spot, because most will be inconsistent with the information obtained earlier from the shorter forms. In some cases, it is possible to learn the correct grammar entirely from the shorter forms, so that no errors ever occur with respect to the longer forms, completely eliminating any effect of the ambiguity of the longer forms. This suggests a more general strategy: the learner should focus on forms with low ambiguity early on. Note that this strategy does not require anything like the existence of completely unambiguous forms.

The entire MRCD approach is motivated by the idea that a great deal of interrestrictiveness exists among the forms of a language. Even if two forms are both highly ambiguous, the expectation is that only a small number of combinations of interpretations of the two forms should be consistent (simultaneously optimal). It is not necessary for any one form to give a full specification of the position of any one constraint, so long as a modest number of forms combine to determine the entire ranking. The greater the interdependence of the forms, the more quickly the learner can detect and eliminate incorrect interpretations of overt forms, thus avoiding having to consider combinations of those (incorrect) interpretations with possible interpretations of later forms. Thus, this approach to learning not only tolerates, but is in fact enhanced by, linguistic analyses in which every single form is the result of interactions among a number of constraints. This is clearly a benefit, given that constraints which each play a role in the explanation of a variety of forms are independently desirable with respect to the concerns of linguistic theory.

6. Acknowledgments

The author would like to thank Eric Bakovic and Alan Prince for useful discussions. The references with a listed ROA number can be obtained electronically, from the Rutgers Optimality Archive, at http://ruccs.rutgers.edu/roa.html.

References

Chomsky, N. 1981. *Lectures on Government and Binding*. Dordrecht: Foris Publications.

Daelemans, W., S. Gillis, & G. Durieux. 1994. The acquisition of stress: A data-oriented approach. *Computational Linguistics*, 20(3):421-451.

Dresher, B. E. 1996. Charting the learning path: Cues to parameter setting. Ms., Department of Linguistics, University of Toronto. Revised version to appear in *Linguistic Inquiry*.

Dresher, B. E. & J. Kaye. 1990. A computational learning model for metrical phonology. *Cognition*, 34:137-195.

Gibson, E. & K. Wexler. 1994. Triggers. *Linguistic Inquiry* 25(4).

Gupta, P. & D. Touretzky. 1994. Connectionist models and linguistic theory: Investigations of stress systems in language. *Cognitive Science*, 18(1):1-50.

Hale, M. & C. Reiss. 1996. Formal and empirical arguments concerning phonological acquisition. Ms., Concordia University (ROA-170).

Hammond, M. 1990. Deriving ternarity. Ms., University of Arizona, Tucson.

Hayes, B. 1980. *A Metrical Theory of Stress Rules*. Doctoral dissertation, Department of Linguistics, Massachusetts Institute of Technology, Cambridge.

Hayes, B. 1995. *Metrical Stress Theory: Principles and Case Studies*. Chicago: University of Chicago Press.

Kager, R. 1994. Ternary rhythm in alignment theory. Ms., Research Institute for language and Speech, Utrecht University (ROA-35).

McCarthy, J. & A. Prince. 1993. Generalized alignment. In G. Booij & J. van Marle (eds.), *Yearbook of Morphology*, 79-154. Dordrecht: Kluwer.

Prince, A. 1990. Quantitative consequences of rhythmic organization. In K. Deaton, M. Noske, & M. Ziolkowski (eds.), *CLS26-II: Papers from the Parasession on the Syllable in Phonetics and Phonology*, 355-398.

Prince, A. & P. Smolensky. 1993. Optimality Theory: Constraint interaction in generative grammar. Technical report, TR-2, Rutgers University Center for Cognitive Science, and CU-CS-696-93, Department of Computer Science, University of Colorado at Boulder. To appear in the Linguistic Inquiry Monograph Series, MIT Press.

Smolensky, P. 1996. On the comprehension/production dilemma in child language. *Linguistic Inquiry*, 27 (ROA-118).

Tesar, B. 1995. Computational Optimality Theory. Doctoral dissertation, Department of Computer Science, University of Colorado, Boulder (ROA-90).

Tesar, B. & P. Smolensky. 1998. Learnability in Optimality Theory. *Linguistic Inquiry*, Spring.

Tesar, B. & P. Smolensky.1995. The learnability of Optimality Theory. In *Proceedings of the Thirteenth West Coast Conference on Formal Linguistics*, 122-137.

Tesar, B. & P. Smolensky. 1996. Learnability in Optimality Theory (long version). Technical Report JHU-CogSci-96-4, Department of Cognitive Science, the Johns Hopkins University (ROA-156).

Wexler, K. & M. R. Manzini. 1987. Parameters and learnability in the binding theory. In T. Roeper & E. Williams (eds.), *Parameter Setting*. Dordrecht: Reidel.

Department of Linguistics / Rutgers Center for Cognitive Science
Rutgers University
18 Seminary Place
New Brunswick, NJ 08903
USA

tesar@ruccs.rutgers.edu