Using Inconsistency Detection to Overcome Structural Ambiguity

Bruce Tesar

The Inconsistency Detection Learner (IDL) is an algorithm for language learning that addresses the problem of structural ambiguity. If an overt form is structurally ambiguous, the learner must be capable of inferring which interpretation of the overt form is correct by reference to other overt data of the language. The IDL does this by attempting to construct grammars for combinations of interpretations of the overt forms, and discarding those combinations that are inconsistent. The potential of this algorithm for overcoming the combinatorial growth in combinations of interpretations is supported by computational results from an implementation of the IDL using an optimality-theoretic system of metrical stress grammars.

Keywords: learnability, metrical phonology, language acquisition, Optimality Theory

1 Structural Ambiguity in Language Learning

1.1 Mutual Entanglement

A central challenge of learning natural languages is that of contending with input data that are structurally ambiguous. The portion of an utterance that is directly perceivable by the learner, labeled here the *overt form,* is *structurally ambiguous* if there is more than one complete *structural description* that may be assigned to it. By structural description, I mean the full constituent structure representation of a candidate. The situation we are concerned with in this article is that where the different structural descriptions are grammatical in different languages.[1]

Structural ambiguity can be illustrated with metrical stress theory. For present purposes, assume that a structural description of a word consists of the ordered sequence of syllables of the word, a grouping of syllables into feet, and an assignment of a stress level to each syllable. The overt form corresponding to a structural description is the ordered string of syllables, along

I would like to thank Jason Eisner, Janet Fodor, Brett Hyde, Jacques Mehler, Joe Pater, Alan Prince, Ken Safir, William Sakas, Vieri Samek-Lodovici, Paul Smolensky, the students of the spring 2000 Rutgers University Learnability and Linguistic Theory seminar, several anonymous reviewers, and the audiences at HOT'97, NELS 28, the CUNY Graduate Center, Carnegie Mellon University, the NELS 30 Workshop on Language Learnability, NYU, MIT, Rochester University, Western Michigan University, and SUNY Stony Brook, for useful comments. Alan Prince also provided many useful comments on multiple drafts of this article. Part of this research was funded by postdoctoral support from the Department of Linguistics, Rutgers University, and the Rutgers Center for Cognitive Science.

[1] Intralinguistic ambiguity, where the same utterance can be assigned more than one description by the same grammar (e.g., *I saw the spy with the binoculars*), can have other interesting implications for language learning, but is beyond the scope of this article.

Linguistic Inquiry, Volume 35, Number 2, Spring 2004
219–253

with the stress levels of the syllables. An overt form is not itself a structural description; it only contains structures for elements that are presumed to be directly observable when a child hears a word uttered.[2] What is missing from the overt form is the foot structure; the child cannot directly ''hear'' foot boundaries. An overt form is ambiguous when more than one structural description shares that overt form. I will refer to a full structural description consistent with an overt form as an *interpretation* of that overt form. An ambiguous overt form has more than one interpretation. A simple example is a three-syllable word with medial main stress: σσ́σ (I use σ to denote a syllable). This overt form is ambiguous between at least two interpretations,[3] including

- an iambic analysis, grouping the first two syllables to form an iambic foot: (σσ́)σ; and
- a trochaic analysis, grouping the final two syllables to form a trochaic foot: σ(σ́σ).

This ambiguity would not necessarily seem consequential if one of the interpretations wasn't really a possible form in a human language; a learner armed with such knowledge could detect the universally ill formed interpretation, or avoid constructing it altogether. But in many instances, including the one just given, more than one of the possible interpretations is a possible interpretation of a human language. The iambic interpretation is that employed by Araucanian: *tipánto* 'year' is analyzed as (*tipán*)*to* (Echeverria and Contreras 1965, Hayes 1995).[4] The trochaic interpretation is that employed by Warao: *koránu* 'drink it!' is analyzed as *ko*(*ránu*) (Hayes 1995, Osborn 1966). The different interpretations assigned by the same language are not motivated by elements intrinsic to that particular trisyllabic form, but by the stressing of other forms of the respective languages (some examples are shown in table 1).

The learner cannot determine the correct interpretation of the overt form on the basis of that overt form alone. This matters to the learner, because the foot structure necessarily mediates between the overt data and the central principles of stress grammars. The principles that can vary among the possible human grammars evaluate full structural descriptions, not overt forms, and they make crucial reference to *hidden structure* such as feet. Dresher (1999) refers to this gap between overt forms and the full structural descriptions evaluated by grammars as the *epistemological problem.* The learner will draw very different conclusions about the grammar it is attempting to learn if it assumes the iambic interpretation than if it assumes the trochaic interpretation.

Note that competent adult speakers of one of the above languages have no problem disambiguating the stress pattern of the overt form, because they possess the correct grammar for their language. But this correct grammar is precisely what the language learner lacks, at least at the

[2] The stress levels in the overt form are a direct translation of the relative prominence of the syllables as expressed in acoustic, observable properties: duration, pitch, and amplitude. The syllable structure itself is, of course, constructed by the learner, based upon the acoustic signal. Syllable structure construction will simply be assumed for the discussion of stress in this article, but in general elements of syllable structure can also be subject to crosslinguistic structural ambiguity.

[3] Another possible interpretation is one with the stressed syllable as a foot by itself, σ(σ́)σ. Such a foot is not unexpected in trochaic, quantity-sensitive languages, when the syllable is heavy.

[4] The Araucanian data are taken from Echeverria and Contreras, while the iambic analysis of the data is due to Hayes. Similarly, the Warao data are taken from Osborn, while the trochaic analysis of the data is due to Hayes.

Table 1
Stress patterns (with structural descriptions) for Araucanian and Warao

Araucanian		Warao	
(wulé)	'tomorrow'	(tíra)	'woman'
(tipán)to	'year'	ko(ránu)	'drink it!'
(elú)(muyù)	'give us'	(rùhu)(náe)	'he sat down'
(elú)(aè)new	'he will give me'	ni(hàra)(páka)	'alligator'

outset. Neither the complete grammar nor the full structural analyses of the forms are directly apparent to the learner; they must be inferred. However, they cannot be inferred independently, because each has strong implications for the other: a grammar assigning a trochaic foot at the right edge goes hand in hand with an interpretation like σ(σ́σ). They must be learned together. This is referred to here as the *mutual entanglement* of grammar and analysis.

It is worth pointing out that the mutual entanglement problem is not parochial to metrical stress, or even to phonology, but is in fact endemic to language learning. Pinker (1987) referred to this problem as the *bootstrapping problem* and specifically discussed it in the context of syntax acquisition. The mutual entanglement there is between the interpretation (syntactic analysis) of sentences and the syntactic grammar of a language: the hidden structure of the interpretations includes the syntactic categories of the words in the sentence and their roles in argument structure. In fact, in syntax, problems of this sort remain even if it is assumed the learner has access to the syntactic categories and argument structure roles of the words, because frequently different tree structures can be assigned by different grammars (see, e.g., Gibson and Wexler 1994). A sentence with SVO word order could have the subject in Spec,IP, or it could have a verb-second description with the subject in Spec,CP. Other possible sources of structural ambiguity in phonology include syllable structure and moraic structure. Structural ambiguity can arise any place where raw overt data can receive more than one interpretation in terms of linguistic representation, and mutual entanglement follows when different possible grammars assign the different interpretations.

Previous proposals for contending with structural ambiguity are discussed in section 1.2. A new approach, one based on inconsistency detection, is described in section 1.3. In this article, the inconsistency detection approach will be instantiated in a learner called the *Inconsistency Detection Learner* (IDL). The IDL is based upon Optimality Theory (Prince and Smolensky 1993), and in particular builds on the constraint demotion approach to learning in Optimality Theory (Tesar and Smolensky 1998, 2000). A brief overview of learning in Optimality Theory is presented in section 2. The details of the IDL are presented in section 3. The results of computer simulations evaluating the efficiency of the algorithm are presented in section 4.

1.2 Previous Proposals for Overcoming Mutual Entanglements

This subsection reviews a combination of prior approaches in language learning. Some are prior approaches to learning metrical stress, while others are prior approaches to contending with mutual

entanglement (not necessarily in metrical stress, or even phonology). The focus is on how the different proposals deal with structural ambiguity and mutual entanglement. Section 1.2.6 briefly outlines how the new proposal of this article, inconsistency detection, relates to the prior work with regard to mutual entanglement.

1.2.1 Avoiding (Explicit) Linguistic Abstractions One approach to language-learning phenomena attempts to utilize quite general pattern recognition techniques for mapping and reproducing the overt forms of the language, without detailed attention to what kinds of abstractions the model develops as it is trained. Thus, the question of structural ambiguity is not so much answered as ignored; the question of what structural analysis (in linguistic terms) is assigned to an overt form is intentionally left unasked. Examples of techniques applied in this way to metrical stress are the Nearest Neighbor method combined with a statistically trainable distance metric (Daelemans, Gillis, and Durieux 1994), perceptron networks (Gupta and Touretzky 1994), and recurrent backpropagation networks (Joanisse and Curtin 1999).

The studies of Joanisse and Curtin and of Daelemans, Gillis, and Durieux evaluated their models on only a single language (Dutch), focusing on the problem of reproducing both regular and irregular forms within a single language, rather than on a learner equally capable of learning the predominant pattern for any of a number of languages. Gupta and Touretzky analyzed the trained weights of the perceptron for 19 different languages, but they focused on weight patterns that correlated with descriptive properties of the stress patterns (such as ''inconsistent primary stress,'' ''stress clash avoidance'') and were primarily interested in explaining differences in perceptron learning times, not in explaining how humans succeed in learning stress patterns.

It should be emphasized that just because the notion ''structural description'' is not commonly discussed with respect to a certain type of model does not automatically mean that the concept does not apply. For example, networks such as the backpropagation network used by Joanisse and Curtin have hidden units, and the activation values of those units for a particular input form constitute an analysis, or ''description,'' of that input. Any model that generalizes beyond a memorized list of input/output pairs has an abstraction mechanism of some kind, and therein lie its ways of interpreting particular input properties for the purposes of assigning an output. Further, one could try to analyze and understand the abstractions used by such models, even in linguistic terms. While the analysis of hidden unit activations in connectionist networks is notoriously difficult, it is not in principle impossible, as a number of specifically constructed example networks demonstrate.

What is missing from all of the studies just cited is any exploration of the range of possible stress systems predicted by the models; Gupta and Touretzky are quite explicit in their lack of regard for such typological issues. These models lack the capacity to explain the central properties of human stress systems—for example, their fundamentally rhythmic quality (Liberman and Prince 1977). The kinds of structural descriptions (and related theoretical principles) of concern in this article are usually motivated by an interest in capturing what different stress systems have in common, as well as capturing the specific ways in which they differ.

1.2.2 Avoiding Assignment and Use of Structural Descriptions during Learning The cue-based learning approach has been applied to principles-and-parameters theories for metrical stress

(Dresher and Kaye 1990, Dresher 1999). In this approach, the problem of structural ambiguity in learning is dealt with by completely avoiding the assignment and use of structural descriptions by the learner while learning. Learners instead invoke mechanisms for evaluating overt forms that are specific to learning and separate from the standard parsing of overt forms involved in language use. These learning-specific mechanisms search for *cues,* which are specified patterns in overt forms. Each parameter has at least one cue associated with it, and if the cue is observed in some sufficient quantity, the corresponding parameter is set to a specific value. Typically, the nature of the cues is such that the parameters must be set in a specific order. The learner cannot set a particular parameter until it has set the other parameters ordered before it (the precise form of a cue may depend upon the values of earlier parameters). This learning approach is decidedly not error driven;[5] throughout learning, the learner remains oblivious to whether it can actually parse the overt forms it is seeing. Indeed, depending upon how one interprets the not-yet-set parameters, the learner may not have a properly determined, usable grammar until the very end of learning.

The definitions of some cues involve the construction of what might be called ''partial descriptions,'' structures that organize certain aspects of the overt form being checked. These constructions can sometimes look suspiciously similar to components of the structural descriptions assigned by grammars; for example, the cue for the main stress parameter is ''Scan a *constituent-sized window* at the edge of a word. Main stress should consistently appear in either the left or right window'' (emphasis added) (Dresher 1999:34). However, they are necessarily different from full structural descriptions, as the order and design of these analytic structures are designed to avoid the structural ambiguity of the full descriptions actually licensed by the grammars (and the linguistic theory). In other words, although the regular structural descriptions are posited by linguistic theory precisely because they are so effective at mediating between overt forms and central linguistic principles, the learner uses an entirely separate class of analytic devices for mediating between overt forms and linguistic principles for the purposes of learning. This can be a cause of dissatisfaction, especially when the complexity of the learning-specific system can rival (or even exceed) that of the linguistic theory itself. See Bertolo et al. 1997a for further discussion and results on cue-based learners.

1.2.3 Enumerative Search and Random Search Linguistically plausible spaces of grammars are quite large, making exhaustive search techniques like simple enumerative learning (Gold 1967) hopelessly inefficient. An alternative is random search with *error-driven learning* (Wexler and Culicover 1980). Such an error-driven algorithm generally holds a single grammar hypothesis and has a mechanism for changing its hypothesis to a different grammar whenever it encounters an overt form that its current hypothesis cannot parse (such an occurrence is deemed ''an error''). One such algorithm is the Triggering Learning Algorithm (TLA) (Gibson and Wexler 1994). Grounded in the principles-and-parameters framework, the TLA is error driven, responding to an

[5] For a description of error-driven learning, see section 1.2.3.

error on an overt form by changing the value of a single parameter at random. If the parameter change results in a grammar that can parse the overt form without error, then the change is kept. Unfortunately, in general the randomness of the search and the constraining topology of the parametric theory do little to prevent learning times from rivaling those of exhaustive search.[6] In fact, the use of parametric theory to constrain the direction of search can be a significant hindrance, making learning impossible in some cases (Niyogi and Berwick 1996). For further discussion of the limitations of the TLA, see Fodor 1998b.

More complex forms of random search, such as genetic algorithms, have also been applied to grammar learning (Clark 1992, Clark and Roberts 1993, Pulleyblank and Turkel 1998). Such algorithms can avoid getting stuck in the way that the TLA can, but have high costs in terms of learning time and required computational effort.

What these enumerative and random search approaches have in common is that they use structural analyses of overt forms, but only for the purpose of determining if an overt form is possible for a particular grammar. When a grammar fails to successfully parse an overt form (when an error occurs), nothing about the overt form is used to determine how the grammar might be changed to achieve success; other extrinsic devices (ordered grammar lists, random grammar changes) are used instead. The overt forms figure only in the evaluation of the grammars. The analyses of overt forms are relevant only in verifying that a grammar is capable of parsing those overt forms. Ambiguity of interpretation for overt forms is certainly possible, but the learners themselves are relatively oblivious to it; either a grammar can parse an overt form (by some structural interpretation or other) or it cannot. The price is paid in terms of learning times; for example, see Sakas and Fodor 2001 for an evaluation of the performance of the TLA.

1.2.4 Insisting on Unambiguous Forms A more explicit confrontation with ambiguity appears in the Structural Triggers Learner (STL) (Fodor 1998b, Sakas and Fodor 2001). The proposal is for the learner to learn only from *unambiguous* forms. The term ‘‘unambiguous’’ has a different meaning for the STL than it does for much of the rest of this article. The STL is more immediately concerned with whether an overt form is consistent with more than one setting of a parameter (parametric ambiguity) than with whether an overt form has more than one structural interpretation (structural ambiguity). In practice, the two types of ambiguity frequently go together. The STL interprets an overt form on the basis of whatever parameters have already been set (initially, none). For the learner to get started, it must encounter an overt form that acts as a *trigger* for at least one parameter (the form *requires* the parameter to be set to a specific value). The learner can then make use of those parameter settings, which may serve to disambiguate another overt form: the overt form may have several interpretations, but if all but one of them contradict at least one of the learner’s set parameters, the learner can presume the consistent interpretation.

[6] A topology is usually imposed on the parameter space via the Single-Value Constraint (Gibson and Wexler 1994), mandating that the learner change the value of only one of the parameters when considering an alternative grammar.

Thus, for the STL, a trigger can be ''unambiguous'' relative to what other prior parameters have been set. The learner can succeed if it can find a critical sequence of triggers, each trigger rendering the next unambiguous, that ultimately sets all of the parameters.

For the STL to work, it must be capable of parsing overt forms while having only some (or none) of its parameters set. This turns out to be a quite nontrivial matter. The STL proposes a mechanism known as a *superparser.* It is not at all clear what a superparser would be like in a more traditional principles-and-parameters system. Sakas and Fodor instead cast the TLA within a framework based upon the Minimalist Program (Chomsky 1995); in their view, a parameter is a small piece of syntactic structure, a tree fragment, and learning a grammar means identifying which of the possible parameters are in use by the target language. Constructing such a superparser is a challenge, and is the topic of current research (Fodor 1998a).

Another potential problem is the availability of unambiguous triggers; the learner could end up waiting a long time, allowing lots of potentially informative input data to go by, waiting for an unambiguous trigger to appear (Bertolo et al. 1997b). The STL might get stuck forever if it reaches a point where each overt form relevant to an unset parameter is ambiguous for that parameter, even if a small set of overt forms would be collectively unambiguous.

The STL responds to an overt form by attempting to assign it structural interpretations, using whatever parameter settings it already has. If only a single interpretation remains, or perhaps if the remaining interpretations all agree in their required setting of a not-yet-set parameter, then that interpretation is used to learn more about the target grammar. If not, the STL refuses to learn from the form. Thus, the STL makes more use of the structural interpretations in directing learning than the previous methods discussed, but places strong restrictions upon the structural interpretations it can actually use.

1.2.5 Iterative Refinement of Analysis and Grammar The Robust Interpretive Parsing/Constraint Demotion algorithm (RIP/CD) (Tesar 1998b, Tesar and Smolensky 2000) is cast within the framework of Optimality Theory (OT) and takes advantage of the optimizing nature of OT. The RIP part uses the learner's working constraint hierarchy to interpret an overt form. If the form has more than one interpretation, the learner selects the interpretation that fares best on the learner's current constraint hierarchy. In this way, the learner can use its current grammar to assign an interpretation to an overt form even when that overt form does not correspond to any description that is grammatical according to that same grammar. This still permits error-driven learning, however, because the learner can then take the underlying form of the interpretation and see how the current grammar would have analyzed it. If the current grammar's analysis matches the learner's interpretation of the overt form—that is, if the learner would say it the way the learner heard it—then no error has occurred for learning purposes. If the two do not match—that is, if the learner would say it differently from the way the learner heard it—then an error has occurred, and learning should take place. For OT, the ''mismatch'' takes the place of ''failing to parse'' in defining an error for the purposes of error-driven learning.

When an error has been detected, the learner changes its grammar in an attempt to make its interpretation of the overt form optimal. The algorithm for performing this grammar change is

Constraint Demotion,[7] the CD part of RIP/CD. Having now modified its grammar, the learner can repeat the process, using the new grammar to robustly parse the overt form and check for an error. The idea is for the learner to iterate back and forth between the two stages, parsing (RIP) and grammar learning (CD), until a convergent grammar is arrived at in which no error is detected.

RIP/CD has been tested computationally on some OT systems for metrical stress (Tesar 1998b, Tesar and Smolensky 2000). When it succeeds, it is very fast, arriving at a correct grammar in very few steps. The challenge to this learner posed by structural ambiguity is the possibility of having RIP assign the wrong interpretation to an overt form. This will lead CD to attempt a different grammar than the target grammar. While it is capable of overcoming the incorrectly assigned interpretation in some circumstances, success is not guaranteed, and some conditions have been determined that can cause the algorithm to get stuck, doomed to never converge on a correct grammar (Tesar 1998b, Tesar and Smolensky 2000). These conditions pose a threat to the viability of RIP/CD as a model of human language acquisition, absent some method of avoiding or recovering from these conditions.

1.2.6 A New Approach: Inconsistency Detection What is desirable is a learning algorithm that can use the observed overt forms to actively guide the learner toward the correct grammar, does not need to wait indefinitely for strictly unambiguous forms, but can proceed in a way that avoids getting stuck. RIP/CD gets stuck in part because it deals with structurally ambiguous overt forms by always limiting itself to one of the possible interpretations. The alternative proposed here is that the learner can benefit greatly from actively considering more than one possible interpretation, using other overt forms of the language to determine for sure which interpretation is the correct one. The claim is that the learner can do this in a way that is guaranteed to succeed, but is nevertheless computationally efficient. The key to the approach is inconsistency detection.

1.3 Detecting Inconsistencies between Analyses

1.3.1 The Concept of (In)consistency If a single overt form is structurally ambiguous, that ambiguity can only be resolved by reference to other forms in the language. This is what linguists do when analyzing a language; they draw conclusions about things like foot structure from a variety of forms in the language. This article proposes a learning algorithm that brings other overt forms to bear on an ambiguous form in a rather direct way, by looking for interpretations of overt forms that are mutually consistent.

An interpretation of an overt form is *consistent* with some interpretation of another overt form if there exists a possible grammar in which both interpretations are grammatical. Therefore,

[7] This Constraint Demotion (CD) algorithm is distinct from, although fundamentally related to, the Recursive Constraint Demotion (RCD) algorithm. CD takes an existing constraint hierarchy and modifies it in response to a single piece of information; it can be invoked repeatedly on different pieces of information to perform a sequence of changes to the hierarchy. In contrast, RCD takes a complete set of information at once and constructs an entire hierarchy from scratch based upon all of the information in the set. See Tesar and Smolensky 2000 for further discussion of the relationship between the two algorithms.

consistency is heavily dependent upon the linguistic theory being used, because it is the linguistic theory that determines what the possible grammars are. Assume (for purposes of illustration) a very simple metrical theory in which any given language must choose a consistent, bisyllabic foot form, a consistent direction of foot alignment, whether or not to iterate feet (include secondary stressing), and a consistent edge for alignment of main stress (a much more sophisticated metrical theory, described in appendix A, was used for the actual simulations described in section 4). The interpretations σ(σ́σ) and (σ́σ)(σ̀σ) are consistent with each other, because there exists a grammar that admits both as grammatical: the grammar that iterates trochaic feet from right to left and places main stress on the leftmost stressed syllable. The interpretations σ(σ́σ) and σσ(σ́σ) are also consistent with each other, because there exists a grammar that places a single trochaic foot at the right edge of the word (with no secondary stresses). However, σ(σ́σ) is *inconsistent* with (σσ́)σσ; no grammars in our assumed system exist that assign right-aligned trochaic feet to three-syllable words and left-aligned iambic feet to four-syllable words.

Given two overt forms from the target language, and given an interpretation of each of those overt forms, if the two interpretations are inconsistent, then the learner may conclude that at least one of the interpretations is not the correct analysis of its corresponding overt form. Given the overt forms σσ́σ and σσ́σσ, the learner can rule out the combination of σ(σ́σ) and (σσ́)σσ, as shown above. The learner can completely rule out the interpretation σ(σ́σ) by going on to notice that every other interpretation of σσ́σσ—namely, σ(σ́σ)σ—is also inconsistent with σ(σ́σ). By eliminating σ(σ́σ), the learner could use the observation of the overt form σσ́σσ to effectively disambiguate the overt form σσ́σ in favor of the interpretation (σσ́)σ.

A few points are worth emphasizing right away. The nonexistence of a grammar can only directly determine the incompatibility of two interpretations, that is, two full structural descriptions. Determining that one interpretation of an overt form, like σ(σ́σ) above, is inconsistent with a different overt form, like σσ́σσ above, requires showing that the former interpretation is inconsistent with every possible interpretation of the latter overt form—that is, that the sets {σ(σ́σ),(σσ́)σσ}, {σ(σ́σ),σ(σ́)σσ}, and {σ(σ́σ),σ(σ́σ)σ} are each inconsistent. Further, (in)consistency is a property not just of pairs of structural descriptions, but of sets of arbitrary size. Even a single structural description is effectively inconsistent if it is not admitted as grammatical by any possible grammar. It is also possible to have two structural descriptions that are consistent as a pair, but when combined with another structural description form an inconsistent triple.

1.3.2 Using Inconsistency Detection in Learning Given a set of overt forms from the target language, the learner could use inconsistency detection to rule out combinations of interpretations of the overt forms. Given enough data, the learner should in principle be able to eliminate all but one interpretation of each overt form. The grammar admitting each of the (noneliminated) interpretations as grammatical should be the correct grammar. That is the inspiration behind the learning proposal of this article, the *Inconsistency Detection Learner* (IDL).

Forging that inspiration into a viable learning theory requires several things. It requires (a) a strong linguistic theory, making specific claims about possible grammars, and what structural descriptions are grammatical for each grammar; (b) an efficient procedure for determining when

a set of structural descriptions is inconsistent; (c) an efficient procedure for determining a grammar corresponding to a consistent set of structural descriptions. If determining whether a set of descriptions is consistent required separately examining the extensional consequences of every possible grammar, then this approach would be woefully inadequate, given the size of realistic grammar spaces, in the same way as the enumerative and random search approaches discussed earlier. Fortunately, it is in fact possible to determine the (in)consistency of a set of descriptions far more efficiently, given the proper linguistic theory; this will be shown below, in section 2.

2 Learning in Optimality Theory

2.1 Learning Constraint Rankings: Recursive Constraint Demotion

With constraint demotion, learning is based upon *winner-loser pairs.* A winner-loser pair represents a comparison between a target candidate (presumed to be grammatical), referred to as the *winner,* and a competitor, referred to as the *loser.* Tableau (1) shows the constraint violations for both members of an example winner-loser pair.

(1) Candidates of an example winner-loser pair (see appendix A for constraint names)

	/σσσσ/	PARSE	MR	ML	AFR	AFL	FNF	IAMB	FTBIN	NF
Winner	(σ̀σ)(σ́σ)			**	**	**		**		*
Loser	(σσ́)σσ	**	**		**		*			

What concerns the learner is the relative number of violations of a constraint between the winner and the loser. These relationships can be seen quite a bit more clearly if this winner-loser pair is represented with a comparative tableau (Prince 2000). Tableau (2) presents the winner-loser pair of (1) in this way.

(2) Comparative tableau for the example winner-loser pair in (1)

Loser	Winner	PARSE	MR	ML	AFR	AFL	FNF	IAMB	FTBIN	NF
(σσ́)σσ	(σ̀σ)(σ́σ)	W	W	L		L	W	L		L

In a comparative tableau, a single row corresponds to a winner-loser pair, and each constraint is marked in its column to indicate which candidate it prefers. A W mark in the column for the PARSE constraint indicates that PARSE *prefers the winner* over the loser, meaning that the loser violates PARSE more times than the winner does; how many more times the loser violates PARSE is not important. The L mark in the column for ML indicates that ML *prefers the loser.* The blank in the column for AFR indicates that it has no preference between the two candidates. Notice that this does not necessarily mean that both candidates satisfy the constraint (in this instance, both candidates violate AFR twice), only that both violate the constraint an equal number of times.

The winner-loser pair is the fundamental unit of information about a constraint ranking. The Recursive Constraint Demotion algorithm (Tesar and Smolensky 1994, 2000) is an efficient means

for deriving a constraint hierarchy that is consistent with all of the information contained in a set of winner-loser pairs. A consistent hierarchy orders the constraint columns so that, for each row, the leftmost nonblank cell contains a W mark. Recursive Constraint Demotion (RCD) differs from the ''online'' version of constraint demotion (Tesar and Smolensky 1998) in that it works with a set of winner-loser pairs all at once, rather than processing winner-loser pairs serially (discarding one before processing the next) as online Constraint Demotion (online CD) does.

RCD can be illustrated using the data in (3) as a starting point.

(3) Illustrating RCD: the initial data set

Loser	Winner	PARSE	MR	ML	AFR	AFL	FNF	IAMB	FTBIN	NF
(a) (σ́σ)σ	σ(σ́σ)		W	L	W	L				L
(b) (σ̀)(σ́σ)	σ(σ́σ)	L			W		W		W	
(c) σσ(σ́σ)	(σ̀σ)(σ́σ)	W			L			L		
(d) σ(σ̀σ)(σ́σ)	(σ̀σ)σ(σ́σ)				L	W				

RCD proceeds by identifying those constraints that can be ranked highest. They are the ones that do not prefer any losers, identifiable by the lack of an L mark in their column in the tableau. RCD ranks those constraints in the top stratum of the hierarchy it is constructing. Here, the stratum is {MR FNF FTBIN}; those constraints will dominate the others. RCD then removes from the tableau those winner-loser pairs accounted for by the constraints just ranked: those pairs assessed with a W mark by one of the constraints just ranked. In our example, the first two pairs, (a) and (b), are removed. The learner then removes the columns for the constraints just ranked from the tableau, leaving the reduced tableau shown in (4).

(4) Illustrating RCD: after the first pass

Loser	Winner	PARSE	ML	AFR	AFL	IAMB	NF
(c) σσ(σ́σ)	(σ̀σ)(σ́σ)	W		L		L	
(d) σ(σ̀σ)(σ́σ)	(σ̀σ)σ(σ́σ)			L	W		

The learner now repeats the same procedure on the reduced tableau, placing the now-rankable constraints into the next stratum down in the hierarchy. In (4), there are four rankable constraints (those not preferring any losers), and ranking them in the next stratum creates the partial hierarchy {MR FNF FTBIN} ≫ {PARSE ML AFL NF}. Both of the remaining winner-loser pairs can be accounted for by the constraints just ranked, so they may be removed, leaving an empty tableau. The remaining constraints are then ranked at the bottom of the hierarchy, and RCD terminates, returning the hierarchy {MR FNF FTBIN} ≫ {PARSE ML AFL NF} ≫ {AFR IAMB}.

RCD has a particular property that makes it very attractive for use with an inconsistency detection approach to learning. When given an inconsistent set of winner-loser pairs, RCD does not run on and on in a futile effort to find a ranking that will work; it detects the inconsistency and halts. Consider the list of winner-loser pairs in (5).

(5) Inconsistent winner-loser pairs

Loser	Winner	AFR	AFL	IAMB	FNF	MR	ML	NF	PARSE	FTBIN
(σσ́)σ	σ(σ́σ)	W	L	L	W	W	L	L		
σσ(σ́σ)	(σσ́)σσ	L	W	W	L	L	W	W		

RCD, as always, starts by identifying which constraints can be ranked highest, which means finding the constraints with no L mark in their column. In this case, two constraints have empty columns and can be ranked on the first pass. No winner-loser pairs are removed on the first pass, however, because the ranked constraints prefer no winners. On the second pass, every remaining constraint has an L mark. This means that the winner-loser pairs collectively require that every remaining constraint be dominated by some other (remaining) constraint. That cannot happen in a strict dominance hierarchy, immediately revealing that the list is inconsistent. For the particular data involved, what it means is that, while there are some rankings by which σ(σ́σ) is grammatical, and there are some rankings by which (σσ́)σσ is grammatical, there are no rankings by which both are simultaneously grammatical.

2.2 Loser Selection with RCD: Multirecursive Constraint Demotion

While the winners are derived directly from data, the losers must be generated by the learner. Given a winner, creating a separate winner-loser pair for every loser is computationally infeasible, and most of the losers will not be *informative;* that is, they will not provide the learner with new information about the ranking. Fortunately, a learner can find an informative loser (if one exists) for a given winner by using parsing (Tesar 1998a). Using the underlying form for the winner, the learner computes the candidate that is optimal according to its current grammar. This is the function defined by the grammar, and the process of computing it is known as production-directed parsing (Tesar and Smolensky 1998). If the currently optimal candidate is not identical to the winner, the learner adopts it as a loser and forms a winner-loser pair.

Previously, the parsing-based approach to forming winner-loser pairs has been described in conjunction with online constraint demotion. However, it is possible to use the same approach to loser selection with RCD, forming an algorithm called *Multirecursive Constraint Demotion* (MRCD) (Tesar 1997). In online CD, whenever a winner-loser pair is formed, it is used to directly manipulate a constraint hierarchy, demoting constraints within the hierarchy. In MRCD, the newly formed winner-loser pair is added to an existing list of winner-loser pairs, and the learner applies RCD to the entire list, deriving an entire new constraint hierarchy.

MRCD maintains at all times a list of winner-loser pairs, from which a corresponding constraint hierarchy may be derived by applying RCD to the list. At the outset, the learner has no winner-loser pairs, so the initial hypothesis is the result of applying RCD to an empty list, namely, a monostratal hierarchy with all constraints tied in a single stratum. As each target description is encountered, the learner consults the current hypothesis to see if it would select a different description as optimal for the same underlying form. If it would, then that description selected

by the current hypothesis is paired (as the loser) with the target description (as the winner). This winner-loser pair is added to the learner's list of winner-loser pairs. RCD is then applied to the (newly extended) list to derive a new constraint hierarchy. In this way, MRCD can interact with the environment in an ''online'' fashion, receiving data one form at a time, while retaining internally a list of the winner-loser pairs it has constructed. The retained list of pairs is crucial to the ability to detect inconsistencies. MRCD also shares the correctness and data complexity properties of other instantiations of constraint demotion.[8] Because the winner-loser pairs are constructed via error-driven learning, pairs are constructed only for forms that are not already optimal. Thus, MRCD stores only as many pairs as necessary (less than the square of the number of constraints) to determine the correct ranking, regardless of how much data it receives.

MRCD, then, provides two of the algorithmic requirements stated in section 1.3.2: it can construct a grammar corresponding to a consistent set of structural descriptions, and it can determine when a set of structural descriptions is mutually inconsistent. Section 3 will demonstrate how MRCD can be used to overcome structural ambiguity.

3 Learning with Inconsistency Detection

I have established that MRCD is capable of both efficiently determining if a set of structural descriptions is consistent, and efficiently finding a correct grammar for a consistent set of structural descriptions. Now we want to consider how to apply these properties in dealing with structural ambiguity.

3.1 The Combinatorics of the Problem

To put things in perspective, let us consider a simple approach that in general will not produce satisfactory results. MRCD can be applied to a set of full structural descriptions, and it will either determine that the set is inconsistent or return a grammar consistent with all of the descriptions. This means that we could try to deal with structural ambiguity by collecting a set of overt forms, and for each overt form generate all possible interpretations of the form. If the overt forms are adequate to uniquely determine the target grammar, then there should be only one combination of interpretations, one interpretation per overt form, that is consistent, namely, the set of interpretations assigned by the target grammar. All other combinations of interpretations should be inconsistent. Therefore, MRCD can be applied to each combination of interpretations; all but one combination should result in inconsistency, and the correct combination will result in the target grammar. I will call this the *Batch All Data* (BAD) algorithm.

[8] If given a sequence of structural descriptions all of which are consistent with each other, MRCD is guaranteed to find a grammar generating all of the descriptions. If N is the number of constraints, not more than $N(N-1)/2$ winner-loser pairs are necessary for MRCD to determine the grammar. These results follow directly from the analogous existing results for RCD and parsing-based loser selection. See Tesar and Smolensky 2000 for proofs of the relevant prior results, along with further analysis.

Suppose we have three overt forms, each ambiguous between two possible interpretations. The number of combinations of interpretations is 2 * 2 * 2 = 8 combinations. As long as there are a finite number of overt forms, and a finite number of interpretations of each overt form, the BAD algorithm will be possible. But it will not in general be practical. The problem is that the number of combinations grows exponentially in the number of ambiguous overt forms. If there are M ambiguous overt forms, and each has only two possible interpretations, the number of combinations is 2^M. Increasing the degree of ambiguity only makes things worse. Once the number of ambiguous overt forms to be used gets much larger than the number of constraints, the number of combinations of interpretations will grow much larger than the number of constraint rankings. At that point, it would be more efficient to simply test directly each total ranking of the constraints. Fortunately, there is a better way to make use of inconsistency detection, described in section 3.2.

3.2 Considering Multiple Interpretations

The IDL searches the same space that the BAD algorithm does, but it goes about the search much more intelligently. The goal is to find the correct combination of interpretations, while evaluating only a fraction of the combinatorial possibilities along the way. Here is the IDL in a nutshell. When the IDL encounters an overt form, it separately processes the form with each of its existing grammar hypotheses (the learner starts with a single hypothesis). A hypothesis processes an overt form by first performing error detection. If no error is detected, the hypothesis is retained unchanged. If an error is detected, the possible interpretations of the overt form are generated. For each interpretation, the IDL separately constructs a new hypothesis by using MRCD to extend the prior hypothesis with the interpretation. If the resulting new hypothesis is inconsistent, it is discarded; otherwise, it is retained. All retained hypotheses are used when processing the next overt form the learner receives. Thus, the IDL depends upon the claim that a learner is capable of maintaining and evaluating multiple grammar hypotheses during learning. The IDL crucially differs from the BAD algorithm in trying to eliminate from consideration some interpretations of one overt form before working on the next, thus reducing the overall number of combinations considered.

I will now describe and illustrate the IDL using the primary word stress pattern of Polish, a trochaic, quantity-insensitive language. In Polish word stress (Rubach and Booij 1985), main stress goes on the penultimate syllable, and secondary stresses iterate from the beginning of the word, starting with the first syllable. Examples are shown in table 2. In words with an odd number of syllables, no secondary stress is placed on the antepenultimate syllable, avoiding stress clash. This illustration uses the metrical theory described in appendix A, but with one simplification: the possibility of quantity sensitivity is not included. Accordingly, the constraint WSP is left out of the tableaux of this section, and the syllables are not distinguished as light or heavy. All key aspects of the algorithm are still shown in this (simpler) case, and quantity sensitivity does not play a role in the Polish stress pattern.

The learner starts with a single grammar hypothesis, containing no winner-loser pairs. When the first overt form is encountered, error detection takes place. Relative to any hypothesized

Table 2
The Polish word stress pattern (Rubach and Booij 1985)

Stress pattern	Example	Gloss
(σ́σ)	dómu	'house'
σ(σ́σ)	spokójny	'quiet'
(σ̀σ)(σ́σ)	pròpagánda	'propaganda'
(σ̀σ)σ(σ́σ)	sàksofonísta	'saxophone player'
(σ̀σ)(σ̀σ)(σ́σ)	rèwolùcjonísta	'revolutionary'
(σ̀σ)(σ̀σ)σ(σ́σ)	rèwolùcjonistámi	'revolutionary' (instr. pl.)

grammar, the learner can perform error detection with an overt form by obtaining the corresponding underlying form[9] and parsing the underlying form with the hypothesized grammar. If the grammar currently assigns, to that underlying form, a description that does not overtly match the received overt form, an error has occurred, and learning needs to take place.[10] That very same (nonmatching) optimal description will serve as an informative loser, as described in section 2.2. If the form does not cause an error, no learning takes place, and the learner moves on to the next overt form. In all likelihood, an error will occur for the first overt form. At that point, the learner must contend with the structural ambiguity of the overt form. The IDL generates every possible full interpretation of the overt form and separately applies MRCD to each one.[11] If an interpretation is not a possible optimum (it cannot be optimal under any ranking), then MRCD will detect inconsistency and that interpretation will be discarded. Any interpretation that is possibly optimal will result in a grammar hypothesis with one or more winner-loser pairs. Each such hypothesis is retained by the learner when it goes on to consider the next overt form.[12] This means that the learner must have the capacity to maintain more than one grammar hypothesis across forms.

[9] This crucially assumes that the learner can recover the correct underlying form from the overt form. In the stress system discussed in this article, obtaining the underlying form is simple, because the underlying form is simply the syllable string of the overt form without the stress levels; the overt form contains the underlying form. For other linguistic systems involving violable correspondence between underlying and surface forms, the problem of recovering the correct input is far from trivial; this is discussed further in section 5.5.

[10] It is possible that the learner fails to detect an error because its grammar is able to analyze the overt form, but assigns a different interpretation from the target grammar. This will not cause any serious difficulty: because no error was detected, the learner doesn't perform any learning based upon a commitment to the (incorrect) interpretation of the overt form.

[11] The possible interpretations of a given overt form can be computed using the same type of computational machinery already in place for parsing. Robust interpretive parsing optimizes over the set of interpretations of an overt form with respect to some constraint hierarchy. For the computer simulations described in section 4, an empty constraint hierarchy was used with robust interpretive parsing: in such a circumstance, all candidates ''tie,'' because no violations are assessed. Performing robust interpretive parsing with an empty constraint hierarchy returns all of the tied candidates, which are precisely the interpretations of the overt form. A more sophisticated approach would be to filter out harmonically bounded interpretations. See section 5.5 for further discussion of this.

[12] In principle, all that the learner crucially needs to store are the interpretations (the winners of the winner-loser pairs); the winner-loser pairs and corresponding constraint hierarchy can be reconstructed via MRCD. It is the set of interpretations that is the basis for the hypothesis. In practice, that could mean a great deal of duplication of effort, and storing the winner-loser pairs seems much more efficient, especially considering the modest number of winner-loser pairs that will ever be associated with any single hypothesis.

Suppose that the first word the learner hears is *spokójny*. This is a trisyllabic word with medial main stress. The initial hierarchy has all constraints tied, so no single candidate will be optimal. In particular, the candidate (*spókoj*)*ny* will do at least as well as any interpretation of *spokójny*. Thus, an error is detected (an interpretation matching the overt form must be the *sole* optimum in order to avoid being flagged as an error). Under the metrical theory adopted here, it has three possible interpretations, here labeled *A1, A2,* and *A3*.

A1: spo(kójny)
A2: spo(kój)ny
A3: (spokój)ny

The learner attempts to construct a hypothesis for each of the three interpretations. It succeeds for A1 after accumulating two winner-loser pairs, constructing hypothesis H(A1), shown in (6). It also succeeds for A3 after accumulating three winner-loser pairs, constructing hypothesis H(A3), shown in (8). However, A2 proves to be inconsistent after one winner-loser pair, as shown in (7). Interpretation A2 is not a possible optimal form; it will be suboptimal under any ranking. Thus, the learner can discard interpretation A2. After processing *spokójny,* the learner is maintaining two hypotheses, H(A1) and H(A3).

(6) H(A1): {FtBin MR AFR FNF} ≫ {Parse ML Iamb AFL NF}

Loser	Winner	FtBin	MR	AFR	FNF	Parse	ML	Iamb	AFL	NF
(spokój)ny	spo(kójny)		W	W	W		L	L	L	L
(spòkoj)(ný)	spo(kójny)	W		W	W	L	W		W	

(7) Interpretation A2: *spo*(*kój*)*ny* is harmonically bounded by (*spokój*)*ny*

Loser	Winner	FtBin	MR	AFR	FNF	Parse	ML	Iamb	AFL	NF
(spokój)ny	spo(kój)ny	L				L	L		L	

(8) H(A3): {FtBin ML AFL Iamb NF} ≫ {Parse MR AFR FNF}

Loser	Winner	FtBin	ML	AFL	Iamb	NF	Parse	MR	AFR	FNF
(sp koj)ny	(spok j)ny				W					L
(spok j)(n yx)	(spok j)ny	W	W	W		W	L	L		W
spo(kojnyx)	(spok j)ny		W	W		W		L	L	

Figure 1 gives a diagrammatic overview of the Polish example described in this section; the presentation thus far, of the learner's response to *spokójny,* is depicted in the top part of the figure.

In the figure, each downward arrow from an interpretation, like A1, to a hypothesis, like H(A1), represents the application of MRCD. Each arrow is accompanied by an indication of ''steps.'' A *learning step* is the addition of a new winner-loser pair to a list; thus, H(A1) contains

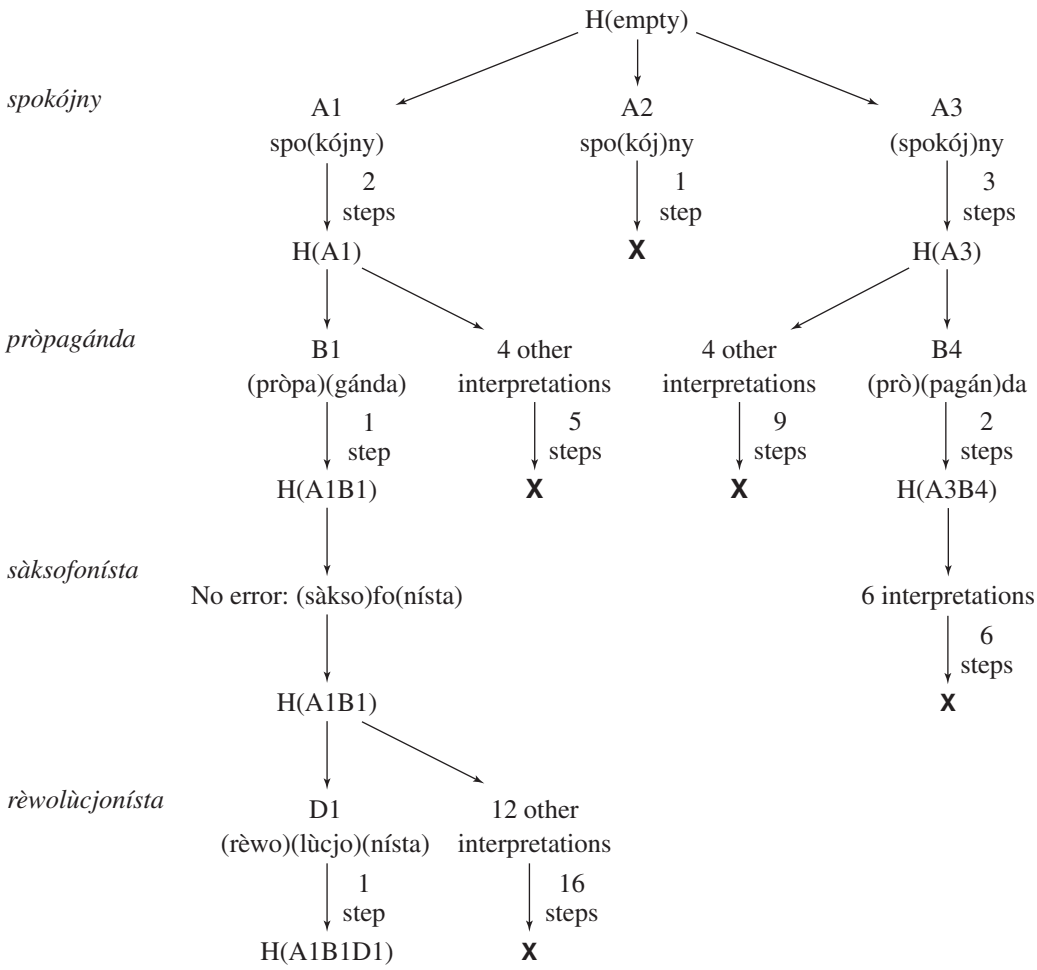

Figure 1
Learning Polish (46 learning steps are shown)

two winner-loser pairs. This definition of learning step provides the unit of effort used to measure the performance of the algorithm.

The learner then proceeds to the next overt form. Error detection takes place again, only now the learner is separately checking the overt form against each grammar hypothesis it possesses. The learner proceeds as described for the first overt form, only with a parallel process for each grammar hypothesis. For each hypothesis on which an error occurs for the new overt form, all interpretations of the new overt form are considered. In the Polish example, the next overt form the learner encounters is *pròpagánda.* This overt form has five possible interpretations, shown

Table 3
Five interpretations for */pròpagánda/*

B1	B2	B3	B4	B5
(pròpa)(gánda)	(pròpa)(gán)da	(prò)pa(gán)da	(prò)(pagán)da	(prò)pa(gánda)

in table 3. Hypothesis H(A1) does not uniquely stress the word in the attested manner, so an error is detected. Specifically, H(A1) has MR and AFR dominating PARSE, so it prefers the candidate *propa*(*gánda*). The learner attempts to separately reconcile each of the five interpretations with hypothesis H(A1). The learner succeeds with interpretation B1 after adding one additional winner-loser pair, yielding the hypothesis H(A1B1), shown in (9). Each of the other four interpretations (B2–B5) proves to be inconsistent with H(A1), with a total of five learning steps needed to determine that. An example of this is the inconsistency of H(A1) with interpretation B4, (*prò*)(*pagán*)*da.* After only one additional winner-loser pair, the inconsistency is revealed. The hypothesis is shown in (10). Note that every single constraint prefers at least one loser; no matter which constraint is ranked at the top, at least one of the winner-loser pairs will not be satisfied.

(9) H(A1B1): {FTBIN MR FNF} >> {PARSE ML AFL NF} >> {AFR IAMB}

Loser	Winner	FTBIN	MR	FNF	PARSE	ML	AFL	NF	AFR	IAMB
(spok j)ny	(spo)k jny		W	W		L	L	L	W	L
(sp koj)(n yk)	spo(k jny)	W		W	L	W	W		W	
propa(g nda)	(pr pa)(g nda)				W				L	L

(10) H(A1B4): inconsistent hypothesis

Loser	Winner	FTBIN	MR	FNF	PARSE	ML	AFL	NF	AFR	IAMB
(spok j)ny	(spo)k jny		W	W		L	L	L	W	L
(sp koj)(n yk)	spo(k jny)	W		W	L	W	W		W	
propa(g nda)	(pr)(pag n)da	L	L	L	W	W	W	L	L	W

Hypothesis H(A3) stresses the word as (*propá*)*ganda,* which also does not match the observed overt form, so it detects an error. The learner attempts to separately reconcile each of the five interpretations with hypothesis H(A3). The learner succeeds with interpretation B4 after adding two additional winner-loser pairs, yielding the hypothesis H(A3B4). Each of the other four interpretations (B1–B3,B5) proves to be inconsistent with H(A3), with a total of nine learning steps needed to determine that.

Notice that inconsistency detection has cut down the possible combinations considerably. Over the first two forms, there are 3 * 5 = 15 possible combinations of interpretations, but only two combinations are actually consistent, A1B1 and A3B4. After two pieces of data, the learner

has 2 hypotheses. When the learner confronts a new ambiguous form, it will be combining interpretations of the new form with only 2 prior hypotheses, not 15.

The processing of *pròpagánda* illustrates the potential for combinatorial growth: the number of separate combinations being evaluated by the learner in parallel could imaginably be as great as the number of learner grammar hypotheses multiplied by the number of interpretations of the new overt form. Inconsistency between forms is crucial to preventing the number of hypotheses simultaneously maintained by the learner from growing rapidly. If MRCD is attempting to reconcile an interpretation with a hypothesis, and the two together are inconsistent, then that combination is discarded. This means that the learner will never have to consider that combination of interpretations together with any of the interpretations of subsequent overt forms; a large piece of the combinatorial growth is thus cut off. As more overt forms are observed, the hypotheses maintained by the learner become more constrained: more forms means more potential for inconsistency to be detected. This is the IDL: it determines the correct interpretations of the overt forms by detecting inconsistencies between incorrect interpretations and other data.

Suppose that the next overt form encountered by the learner is *sàksofonísta.* This overt form has six possible interpretations. Hypothesis H(A1B1), when used to generate a stress pattern for this word, assigns the attested stress pattern. Thus, no learning need take place *for H(A1B1)* with this word. This is a case where the new datum is uninformative with respect to the hypothesis, so the learner does not bother constructing any new winner-loser pairs. The same is not true for hypothesis H(A3B4), which assigns (*saksò*)(*foní*)*sta.* The learner attempts to reconcile each of the six interpretations of *sàksofonísta* with H(A3B4). However, each of the six interpretations proves inconsistent with H(A3B4), a fact determined after 6 learning steps (one per interpretation). Thus, after only three pieces of data, and 29 learning steps, the learner has reduced the possibilities to a single hypothesis, H(A1B1).

Hypothesis H(A1B1) will remain unchanged until the learner encounters a longer form, like *rèwolùcjonísta.* H(A1B1) prefers the nonmatching (*rèwo*)*lucjo*(*nísta*), in part owing to the unresolved conflict between PARSE and AFL, so the learner must consider all 13 interpretations. Interpretation D1, (*rèwo*)(*lùcjo*)(*nísta*), is consistent with the earlier data, forming hypothesis H(A1B1D1) after just one additional winner-loser pair. The other 12 interpretations are all inconsistent, which can be determined with 16 learning steps of effort.

The learner now possesses a grammar for Polish, H(A1B1D1), having taken a total of 46 learning steps to find it. The learner never had to carry more than two simultaneous hypotheses from one data form to the next. This example is somewhat simplified. In the full system, 59 learning steps were used to learn the complete Polish pattern (see section 4); the maximum simultaneous hypothesis count was still two.

To summarize, there are three ways in which the IDL can cut down on the combinatorial explosion in combinations of interpretations:

- Eliminate interpretations that cannot possibly be optimal. This was illustrated by the elimination of candidate A2.
- Detect inconsistent combinations of interpretations. This occurred when the combination of hypothesis H(A1) and interpretation B4 was found to be inconsistent.

- Correctly parse the overt form with a hypothesis, thus avoiding the consideration of multiple interpretations. This occurred when hypothesis H(A1B1) encountered the overt form *sàksofonísta.*

In a sense, the IDL carries out a race between two forces, one trying to increase the number of hypotheses the learner must maintain, and the other trying to reduce the number of hypotheses maintained. The feasibility of the IDL depends upon the relative strength of the two forces. If the linguistic theory being employed is such that the interpretations assigned to different forms are strongly interrestrictive, then the IDL will be efficient, because when an error occurs, most of the interpretations of the overt form will be discarded because of inconsistency with data already observed. If the forms are not so strongly interrestrictive, the IDL may get swamped by an unrealistically large number of hypotheses to maintain. Advocates of the IDL are thus ''putting their money'' on linguistic theory, counting on it to be restrictive enough to overcome the exponential growth implied by the ambiguity of the overt forms. The relationship between linguistic theory and interrestrictiveness is discussed further in section 5.4.

4 The Computer Simulations

4.1 The Stress Data

The definitional details of the OT stress system are presented in appendix A. The system defines languages that assign stress patterns to words of arbitrary length. For the simulations, 62 words of each language were presented: words with length between two and five syllables using all possible ordered combinations of light and heavy syllables, a word of six light syllables, and a word of seven light syllables.

Given this base of 62 underlying forms, in this system there are 2,140 distinct sets of structural descriptions (languages), each realizable with different constraint rankings. However, some languages completely match in overt forms: there are cases where two different grammars, determined by two different constraint rankings, define languages with identical sets of overt forms, even though the structural descriptions for some of the overt forms differ. Because the learner is only given overt forms as data, these languages cannot be distinguished by the learner. To avoid uninformative duplication in the simulations, the entire set of possible languages was evaluated. For each collection of languages with duplicate sets of overt forms, all but one were removed, so that in the resulting set of languages, only one language remained for each possible set of overt forms. With the duplicates removed, a total of 1,527 languages remained for the learner to be tested on. The learner was separately tested on all 1,527 languages.[13]

The overt forms vary widely in their degree of ambiguity. A form like σ́σ is only two-ways ambiguous, permitting interpretations (σ́)σ and (σ́σ). The form σσ́σσ̀σσ̀σ is 21-ways ambiguous, permitting 21 different interpretations.

[13] It should be emphasized that the learning algorithm itself does not possess, and thus cannot exploit, any specific knowledge of the existence of such weakly equivalent grammars.

Now suppose for a moment that every form in a language is only two-ways ambiguous. Then the number of combinations of interpretations that could be realized is $2^{62} \cong 5 * 10^{18}$. If the number of learning steps the learner must take to learn a language via inconsistency detection scales at all like the number of combinations (as with the BAD algorithm of section 3.1), then this approach will be hopelessly intractable. Every overt form in every language has at least two possible interpretations, so 2^{62} is a lower bound on all languages in the system. For languages including forms that are more than two-ways ambiguous, the number of possible combinations will be much, much larger still. The number of total rankings of the constraints is tiny by comparison, $10! = 3{,}628{,}800$.[14] For the IDL to be an improvement over simple exhaustive search of all total rankings, it will have to completely escape the combinatorial explosion of combinations of interpretations.

4.2 Purpose and Design of the Simulations

The primary purpose of the simulations is to see, using a linguistically plausible OT system, whether the IDL can overcome the combinatorics of the system and efficiently learn correct grammars from sets of overt forms. The IDL is guaranteed to eventually find a correct grammar, so the issue is the amount of computational effort required by the IDL to learn. The measures of computational effort that were used are described in section 4.2.2.

A secondary purpose of the simulations is to examine the effect of the presentation order of the data; the significance of order is discussed in section 4.2.1. For each language, once an ordering of the overt forms for the language was established, the forms were presented to the learner in that order. Once the end of the list of forms was reached, presentation continued starting over at the beginning of the list of forms (using the same order). This continued until the learner made a pass through the forms without encountering an error (a mismatch between the overt form and the optimal candidate for one of the learner's hypotheses) on any of the overt forms.

4.2.1 Data Order The data for each language consists of a set of overt forms. The overt forms within a language can vary in degree of ambiguity, and it is intuitively plausible that using forms with lower ambiguity early could benefit learning. With metrical stress, degree of ambiguity correlates to some extent with the length (in syllables) of the form: more syllables means more ways of grouping syllables into feet if there are secondary stresses present. Thus, the length of a form is an overtly available indicator that a learner could use to select low-ambiguity forms earlier.

Two data orders were used. In the ''random'' case, the overt form order was fully randomized prior to presenting them to the learner. In the ''block'' case, the overt forms were grouped into blocks by length: the forms of lengths of two and three syllables were the first block, the forms of four syllables were the second block, the forms of five syllables were the third block, the six-

[14] For those interested in combinatorial comparison with parametric theories, the number of total rankings of 10 constraints, $10! = 3{,}628{,}800$, is roughly comparable to the number of distinct parameter-setting combinations for a system with 22 binary parameters, $2^{22} = 4{,}194{,}304$.

syllable form was the fourth block, and the seven-syllable form was the fifth block. The order of the forms within each block was randomized, and the overall order was established by ordering the blocks (first block first, up to the fifth block last). This is a simple approximation to a strategy whereby the learner selects short forms early.

4.2.2 Measuring Learning Efficiency Two quantities were measured during the learning of each language. The measure of cumulative computational effort by the learner is the number of learning steps as described in section 3.2: the number of times a winner-loser pair is added to a hypothesis (and RCD applied). The number of learning steps includes not just the number of winner-loser pairs in the hypothesis that survives to the end as the learned grammar, but all such operations performed on other hypotheses prior to their being discarded as inconsistent. Detecting inconsistency in a hypothesis necessarily requires at least one winner-loser pair, so if the learner must consider a large number of hypotheses during learning, there will be a correspondingly large number of learning steps added to the total for the language.

The other quantity used to evaluate learning performance is the maximum number of simultaneous hypotheses maintained by the learner from one overt form to the next during learning. After processing an overt form, the learner possesses some number of hypotheses in memory. That number was examined after the processing of each form during the learning of a language, and the maximum value over the entire learning process for that language was determined.

4.3 The Simulation Results

For each language, in each data order, the total number of learning steps and the maximum number of simultaneous hypotheses were recorded.[15] The results are summarized in table 4. The algorithm proves to be quite fast on this OT system. For each data order, a majority of the languages required fewer total learning steps than the square of the number of constraints ($10^2 = 100$). The IDL's performance compares quite favorably to the scale of the core hypothesis space ($10! = 3{,}628{,}800$ total rankings). It clearly has escaped the combinatorial growth in number of combinations of interpretations.

Data order affected the number of learning steps, with the block data order causing improvement. The number of languages ($N = 1{,}527$) leaves little room for doubt of the statistical significance of the differences. A statistical comparison was performed to verify this.[16] Not only is the

[15] A fully detailed description of the algorithm used in these simulations would require addressing some computational issues outside the scope of this article. A discussion of these details, and their role in the results presented here, can be found in Tesar 2000b. To summarize briefly, the error detection component included a conflict detection criterion called Conflicts Tie (or CTie), used to deal with strata containing multiple constraints when parsing for error detection. Also, prior winner checking was employed: whenever a new winner-loser pair was added to a list, error detection was repeated with the winners of the existing winner-loser pairs of the list, to ensure that they were still optimal. Any additional winner-loser pairs resulting from prior winner checking were of course included in the measures of computational effort.

[16] Because the same set of languages was used with each data order, the difference in learning steps between the compared orders was computed for each language, and then the mean of the differences was computed, thus eliminating crosslanguage variance (which is considerable). With a null hypothesis that the mean of the differences in learning steps is zero, the Z-value is 27.6 and the p-value is less than 10^{-167}.

Table 4
Learning simulation results: $N = 1{,}527$

	Total learning steps			Max # simultaneous hypotheses		
Data order	Mean	Median	Range	Mean	Median	Range
Random	62.6	57	2..328	2.5	2	1..8
Block	39.8	37	4..141	2.2	2	1..6

difference in the number of learning steps statistically significant, but the block order median number of learning steps is only about 65% of the median number of steps for the random order. In fact, in the block order there were only 14 languages (out of 1,527) on which the learner took more than 100 learning steps; in the random order, there were 192 languages on which the learner took more than 100 learning steps. Biasing the learner to select shorter forms early has a noticeable effect.

For most of the languages, the learner never maintained more than two simultaneous hypotheses. For the block order, the largest number of simultaneous hypotheses ever maintained was six, and there was only one language that ever reached six (and only eight others that ever reached five). Further, these outliers appear to be at least partially a function of specific data orders. For the language that resulted in six simultaneous hypotheses in the main simulations, when a different randomization within the blocks was run, the number of simultaneous hypotheses never exceeded two. It is certainly possible that, for some of the languages reported in the simulations as having a maximum of two simultaneous hypotheses, there exist fluke data orders that could cause a larger number of simultaneous hypotheses. The simulation results suggest that (a) such events would be expected to be quite rare, and (b) even the least generous data orders will not result in excessively large numbers of simultaneous hypotheses.

The very small number of simultaneous hypotheses maintained by the learner reflects two things. First, for many of the overt forms with large numbers of possible interpretations, quite a number of those interpretations are not possibly optimal. The IDL detected those forms quickly and discarded them. Second, the overt forms are heavily interrestrictive, and the learner is able to capitalize quickly upon the restrictions between forms imposed by linguistic theory. This is discussed further in section 5.4.2.

5 Discussion

5.1 The Role of Optimality Theory

Nothing about the IDL is specific to metrical stress. The approach could certainly be applied to other linguistic phenomena. In principle, the IDL approach does not even require that linguistic knowledge be analyzed in optimality-theoretic terms. What the IDL requires are

- an explicit space of possible grammars,
- an explicit set of possible interpretations for any overt form,

- a method for determining when a set of interpretations is inconsistent, and
- a method for finding a grammar that licenses a consistent set of interpretations.

OT makes the space of possible grammars quite explicit: it is the set of grammars that arise from total rankings of the constraints. This explicitness is not notably different from that of other typologically rigorous approaches to linguistic theory, such as the principles-and-parameters framework.

However, some of the other requirements make OT particularly well suited to the IDL approach to learning. OT's reliance on strict domination among constraints is the key to the guaranteed efficiency of the MRCD algorithm (Tesar 2000a). MRCD works both for determining whether a set of interpretations is inconsistent and for finding a grammar (a constraint hierarchy) licensing a consistent set of interpretations. Somewhat more loosely, OT forces on the analyst a precise specification of the possible structural descriptions of various underlying forms and overt forms, even in the analysis of a single form, because OT is defined in terms of optimization over a space of candidate structural descriptions.

5.2 The Value of Candidate Interpretations

It can be instructive to ask why the learner needs to bother with considering different interpretations of overt forms. Why not just test the compatibility of overt forms with hypotheses directly? After all, that is how error-driven learning has been more traditionally employed: the learner has a hypothesis grammar and attempts to parse an overt form; if parsing fails, the grammar is wrong and should be changed. But that immediately raises the question, *how* should the grammar be changed? Merely observing that the current grammar is incorrect leaves one with the (exponentially large) set of alternative grammars to choose from in attempting to find one that will parse the overt form. Focusing on different interpretations of the overt form allows one to initially shrink from a decision space of the *possible grammars* to a (much, much smaller) decision space of the *plausible interpretations* of an overt form. A candidate interpretation is quite specific in what it requires of a grammar, enough that the learner (via MRCD) can limit the set of possible grammars consistent with the interpretation in a few steps. In the metrical stress system examined here, the space of possible interpretations for an overt form is small enough to be plausibly exhaustively searched. Once interpretation space has been searched, projecting the result back into grammar space is easily done with MRCD. This way, the overt form causing an error can provide much more information than just the fact that the learner's current grammar is incorrect; it can also give strong indications of what the correct grammar must be like, and in a way that can be used in a computationally efficient manner.

The computational evaluation of the different candidate interpretations can take place independently; the computation for evaluating one candidate interpretation does not depend upon the outcome of the evaluation of another. This opens the way for parallel computation: the different candidate interpretations can be evaluated simultaneously. The same applies to the evaluation of a form by multiple hypotheses: the different hypotheses independently interpret the overt form, and those computations could be performed in parallel. As long as the number of hypotheses

simultaneously maintained by the learner remains reasonably small, there will not be excessive demands on computational resources.

5.3 The Structure of Hypotheses

Any serious approach to learning must make commitments about the nature of the intermediate information states, or hypotheses, that the learner will have during the course of learning. The structure of a learner's hypotheses will be closely connected to how the learner acts and can play a big role in the effectiveness of a learner.

In the traditional learnability-in-the-limit framework (Gold 1967), the learner always has a single fully specified grammar as a hypothesis. This view of learning hypotheses is preserved faithfully in the TLA, which at any time has a setting for each parameter (see section 1.2.3). This straightforwardly allows error-driven learning, because the learner always has a full specification of parameter settings. However, the learner has no way of knowing which of its parameter settings are supported by previously observed empirical evidence and which are just arbitrary initial selections. To avoid such confusion, the cue-based learning approach prefers to ''set no parameter before its time.'' At intermediate stages, the learner will have assigned settings to some parameters, while others are unset. This makes error-driven learning difficult, and cue-based learners abandon it. The price for this is other learning-specific processing mechanisms, which take the place of the regular parser in analyzing overt forms for the learner.

In the OT-based IDL, a hypothesis is represented by a set of winner-loser pairs. The winner-loser pairs are a specific representation of what the hypothesis is committed to. This avoids the overcommitment problem of the TLA; the learner keeps track of what has been learned from prior data. The stratified hierarchy resulting from applying RCD to the winner-loser pairs need not be a fully formed grammar (it need not be equivalent to a total ranking), but nevertheless puts the information of the winner-loser pairs into a form that can be used for parsing and error-driven learning. Further, the learner is able to parse with stratified hierarchies using the same optimizing parse mechanism that is used to parse with fully formed grammars. The use of stratified hierarchies allows the learner to parse overt forms during learning, without having to make premature full grammar commitments, and without needing separate, learning-specific parse mechanisms.

Nothing in life is absolutely free. For the IDL, the price of basing hypotheses upon consistent sets of interpretations is that sometimes more than one hypothesis must be simultaneously maintained in memory. Uncertainty resulting from unresolved ambiguity of interpretation of overt forms is represented not by introducing further complexity into an individual hypothesis, but by creating different hypotheses for the different interpretations. This allows the learner to benefit from the grammarlike form of the individual hypotheses (e.g., they can be used for parsing) without having to prematurely commit to a particular interpretation (as is done in RIP/CD). It also allows the learner to proceed with learning in the face of (temporary) unresolved ambiguity, rather than waiting indefinitely for fully unambiguous forms (as is done in the STL). The risk of this approach is the possibility that the learner will maintain a large number of hypotheses

simultaneously. The simulation results suggest that this risk may be quite manageable for spaces of human linguistic grammars.

5.4 Relevant Properties of Grammars

5.4.1 The Degree of Ambiguity of Overt Forms One property that can affect the success of the IDL is the degree of ambiguity of overt forms. There are really two properties here. One is the number of candidate interpretations of an overt form. The other is the number of *possibly optimal* interpretations of an overt form.

Conceivably, a system could assign a huge number of candidate interpretations to an overt form, with most of them being suboptimal under all grammars.[17] For a linguist, this is not problematic; the grammar is doing its job. But it could be problematic for the learner if the form causes an error, and the learner has to enumerate and evaluate all of the possible interpretations in order to determine which are feasible. For such systems, the problem might be dealt with in part by avoiding processing high-ambiguity forms early on. This was done heuristically in the stress simulations when the learner processed short forms before long ones, so that much of the learning was done before longer forms were encountered, and fewer errors occurred on overt forms with larger numbers of interpretations. One could imagine a learner doing this more directly, computing the number of interpretations of an overt form and then discarding them all if the number is above some threshold. Of course, in a system where forms with higher ambiguity are the crucial ones in determining the grammar, the learner will not be able to avoid processing them altogether. If important forms have too many interpretations, the IDL would not be able to succeed without some more efficient means of identifying which interpretations are possibly optimal. This is a possible area of future research, discussed in section 5.5.

The other threatening property mentioned above is the inclusion of overt forms with large numbers of possibly optimal interpretations. The theoretical worst case would be a system in which there existed an overt form that was given a different interpretation in every language (the number of possibly optimal interpretations is equal to the number of languages). This implies that the overt form is present in every language, making it completely uninformative for learning. If the learner happened to process that overt form first, it would create a separate hypothesis for every possible grammar, making the tractability no better than exhaustive search. Clearly, such a case is quite implausible, but it demonstrates the limits of what can be said about the IDL under fully unconstrained circumstances. Note that this discussion indicates the theoretical bound on the number of simultaneous hypotheses: one for each language in the system.

The stress system investigated in the simulations had overt forms with a reasonably bounded degree of ambiguity; the most highly ambiguous overt forms had around 21 interpretations (not

[17] If one abandons linguistic plausibility entirely and allows arbitrary relationships between an overt form and its interpretations, then one can make the learning task equivalent to intractable optimization problems. Eisner (2000) gives an example of such a case, where the interpretations are paths through a graph, and determining whether there is a ranking making one of the interpretations optimal is equivalent to determining whether the graph contains a Hamiltonian path (an NP-complete problem).

21,000). Further, the forms with high degrees of raw ambiguity had far fewer interpretations that were actual possible optima, so if an error occurred on the form, most of the interpretations did not lead to viable hypotheses. Finally, the learner was able to improve efficiency by processing shorter forms first, which generally meant processing less ambiguous forms early. In this data-ordering condition, the learner encountered fewer errors on forms of high ambiguity and so was evaluating the interpretations of those forms less frequently. It also was armed with more data from the shorter forms, so that when an error did occur on a form with a larger number of interpretations, there was greater opportunity to detect inconsistency between the possibly optimal interpretations of the new overt form and the previous data.

5.4.2 The Interrestrictiveness of Overt Forms Two overt forms are interrestrictive if at least one combination of interpretations of each of the forms is inconsistent. If there are no inconsistent combinations of interpretations of a pair (in general, a set) of overt forms, then they are not interrestrictive, and if each overt form has several interpretations, there will be a significant number of combinations. On the other hand, if two overt forms have a significant number of inconsistent combinations of interpretations, they are highly interrestrictive.

When an overt form is ambiguous, it has different interpretations that make different requirements of the principles defining the possible grammars; in OT, the different interpretations impose conflicting requirements on the constraint ranking. Two overt forms are likely to be interrestrictive when the grammatical principles at issue in the interpretations of each form overlap. Inconsistency detection requires that contradictory ranking conditions be imposed on the same constraints by different forms, which can only happen when different forms impose conditions on the same constraints. A system could be expected to be highly interrestrictive if, for most small sets of overt forms, the interpretations of those overt forms imposed requirements on some of the same principles. In OT, this means that the interpretations of the overt forms put ranking conditions on the same constraints.

This kind of overlap is certainly the case in the metrical stress system studied in this article. For example, consider (10) again, giving the winner-loser pairs for the pair of interpretations *spo*(*kójny*) and (*prò*)(*pagán*)*da.* The two winner-loser pairs for *spo*(*kójny*) jointly refer to every single constraint in the tableau (as indicated by the fact that every constraint has an L or W mark in one of the winner-loser pairs); the same is true for the winner-loser pair of (*prò*)(*pagán*)*da.* There is total overlap in the constraints being referred to.

The two interpretations *spo*(*kójny*) and (*spokój*)*ny* make conflicting demands on several constraints. In particular, they conflict on foot form, the first requiring trochaic feet and the second requiring iambic feet. The first also requires that right-alignment dominate left-alignment and nonfinality, while the second requires that right-alignment be dominated by either left-alignment or nonfinality. The interpretations of *pròpagánda* make demands of the same sort and thus involve the same constraints. A trochee-demanding interpretation like (*pròpa*)(*gánda*) will not be consistent with an iamb-demanding interpretation like (*spokój*)*ny.* The interrestrictiveness results from the fact that the same constraints are implicated by the interpretations of each overt form; they are all involved in determining the placement of stress.

A system could fail to adequately interrestrict if it had several forms of structural ambiguity, each involving unrelated linguistic principles and overt data patterns. If a language had three domains of overt ambiguity, A, B, and C, and the three involved relationships over disjoint principles (e.g., they involved ranking relations among three disjoint sets of constraints), the IDL could encounter difficulty in two ways. One way would have the learner process a form that is ambiguous with respect to domain A, and then process a form that is ambiguous with respect to domain B before processing a form that restricts the domain A ambiguity. If that happened, the learner would end up with a number of hypotheses equal to the degree of ambiguity in domain A multiplied by the degree of ambiguity in domain B. This multiplication could continue further if the learner then processed a form that was ambiguous in domain C, but totally uninformative in domains A and B. Again, the processing order of forms would be crucial; the problem would result because the learner processed this form before other forms that were relevant and informative in domains A and B.

The other and perhaps more plausible way the IDL could encounter difficulty would be for the language to have a high percentage of forms that simultaneously exhibited ambiguities from each of domains A, B, and C. In this case, the overt forms would likely be interrestrictive, but each individual overt form would have a high degree of ambiguity; the plausible analyses of a single form would include various combinations of the different structural possibilities for each of the domains A, B, and C.

Both forms of the hazard for the IDL just described depend on the domains of overt ambiguity (A, B, and C) having no overlap in the linguistic principles that govern them. In OT terms, the constraints relevant to each domain must not interact with the constraints relevant to the other domains. Thus, it is beneficial to inconsistency detection learning to have linguistic principles that interact widely in determining overt forms. Fortunately, this property is independently desirable for linguistic theory in general. In essence, to avoid causing problems for inconsistency detection learning, it is desirable to avoid blatantly descriptive theories, at least in those areas where the theory predicts structural ambiguity.

It may be at least plausible that some empirical domains do not interact (e.g., segmental phonology and basic word order syntax). For such cases, one could imagine a learner being endowed with such knowledge, and separately learning each of the domains. This would avoid the multiplicative combination of the representational possibilities across the domains. However, it would not deal with the more sinister possibility of two empirical domains that were capable of interacting, but only rarely did. The capacity for interaction would require the learner to learn the domains in tandem.

5.5 Directions for Current and Future Research

In the simulations presented in section 4, the learner exhaustively checked all possible interpretations of an overt form that caused an error on some hypothesis. For that OT system, the number of interpretations of the overt forms is reasonable enough to permit this. When more complex linguistic systems are attempted, it is likely that the learner will not realistically be able to sepa-

rately generate and examine all possible interpretations of overt forms and will instead have to use some means for selecting the most plausible interpretations to try.

One basis for rejecting a candidate interpretation is that the interpretation is not possibly optimal. As explained above, MRCD is guaranteed, when attempting to find a ranking supporting an interpretation, to detect whether that interpretation is not possibly optimal. Sometimes, MRCD is not even necessary: if the interpretation is harmonically bounded by a single competitor, and that competitor happens to be selected as the losing candidate for the formation of a winner-loser pair, the learner can tell just from the single winner-loser pair that the interpretation is doomed (no possible ranking can cause a description to beat a competitor that harmonically bounds it). In the simulations reported here, such cases were still counted as applications of MRCD, so that the reported counts more fully reflected the extent of the generation and evaluation of candidate interpretations of the overt forms; but in practice, simple tests of harmonic bounding might significantly reduce the number of interpretations actually subjected to MRCD.

If linguistic systems with larger numbers of interpretations for some overt forms are considered, there may be other efficient means of identifying and eliminating interpretations that are not possibly optimal. See Samek-Lodovici and Prince 1999 for relevant work on the structure of candidate spaces, in particular the relationship of possibly optimal candidates to others.

One major issue beyond the scope of the present work is the learning of underlying forms that surface differently in different contexts. Simple exhaustive search is clearly out of the question. However, there may be general strategies available that allow the learner to further restrict active consideration to just a few possible underlying forms for each morpheme. The inclusion of underlying forms as another dimension of ambiguity in structural descriptions, and the consequences for learning, are a topic of current research.

Related to underlying forms is the question of grammatical restrictiveness. MRCD is guaranteed to give the learner *a* constraint hierarchy consistent with all of the data presented to it. However, it is possible for more than one (fully formed) grammar to be consistent with a set of data, and an adequate learner must be capable of selecting the most restrictive of those grammars. In the stress system of this article, every underlying form maps to a distinct overt form (distinct from that of any other underlying form), identifiable by the weights of the syllables in the form, so the concern about more/less restrictive grammars does not arise. Neutralization does not occur in this system. But it could if the system were expanded to include lexical stress systems, where different words with the same syllable weight pattern can have somewhat different stress patterns, or to include the possibility of faithfulness-violating changes to syllable weight. Learning algorithms have been proposed for OT systems that attempt to bias the learner toward the most restrictive grammar; for details, see Hayes, to appear, Prince and Tesar, to appear, Smolensky 1996. The incorporation of this kind of learning into an IDL is a topic of current research (see Tesar 2002).

Other challenging topics not directly addressed here include robustness of learning in the face of input with ungrammatical utterances, and the modeling and acquisition of linguistic variation. These pose challenges for any approach to learning. For the IDL in particular, input including ungrammatical utterances could cause the algorithm to eliminate all hypotheses, concluding that

no grammar is consistent with all of the data. An approach to this would be to have the learner work toward finding a grammar consistent with the largest possible subset of the input data (another dimension of hypothesis competition). One can imagine several ways to attempt this; formalizing and evaluating them will require significant future research. For a promising approach to dealing with ungrammatical input and variation in the problem of learning OT constraint rankings from full descriptions, see recent work on the Gradual Learning Algorithm (Boersma 1998, Boersma and Hayes 2001).

6 Conclusions

One way or another, a human language learner must combine information from multiple overt forms in order to resolve structural ambiguity and infer the correct grammar. The Inconsistency Detection Learner does this in a particularly direct way, by attempting to construct hypothesis grammars for combinations of interpretations of the overt forms and discarding those combinations that are shown to be inconsistent. The simulation results presented in this article show that, for a linguistically motivated OT system for metrical stress, the IDL algorithm overcomes structural ambiguity without requiring the learner to evaluate more than a reasonable number of sets of structural descriptions. Further, the IDL is not defined in a way that is specific to metrical stress, so it is reasonable to believe that inconsistency detection learning could be effectively applied to language learning in a variety of linguistic domains and that this approach may play an important part in the acquisition mechanisms of human learners.

Appendix A: The Metrical Stress System Used in the Simulations

Candidate structural descriptions are assignments of foot structure and stress levels to a string of syllables, constituting a prosodic word. Feet are nonoverlapping and consist of either one or two contiguous syllables. Each foot has precisely one head syllable, which bears stress. A prosodic word must have a head foot, whose head syllable bears main stress. Syllables may be unfooted, and the heads of feet other than the head foot bear secondary stress. Syllables are distinguished only by their weight: as light or heavy.

Candidate structural descriptions of an underlying form are obliged to preserve the number, order, and weight of the syllables in the output. Underlying forms have no specification of stress levels or accents. Thus, the system has no violable faithfulness component, with the consequence that the system cannot provide an analysis of lexical stress systems. This is an intentional idealization, abstracting away from the complex issue of learning underlying forms so that structural ambiguity can be focused upon.

This system idealizes syllable weight as a single, fixed notion. This is done to avoid dragging in details of segmental syllable structure that are relevant to crosslinguistic distinctions of syllable weight. This idealization is *not* based on an assumption that the learner will have somehow preanalyzed the data to determine what types of syllables (if any) count as heavy. The notion of overt form is not fluid or subject to change during the course of learning. A proper treatment of crosslinguistic variation in syllable weight would require violable faithfulness to account for distributional restrictions on different syllable types; again, this is beyond the scope of the present

Table 5
The constraints of the stress system

Constraint	Abbrev.	Description
PARSE	PARSE	a syllable must be footed
MAIN-RIGHT	MR	align the head foot with the right edge of the word
MAIN-LEFT	ML	align the head foot with the left edge of the word
ALL-FEET-RIGHT	AFR	align each foot with the right edge of the word
ALL-FEET-LEFT	AFL	align each foot with the left edge of the word
FOOT-NONFINAL	FNF	a head (stressed) syllable must not be final in its foot
IAMBIC	IAMB	the final syllable of a foot must be the head (stressed)
FOOT BINARITY	FTBIN	a foot cannot consist of a single light syllable
NONFINAL	NF	the final syllable of a word must not be footed
WEIGHT-TO-STRESS	WSP	a heavy syllable must be stressed

work. Such an analysis would use a more detailed representation of the overt form, including segmental structure and length, and would include constraints that relate those representational elements to stress.

The stress system has 10 constraints, shown in table 5. To very roughly characterize it in more traditional terms, the system allows alignment of main stress to either edge, the possibility of iterative footing, directional alignment of iterating feet, final syllable extrametricality, trochaic and iambic foot types, and the possibility of allowing degenerate feet. Quantity sensitivity can have varying degrees of effect (not just sensitive vs. insensitive).

This system for metrical stress incorporates theoretical ideas from a number of sources, including Hammond 1990, Hayes 1995, Kager 1994, McCarthy and Prince 1993, Prince 1990, Prince and Smolensky 1993. It is also less directly but no less significantly influenced by other research on metrical stress: see, among many other works, Halle and Vergnaud 1987, Hayes 1980, Liberman and Prince 1977, Prince 1983.

Appendix B: Parsing for Error Detection and Loser Selection

The use of parsing is illustrated here with a particular stage of the Polish example of section 3.2. In tableau (9) of that example, the learner uses hypothesis H(A1) (see (6)) to analyze the new overt form *pròpagánda.* Several candidate structural descriptions for the underlying form */propaganda/* are shown here in (11). The constraints are ordered as they are in hypothesis H(A1).

(11) Parsing */propaganda/* with hypothesis H(A1)

/propaganda/	FTBIN	MR	AFR	FNF	PARSE	ML	IAMB	AFL	NF
(a) propa(gánda)					**	**	*	**	*
(b) (pròpa)(gánda)			**			**	**	**	*
(c) (propá)(gandà)		**	**	**				**	*

The form labeled (a) is the optimal candidate for the hierarchy of H(A1), as it is the only candidate that does not violate any constraints of the top stratum. An error is detected here, because the stressing of this candidate does not match the observed overt form *pròpagánda.* This triggers the generation of possible interpretations of the overt form, and the pursuit of each with MRCD. When an interpretation is pursued, a winner-loser pair is constructed, using that interpretation as the winner and form (a) of (11) as the loser. Parsing constructs the loser, and that loser's existence constitutes an error.

An important virtue is the ability to parse using hierarchies that are not total rankings (Tesar 1995). This OT-specific capacity helps the learner avoid arbitrary commitments to grammar specification just to permit error-driven learning. A standard way to extend harmonic evaluation is to pool the marks assigned by all constraints in a stratum and then compare candidates on the total number of violations in that stratum. If one candidate has more total violations on the stratum, it loses to the other. If two candidates have an equal number of violations on a stratum, the decision passes down to the next stratum. This is illustrated schematically in (12). Candidate D5 is eliminated by constraint C1. Candidates D3 and D4 each have two violations on the second stratum (C2 and C3), while D1 and D2 only have one violation each. Only D1 and D2 survive at that point, and the decision passes to the next stratum, containing C4, which decides in favor of D1.

(12) Parsing with constraint ties: constraints C2 and C3 occupy the same stratum

	C1	C2	C3	C4	C5
D1		*			*
D2			*	*	
D3		**			
D4		*	*		
D5	*				

Another method for comparing candidates is to declare pairs of candidates that have conflicting violations a full tie, and not consider lower strata. This method is known as Conflicts Tie (Tesar 2000b). Applying this method to (12), candidates D1 and D2 conflict on the second stratum (C2 prefers D2, C3 prefers D1), so they would tie overall, without giving C4 or C5 a chance to break the tie. With either mark pooling or Conflicts Tie, if multiple candidates tie for optimality on a hierarchy, any one of those that does not match the attested overt form will function adequately as a loser.

References

Bertolo, Stefano, Kevin Broihier, Edward Gibson, and Kenneth Wexler. 1997a. Characterizing learnability conditions for cue-based learners in parametric language systems. In *Proceedings of the Fifth Meeting on the Mathematics of Language,* ed. by Tilman Becker and Hans-Ulrich Krieger. http://www.dfki.de/events/mol/.

Bertolo, Stefano, Kevin Broihier, Edward Gibson, and Kenneth Wexler. 1997b. Cue-based learners in parametric language systems: Application of general results to a recently proposed learning algorithm based on unambiguous ''superparsing.'' In *Proceedings of the 19th Annual Conference of the Cognitive Science Society,* ed. by Michael G. Shafto and Pat Langley, 49–54. Mahwah, N.J.: Lawrence Erlbaum.

Boersma, Paul. 1998. *Functional phonology.* The Hague: Holland Academic Graphics.

Boersma, Paul, and Bruce Hayes. 2001. Empirical tests of the Gradual Learning Algorithm. *Linguistic Inquiry* 32:45–86.

Chomsky, Noam. 1995. *The Minimalist Program.* Cambridge, Mass.: MIT Press.

Clark, Robin. 1992. The selection of syntactic knowledge. *Language Acquisition* 2:83–149.

Clark, Robin, and Ian Roberts. 1993. A computational model of language learnability and language change. *Linguistic Inquiry* 24:299–345.

Daelemans, Walter, Steven Gillis, and Gert Durieux. 1994. The acquisition of stress: A data-oriented approach. *Computational Linguistics* 20:421–451.

Dresher, B. Elan. 1999. Charting the learning path: Cues to parameter setting. *Linguistic Inquiry* 30:27–67.

Dresher, B. Elan, and Jonathan Kaye. 1990. A computational learning model for metrical phonology. *Cognition* 34:137–195.

Echeverria, Max S., and Heles Contreras. 1965. Araucanian phonemics. *International Journal of American Linguistics* 31:132–135.

Eisner, Jason. 2000. Easy and hard constraint ranking in Optimality Theory: Algorithms and complexity. In *Finite-state phonology: Proceedings of the 5th Workshop of the ACL Special Interest Group in Computational Phonology,* ed. by Jason Eisner, Lauri Karttunen, and Alain Theriault, 22–33. Association for Computational Linguistics.

Fodor, Janet Dean. 1998a. Parsing to learn. *Journal of Psycholinguistic Research* 27:339–374.

Fodor, Janet Dean. 1998b. Unambiguous triggers. *Linguistic Inquiry* 29:1–36.

Gibson, Edward, and Kenneth Wexler. 1994. Triggers. *Linguistic Inquiry* 25:407–454.

Gold, E. Mark. 1967. Language identification in the limit. *Information and Control* 10:447–474.

Gupta, Prahlad, and David Touretzky. 1994. Connectionist models and linguistic theory: Investigations of stress systems in language. *Cognitive Science* 18:1–50.

Halle, Morris, and Jean-Roger Vergnaud. 1987. *An essay on stress.* Cambridge, Mass.: MIT Press.

Hammond, Michael. 1990. Metrical theory and learnability. Ms., University of Arizona, Tucson.

Hayes, Bruce. 1980. A metrical theory of stress rules. Doctoral dissertation, MIT, Cambridge, Mass.

Hayes, Bruce. 1995. *Metrical stress theory: Principles and case studies.* Chicago: University of Chicago Press.

Hayes, Bruce. To appear. Phonological acquisition in Optimality Theory: The early stages. In *Fixing priorities: Constraints in phonological acquisition,* ed. by René Kager, Joe Pater, and Wim Zonneveld. Cambridge: Cambridge University Press.

Joanisse, Marc, and Suzanne Curtin. 1999. Dutch stress acquisition: OT and connectionist approaches. Ms., University of Southern California, Los Angeles.

Kager, René. 1994. Ternary rhythm in alignment theory. Ms., Utrecht University.

Liberman, Mark, and Alan Prince. 1977. On stress and linguistic rhythm. *Linguistic Inquiry* 8:249–336.

McCarthy, John, and Alan Prince. 1993. Generalized alignment. In *Yearbook of morphology,* ed. by Geert Booij and Jaap van Marle, 79–154. Dordrecht: Kluwer.

Niyogi, Partha, and Robert C. Berwick. 1996. A language learning model for finite parameter spaces. In *Computational approaches to language acquisition,* ed. by Michael R. Brent, 161–193. Cambridge, Mass.: MIT Press.

Osborn, Henry. 1966. Warao I: Phonology and morphophonemics. *International Journal of American Linguistics* 32:108–123.

Pinker, Steven. 1987. The bootstrapping problem in language acquisition. In *Mechanisms of language acquisition,* ed. by Brian MacWhinney, 399–441. Hillsdale, N.J.: Lawrence Erlbaum.

Prince, Alan. 1983. Relating to the grid. *Linguistic Inquiry* 14:19–100.

Prince, Alan. 1990. Quantitative consequences of rhythmic organization. In *CLS 26.* Vol. 2, *Papers from the Parasession on the Syllable in Phonetics and Phonology,* ed. by Karen Deaton, Manuela Noske, and Michael Ziolkowski, 355–398. Chicago: University of Chicago, Chicago Linguistic Society.

Prince, Alan. 2000. Comparative tableaux. Ms., Rutgers University, New Brunswick, N.J. Rutgers Optimality Archive ROA-376. http://roa.rutgers.edu.

Prince, Alan, and Paul Smolensky. 1993. Optimality Theory: Constraint interaction in generative grammar. Ms., Rutgers University, New Brunswick, N.J., and the University of Colorado, Boulder. Rutgers Optimality Archive ROA-537. http://roa.rutgers.edu.

Prince, Alan, and Bruce Tesar. To appear. Learning phonotactic distributions. In *Fixing priorities: Constraints in phonological acquisition,* ed. by René Kager, Joe Pater, and Wim Zonneveld. Cambridge: Cambridge University Press.

Pulleyblank, Douglas, and William J. Turkel. 1998. The logical problem of language acquisition in Optimality Theory. In *Is the best good enough? Optimality and competition in syntax,* ed. by Pilar Barbosa, Danny Fox, Paul Hagstrom, Martha McGinnis, and David Pesetsky, 399–420. Cambridge, Mass.: MIT Press.

Rubach, Jerzy, and Geert E. Booij. 1985. A grid theory of stress in Polish. *Lingua* 66:281–319.

Sakas, William, and Janet Dean Fodor. 2001. The Structural Triggers Learner. In *Language acquisition and learnability,* ed. by Stefano Bertolo, 172–233. Cambridge: Cambridge University Press.

Samek-Lodovici, Vieri, and Alan Prince. 1999. Optima. Technical report RuCCS-TR-57, Center for Cognitive Science, Rutgers University, New Brunswick, N.J. Rutgers Optimality Archive ROA-363. http://roa.rutgers.edu.

Smolensky, Paul. 1996. The initial state and ''richness of the base'' in Optimality Theory. Technical report JHU-CogSci-96-4, The Johns Hopkins University, Baltimore, Md. Rutgers Optimality Archive ROA-154. http://roa.rutgers.edu.

Tesar, Bruce. 1995. Computational Optimality Theory. Doctoral dissertation, University of Colorado, Boulder.

Tesar, Bruce. 1997. Multi-Recursive Constraint Demotion. Ms., Rutgers University, New Brunswick, N.J. Rutgers Optimality Archive ROA-197. http://roa.rutgers.edu.

Tesar, Bruce. 1998a. Error-driven learning in Optimality Theory via the efficient computation of optimal forms. In *Is the best good enough? Optimality and competition in syntax,* ed. by Pilar Barbosa, Danny Fox, Paul Hagstrom, Martha McGinnis, and David Pesetsky, 421–435. Cambridge, Mass.: MIT Press.

Tesar, Bruce. 1998b. An iterative strategy for language learning. *Lingua* 104:131–145.

Tesar, Bruce. 2000a. On the roles of optimality and strict domination in language learning. In *Optimality Theory: Phonology, syntax, and acquisition,* ed. by Joost Dekkers, Frank van der Leeuw, and Jeroen van de Weijer, 592–620. Oxford: Oxford University Press.

Tesar, Bruce. 2000b. Using inconsistency detection to overcome structural ambiguity in language learning. Technical report RuCCS-TR-58, Center for Cognitive Science, Rutgers University, New Brunswick, N.J. Rutgers Optimality Archive ROA-426. http://roa.rutgers.edu.

Tesar, Bruce. 2002. Enforcing grammatical restrictiveness can help resolve structural ambiguity. In *Proceedings of the Twenty-first West Coast Conference on Formal Linguistics,* ed. by Line Mikkelsen and Christopher Potts, 443–456. Somerville, Mass.: Cascadilla Press. Rutgers Optimality Archive ROA-618. http://roa.rutgers.edu.

Tesar, Bruce, and Paul Smolensky. 1994. The learnability of Optimality Theory. In *Proceedings of the Thirteenth West Coast Conference on Formal Linguistics,* ed. by Raul Aranovich, William Byrne, Susanne Preuss, and Martha Senturia, 122–137. Stanford, Calif.: CSLI Publications.

Tesar, Bruce, and Paul Smolensky. 1998. Learnability in Optimality Theory. *Linguistic Inquiry* 29:229–268.
Tesar, Bruce, and Paul Smolensky. 2000. *Learnability in Optimality Theory.* Cambridge, Mass.: MIT Press.
Wexler, Kenneth, and Peter Culicover. 1980. *Formal principles of language acquisition.* Cambridge, Mass.: MIT Press.

Department of Linguistics
18 Seminary Place
Rutgers University
New Brunswick, New Jersey 08901-1184

tesar@ruccs.rutgers.edu