Case Studies in Language Learnability

by

Kevin J. Broihier
B.A., Psychology
University of Chicago, 1991

Submitted to the Department of Brain and Cognitive Sciences in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy in Brain and Cognitive Sciences
at the
Massachusetts Institute of Technology

September 1996

Signature of Author .....
Department of Brain and Cognitive Sciences
September 9, 1996

Certified by .....
Kenneth Wexler
Professor of Brain and Cognitive Sciences
Professor of Linguistics and Philosophy
Thesis Supervisor

Accepted by .....
Gerald E. Schneider
Professor of Neuroscience
Chairman, Department Graduate Committee

Case Studies in Language Learnability

by

Kevin J. Broihier

Submitted to the Department of Brain and Cognitive Sciences on September 9, 1996 in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Brain and Cognitive Sciences

ABSTRACT

In the linguistic framework of Principles and Parameters theory (Chomsky 1981), acquisition of a natural language grammar involves fixing the value of a finite set of finite-valued parameters. Theoretical and computational analyses of several proposed classes of algorithms for natural language parameter setting are reported here. These include cue-based algorithms (Dresher and Kaye, 1990; Dresher 1994), the Triggering Learning Algorithm (Gibson and Wexler 1994) and a class of algorithms that deduce parameter values from the output of the parser (Fodor 1995).

Properties of parametric spaces that will allow for successful application of each type of algorithm are identified. Computational analyses of some simplified natural language parametric systems are performed to indicate whether there is preliminary evidence to suggest that these properties can be expected to hold of the parameter spaces that underlie human linguistic competence.

Thesis Supervisor: Kenneth Wexler

Title: Professor of Brain and Cognitive Sciences
Professor of Linguistics and Philosophy

Acknowledgments

After five years, there are many people to thank. First and foremost, I would like to thank my family for a lifetime of love, support and encouragement to pursue my interests wherever they have taken me. My wife and fellow graduate student, Heather, and my mother deserve special credit for assuring me it would be all right if I didn't make it out, but then making sure that I did.

Thanks too, to my committee members: Ted Gibson, Gary Marcus, David Pesetsky and, especially, my thesis advisor, Ken Wexler. A glance through my references gives only a hint of my intellectual debt to all of them. I am also grateful to all those who instructed me along the way in both our department and the Department of Linguistics and Philosophy.

I have also benefitted greatly from my interaction with my fellow students. Thanks especially to some of those who have gone before me—Sergey Avrutin, John Kim, David Poeppel and Sandeep Prasada—and to my classmates—Philip Sabes and Fei Xu—who provided me with a orientation to life, intellectual and otherwise, at MIT. James Thomas was always an engaging conversational partner when the topic turned to the parameter-setting problem, and Kelly Olguin and Gregg Solomon always provided a helpful perspective on less technical matters.

Thanks too, to Ellie Bonsaint, Pat Claffey and Jan Ellertsen for their administrative support.

Portions of this work were presented as part of the departmental Cog Lunch seminar series. I am thankful both for the feedback from those who attended my presentations, as well as the opportunity, in turn, to learn more about the work of other students in the department.

Outside of MIT, portions of this work were presented at the Boston University Conference on Language Development, the Maryland Mayfest: Workshop on Formal Learnability and the University of British Columbia International Conference on Phonological Development. My thanks to those who all those involved in organizing these meetings including Stephen Crain, David Lightfoot, Rozz Thornton and Barbara Bernhardt. Thanks also

to Doug Pulleyblank and family and Bill Turkel for being such excellent hosts during the UBC meeting.

Two households deserve special thanks for enduring my company in the final days of my graduate career when I was "commuting" from New York. Patrick McCaw and Andrea Page hosted me during the final phases of the research reported in Chapter 6. Stefano Bertolo, Li-Fei Chen and Mila Bertolo saw me out through the last push to complete the manuscript.

Collaborating with Stefano during my last year has been one of the most enjoyable experiences in my time here.

I was extremely grateful to have some company during these last days here. Thanks and congratulations to my fellow September thesis-writers: John Houde, Adee Matan, Philip Sabes and Claudia Uller.

Table of Contents

Chapter 1
Introduction

What does a speaker of a natural language know about that language that they didn't know already when they were born? Historically (see, for example, Joos 1957), some have claimed that natural languages could differ from one another arbitrarily. On such a view, languages would be entirely unrestricted in the way they group linguistic units into larger structures, and in the way that they interpret these structures phonetically and semantically. If this claim were correct, then natural language learners would clearly have to start from scratch. Before exposure to their target language, learners could have no knowledge of the shape of the linguistic system that awaited them. The forty year history of generative grammar can be read as an extended refutation of such claims of unconstrained linguistic variety. Arguments from the poverty of the stimulus, from the rapidity and uniformity of language acquisition, and from the existence of linguistic universals have been used to motivate the claim that language learning engages a rich, innate endowment that constrains linguistic variation. Logically, the finite sets of linguistic data that learners are exposed to could be compatible with any number of generalizations. Empirically, however, learners of a language appear to draw the same generalizations rapidly and uniformly (See Crain 1991 for just one set of examples of this type of evidence and argumentation based on experimental studies with children. See also, of course, the vast generative linguistic literature that draws on native speaker intuitions.)

In the late 1970s and early 1980s, a particularly restrictive view of this endowment was articulated. The working hypothesis adopted by researchers developing this Principles and Parameters framework (see, for example, Chomsky 1981) is that the child's innate endowment universally specifies the basic cognitive machinery or principles used to build, interpret and license linguistic representations.[1] On this model, cross-linguistic variation arises because the detailed operation of the machinery that this Universal Grammar (UG) provides is sensitive to both the contents of the lexicon and, crucially, to the setting of a finite set of "switches" or parameters that can take on a finite number of values. Under the Principles and Parameters approach to linguistic explanation, then, our opening question receives a very different answer. Besides the vocabulary of the language, there is surprisingly little for the natural language learner of, say, a phonological or syntactic system to acquire.[2]

If the Principles and Parameters framework provides a correct description of the learner's innate endowment, then developmental psycholinguists face an obvious question.[3]

---

[1] Generally, linguistic theories are stated "declaratively" without any strong commitment about the details of the generative processes that underlie human linguistic ability.

[2] Depending on your view of the lexicon, the answer could even turn out to be: nothing but the vocabulary. For example, as the Principles and Parameters approach is developed in the Minimalist Program (Chomsky 1992), cross-linguistic variation in word order is explained largely by appeal to differences in the features associated with inflectional categories stored in the lexicon. The lexical learning that needs to go on in such a system, however, has a very parametric flavor to it. See Snyder (1995) for presentation of some parametrically varying properties of language that do not seem to lend themselves to such an analysis.

None of this is meant to suggest that lexical learning is even a remotely trivial problem, but only to indicate the dramatic simplification of the learning problem from, say, the point of view of a syntactician or a phonologist. Clearly, a difficult situation faces the learner attempting to map words to meanings. Moreover, the learner also faces the problem of mapping sounds onto lexical entries in the face of context-sensitive morphologically, phonologically and phonetically driven sound changes.

[3] The Principles and Parameters framework is not the only competitor in the field of generative grammar. For example, there has been an explosive growth of late in the development of Optimality Theoretic approaches to linguistic constraints. See Prince and Smolensky (1993) for

How do human learners use the linguistic (and extra-linguistic) input that they receive to set their parameters so that the grammar they end up with generates the target language that they are exposed to?

Any answer to this question must satisfy certain criteria.

First and foremost, it must account for the overwhelming empirical generalization that neurologically unimpaired children master their native languages. That is to say, it must satisfy Chomsky's (1965) criteria of *explanatory adequacy*. In the language of the Principles and Parameters framework, this means that for a learning algorithm to be considered as a serious candidate description of the algorithm that humans use, it must (at least with a *very* high probability) eventually converge to a set of parameter values that is capable of generating the target language.[4]

Satisfaction of this convergence requirement, while necessary, is clearly not sufficient. Any serious candidate description of the human language acquisition device must also be capable of acting under any informational and cognitive restrictions that the human language acquisition device can be shown to labor under.

For example, the learning of a native language (with the exception of the learning of new vocabulary items) seems to end, at the *very* latest, by adolescence. A learning device that requires more linguistic (or extra-linguistic) information than would be available to a human

---

an introduction to this approach, which has seen greatest application in the area of phonology. See also Grimshaw (1993) and Pesetsky (1993)—two prominent examples of its initial application in syntax.

[4] Obviously, if this always happened it would be difficult to explain language change in linguistically homogenous environments. Moreover, there do seem to be minor variations in the idiolects of speakers in the same linguistic community. For the purposes of this thesis, however, I will abstract away from these possibilities with the recognition that seriously accounting for them could have important consequences for learning theory.

learner during his or her developmental phase clearly cannot be considered as an explanation for human linguistic competence. Moreover, it is not enough to simply place a time bound on a candidate algorithm; the learning device must be constrained to make do with the same types of distributions of inputs found in the environments that young human learners are exposed to. In particular, properties of the target language that are esoteric and not readily available in the linguistic input that is available to almost all learners cannot be relied on. On the contrary, it is, in part, the fact that learners come to know such properties in the absence of relevant information that drives the poverty of the stimulus argument for innate constraints.

In addition to these *external* constraints imposed by the learner's linguistic environment, candidate human language acquisition devices must also satisfy any other constraints that can be derived from our knowledge of human psychology. For example, candidate algorithms must satisfy any theoretically or empirically established restrictions on the language learner's ability to process or store information.

In the absence of detailed psychological evidence, it seems a reasonable approach to start by proposing fairly tight limitations on the properties attributed to the human learner, and to abandon these limitations when forced to by consideration of further psychological evidence or concrete consideration of the learning problem. As an example of the types of limitations that have been proposed, Wexler and Culicover (1980) assume both that the learner has no memory for past data and that the complexity of the class of transformational grammars that they studied is bound in a particular way. Throughout, I will draw attention to features of the learning algorithms that I consider that require psychological mechanisms that go beyond the conceptually necessary abilities to change parameter values and to determine whether a candidate hypothesis is in error.

Finally, the candidate language acquisition device must contribute to (or at least be compatible with) an explanation of what can be observed about the learner's developing grammatical knowledge. For example, if empirical documentation could be provided that traced out sequences of parametrically varying hypotheses for learners of particular target languages, then a candidate human learning algorithm should contribute to an explanation of these learning trajectories. If, of course, it were to turn out to be true that, after some point in development, no evidence could be found that children misset certain parameters, a satisfactory parametric learning theory would need to match this accomplishment. If Wexler's (1996) Very Early Parameter Setting hypothesis can be maintained, then the upper bound on the amount of time to set (at least certain) parameters falls somewhere before children begin producing utterances complicated enough to potentially reveal a missetting.

The question of how parameters are set has received widely contrasting answers. From my point of view, the most dramatic divide in the parametric learning literature lies between cue-based approaches of the sort exemplified by Dresher and Kaye (1990) and Dresher (1994) and approaches that perform what I will call implicitly-guided mechanistic search (IGMS) through the space of parametric possibilities (see, for example, Nyberg 1991; Clark 1992; Clark and Roberts 1993; Niyogi and Berwick 1993; Gibson and Wexler 1994; Bertolo 1995).

As linguistic theory has developed, (at least certain) parameters typically interact extensively. Changing even a single parameter can dramatically change the set of structures that a language generates. Moreover, a single parameter value rarely suffices to guarantee that a grammar will license a particular structure. Rather, the generation of structures typically requires the contribution of several parametrically provided options. From the point of view of a researcher interested in providing a compact description of linguistic variation, a massive

interaction between parameters could turn out to be highly desirable. The Principles and Parameters approach holds out the hope that differences between languages that, on the surface, look radically different will reduce to a small set of differences in the settings of highly interactive parameters. The learner, however, faces the problem of "seeing through" this intricate interaction to the target parameter values.

A leading motivation of the cue-based approach is a belief that the interaction between parameters is complicated enough that the component of UG that outlines the parametric system (thus limiting the range of possible human languages) must be supplemented with a sort of instruction manual that guides the learner through the parameter-setting process. This additional UG component provides the learner with an ordered "flow chart" of tests to apply to the input stream from the target language. The results of each test serve to constrain the settings of the learner's parameters, until, at the end of some branch of the flow chart, the learner has set them all —hopefully to their target values.

What I am calling IGMS algorithms, on the other hand, embody a more optimistic belief and a stronger hypothesis that the availability of the parametric system, in and of itself, is largely sufficient for the work of language learning. Given the parametric system, proponents of IGMS algorithms hypothesize that simple search strategies will lead the learner to the target.

The typical IGMS algorithm relies primarily on cognitive mechanisms that, unlike an instruction manual for parameter-setting, can receive some motivation that is independent of the need to solve the learning problem. The key cognitive components in the IGMS-style work referenced above are parsers capable of taking linguistic input and assigning structure to that input (or failing to do so) in accordance with the setting of UG parameters. I take it that the

claim that such parsers exist is relatively uncontroversial in any current version of psycholinguistics that makes contact with work in generative grammar.[5] IGMS algorithms use the output of these parsers in a mechanistic fashion to revise the learner's hypotheses about the way in which their grammar's parameters should be set so as to generate the target language. By use of the term *mechanistic*, I mean to imply that the processes that operate on the parser output to produce parametric hypothesis changes do not have access to any domain-specific information about the contents of UG, nor do they have any ability to make deductions about the way that parameter values interact in the generation of the well-formed representations of a language. By the use of the term *implicitly guided*, I mean to point out that such algorithms rely primarily on the parameter space having an appropriate configuration so that the pursuit of their mechanistic policies will lead to convergence.

This thesis will focus on Gibson and Wexler's (1994) Triggering Learning Algorithm (TLA) as an exemplar of the IGMS approach. The TLA attempts to change hypotheses whenever it comes across an input that it is not able to parse with its current parameter settings; the algorithm selects a potential new hypothesis by simply randomly choosing a single parameter to "flip". The new hypothesis is only adopted if it allows the learner to process the previously unanalyzable input.

Following earlier work in learnability (e.g., Wexler and Culicover 1980), the TLA is an error-driven IGMS algorithm; the learner only revises its hypothesis when it encounters inputs that it cannot accommodate with the current hypothesis. This, however, is not a defining

---

[5] Some IGMS algorithms, such as the genetic algorithms in Clark's (1992) proposal, also require the parser to return some indication of what parameters, or at least how many parameters, a particular structure violates. This, however, seems to beg a number of important questions.

property of the IGMS approach, as I have presented it. For example, genetic algorithms for language learning (Clark 1992; Clark and Roberts 1993), another branch of the IGMS family, form new hypotheses by splicing together parameter values from a pool of old hypotheses according to a mechanistic scheme that reflects the parsing success of the old hypotheses; relatively successful hypotheses contribute more of the raw material used to create hypotheses in the following generation. The learner could change hypotheses after a stretch of input even if all hypotheses in the current pool successfully parsed the input, so the scheme is not entirely error-driven.

Unlike cue-based algorithms, which explicitly build in chains of deductive logic for the learner to follow *en route* to the acquirable natural languages, IGMS algorithms essentially rely on the parameter space having a structure that pulls the learner to the target grammar, perhaps with some errors and missteps along the way. Again, this is what I intend by the term *implicitly guided*. The particular structure that an IGMS algorithm requires will clearly vary somewhat with the details of the algorithm's hypothesis revision policy, and an important program for parametric learning theory is to establish general results about the properties of such learners.

In the sense sketched above, IGMS algorithms are more parsimonious than their cue-based competitors. Both cue-based and IGMS algorithms rely on a parametric specification of the space of natural languages. IGMS algorithms, however, seek to do without any explicit road map to the target grammar. Occam's Razor, then, would dictate that, all else being equal, IGMS algorithms should be preferred. Of course, all else is rarely ever equal when attention turns to the time and resource requirements that different algorithms impose. At present, a major consideration in favor of the cue-based approach is that, as will be seen below, its basic

design philosophy, when it can be successfully applied, leads to a learning algorithm capable of rapidly zeroing in on the target parameters. A key question here, of course, is what properties of a parametric space make it is possible to successfully develop a cue-based instruction manual. With IGMS algorithms, there is little room for clever devices that exploit the particular content of linguistic parameters or the pattern of their interaction. Instead, the adoption of a locally guided search algorithm requires the strong hypothesis that the pursuit of simple policies for revising hypotheses on the basis of the parser's output will lead the learner quickly through a potentially vast parameter space.

As discussed above, the ultimate test for any proposed learning algorithm is to explain (in concert with our other theories of human cognition) what we can observe about both the time course of acquisition and the structure of the human mind/brain. The aims of this thesis are considerably more modest. I intend to further lay out some issues involved in choosing between approaches to the problem of parametric language learning. I will also provide some concrete examples of their application to simplified phonological and syntactic spaces, and engage the question of how some of the time and space requirements that these approaches demand might scale up to succeed in the *real* parametric space—yet to be specified—that researchers in the Principles and Parameters framework believe underlies human language. In particular, since the size of parameter spaces are exponentially dependent on the number of parameters that constitute them, a key concern will be the need to avoid learning strategies that are linearly dependent on the size of these exponentially large parameter spaces.

In Chapter 2, I will briefly make the case that search algorithms that do not allow data from the target language to provide *any* useful implicit guidance from the parameter space as to what hypothesis to consider next are unlikely candidates for the human learning

algorithm. Such algorithms appear regularly in literature that emphasizes certainty of convergence in the learnability framework of identification in the limit (Gold 1967), but are almost certainly inadequate in large parametric spaces.

In Chapter 3, the focus will be on cue-based algorithms. I will begin by articulating the cue-based approach with an eye towards extracting what I take to be the key components of its design philosophy. I will then lay out abstractly what I take to be the best possible and worst possible types of parameter spaces that might confront the class of cue-based algorithms. The focus will then turn to concrete examples of cue-based algorithms for the acquisition of simplified, model parameter spaces. Dresher (1994) presents an instruction set designed to acquire a fragment of the system of metrical phonological parameters developed in Idsardi (1992) and Halle and Idsardi (1994). I will present the phonological system and the cue-based algorithm, evaluate the success of the cue-based algorithm in satisfying the design philosophy of the cue-based approach, and identify issues involved in extending the system to more realistic approximations to the space that proponents of the Principles and Parameters approach to metrical phonology might eventually develop. Next, I will briefly discuss possible cue-based algorithms that successfully acquires languages in the 3-parameter syntactic space that Gibson and Wexler (1994) apply the TLA to.

In Chapter 4, the focus turns directly to Gibson and Wexler's TLA—this thesis's running example of the IGMS approach. I will explicitly present the algorithm and briefly describe some of the learning results that Gibson and Wexler (1994) obtained. In particular, I will describe their maturational extension of the TLA (MTLA), which suffices to ensure identification in the limit of all languages in the space they investigate with arbitrarily high, but not certain, probability. Next, I will demonstrate how a similar maturational

extension, in combination with a linguistically reasonable restriction of the space, allow for in-the-limit TLA acquisition of a simplified phonological system based on Idsardi (1992) and Halle and Idsardi (1994) .

As discussed above, however, identification in the limit, while important, is a very modest goal in the context of the systems of phonological and syntactic parameters considered here, and is, in fact, easily guaranteed. As I will point out along the way, both of the finite parametric spaces presented in Chapter 4 are free of the proper subset/superset relations among languages that could potentially contribute to a general negative result for error-driven, IGMS learners with no domain-specific knowledge about the effects of particular parameters. While the unmodified TLA is actually restricted in such a way that identification in the limit is not guaranteed even in the absence of subset/superset relations, very minor alterations to the algorithm make it possible to ensure that, in the limit, the learner converges to the target with probability *1* without greatly altering the characterization of spaces that would allow for tractable acquisition. I will suggest one such set of alterations here. In this chapter, I will also briefly discuss the issues that arise in spaces where proper subset/superset relations do exist.

With the problem of identification in the limit addressed and with a recognition that the TLA makes minimal demands on the learner's cognitive resources, it is possible to focus on what I take to be more crucial questions about TLA-type algorithms' expected time to convergence. In Chapter 5, I will follow the lead of Niyogi and Berwick (1993) (who in turn follow up a suggestion of Gibson and Wexler (1994)) and apply the mathematical theory of Markov chains to gain some preliminary insights into the plausibility of applying the TLA to systems like our models. Niyogi and Berwick's (1993) paper provides an excellent overview of the range of questions that can be addressed within the context of Markov chain theory.

In Chapter 6, I will turn to a new set of syntactic parameters and examine the promise of a novel approach to parameter setting that falls somewhere in the gap between cue-based and IGMS algorithms. This approach, advocated in Fodor (1995) in a restricted form, aims to make use of cues and to set parameters in a deductive fashion, unlike the typical IGMS algorithms referred to above which are willing (at least in theory) to pursue a series of incorrect hypotheses and parameter revisions on the way to the target grammar. However, it aims to do so without explicitly storing the cues in UG, or, in fact, without making use of much more than the constraints encoded in UG and the "logic" of the parser. In particular, for systems where parameters can be identified with local configurations of a parse structure, a generalized family of Fodorian learners, which I will call On-line Parsing Logic (OPL) learners, aims to set parameters by noticing which components of linguistic structure, and, therefore, which parameter values, are necessary to accomplish a parse of an input form. This involves a denial of a typical assumption of proponents of cue-based learning, who argue that, at least initially, there is an epistemological gap between the abstract mental "language" that parameters are couched in and the perceptually available properties of the input. The parser, Fodor essentially claims, allows the learner to directly perceive the parameter values that are compatible with the linguistic forms that a learner is exposed to. In a sense, an OPL's cues are generated "on the fly" as a byproduct of the parser's computations, voiding the need for explicit representation in UG. This approach, however, requires that the learner parses in a special *exhaustive supergrammar mode*. In this mode, the learner attempts to provide all humanly possible parses of the sentences that it encounters. This approach is required, so that the learner does not reach mistaken conclusions about the necessity of some parametrically available structural component. I will discuss the convergence properties and resource

requirements of OPL learners in terms of the vocabulary developed in Chapter 3. Chapter 6 will also contain an illustrative application of OPL learning to our extended syntactic space and point to several problems that arise there and some open issues concerning the OPL approach.

Chapter 2
Parameter Setting as Search

As noted in the introduction, the move to the Principles and Parameters approach is a move to a world where, from the point of view of the parametrized system, there are only finitely many possible natural human languages. For example, in the domain of syntax, since there are only a finite number of parameters to be set, and since these parameters can only take on a finite number of values, it follows that (abstracting away from vocabulary and all the complications that arbitrarily long words might introduce) there are only a finite number of possible natural languages. Does this restriction trivialize the learning problem?

In a logical sense, the answer would seem to be yes. From the point of view of the traditional learnability criterion of identification in the limit articulated by Gold (1967), the move to a finite class of natural languages has a dramatic effect. The question of identification in the limit is the question of whether a learning algorithm, exposed to a sequence of data generated from a target language, will, at some point, hypothesize a grammar that generates the target and, then, maintain that hypothesis forever after. In keeping with standard views about the linguistic evidence available to children (see for example, Braine (1971); Brown and Hanlon (1970); McNeill (1966); Marcus (1993); Wexler and Hamburger (1973)), learning algorithms in this framework receive no direct evidence that would indicate that a particular form was *not* generated by the target language. The standard identification in the limit criterion also requires that an algorithm be able to converge to target languages independently of the particular order of the input data that it receives, although the data stream must not

systematically and eternally withhold evidence. If an adversarial caretaker chose to simply repeat the same perfectly grammatical sentence over and over to a learner, the learner could hardly be expected to demonstrate mastery of their target language. If a learning algorithm had an unbounded sequence of input data to work with, it is possible to imagine that it might undertake an exhaustive search for the target grammar in the parameter space by simply enumerating and testing every member of the finite class of languages, either in serial or in parallel, and seeing which ones fit the data that streamed in from their caretakers and other members of their linguistic community.[6]

In the absence of noise—linguistic input data that the learner receives that does not come from the target—the main *logical* complication that arises for such approaches is the possible existence of subset/superset relations between languages.[7] Recall from work such as Angluin (1978), Berwick (1985) and Manzini and Wexler (1987) that a superset language for a target language is a language, other than the target, which generates all the linguistic patterns that the target generates. Clearly, if the set of possible languages included languages that were supersets for a target, then these superset languages could never be deductively discarded through a process of elimination that simply considered whether a grammar could cover all the examples it had seen so far. Problems with supersets arise in both parallel and serial schemes of enumeration and elimination.

---

[6] If data came from a mix of several target languages and the learner can reliably separate the input stream, the problem for the multiingual learner is reduced to several instances of the original problem.

[7] The algorithms presented and developed in this thesis do not have any special mechanisms for accommodating noise. In my mind, this is a major shortcoming. I will not, however, remedy this shortcoming, but will attempt to provide discussion of the particular noise sensitivities that the different algorithms have.

In a parallel evaluation scheme, the learner would begin by considering all grammars in the space as viable candidates. When data from the target could no longer be accommodated by a candidate grammar, that candidate grammar would effectively drop out of the competition. If there are several grammars that generate exactly the set of forms allowed by the target language—that is to say if several grammars generate *weakly equivalent* languages—, then the learner actually has no principled way of selecting a single grammar.[8] Presumably, if the learner eventually does zero in on a unique hypothesis, it can only do so by making a choice that is arbitrary from the point of view of the generative power of the grammar. If, however, the languages generated by some grammars are proper supersets of the target, such an arbitrary choice could be a mistake. In particular, it would be a mistake to select a grammar that generated a superset of the target rather than the target itself.

In the more traditional serial scheme of enumeration and elimination, a grammar's position in a list of grammars crucially determines whether it will be chosen over other grammars that are also capable of generating the data from the target language. Since the learner tries out hypothesis grammars in a specified order, they will adopt the first grammar that they encounter in the list that generates all the data from the target language. To avoid mistakenly selecting a grammar that generates a superset of the target language, consideration of hypotheses must be organized in such a way that a learner does not hypothesize a grammar that generates a superset language until it has ruled out all of the corresponding subset languages.

---

[8] For my purposes, languages are weakly equivalent if they result in perceptually identical sets of input data for the learner.

To avoid superset mistakes, a learner engaged in exhaustive search must have some additional capabilities that allow it to identify subset/superset relations among languages and somehow decide that evidence that would argue in favor of the superset language was lacking. Of course, this is really true of *any* learner. The learner must either avoid ever hypothesizing a grammar that generates a superset of the target, or the learner must possess some means of monitoring that evidence that would argue in favor of a superset of the target is lacking—allowing the possibility of "retraction" from the superset. In the second case, the learner will not strictly satisfy the criterion of identification in the limit. Identification in the limit does not allow the input text to eternally withhold evidence, but it is impossible to rely on crucial evidence occurring with any fixed interval; the learner may be able to wait long enough to reject a superset grammar with a great deal of confidence, but never with certainty.

Enumerative approaches to learnability predate the Principles and Parameters framework and many theorems describing their applicability to classes of infinite languages are to be found in Wexler and Culicover (1980) and the references cited there. More recently, Bertolo (1995) provides a formal proof that ensures that a serial scheme that uses an enumerative listing that places targets generating subset languages before targets generating the corresponding superset languages will succeed for finite classes of languages even if the membership of linguistic forms in languages is not decidable. This clearly covers the situation that confronts a learner of a finite parametric system.

Wu (1994) provides a recent example of a fairly direct attempt to implement the first strategy of superset avoidance mentioned above. His learner performs an exhaustive serial search through a listing of grammars in a Minimalist (see Chomsky 1992) syntactic parameter space consisting of fourteen parameters. Wu provides a simple algorithm that constructs a

listing of grammars that is guaranteed to eliminate any difficulties due to subset/superset relations among the languages that the grammar generates. As Bertolo (1995) notes, it is a non-trivial finding that such a simple algorithm for computing an effective listing exists. Wu's learner has the virtue of not needing to store a precomputed list of all the grammars in the parametric space; instead, it can generate elements of the sequence on-line, as necessary. In the sense of identification in the limit, then, Wu's example, indicates how the move to finite parametric systems can meet with success; a learner with very limited resources can effectively compute an enumeration that can lead to identification in the limit.

In the parametric spaces that will occupy us in the rest of the thesis, matters are even simpler. In the phonological parametric spaces derived from Idsardi (1992) and Halle and Idsardi (1994), in Gibson and Wexler's (1994) three-parameter syntactic space, and in the extension of Gibson and Wexler's syntactic space that is the focus of Chapter 6, there are no proper superset/subset relations among languages. An exhaustive search approach is guaranteed to succeed in such spaces, provided that any grammar that generates the target language is deemed acceptable.

However, this exhaustive search approach, as exemplified by Wu (1994), will clearly break down if natural language parameter spaces are very large, which they easily could be since the number of possible grammars grows exponentially with the number of parameters. With n parameters, and with some fixed upper bound on the number of parameter values, the number of candidate grammars grows exponentially as $\Omega(2^n)$. This *order of growth* notation adopted from computer science indicates that there are constants m and c, positive, such that if $n > m$, then the number of grammars is greater than $c*2^n$. (If, an upper bound rather than a lower bound is desired then, *greater than* is replaced with *less than* and O(.) notation is used.)

With n binary parameters, the space of possible parameter settings grows as exactly 2^n. With multi-valued parameters the absolute rate of growth of the parameter space can only grow larger. Computational problems whose resource requirements grow as $\Omega(2^n)$, where n reflects some indication of the size of the problem, are generally considered intractable. Obviously, the "real" parameter space encoded in UG has some fixed constant size, so that the number of candidate grammars can be upper-bounded by a constant. Nonetheless, the exponential growth "hidden" in this constant size is likely to be quite prohibitive if n is at all large. Berwick and Weinberg (1984) and Barton, Berwick and Ristad (1987) provide a thorough discussion of the dangers of operating in the exponential regime.

With this caveat about the fixed size of the real system in mind, I take the need to avoid resource requirements that depend exponentially on the number of parameters in the system as a key constraint on theories of human language learning. In the enumerative approach outlined above, and adopted by Wu as a proposal about the human natural language learner, successful acquisition requires the learner to perform a certain fixed amount of computation for the target grammar and for each candidate grammar in the enumeration that precedes it. Minimally, before rejecting a grammar, the learner must check to see whether or not it generates the current input. If attested human languages populate the tail end of the enumeration, then the learner will clearly have to perform $\Omega(2^n)$ computations in order to reach it—an unacceptable result. If the learner takes 2^{n-1} steps to reach the target, the learning scheme still requires $\Omega(2^n)$; choose c less than $1/2$. These arguments against enumerative approaches, of course, do not depend on the claim that linguistic variation is parametric. Wexler and Culicover (1980) provide an example of pre-Principles and Parameters

argumentation against enumeration based on similar considerations of the resources available to the learner.

Our current knowledge of the resource limitations confronting the human learner does not allow for anything resembling a precise estimate of the point at which an exponentially growing parametric space would definitively break an enumerative search scheme for parameter setting. Nonetheless, consideration of common, although perhaps undermotivated, speculation about the number of parameters in natural language suggests that parameter spaces are large enough that any learning scheme that requires a learner to consider a sizable fraction of the set of candidate grammars would be doomed.[9] Clark (1992) suggests a conservative consensus estimate of between 30 to 40 parameters. This estimate translates directly into an estimate of roughly a billion to a trillion possible natural language grammars. If such estimates of the size of parameter spaces turn out to be overly pessimistic, either because there simply are not that many parameters or because parameters are partitioned into manageably sized, non-interacting modules that can be learned independently, then investigation of the properties of learning algorithms in large parametric spaces may turn out to be something of a scientific dead end. If, on the other hand, these estimates are correct or overly optimistic, it seems most likely that exhaustive search approaches will not be viable, even though they might be perfectly successful from the point of view of identification in the limit. Approaches based on a random, error-driven search through the parameter space will not succeed for similar reasons. With no *a priori* knowledge of the identity of the target grammar, a random search must expect, on

[9] An estimate of the number of grammatically distinct languages of the world, past and present, would straightforwardly provide a *very* conservative lower bound on the number of parameters, on the assumption that parameters were binary.

average, to consider half of the grammars in the space before finding the target.[10] Again, this halving of the size of the space does not change the fact that this search will grow as $\Omega(2^n)$—an unacceptable result if parameter spaces are large.

Nonetheless, something is conceptually appealing about the trial-and-error approach. If nature has essentially provided learners with a complete model of possible human languages, shouldn't that be enough? The Principles and Parameters view already includes the belief that the learner comes equipped with a reconfigurable grammatical system. The learner must be able to access and change parameter values if learning is to take place at all. Under relatively straightforward views, the parser also directly provides the learner with the ability to decide whether or not patterns are generated by a particular grammar.[11] Clearly, some way of rejecting certain hypotheses and adopting others is required. How much more does the learner require than these abilities and a modest amount of control structure of the sort familiar to any beginning computer programmer? The rigidity that governs an enumerative or strictly random search's choice of new candidate hypotheses makes it impossible to avoid search times that grow exponentially with the number of parameters. The itinerary of hypothesis changes that the learner goes through is not dependent on the target language in any useful way. Does the clear inadequacy of enumerative or random searches, however, mean that the entire trial-and-error approach, which they are a special case of, is bankrupt? Could the learner's successes and failures in analyzing data from the target language be used to guide

---

[10] If there are m weak equivalents of the target that are equally as acceptable than this lower bound on the expected search time should be adjusted appropriately.

[11] The parser may not be able to provide a definitive decision in all cases. For example, in the domain of syntax, it may simply break down in the face of deeply-nested examples of center-embedding. Presumably, such examples are not part of the core of sentences that are used to fix parameters.

this search in a simple, non-domain-specific way that did not require a non-zero amount of processing to be performed on a significant fraction of grammars in the system? Could a simple learner avoid considering vast regions of parameter space? To put it another way, does domain-specific knowledge about the solution of the parameter setting problem need to be built into UG or will an IGMS algorithm like the TLA suffice? Alternatively, could the same domain-specific knowledge that constrains the parameter space be used in a simple fashion to se parameters.

Chapter 3
Cue-based Approaches to Parameter Setting

3.1 Motivation

As will be discussed further in Chapter 4, IGMS algorithms are willing to tolerate incorrect hypotheses *en route* to the target grammar. In the cue-based approach exemplified by Dresher and Kaye (1990) and Dresher (1994), there is a much greater emphasis on the correctness of intermediate partial hypotheses about the identity of the target grammar. In addition to a general sense of the intricacy of parameter interaction, the need for an extensive set of built-in instructions for parameter setting is further motivated by the perceived difficulty of otherwise avoiding false beliefs about the target grammar in light of the claims such as those in (3.1.1), adapted from Dresher and Kaye (1990).

(3.1.1)

a. The credit problem: When there is a mismatch between a target form and a learner's grammar, there is no way of reliably knowing which parameters must be reset to yield a correct output.

b. The epistemological problem: There is a gap between the vocabulary in terms of which parameters are couched and the learner's analysis of the input.[12]

---

[12] We will see in Chapter 6 that Fodor's approach essentially denies the existence of this second problem. OPL learners explicitly identify the vocabulary in terms of which parameters are couched with features of the learner's analysis of the input.

To expand somewhat: a major problem for the learner of a parametric system is that linguistic forms that the learner samples from the target language do not come labeled with the parameter settings that were used to generate them. Somehow, the learner must work backwards from the input they receive to discover parameter values that will generate the target. It could have been otherwise, but, in the real world, this problem seems to be quite complicated due to the extensive interaction between linguistic parameters.

Consider, for example, the problem that confronts a learner attempting to learn the mapping from syllables to stress patterns when that learner hears the following schematic word and its associated stress pattern. (L's indicate light syllables. H's indicate heavy syllables. Syllables in a language can be categorized as light or heavy on the basis of the segments that make them up. The height of the column of "grid marks" above a syllable indicates its relative prominence in the stress contour. A parametric system for stress will be developed in more detail in following sections.)

(3.1.2)
```
    *   *
*   * * *
L H L H
```

In particular, consider how the learner might go about determining how to set the parameters so as to capture the fact that the second and fourth syllables in this word receive stress. It is conceivable that the learner approaches this task in the same way that a student in an introductory phonology class might—armed with their knowledge of the parametric system governing stress assignment. Such a student would know that the process of binary foot formation provides one parametrically available device for forming metrical groupings. They might, then, surmise that the language responsible for generating (3.1.2) is just such a language.

With the auxiliary hypothesis that the target language stresses the rightmost member of these binary constituents, it would be possible to generate the stresses in (3.1.2).

However, while this is one possible analysis, a careful student would note that there are alternative analyses of the stress pattern in (3.1.2) that do not require binary constituents. For example, in some languages all heavy syllables bear stress. Since both of the stressed syllables in (3.1.2)happen to be heavy, it is possible to generate the appropriate stress without any use of binary feet. Of course, these alternative analyses will diverge on other input patterns. Here the complication is that the same pattern of data can be generated with very different parameter settings. Parameter setting is also complicated because a small number of changes in parameter settings can have drastic effects on the shape of a grammar's outputs.

Hearing form in (3.1.2), then, does not allow the learner to draw any definite conclusions about the setting of the parameters governing binary constituent construction. It is easy to imagine, however, that there could be other data for the learner to use, either individually or in combination with (3.1.2), to get more inferential leverage on the problem. Moreover, the learner might make even more progress by taking previously established parameter values into account. This is essentially the cue-based approach to parameter setting that Dresher and Kaye (1990) and Dresher (1994) advocate. Of course, these authors dispense with my fiction that the learner is actively reasoning about parameter setting. Instead, they propose that all the necessary chains of inference are, in some sense, pre-compiled and encoded in UG—essentially providing the learner with a set of step-by-step instructions detailing what types of patterns of input data to look for and what conclusions to draw when these patterns are or are not discovered. The ultimate aim of such cue-based algorithms is to provide a set of

instructions that suffices for the acquisition of any learnable language in the linguistic space at hand.

What are the circumstances under which this aim can be met in a way conducive to learning in large parametric spaces? This question will be the focus of this chapter.

In Section 3.2 I will develop some vocabulary for the description of cue-based algorithms. The development of the cue-based approach that I present in Section 3.2 will be quite general. So general, in fact, that it is capable of describing exhaustive search algorithms such as Wu's which we have already rejected as potential candidates for natural language learners in large parametric spaces.

In Section 3.3 I will briefly detail the points of difference between my general development of cue-based algorithms and the more restrictive development held up as the model in Dresher and Kaye (1990) and Dresher (1994). The more restrictive criteria stated in what I will call Dresher and Kaye's "design philosophy" for cue-based algorithms further restricts the class of cue-based learners in a way that rules out exhaustive search algorithms. Moreover, it can be shown to lead to rapid learning for large parametric systems that it can be successfully applied to.

This notion of rapidity will be made precise in Section 3.4. In Section 3.4, I will also point to a potential problem with cue-based approaches that, as far as I know, has not been discussed in any detail elsewhere: since cues need to be explicitly encoded in UG, it is necessary to ensure that the problem of dealing with an exponential time-dependence on the number of parameters in the system is not simply traded for an exponential space or memory-dependence on the number of parameters.

After the discussion in Sections 3.1–3.4, the chapter will take a decidedly empirical turn. Section 3.5 will be the site for an introduction to the Idsardi (1992) and Halle and Idsardi (1994) system of phonological parameters. Section 3.6 will present Dresher's (1994) outline of a cue-based algorithm for acquisition of this system. The informal sketch in Dresher (1994) is, as Dresher himself indicates, not complete. Dresher focuses on the learning paths for several classes of languages in the space, but does not provide a general proof that the learner will successfully acquire all languages in the space. I will attempt to fill in the gaps. Section 3.7 will evaluate the extended solution in terms of its satisfaction of Dresher and Kaye's design philosophy. Section 3.8 will discuss issues involved in extending the system to handle a larger phonological fragment.

Finally, Section 3.9 will briefly present some possible simple cue-based learners for Gibson and Wexler's 3-parameter syntactic space, which, as we will see in the following chapter, is a space that the simplest version of the TLA cannot be guaranteed to succeed in; the algorithm can be made to succeed in this space if certain natural extensions are made. This result is intended to foreshadow some issues that arise with the OPL learners presented in Chapter 6, by drawing attention to a property of the parameter space that could lead to rapid OPL acquisition. In particular, each language in the Gibson and Wexler space will be seen to contain strings that are unique to it.

3.2 Cue-based algorithms

Like any inferential engine, cue-based algorithms are fueled by beliefs. The beliefs that are important to cue-based algorithms acquiring natural language are beliefs about the

membership and properties of the set of linguistic forms that make up the data stream from the target language, or *text* in the sense of Gold (1967).

Definition 3.2.1

Call the beliefs that the cue-based algorithm forms about the target language on the basis of the data stream *hypotheses*. Call hypotheses that are deductively warranted on the basis of the segment of the data stream that the learner has already observed *observations*. Call all other hypotheses that are not contradicted by the data stream *conjectures*.[13] Call the aspects of the data that could potentially cause the learner to appropriately form an observation or a conjecture *possible cues* for those observations or conjectures.

A particular cue-based learning algorithm will, in the course of acquiring a grammar, actually form observations and conjectures in response to detecting certain possible cues. Some of those observations and conjectures, in turn, will cause the learner to set parameters, or otherwise reduce the set of candidate grammars. If, given any reduction in the set of candidate grammars that the learner has already achieved, the learner's language acquisition device, as implemented, could, for some further segment of the data stream, actually reduce the set of candidate grammars further (parametrically or otherwise) on the basis of a possible cue (and if this reduction is deductively licensed given the learner's current candidate set and the given observation or conjecture), then call the possible cue that sets off this chain of events an

---

[13] The learner might actually base their conjectures on a window of input that is somewhat smaller than the larger text, but I will not generally make much of this distinction. I will assume that no initial segment of the text will contain critical information that the learner could not extract by beginning later in the text.

implemented cue for the reduction, in the context of the particular learning algorithm and an appropriately reduced candidate set.

Any particular algorithm will implement some subset of the vast logical space of possible cues. In cases where confusion will not result, I will often refer to both possible and implemented cues simply as cues. Note that cues do not have to be considered in combination with the full set of beliefs that a learner has acquired, but simply on their current hypothesis space.

A learner whose beliefs consist entirely of observations, in other words a learner whose beliefs about the target language are based on positive evidence, does not resort to conjecture and, in the absence of noise, cannot form any false beliefs. A learner that relies on indirect negative evidence, by definition, does rely on conjecture. The absence of some feature of the data stream in some initial segment of the text is used to infer its absence in the target language. A false inference could result in an incorrect reduction of the set of possible grammars and prevent convergence to the target. In keeping with our desiderata for the language learner, for cue-based systems that use indirect negative evidence, false inferences must be exceedingly rare.

Some distinctions based on the content of hypotheses will prove useful in later discussion. In the absence of noise, it is convenient, and harmless, to gloss over the distinction between languages and texts for languages.

Definition 3.2.2

Define a *form observation* as an observation that a particular form occurs in the target language.

Define a *property observation* as an observation that some form in the target language has the specified property.

Define a *form-presence conjecture* as a conjecture that a particular form occurs in the target language.

Define a *form-absence conjecture* as a conjecture that a particular form does not occur in the target language.

Define a *property-presence conjecture* as a conjecture that some form in the target language has the specified property.

Define a *property-absence conjecture* as a conjecture that no form in the target language has the specified property.

As an example of the application of these definitions, consider the conclusions that a learner might draw when it encounters the form, "John saw Bill". The learner might develop a form observation corresponding to the belief that "John saw Bill" was a sentence of the language that they were learning.[14] Upon hearing the same sentence, "John saw Bill", the learner would also be free to develop a property observation reflecting the belief that the grammar that they were acquiring allowed sentences that were three words long. This is, of

[14]The notion of form observation might be relaxed somewhat to abstract away from lexical particularities that do not matter for the syntax. After hearing the sentence "John saw Frank", hearing the sentence "John saw Bill" in a similar context does not seem to warrant a conceptually distinct observation. For example, if the learner had figured out the syntactic categories of the words involved here and could also determine which noun phrase was the subject and which was the object, on the basis of semantic context, then both of these utterances might receive the same encoding—"Subject Verb Object".

course, only one among infinitely many property observations that a learner might be built to form in response to this input. A learner might also develop any number of conjectures.

Form-presence and property-presence conjectures are included for completeness, but, for obvious reasons, do not seem likely to play an important logical role in the development of cue-based algorithms. There is little reason to conjecture about the presence of forms or properties when they will eventually be revealed by the data stream if the learner simply waits. Generalization of these definitions to sets of forms and properties should be straightforward.

How does the cue-based learner translate its observations and conjectures into beliefs about the identity of the target grammar? Under the Principles and Parameters approach, the learner begins life knowing that the grammar that generates their target language lies somewhere in a finite set of candidate grammars. Initially, assuming all the languages in the space are learnable, the learner has no reliable information about which grammar will turn out to be the target.[15] As observations and conjectures are formed, certain grammars become incompatible with the learner's beliefs and can be eliminated from the candidate set. Eliminations that are deductively licensed by observations are guaranteed to be correct; the learner who relies solely on observations will never discover that a grammar was erroneously eliminated. Eliminations based on conjectures stand some non-zero, hopefully small, chance of leading the learner astray. At the point that they form a conjecture, the learner may not yet have encountered the crucial evidence that would contradict it.[16] In the end, of course, if the

---

[15] Universal grammar could conceivably encode some type of probabilistic, *a priori* beliefs about the likelihood of different grammars.

[16] The learner might also only consider a fixed window of evidence, so that contradictory evidence might actually have been previously encountered in the text, but at a point that was too early for it to do any good.

cue-based hypothesis is correct the learner ideally selects a single grammar that generates the target language.

I take this reasonably reliable, deductive narrowing of the candidate set as a central, perhaps the central, property of the cue-based approach. It potentially stands in stark contrast to the behavior of IGMS algorithms which, depending on the parameter space, are quite free to form intermediate hypotheses that are inconsistent with the target grammar.

Notice, however, that at this level of generality it is possible, with only slight modifications, to describe an enumerative learner like Wu's, which plausibility considerations based on the size of natural language parametric spaces have already led us to reject. At the outset, all possibilities are open to an enumerative learner. As the learner forms observations corresponding to the belief that sentences that it encounters are part of the target language, it is able to notice that these same sentences cannot be generated by certain grammars in the enumeration. As a result, the learner proceeds forward in the enumeration and eliminates all grammars that fall earlier in the enumeration from consideration. Eventually, the learner will come to the a grammar capable of generating the target language and stop making progress through the enumeration. In a strict identification in the limit paradigm, the learner would never conclusively rule out grammars later in the enumeration. However, it is easy to see that the learner might eventually conjecture the absence of forms that are not generated by the most recently selected grammar in the enumeration. At this point, all later grammars could be eliminated from consideration.

Given the extreme generality of this notion of cue-based learning, the interest in the discussion of Dresher and Kaye's design philosophy will lie in noting how the additional constraints that they impose restrict the notion of cue-based learning in a way that, if they

were right, would make for tractable acquisition. It will also be of interest to identify some properties of systems where their approach can be successfully applied and some properties of systems that will lead to breakdown. The development of some vocabulary, in this case vocabulary for talking about the effect that observations and conjectures have on reducing the candidate set of grammars, will facilitate later discussion both in this chapter and also, conveniently, in discussion in Chapter 6, when I turn to consideration of Fodor's proposal and the more general class of OPL learners.

The enumerative learner performs a non-zero amount of work for each grammar that it eliminates. A more efficient learner might eliminate large portions of the parameter space in a single step. Ideally, for reasons that will be made explicit in section 3.4, candidate grammars can be eliminated a parameter (or at least a parameter value) at a time. The following definition provides some terms for talking about this desirable state of affairs.

Definition 3.2.3

Divide the parameters in a space into three disjoint (possibly empty) sets: 1) a set of *fixed parameters* whose values have already been set, 2) a set of previously unset, but *to-be-set* parameters with a corresponding set of *candidate values* that they will be set to, and 3) a set of *open parameters* whose values are as of yet unset, and which are not currently being considered for setting. The *candidate values uniquely explain a hypothesis* about the target language, in the context of the fixed parameters, just in case the hypothesis holds true of the target language only if, given the fixed parameters, the to-be-set parameters receive the candidate values. Call a cue that leads to such a hypothesis (without contradiction from the data stream)

a *sufficient parametric cue* for the candidate values of the to-be-set parameters in the context of the fixed parameters.

Clearly, a sufficient parametric cue for the candidate values of a set of to-be-set parameters (if such things exists for a space) can be used to extend the set of fixed parameters. The cue that indicates that they should be set, by definition, would only be found in the target language if they take on the indicated values. Fixing additional parameters eliminates entire regions of the parameter space from consideration in a single step. The provision of a set of previously fixed parameters allows previous conclusions about the identity of the target grammar to provide inferential leverage on later decisions, since the candidate values of the to-be-set parameters only need be justified in the more restrictive context of the previously set parameters. A piece of evidence that would not conclusively allow a parameter to be set in the early stages of acquisition could prove quite definitive once a number of other parameters have been set. For example, in the absence of any information a particular form might be compatible with several values of a given parameter, A. Once parameter B's value is fixed, however, that form might only be compatible with a single value of parameter A. If the form were observed at that point it would be safe to set parameter A to the appropriate value, although it would not have been safe to do so at the outset.

If a sufficient parametric cue relies on a fixed context, then, crucially, a learner who makes use of such a cue must have some way of ordering its use of cues so that they do not trigger parameter setting until the appropriate context of fixed parameters has been established.

Several features of Definition 3.2.3 deserve further discussion.

First, note that the set of to-be-set parameters that can get set through appropriate use of a sufficient parametric cue will always be minimal. If parameters are included in the set of to-be-set parameters that do not actually need to be fixed in order to accommodate the data, then there is a contradiction with the requirement that the form be licensed by the grammar *only* in case all the to-be-set parameters receive their candidate values.

Second, note that such cues are only *sufficient* cues for assignment of those values that they indicate. It is possible that the cue will only be available for some, but not all, of the target languages that should fix the to-be-set parameters to those candidate values. In this case, additional cues would need to be implemented to cover all of the possibilities, and ensure learning of the other languages that should have their parameters fixed in this way. The following definition provides the terms used to describe the situation when, given a set of fixed parameters, a cue is guaranteed to work for *all* target languages that should receive a candidate assignment of parameter values to the to-be-set parameters. This is clearly the more desirable case if the aim is to make do with a small set of explicitly encoded cues. The key change is changing "only if" to "if and only if" in the second sentence.[17]

---

[17] A final complication arises in spaces where grammars generate languages that are weakly equivalent. Here a cue might be prevented from functioning parametrically, as defined, because there is a grammar generating a weak equivalent that would be inappropriately eliminated by applying the candidate assignment to the candidate extension of the fixed parameters. The definitions above should be extended so that this state of affairs does not block such evidence from counting as a parametric cue. If the learner has no principled way of choosing between weak equivalents, one is as good as any other, and it is acceptable to let a cue-based algorithm make the decision arbitrarily.

Definition 3.2.4

Divide the parameters in a space into three disjoint (possibly empty) sets: 1) a set of *fixed parameters* whose values have already been set, 2) a set of previously unset, but *to-be-set* parameters with a corresponding set of *candidate values* that they will be set to, and 3) a set of *open parameters* whose values are as of yet unset, and which are not currently being considered for setting. The *candidate values always explain a hypothesis* about the target language, in the context of the fixed parameters, just in case the hypothesis holds true of the target language if and only if, given the fixed parameters, the to-be-set parameters receive the candidate values. Call a cue that leads to such a hypothesis (without contradiction from the data stream) a *generally available parametric cue* for the candidate values of the to-be-set parameters in the context of the fixed parameters.

Note that the question of whether a candidate assignment uniquely explains a hypothesis about the target language is independent of the question of whether or not that hypothesis actually turns out to be true. An extremely ill-designed learner might form wildly false beliefs about the target grammar. More importantly, though, a learner who makes use of conjectures always stands some non-zero chance of adopting a false belief.

Third, note that in order for a learner to get off the ground using parametric cues, it is necessary that some parametric cue be able to work when the set of fixed parameters is empty. The following definition distinguishes those cues from the rest.

Definition 3.2.5

Label cues which apply with an empty set of fixed parameters *independent*.

Label cues which require a non-empty set of fixed parameters *dependent.*

It is also possible to generalize these definitions to capture the notions of two cues being independent of one another. Clearly, two cues that are independent are independent of one another.

As the discussion of exhaustive search style learning above makes plain, cue-based learning (in the broad sense) can proceed without extensive reliance on parametric cues. Such a learner rules out hypotheses a grammar at a time rather than a parameter at a time. Of course, some of the cues used to eliminate a single grammar could accidentally turn out to be parametric cues as well if they happened to require a parameter to be set to a certain value because all other grammars bearing different values of the parameter had already been eliminated.

The following definition provides for this more piecemeal approach to cue-based learning and indicates the more general effect that a cue can have on the learner's hypothesis space.

Definition 3.2.6

A hypothesis *rules out* a grammar if and only if the observation is not true of the language generated by the grammar. Call a cue that leads to such a hypothesis an *eliminator* for that grammar.

As will be seen below, *eliminators* are not, in principle, welcome components of cue-based systems for Dresher and Kaye (1990) and Dresher (1994). These systems claim to aim for a

system based on parametric cues. In fact, they are even more restrictive in that they aim to provide a single cue per parameter value.

With these definitions in place, it is possible tc provide two definitions that identify different families of cue-based algorithms—one maximally general, the other a subset of the first and potentially quite restrictive. It is an open question whether human parametric systems are compatible with some member of the more restrictive class. These definitions essentially formalize the notion of a cue-based algorithm as a sort of flow chart for parameter setting.

First, consider the most general instantiation of the notion of cue-based learning.

Definition 3.2.7

Represent an *unrestricted cue-based algorithm* as a tree whose nodes have the following properties:

i) each node has a set of grammars associated with it; call these grammars *compatible* with the node.

ii) all grammars in the finite system to be learned are compatible with the root node.

iii) the terminal nodes of the tree have sets with a single member associated with them—weak equivalents can be eliminated arbitrarily; the grammars associated with terminal nodes are the learnable languages in the space.

iv) also associated with each non-root node is an eliminator for some non-empty subset of grammars in the set of grammars compatible with the node.

v) the set of grammars compatible with a daughter node is equal to the set difference of the set of grammars compatible with the parent node and the set of grammars that the daughter node's eliminator rules out.

It should be clear how this representation translates into an algorithm. The learner begins at the root node and then proceeds to monitor the input for cues corresponding to one of the root's daughter nodes. When such a cue is found, the learner moves to the appropriate daughter node and begins once again to monitor for cues that would lead it, in similar fashion, to move further down the tree. If the learner reaches a terminal node, it has converged to its guess about the identity of the target grammar.

Note that this definition allows a node to have an arbitrary number of daughters. As an example of the possibilities that this could afford, consider a parametric space such that each grammar in the space generated at least one form that was unique to it. Each such form could serve as a cue for the elimination of all the other grammars in the space. The corresponding decision tree would be an extremely flat one.

More generally, the tree-based description of cue-based learning is useful because it makes certain aspects of the time and space requirements of the algorithm plain.

The lengths of the paths from the root node to the terminal leaves provide a lower bound on the number of processing steps that need to go on during acquisition. Of course, the time required to gather the evidence that contributes to a particular cue could vary greatly. Some grammars might be eliminated, or parameters cued, after the observation of a single easily observable property of the input. Others might require the learner to amass a wide collection of forms from the input stream, or to observe a long sequence of forms to ensure that none of them have a particular property.

The various annotations on the nodes of the tree need to receive a mental representation, and, therefore, occupy memory. On the cue-based approach, UG explicitly

provides the set of cues that the learner will need to acquire any learnable languages in a parameter space. Moreover, at any step in the acquisition process the learner must somehow explicitly represent its hypothesis. This last requirement may or may not prove onerous. For example, an enumerative learner like Wu's, which generates its enumeration on the fly, has a fairly simple representational requirement.

The following definition picks out a more restricted class of cue-based algorithms that must set a parameter at a time, rather than just eliminate a grammar at a time.[18]

Definition 3.2.8

Represent a *parametric cue-based algorithm* as a tree whose nodes have the following properties:

i) each node has a set of fixed parameters associated with it; call this background context *compatible* with the node.

ii) the background context compatible with the root node is null.

iii) the terminal nodes of the tree have all parameters fixed and each learnable natural language is represented by a terminal node—weak equivalents can be eliminated arbitrarily via a suitable vacuous conjecture.

---

[18] Actually, as things are developed here, the class of parametric cue-based algorithms is not any more restrictive than the class of cue-based algorithms with respect to the class of languages that it can learn. Intuition would suggest that the parametric cue-based learner is more restricted because it can only employ cues that set a parameter at a time, not cues that simply eliminate a grammar at a time. However, any potential restrictiveness is voided by the lack of restriction on possible cues. To see this, consider that a parametric cue-based algorithm's cues could actually internally simulate the workings of an enumerative search. If there is to be a distinction in the classes of languages that the two families can learn, there needs to be some bound on the amount of time and input that it takes to perform a parametric cue-based learning step.

iv) also associated with each non-root node is a set of to-be-fixed parameters, the corresponding set of values that they are to be set to, and an encoded parametric cue that would license the learning algorithm to make corresponding extension from the set of fixed parameters of the parent node to the set of fixed parameters of the daughter node.

Translating this representation into an algorithm is also quite clear.

The intuition behind these algorithms is that after successful piecemeal or parametric elimination of candidate grammars, the learner would be left with a single grammar that generates the target language. If the learner relies on conjecture, this situation cannot be entirely guaranteed even if an algorithm with the appropriate structure is provided. Proponents of cue-based algorithms that, by relying on conjecture, make use of indirect negative evidence can only hope that the systems conjectures will be correct with high probability, and, therefore, lead the algorithm to correctly converge with high probability. Ensuring high reliability of cues based on indirect negative evidence could potentially require the learner to process a considerable number of inputs. This would be the case if the property of the data stream that the learner was monitoring for would be expected to occur infrequently in languages that manifested it. If the learner never resorts to conjecture, then a correctly specified cue-based algorithm will satisfy the requirements of the identification in the limit paradigm. All of the eliminations, piecemeal or parametric, that a learner performs are guaranteed to lead the learner along a path to a suitable target. If a learner resorts to conjecture, then, the learner could make an unwarranted step. A learner who took such an unwarranted step might, in fact, never reach a leaf of the decision tree. Further cues on the incorrect branch could prove

inapplicable. Alternatively, a learner might reach a leaf corresponding to a grammar that did not generate the same set of forms as the target grammar.

With some empirical estimate of the distribution of forms in texts from target languages, it is straightforward to estimate the probability that a learner will pursue a particular branch of the decision tree. Since a learner only ever pursues one branch, the probability of successful acquisition of a target is simply the sum of the probabilities of all paths that lead from the root to a leaf that generates the same language as the target grammar.

A learner who relies only on positive evidence can, in the limit at least, expect to be driven to the target grammar. A learner who relies on the absence of a particular form or property from the data stream can be misled if it does not wait long enough. If the particular form or property can be expected to occur, on average, every n inputs, then it is possible for the learner to wait long enough to build up an arbitrarily high level of confidence in the belief that the form or property will never occur.

3.3 Dresher and Kaye's design philosophy

With a couple of additional wrinkles, Dresher and Kaye (1990) and Dresher (1994) essentially advocate a version of the parametric, cue-based learner that I have sketched above. They provide the following list of properties that they believe hold true of algorithms for natural language parameter setting:

(3.3.1)

A. UG associates every parameter with a cue.

B. A cue is not an input sentence or form but is something that can be derived from input.

C. Cues must be appropriate to their parameters.

D. What the correct cue to any given parameter is must be empirically determined (by the linguist not the learner, to whom it is supplied by UG). There is no parameter-independent general algorithm for parameter setting.

E. Parameter setting proceeds in a (partial) order set by UG: this ordering reflects dependencies among cues, and specifies a learning path.

F. A parameter which has a default state remains in it until the learner detects its cue, which acts as the trigger to move to the marked setting.

G. The learning strategy is loosely speaking 'deterministic', in the sense that the learner may not backtrack or undo parameter settings that have already been set.

H. Determinism does not hold in the following case: when a parameter is set to a new value, all parameters which depend upon it (follow it in the order) revert to default.

I. Cues are local in the sense that each decision depends on finding a specific configuration in the input, and acts on this without regard to the final result. Hence, learners are not trying to match the input.

J. Cues become increasingly abstract and grammar-internal the further along the learning path they are.

My aim here, is to briefly dissect this list of proposed properties of the human language acquisition device in terms of the vocabulary developed in 3.2 with an eye towards both developing a picture of the restrictiveness of the proposal and a means of evaluating how closely our concrete examples of cue-based learners adhere to these guidelines.

Property A reflects the belief that it will be possible to develop a parametric cue-based algorithm as opposed to simply an unrestricted cue-based algorithm—which can be trivially constructed wherever enumeration succeeds. As will be seen in 3.4, this property is really a key

to ensuring reasonable resource usage for the parametric acquisition of large spaces. Moreover, it would appear that the claim is being made that the cues that are encoded in UG are generally available, rather than just sufficient. Every parameter has *a* cue.

Property B's motivation seems to be a belief that the parameters in a system are likely to be highly interactive. Several potentially useful notions of interaction are developed in Definition 3.3.2.

Definition 3.3.2

A subset of parameters fixed to certain values will *interact* to generate data that license a hypothesis if and only if, with no other parameters fixed, the corresponding hypothesis is uniquely (not necessarily, always) explained by the parameter assignment corresponding to the fixed values of the parameters in the designated subset. Designate the *degree of interactivity* of a hypothesis as the size of the non-empty set of interacting parameters. (If the set of interacting parameters is empty, then the data have a *disjunctive explanation.*)

These definitions are straightforwardly generalized to capture the notion of parameter interaction against the background of a set of previously established parameter values.

If the space tends to have a *high degree of interactivity* with respect to form observations then a parametric cue-based learner that relied primarily on form observations could require many highly specialized, encoded cues, all of which would need to be made available in UG. Moreover many particular forms that could potentially serve as cues and that, collectively, would be quite common in natural language data sets, could, individually, prove, rather sparse. A learner might have to represent a large number of forms to look for,

instead of relying on individually infrequent forms. In this case, the burden of supplementing UG with enough cues to ensure a reasonable time course of acquisition could prove rather problematic. Of course, if cues could be made available to the learner at no cost, high interactivity is actually a boon for a learner because it allows for many parameters to be set in one step. We will see in Chapter 6 that OPL learners potentially exploit this possibility. An important question regarding such learners, of course, is whether the relevant cues are really to be had, if not for free then at a reasonable cost.

On the other hand, it is possible that parameters do not interact to generate any of the forms in the space. All the forms in the space could turn out to be generated by odd disjunctions of parameters. Suppose, for example that the sentence "John saw Bill" were possible if and only if parameter 1 received value A or parameter 7 received value B. In this case, then, no particular parameter needs to be set to a particular value in order to generate "John saw Bill". If most forms in the parameter space had a disjunctive explanation, a parametric cue-based learner who relied on form observations could be in trouble. Forms that have a disjunctive explanation do not require any parameter values to be set in a particular way, so the learner cannot use them as evidence for setting a parameter.

Dresher and Kaye's (1990) and Dresher's (1994) motivation for focusing on property observations and property-absence conjectures is a belief that this focus on more abstract properties of the input will lead to the discovery of easily encoded cues that (in the context of a fixed background of set parameters) are not plagued by high interactivity and that can be expected to occur in the input in a reasonable time span.

Property C is primarily an aesthetic criterion from the point of view of questions about the time and space that the learner needs to be devote to language acquisition. It is not really

easy to evaluate the meaning of this requirement, which is intended to prevent cues that are linguistically "unnatural" in some way or other.

Property D is a simple assertion of the need for a cue-based approach and a rejection of alternative approaches, for example those based on IGMS algorithms.

Property E is in keeping with the tree-based representation of cue-based algorithms developed in 3.3.

Property I reemphasizes the claim made in B that Dresher and Kaye intend to limit themselves to observing properties of forms rather than forms.

Property J reflects the belief that there will be a high degree of dependence among cues. Most of them will only operate successfully against a fixed background of previously set parameters. Dresher and Kaye (1990) and Dresher (1994) aim to accommodate this dependence by stating abstract cues that can be concretely instantiated once certain parameters have been fixed.

Properties F, G and H represent a slight departure from my formulation. This limited backtracking is not exercised in the Dresher (1994) paper that will be our focus here. The formalism I have developed here could be easily extended to handle this possibility without significantly impacting the desirable properties of the parametric cue-based learner that I turn to now.

3.4 Cue-based Learning in Large Parametric Spaces

We have already seen that an unrestricted cue-based learner can effectively mimic an enumerative learner. As a result we know that an entirely unrestricted approach will not do for large parametric spaces.

Fortunately, from the point of view of coping with large parametric spaces, parametric cue-based algorithms offer interesting possibilities that, in appropriate parameter spaces, would guarantee that acquisition takes a tractable amount of time. In fact, the parametric cue-based learner implements a design paradigm that is textbook computer science: divide-and-conquer. At each step that a cue is applied to fix a parameter, the learner reduces the problem that they are currently facing—finding the target grammar in a space of m candidates—to a problem that is $1/n$th that size—where n is the number of parameter values for the parameter that gets fixed. If more than one parameter gets fixed in a single step, the reduction in the size of the problem space is even greater.

Figure 3.4.1 provides a schematic view of the reduction of the size of the problem space in a system with binary parameters. At the root node, all grammars in the space are still possible. At the first level down, a parameter has been fixed one way or the other, and, as a result, half of the grammars in the space have been eliminated. Another level down, and the space is halved again.

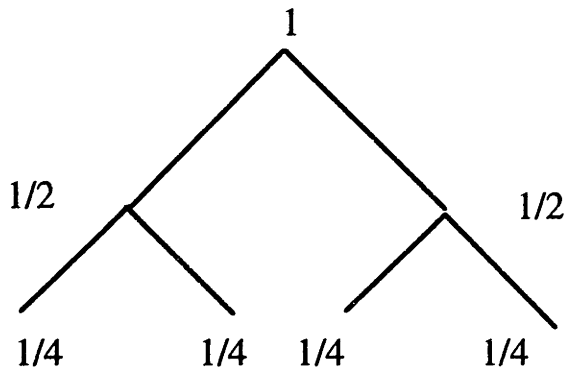


Figure 3.4.1

In this example, since each cue rules out half of the candidate grammars, the number of cues utilized along a learning path *en route* to a target grammar at the terminals will be (proportional to) $\log_2 2^n$—in other words, n. A successful learning path requires a number of cues that is linear in the number of parameters. The number of parameters would need to be quite enormous for this requirement, in and of itself, to overwhelm the learner.

It is easily shown (see for example, Cormen, Leiserson and Rivest 1990) that any divide-and-conquer scheme that reduces the problem space at each step by at least some fixed non-zero fraction will lead to a scheme that takes at worst a number of steps that is logarithmically proportional to the size of an exponentially growing initial problem space. Therefore, the number of steps required will be linear in the number of parameters to be set. (Of course, the constant of proportionality could prove large enough to become somewhat daunting. Eliminating a millionth of the remaining parameter space at a time would not be particularly useful. Again, as discussed in length in Berwick and Weinberg (1984), there is need for some

caution in applying this type of reasoning to the case of natural language learning since the actual parameter space is fixed in size.)

Along these lines, it would also be acceptable, although slower in the absolute sense, for a learner with multi-valued parameters to reject a single value of a parameter at each step. To recognize the tractability of this approach, notice that since there is a maximum finite number of values, v, that a parameter in the space can have, the fraction of the space eliminated every time a parameter value is eliminated is at least $1/v$.

On the other hand, a learner that is only required to reject certain *combinations* of parameter values at each step without definitively setting the values of any parameter does not necessarily maintain the desired property of guaranteeing that fixed fraction of the candidate grammars, independent of the size of the space, get eliminated at any step. In fact, this is exactly what the unrestricted parametric learner of Section 3.3 does. A breakdown in the ability to satisfy the one parameter per cue (or the equally satisfactory, but weaker "eliminate one parameter value" per cue) assumption, thus, leaves the learner potentially unable to guarantee success in exponentially growing parametric spaces.

The learner needs to be able eliminate large regions of the parameter space in a single step in order for cue-based acquisition to work in large parametric spaces. Unfortunately, while ensuring that each learning step eliminates a fixed fraction of the remaining candidate grammars does stave off intractable growth in the number of learning steps that the learner needs to devote to language acquisition, it does not guarantee complete avoidance of the "exponential" problems inherent in large spaces. There could still be "exponential" growth in the contents of UG.

Clearly, the cue-based approach as developed by Dresher and Kaye (1990) and Dresher (1994) explicitly requires UG to provide a learner with all the cues that are *actually* used in learning. It also, however, requires UG to provide the learner with all the cues that are *potentially* used because the learner enters the world not knowing what grammar it will eventually have to acquire. All of these cues must be encoded somehow in the instruction manual for UG. At the very least, there is one cue for each terminal in the tree. In a space with 2^n grammars, then, the learner might potentially have to store 2^n distinct cues just to make the final parameter settings for target languages. Moreover, the cues at each decision point in the tree structure could turn out to be distinct. This unhappy state of affairs is represented in Figure 3.4.2. In fact, every language might require an entirely different chain of cues along its learning path.[19] If the learner has pursued the leftmost branch of the decision tree in Figure 3.4.2, then Cue A will be used to make the first parametric decision, while Cue C gets used to make the second parametric decision. If, instead, the learner had pursued the rightmost branch in the decision tree, then Cues B and F would be deployed. A fully binary tree will contain $2^{n+1} - 1$ nodes.

[19] Moreover, nothing prevents the possibility of target grammars having multiple terminal nodes.

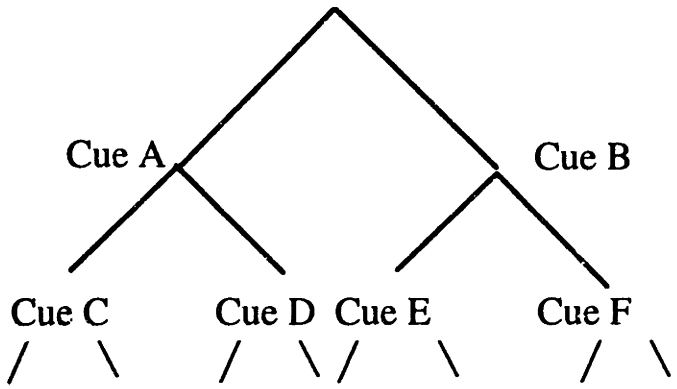


Figure 3.4.2

Here, the learner has bought a time course of acquisition that depends only linearly on the number of parameters and their values at the cost of a storage requirement that is of the same order of the problem space—an unacceptable result given our ground rule about avoiding exponential dependence on the number of parameters in the system.

In the best case, however, it would be possible to, in some sense, share cues throughout the tree structure and achieve a more compact representation. For example, if a single cue could be assigned to each level in the decision tree, it would be possible to ensure that the number of cues required was bounded by the product of the number of parameters and the maximum number of values that any particular parameter could take on. If each cue were able to effectively fix the value of one parameter, then it would take n cues to set n parameters. This possibility is represented in Figure 3.4.3, although it is necessary to give the cues here some further interpretation.

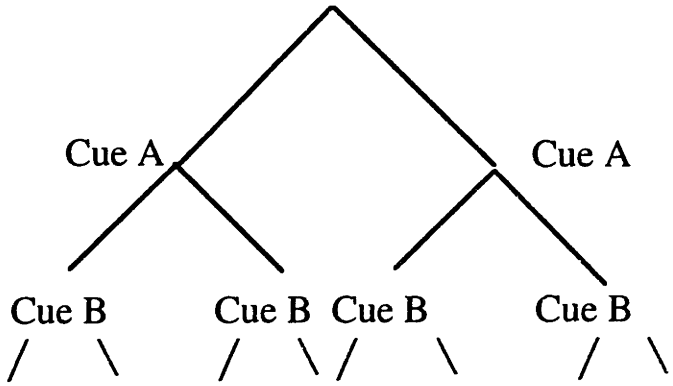


Figure 3.4.3

There are at least two possible ways to make sense of the identical cues in the tree structure. First, and most simply, the space might be such that it is possible to provide cues that show little if any dependence on the background context of fixed parameters. Second, a set of *meta-cues* might be cleverly devised. The idea behind what I am calling the meta-cue approach is to provide a set of abstract cues that, given a background context of fixed parameters, can be used to quickly compute a cue for parameter setting that will work in the context of that background. As always, whether or not this can be done is an open empirical question. Discussion of Dresher's (1994) system will provide some examples of the meta-cue approach.

3.5 An Example System of Phonological Parameters:

Metrical systems in phonology provide an attractive testing ground for parametric learning schemes because quite sizable systems of interacting parameters have been proposed and these systems tend to provide a relatively broad coverage of natural language data in a diverse set of languages. Idsardi (1992) and Halle and Idsardi (1994) develop the system that Dresher (1994) analyzes from the point of view of cue-based learning. There are a number of alternative systems with comparable coverage of this phonological domain, but our focus here will be squarely on this particular variant. I will further analyze this system in Chapters 4 and 5 from the point of view of the TLA. The system provides a parametrized algorithm that maps from a string of syllabified phonemic material to a metrical structure and stress assignment. In all the learning scenarios that I will consider, the learner is assumed to have direct perceptual access to the string of syllabified phonemic material and the stress contour, but cannot directly perceive metrical groupings. This assumption, of course, will eventually need to be justified if any of these approaches are to have a hope of succeeding.

Pedagogically, the language Tübatulabal provides a convenient and rapid introduction to the basic workings of the system since it exercises most of the parametrically available machinery that the system allows. In this section, I will present a version of the system that corresponds to the one that I have implemented in the computer language LISP for analysis of Gibson and Wexler's TLA. Dresher's analysis of the Halle and Idsardi system, unlike Dresher and Kaye's (1990) analysis of an alternative system of metrical parameters does not make use of an explicit implementation; instead, it is based on a series of deductive arguments about the

interaction of the parameters as described in the literature. The fragment that I have implemented is closest to the presentation in Halle and Idsardi (1994), with an important difference motivated by a proposal by Halle (p.c.) for a possible extension of the system tentatively suggested by some facts from Ojibwa. Interestingly, this extension turns out to have dramatic, and unfortunate, consequences for the TLA. This will be discussed in more detail in Chapter 4. Dresher makes some further modifications to the system that I will present in the course of consideration of his cue-based learner for the space.

For simplicity, I begin with the machinery used in generating a single level of stress. L's indicate light syllables. H's indicate heavy syllables. The # symbol indicates word boundaries. Several tiers, or lines, of metrical structure are built "on top" of the string of syllables. The elements on these tiers, or grid marks, are indicated by asterisks. The height of the column of asterisks above a syllable indicates its stress level.

(3.5.1) Halle and Idsardi (1994), as exemplified by Tübatulabal.

a. Initial syllabification and projection to Line 0.

```
# *   *   *   *   *   *  # Line 0
# L   L   L   H   L   L  #
```

b. Quantity Sensitivity: "Project Heavy Syllable".

```
# *   *   * [ *   *   *  # Line 0
# L   L   L   H   L   L  #
```

c. Edge Marking.

```
# *   *   * [ *   * [ *  # Line 0
# L   L   L   H   L   L  #
```

d. Iterative Constituent Construction:
Finish <---------- Start

```
# * [ *   * [ *   * [ * [ # Line 0
# L   L   L   H   L   L  #
```

e. Project Constituent Heads

```
#     *       *       *  # Line 1
# * [ *   * [ *   * [ *[ # Line 0
# L   L   L   H   L   L  #
```

Initially, as shown in (3.5.1.a), all syllables are projected to a tier called Line 0 to indicate that they are potential stress-bearers. The asterisks that appear above them reflect their stress-bearing status. Next, several parametrically regulated pieces of metrical-construction machinery have an opportunity to add brackets to Line 0. These brackets serve to group these grid marks into constituents. The metrical constituents that this system forms are somewhat idiosyncratic. Unlike in more "standard" systems where the elements of a constituent are grouped together by a matching left and right bracket, here a single bracket suffices to form

a constituent. Constituents can be read off the representations in (3.5.1) as follows: All the material on the inside of a bracket—to the right of a left bracket and to the left of a right bracket—up to another bracket or a word boundary form a constituent. For example, in (3.5.1.b) the first three grid marks are initially not grouped into any constituent, while the last three syllables are grouped into a constituent by the left bracket to the left of the grid mark associated with the heavy syllable.

The first bracket-inserting process, which inserted the bracket in (3.5.1.b), places a bracket to the left of the grid mark associated with a heavy syllable. Brackets inserted by this process are always oriented so that they group the heavy syllable into a constituent. Parametrically, languages can choose to place no bracket at all, to place a bracket to the left of the grid mark associated with the heavy syllable, or to place a grid mark to the right of the heavy syllable. Tübatulabal chooses to place a left bracket. In my implementation, I have only represented one notion of heavy syllable. In reality, there is cross-linguistic variation in terms of whether syllables behave as light or heavy. The rough generalization is that in all languages sensitive to syllable weight, a syllable with a long vowel counts as heavy. Languages, however, can also opt to allow closed syllables—syllables with a consonant in their coda—to count as heavy even if the vowel is not long.

In (3.5.1.c), a bracket associated with the edge of the word is put into place. Languages do not have to exercise an edge marking option, but if they do, they can choose to place a left or a right bracket to the left or the right of the leftmost or rightmost grid mark. Tübatulabal places a left bracket to the left of the rightmost grid mark.

In (3.5.1.d), the final bracket insertion process applies. This somewhat more complicated device iterates through the structure, from one end or the other, placing brackets in

any position that will not create a new single syllable constituent between the current position and the starting position. That is to say, the process can look back at the portion of the representation that it has already traversed, but not ahead to the portion of the representation that is yet to come. A typical, but not necessary, effect of this process is to create new binary constituents. So, I will sometimes refer to it as binary constituent construction, rather than the more technically correct name for it—iterative constituent construction. In the version of the system that I will consider, languages can choose to start from either the left or the right of the word, and to place either left or right brackets (if, of course, they adopt iterative constituent construction at all). Tübatulabal proceeds from right to left, placing left brackets. The variants that I'll call the forward grouping iterative possibilities—moving from left to right with right brackets and from right to left with left brackets—are an extension of the system that does not appear in Halle and Idsardi (1994). They are motivated by general considerations of symmetry and a tentative analysis of Ojibwa (Halle p.c.). In the unextended system, the direction of motion in the process of bracket placement is correlated with the orientation of the bracket; only backward grouping insertions were allowed. The system presented by Idsardi (1992) and Halle and Idsardi (1994) also allows for parametrization of the size of iterative constituents to accommodate languages such as Cayuvava which seem to require ternary constituents. This parametric variation is not captured in my implementation.

Finally, once constituents have been formed, one member of each constituent is projected—an additional grid mark is placed above it on Line 1. Languages can choose to select either the leftmost or the rightmost member of a constituent. Tübatulabal chooses to project the leftmost. If such marks remain, they will be interpreted as secondary stresses.

Many languages make a distinction between secondary and primary stress. With the exception of the bracket inserting process that places brackets to the left or right of Line 0 grid marks corresponding to heavy syllables, independently parametrized versions of the bracket inserting processes and head projecting machinery used to build Line 1 from Line 0 are available for the construction of and additional Line 2 on top of Line 1. Languages may impose some additional restrictions on the operation of constituent formation on top of Line 1, since the typical result in the world's languages seems to be a single main stress. This is not discussed in great detail in Idsardi (1992) or Halle and Idsardi (1994),

In some cases, the system requires the machinery used to generate two levels of stress even though only a stressed/unstressed distinction appears on the surface. This phenomenon is captured by allowing for an operation of conflation that essentially eliminates Line 1. For example, stress placement in languages with a single antepenultimate main stress and no secondary stresses can be analyzed in terms of conflation, as shown in (3.5.2).

(3.5.2) Example of the use of conflation:

a. Pre-conflation structure

```
#                *]          # Line 2
#      *         *]          # Line 1
# * [  *    * [  *    * ] *  # Line 0
# S    S    S    S    S    S  #
```

b. Post-conflation gridmarks

```
#                *           #
# *    *    *    *    *    *  #
# S    S    S    S    S    S  #
```

The grammar that generated the example in 3.5.2 is not sensitive to syllable weight, so I have represented syllables simply as "S". In the constituent construction on Line 0, this language places a right bracket to the left of its rightmost syllable. The placement of this bracket has the effect of making the final syllable "extrametrical"; it is not grouped into a constituent. The first constituent that the leftward moving process of iterative constituent construction forms contains the penultimate and the antepenultimate syllable. The language that generated the structure in (3.5.2) is left-headed on Line 0, so the leftmost member of the final constituent—that is to say the antepenultimate syllable has a grid mark projected to Line 1. So too, however, is the grid mark of the second syllable. In order to capture the fact that it is the antepenultimate syllable that actually bears main stress, some additional process of selection needs to occur. Within the context of the system, the selection could be implemented in a number of ways. Here a right bracket is placed to the right of the rightmost syllable on right-headed Line 1. Since the language description calls for no secondary stresses, Line 1 must be suppressed. It is possible to imagine several ways of implementing this.

As mentioned above, the typical interpretation of these bracketed grid structures is that while stress level has some perceptual correlates, the bracket structure itself can only be viewed indirectly through their effect on phonological processes.

The discussion above presents the core components of the parametric system as they apply to the derivation of metrical structure for monomorphemic words. The system, with no further parametric extension, is also capable of deriving metrical structure for multimorphemic words. This part of the system is not modeled in the subsystems discussed in this thesis.

I will, however, discuss some additional issues that are involved in extending learning analyses to data sets that are closer to the "real thing". As always seems to be the case, the "real thing" is not so simple. In addition to this core parametric system, Halle and Idsardi also rely on an additional set of less systematized processes that affect bracket insertion to cover the extensive range of cross-linguistic facts that their system handles. For example the analyses of several languages deploy rules that are language-specific, but generally applicable within a language. These rules insert or delete brackets in specified contexts. I take it that, in keeping with the Principles and Parameters approach, the eventual hope here would be to provide a circumscribed, parametrized set of these rules. Another set of bracket-adjustment processes, however, originates in the lexicon. This does not appear to be entirely amenable to parametrization. In order to capture lexical exceptions to stress generalizations, words are allowed to emerge from the lexicon with some pre-specified information about the bracketing of Line 0. These "extra-parametric" options will be discussed in more detail in Section 3.7, when I turn to questions about the extendibility of cue-based analyses.

3.6 An Instruction Manual for Acquiring the Phonological Fragment in Dresher (1994)

The language Selkup provides the running example for much of Dresher's (1994) sketch of a cue-based acquisition algorithm for a system modeled closely after Halle and Idsardi (1994) and Idsardi (1992). Dresher also discusses the learning paths for several other languages. Of course, as Dresher clearly recognizes, for a complete cue-based algorithm for a system, it is really necessary to eventually provide learning paths that cover *all* attainable targets in the space. In this section, I will say what I can to fill in the necessary learning paths.

In Selkup, stress falls on the rightmost long vowel; if there is no long vowel stress falls on the initial heavy syllable. The analysis of Selkup that Halle and Idsardi provide is summarized in 3.6.1.

(3.6.1) Line 0:
Quantity Sensitivity: left bracket to the left of syllables with long vowels.
Edge Marking: left bracket to the left of the leftmost syllable.
Iterative Constituent Construction: none.
Headedness: left.

Line 1:
Edge Marking: right bracket to the right of the rightmost syllable.
Headedness: right.

Conflation: yes.

The Line 0 parameters ensure that the initial syllable and all heavy syllables project grid marks to Line 1. The Line 1 parameters pick out the rightmost of these Line 1 grid marks and promotes it to Line 2; if there is no heavy syllable, the initial syllable turns out to the rightmost Line 1 grid mark. The setting of the conflation parameter suppresses any secondary stresses.

The chain of encoded cues that lead Dresher's learner to the acquisition of Selkup is shown in (3.6.2). The input data are assumed throughout to be monomorphemic forms drawn from the target grammar. Following Dresher, I will be making the assumption that there are no restrictions on the makeup of the strings of light and heavy that form the input to the structure building/stress assignment algorithm. If this turns out to be false, it will be necessary to make sure that the restrictions do not invalidate any of the system's cues.

(3.6.2) Summary of learning path (results for Selkup)

a.	Look for:	Secondary stress.
	Result:	Fail.
	Conclude:	There is only one stress per word.
b.	Look for:	Identical word types with conflicting stress contours.
	Result:	Succeed.
	Conclude:	Stress is sensitive to quantity (distinguishes H and L syllables), closed syllables are classified as H.
c.	Look for:	Identical word types with conflicting stress contours.
	Result:	Succeed.
	Conclude:	Closed syllables with short vowels are represented as L.
d.	Look for:	Stressed syllable at the left and right edges.
	Result:	Succeed.

	Conclude:	There is no extrametrical syllable on the left or right, i.e. no # *[or]*# on Line 0.
e.	Look for:	Stressed nonedge L always adjacent to H.
	Result:	Fail.
	Conclude:	There is no pre- or post-accenting, i.e. no *[or]* on Line 0.
f.	Look for:	Stressed nonedge L.
	Result:	Fail.
	Conclude:	There is no construction of bounded constituents on Line 0.
g.	Look for:	Main stress in a constituent-sized span at either edge.
	Result:	Succeed on the right.
	Conclude:	Main stress (* on line 2) is assigned to the head of the rightmost Line 0 constituent.
h.	Look for:	Position of stress in words with only light syllables.
	Result:	Succeed on the left.
	Conclude:	Line 0 constituents are left-headed; heavy syllables project (x on Line 0.
j.	Look for:	Homogeneous settings on Line 0.
	Result:	Succeed.
	Conclude:	The left edge begins a Line 0 constituent: #[*.

As noted above, although the focus is on Selkup and several other languages, Dresher's clear intent is that the fully specified algorithm will succeed for *all* grammars in the space. This section will review the developments in Dresher (1994), and, where necessary, fill in the details to construct an algorithm that will acquire all languages in the space. As a perusal of the cues for Selkup will indicate, Dresher's modification of the Halle and Idsardi system actually covers a slightly different range of phenomena. I will draw attention to these differences as I proceed.

I will begin this development by presenting the learning path for Selkup. As I go, I will indicate the dependence of later cues on earlier setting of parameters, and point out portions of the decision tree that will need to be filled out. This filling out will occur after discussion of Selkup is complete.

Step A: Conflation

The first cue in the learning path is, of necessity, independent. It must be applicable even though no other parameters have been fixed. In particular, this cue—the absence of forms with secondary stresses in the observed data stream—licenses a property-absence conjecture. If Dresher's learner does not hear forms with secondary stresses, it eventually assumes that the target grammar does not generate forms with secondary stresses.[20] (Although the learner is required to wait an adequate amount of time before drawing the conclusion that the language it is learning does not contain forms with secondary stress, it need not hold detailed information about the forms that it has seen in memory; a simple register or counter will suffice.) The flip

---

[20] Dresher does not attempt to fully specify many of his cues. For example, he provides no indication of how long a learner should wait before conjecturing the absence of secondary stresses. At this point this seems like a reasonable omission since it seems likely that secondary stresses will occur frequently for those grammars that produce them. It would not be reasonable to expect the learner to use indirect negative evidence based cues for data that was expected to be infrequent. In pre-Principles and Parameters learnability work, Wexler and Culicover (1980) were able to establish a *boundedness* condition on transformations that made in-the-limit acquisition of the class of transformational grammars that they studied possible for a learner exposed only to degree-2 data. They were, therefore, able to rule out the need for their learner to make use of more complex structures.

Estimating "cue durations" would involve extensive empirical consideration of the distribution of particular phonological patterns in a wide variety of languages. Clearly, this omission would have to be filled in for a complete theory.

side of this cue, of course, would be the observation of a form with a secondary stress, which would lead to the corresponding property observation and rejection of conflation. In the absence of noise, a learner who rejects conflation can be sure of its choice—it is driven by positive evidence. A learner who selects conflation, on the other hand, however, does so on the basis of an inductive inference; it could turn out to be wrong. In this particular case, however, it seems likely that languages that make a distinction between primary and secondary stress would provide ample evidence for the distinction.

Here, I simply make the logic behind this cue more explicit than it is in Dresher (1994). Dresher's presentation is made in terms of default settings and cues for parameter change, but this is easily translated into the scheme of observations and conjectures that I have developed. Assume that the default setting for the parameter controlling conflation is +conflate; the learner initially assumes that any secondary stresses that get generated in the course of a derivation get "wiped out" in the phonetic output. Now, the learner will fail to acquire the target if: (a) the target grammar requires –conflate, and whatever cue that the learner uses to switch to the –conflate value doesn't get triggered by the input data, or (b) the learner mistakenly abandons the default setting. The learner will never fail if they use the presence of secondary stresses to motivate a switch to –conflate. Clearly, the presence of secondary stresses indicates that secondary stresses have not been eliminated. The only problem that could potentially arise is that a grammar could have no secondary stresses on the surface and yet require –conflate. This, however, is an impossible situation. The space of languages can be partitioned into those languages that generate more than one mark on Line 1 in the course of some derivation and those that don't. If a language generates more than one mark on Line 1 in the course of some derivation, but no secondary stresses appear on the surface, conflation *must*

apply to remove the "extra" marks. If, on the other hand, a language never generates more than one mark on Line 1 in the course of a derivation, it's safe to apply conflation. The projection from Line 1 to Line 2 is vacuous.

Selkup conflates.

Steps B and C: Quantity Sensitivity

Dresher orders the cue for conflation before the cues governing quantity sensitivity. However, there is no real reason to set the learner's conflation parameter before determining whether or not the learner is quantity sensitive (and, if they are, what counts as a heavy syllable). As will be seen directly, these decisions can also be made independently—that is, they can be made with no fixed parameters. Dresher claims that the learner figures out whether or not their language is quantity sensitive by accumulating information about whether words with the same number of syllables consistently have the same stress pattern. (Again, it seems likely that this type of evidence would be readily available.) If they do, then it is safe to conclude that whether or not a syllable receives stress is wholly a function of position within the word, not of weight. If not, something else must be causing a difference. The only *something else* that this system can make use of is syllable weight, so the discovery of any such differences points to quantity sensitivity. To put it in other words, if syllable weight never has a measurable impact on the system, it is safe for the learner to conclude that syllable weight has no impact on the system.

Note that it is possible to construct a grammar that is quantity sensitive in its parameter settings, but which does not exhibit quantity sensitivity overtly. For example

consider a grammar with invariant initial stress that enforces this initial stress by placing a right bracket to the right of the leftmost Line 0 grid mark, as in 3.6.3.

(3.6.3) Example of the use of conflation:

a. Pre-conflation structure

```
# *   ?   ?   ?   ?   ?  # Line 1
# * ] *   *   *   *   *  # Line 0
# S   S   S   S   S   S  #
```

Any other grid marks that get placed on Line 1, including those attributable to constituent structure reflecting quantity sensitivity, can be suppressed by promoting the first Line 1 grid mark to Line 2 and then conflating. The components of the metrical structure that get erected as a reflex of quantity sensitivity turn out to be irrelevant in the stress contour that ultimately emerges. As a result, there are a number of weakly equivalent grammars that generate this stress-initial language. In all cases, it is safe to choose one that is not quantity sensitive.[21]

As mentioned above, there are at least two possible instantiations of the notion of quantity sensitivity, Dresher proposes a second round of consistency checking to fix the value of this parameter appropriately. The learner assumes that short-vowelled syllables that are closed by a consonant count as light. In other words short-vowelled syllables that are closed by a consonant and short-vowelled syllables that are not should be indistinguishable in their

[21] The possibility presented in (3.6.3) could become of interest in a more enriched theory of the data available to the learner. Other phonological processes sensitive to the metrical constituency constructed by the system could provide the learner with additional information about the bracketing of the structure.

contributions to the stress contour. If this is not the case, then short-vowelled syllables that are closed by a consonant need to be counted as heavy.

There is no principled reason that some version of these cues could not be monitored for in parallel. The first set of cues—one for quantity sensitivity based on positive evidence, its "flip side" based on indirect negative evidence—that determines whether or not a language is quantity-sensitive at all comes before the second set of cues which fixes the boundary between light and heavy syllables. So clearly, they can be deployed independently of the second. The second set of cues requires the learner to assume one of the two possible groupings of syllable types into light and heavy syllables and see if forms that this grouping predicts should be identically stressed do, in fact, receive the same stress. There is no reason this test could not be run at the same time as the first. Moreover, there is no reason that these cues need to wait for the conflation parameter to be set. In other words, these three sets of cues are independent.

As with the conflation parameter, there are asymmetries in the nature of the evidence used to set the quantity sensitivity parameters, a learner who selects for quantity sensitivity does so because they have formed a guaranteed property observation. The learner who rules out quantity sensitivity relies on indirect negative evidence. Similarly, the learner who selects the more restrictive definition of heavy syllables does so on the basis of indirect negative evidence; the learner who selects the less restrictive definition of heavy syllables does so because they have directly observed that the more restrictive definition leads to conflicting stress contour assignments. Unlike the case of the conflation parameter, however, here the learner can't make do with a simple register and really must store more explicit information about the forms that it has seen. Conceptually, the first test requires the learner to maintain a table of stress patterns with separate cells for words of different lengths. The first test requires the learner to

maintain a table of stress patterns with separate cells for the various strings of light and heavy syllables of different length. This may or may not turn out to be psychologically acceptable.

Selkup is quantity sensitive and classifies closed syllables as H.

Step D: Rule out the left-to-the-right-of-the-leftmost and the right-to-the-left-of-the rightmost edge marks

Here, I believe Dresher slightly misrepresents the Halle and Idsardi system. The notion of extrametricality, the property of not being included in a metrical constituent, plays no formal role in the Halle and Idsardi system. For example, no syllable is ever directly labeled extrametrical, as it might be in other theories. Rather, certain patterns of bracket insertion have the effect of preventing certain syllables from being grouped into any metrical constituent. In monomorphemic words, for example, placing a left bracket to the right of the leftmost syllable, as in (3.6.4.a), or a right bracket to the left of the rightmost syllable, as in (3.6.4.b) *can* have the effect of creating an ungrouped syllable at the beginning or end of the word, respectively.

(3.6.4) "Extrametricality" in Halle and Idsardi

a. Left "extrametricality"

```
# * [ *   *   *   *   *  # Line 0
# L   S   S   S   S   S  #
```

a. Right "extrametricality"

```
# *   *   *   *   * ] *  # Line 0
# S   S   S   S   S   L  #
```

Note, however, that if the language is quantity sensitive, these particular edge mark options no longer ensure "extrametricality". In quantity sensitive languages, heavy syllables generate their own brackets in way that guarantees that they are included in a constituent, as shown in (3.6.5)—the bracket due to quantity sensitivity is indicated in bold in each case.

(3.6.5) "Extrametricality" and its interaction with quantity sensitivity in Halle and Idsardi

a. Left "extrametricality" with quantity sensitivity

```
# [*[ *   *   *   *   *  # Line 0
# H   S   S   S   S   S  #
```

a. Right "extrametricality" with quantity sensitivity

```
# *   *   *   *   * ] *] # Line 0
# S   S   S   S   S   H  #
```

Since this is the case, the presence of a stressed syllable at an end of the word can't be used to definitively rule out "extrametricality", in the sense of ruling out either of the edge marking

options shown in (3.6.4). If Dresher had worked with a formal implementation of the system, this issue would probably have received closer discussion.

It could turn out that Dresher's notion of extrametricality, which as I interpret it would make a syllable irrevocably ungroupable into a metrical constituent, is a better fit to the empirical facts. It is in fact, consistent with the metrical theory implemented in Dresher and Kaye (1990). Nonetheless, it is also easy to patch up the cue to work with the Halle and Idsardi system as is. If the learner only considers stressed *light* syllables at the word boundaries as evidence for rejecting the appropriate edge marking option, then it will not be confounded by these cases.

This cue, as I have amended it, now refers to the notion of syllable weight. Strictly speaking, then, the application of this cue must wait until after steps B and C. However, it is possible to restrict the cue even further, so that this is no longer the case. If the learner only considers stressed *open, short-vowelled* syllables at the edge, it is guaranteed not to mistakenly consider a heavy syllable. These *lightest* of light syllables will not be heavy in any system. As stated the cue will work for all languages. This generality, however, comes at a price. Although a learner uses positive evidence in this case, it may have to wait longer for the appropriate evidence to arise if the more general cue is adopted because I have restricted the set of applicable forms.

Unlike the earlier cues, this test for eliminating an "extrametricality" option is only intended to work in one direction. In Dresher's cue-based algorithm, the failure to observe a stressed, light syllable at an edge is not intended to license the conclusion that extrametricality is at work. Dresher's main interest in this cue is that it sets the stage for determining whether languages build iterative constituents. This determination is made in Step F. The cue used in

Step F relies on the assumption that iterative constituent construction is the only source of stressed light syllables that do not fall at word boundaries. The present step rules out a class of languages where this assumption does not hold true; the extrametrical bracket can produce a stress on a *word-internal* syllable regardless of the syllable's weight.

Note also that unlike the cues used in Steps A–C, the cues used here do not fix any particular parameter values. Rather they eliminate certain combinations of values that govern edge marking. This is a noteworthy deviation from the Dresher and Kaye design philosophy.

There is not "extrametricality" in Selkup.

Step E: Rule out pre- or post-accenting

Like the cue used in Step D, the cue used here is intended to rule out a source of stresses on light syllables that don't fall at the edge of the word. Again, the reasoning is that such syllables that would potentially invalidate the cue in Step F. In pre- and post-accenting languages, a heavy syllable results in a stress falling on the syllable next to a heavy syllable.

Here, Dresher is being admirably cautious; the phenomenon of pre- and post-accenting is not even currently modeled by the Halle and Idsardi system, but Dresher wishes to protect the cue-based algorithm against possible extensions of the system. I will not discuss this cue further here. At any rate, it can also be made to be an independent cue if *light* is replaced with *lightest* and *heavy* with *heaviest*.

This option is not exercised in Selkup.

Step F: Rule out iterative constituent construction

As noted above, Steps D and E were intended to set the stage for the cue used at this step by ruling out any options, other than iterative constituent construction, that would result in a stress on a word-internal—that is to say, not initial and not final—light syllable. Dresher also rules out one further complication by fiat. Dresher argues that the empirical intention of quantity sensitivity is to provide a mechanism for stressing heavy syllables. He notes, however, that as this notion is formulated in the Halle and Idsardi system, this is not necessarily the result of quantity sensitive bracket insertion. Consider a right-headed language that places a left bracket to the left of its heavy syllables. In such a language, the heavy syllable will not necessarily, or even typically, receive a stress. As shown, in (3.6.6), it is quite easy for the heavy syllable to "throw" its stress downstream to a light syllable.

(3.6.6) Quantity Sensitivity does not lead necessarily to stressed heavy syllables.

```
#          *            *  # Line 1
# [*   *    *   [*   *    *  # Line 0
# H    L    L   H    L    L  #
```

Dresher explicitly rejects this option by requiring the direction of quantity sensitive bracket insertion to be correlated with headedness so that the stress stays with the heavy syllable that generated the bracket. Relaxing this additional assumption would obviously invalidate this cue.

This cue also explicitly relies on what I'll call Dresher's *contract on stress adjustment*. Dresher (1994) indicates that the empirical record suggests that something like this might be true, but, of course, this is a crucial assumption. Again, admirably aiming to constrain the effect that subsequent additions might have on the current set of cues, Dresher proposes that any

additions to the system will obey the following restrictions: Languages can modify brackets in such a way that syllables that would otherwise be stressed end up not receiving stress. Languages cannot, however, modify brackets in such a way that syllables that would otherwise end up being unstressed end up receiving stress. If this contract holds, the only remaining possible source of a word-internal stress is iterative constituent construction.

As stated, this cue depends on the elimination of extrametricality, accomplished in Step D, and the elimination of pre- and post-accenting, accomplished in Step E. Dresher does not indicate how the decision about iterative constituent construction would get made, if a grammar exhibited "extrametricality". A slightly different cue, however, that is demonstrably independent of the outcome of Steps D and Steps E will also do the trick. Consider a string of light syllables. In fact, consider a longish string of lightest syllables so as to allow for independence of the outcome of Steps B and C as well. If a stressed syllable occurs anywhere other than the initial, second, penultimate or final position, iterative constituent construction has occurred. With the resources of the system and the contract on stress adjustment, this is the only possible source of such a stress. I have avoided dependence on Step D, which eliminates the extrametrical bracketing options, by expanding the notion of the periphery of the word to encompass two syllables. I have avoided dependence on Step E, which fixes quantity sensitivity, by considering strings consisting only of lightest syllables. Such strings are not subject to pre- or post-accenting. Again, doing so comes at a cost since I have restricted the set of sentences that could trigger this decision.

Iterative constituent construction is positively cued. A learner who decides that there is no iterative constituent construction relies on indirect negative evidence. Again, it is worth pondering the cost of the generalized cue that I have constructed since it forces learners to

examine longer words; words with syllables that are not initial, second, penultimate or final, of necessity, are at least five syllables long. If this proves to be unreasonable, as it seems that it might, then there will have to be some alternative means of ruling of determining whether grammars with "extrametricality" have iterative constituent construction.

Selkup does not build constituents iteratively.

Step G: Assign Line 1 headedness

Dresher makes the simplifying assumption that, as a matter of parametric variation, main stress picks out either the leftmost or the rightmost of the marks on Line 1. The grammatical resources built up in Halle and Idsardi potentially allow for further possibilities, but, in the spirit of evaluating the cue-based approach with reference to the model delimited by Dresher, I will ignore these here.

In the cue-based algorithm that Dresher proposes, the cue for main stress location depends on prior decisions about what how the learner groups syllables into constituents. The key idea is that if learners can establish a boundary between material grouped into different constituents, they can determine straightforwardly whether main stress falls within the leftmost foot or the rightmost foot. If stress falls on a position to the left of the boundary, it falls on the leftmost foot. If stress falls on a position to the right of the boundary, it falls on the rightmost foot.

It is possible to employ a slight variant of this idea to develop a cue that doesn't depend on any decision about foot type. If a language never generates more than one mark on Line 1, then the question of whether the leftmost or the rightmost mark on Line 1 receives stress

arises only vacuously. The question of main stress location is only meaningful if more than one mark gets projected to Line 1, or, equivalently, if more than one constituent gets formed one Line 0. If a) there were syllable strings that were universally divided into more than one constituent in any language that formed multiple constituents and b) it were possible to definitively identify a dividing point between the region containing the initial head and the region containing the final head, a learner could use such syllable strings and fix main stress location by following the same straightforward strategy of looking to see where main stress fell. There are syllable strings that satisfy both a) and b), so that it is possible to indicate a "universal division" between the left and the right half of the word. This argument is heavily sensitive to disruption of the brackets by the currently extra-parametric portion of the system.

Given the resources of the system, in order for multiple constituents to arise, a language must either be quantity sensitive or a language must construct iterative, binary constituents. Consider the following syllable string fragment, where the ellipses are filled in by one or more unspecified syllables

(3.6.7) ...S H S S H S...

If a language ever forms more than Line 0 constituent, it will do so here. In order to form more than one constituent, a language must either a) be quantity sensitive or b) make use of iterative constituent construction. If the language forms quantity sensitive constituents, then clearly there will be two constituents here. The left H will fall in one, while the right H will fall in the other. A constituent boundary must lie somewhere between these two boundaries. The constituents formed by quantity sensitive insertion take precedence over those inserted by

iterative constituent construction; in its calculations, iterative constituent construction respects the boundaries placed by quantity sensitive insertion. Therefore, the same argument holds for a language that is *both* quantity sensitive and a builder of binary constituents. If a language is not quantity sensitive, but does build binary constituents, then it is plain that the two H syllables will not be in the same constituent because there would be nothing blocking the insertion of bracket somewhere between the two H's. Again, a constituent boundary must lie between them.

In the case where the language is quantity sensitive, then if the main stress falls on or to the left of the left H, then the language is left-headed. If the main stress falls on or to the right of the right H, then the language is right-headed. Here, I am assuming with Dresher that a quantity-sensitive bracket always results in stress on a heavy syllable. If this assumption is relaxed, it is still possible to get this *universal divider* argument to work with a more complicated boundary region.

If the language is not quantity sensitive, then if the main stress falls on or to the left of the first of the middle S's then the language is left-headed. Otherwise it is right-headed. This can be seen by considering all possible bracketings of the syllables in (3.6.7) that are consistent with the operation of iterative constituent construction.

(3.6.7) ..S H) S S) H S..

..S)H S)S H)S..

..S (H S (S H (S..

..S H (S S (H S..

In all cases, then a division between left and right in the word can be made between the two middle syllables. Again, this modified cue could potentially run afoul of destressing mechanisms if they had the effect of dismantling constituents in this region.

Use of this cue in an independent fashion requires the learner to do without any information about the quantity sensitivity of the target grammar. In particular, the learner cannot rely on any ability to correctly classify closed syllables with short vowels, which are light in some languages, but heavy in others. However, the class of syllables with long vowels, in the context of this system, are guaranteed to be treated as heavy if any syllable is treated as heavy. Therefore, in order for this cue to operate independently, the H's above should be interpreted as these heaviest of the heavy syllable—the syllables that are guaranteed to be heavy. Note that the success of this cue depends on the assumption that languages will not impose restrictions on the strings of syllables that make up the words of its languages.

The type of cue considered here looks a lot less natural natural than others presented above, but it is not entirely clear how to determine what makes a cue sufficiently unnatural that it should be excluded from consideration. For example, it is not clear that Dresher's cue for iterative constituent construction, used in F, is a direct consequence of the content of the parameter. The only reason that stressed, word-internal, light syllables can cue quantity sensitivity is that there are no other mechanisms in the system that could produce them.

There is clearly a price to pay for ensuring the independence of this cue: a rather special string of syllables must occur in the middle of a long word. In fairness to Dresher, his approach is more elegant. The user uses a sort of meta-cue that relies on knowledge of previous parameter settings. First, the constituent formation processes are determined and then the learner simply looks to see whether the leftmost or the rightmost such constituent is stressed. To really be able to determine the constituency of a word, however, it would be necessary to have made a firm decision about whether or not a language used the "extrametrical" options.

Both values of the parameter are positively cued.

Selkup is right-headed on Line 1.

To summarize up to this point: I have traced through the cues that are used to a) set the conflation parameter, b) set the quantity sensitivity parameter, c) fix the notion of heavy syllable, d) rule out LRL and RLR edge marking , e) rule out pre- and post-accenting, f) determine whether a language builds iterative constituents and g) localize main stress to the right or the left of the word. Along, the way I have indicated how to generalize the statement of the cues to make them independent. As a result, it is clear how these decisions for *all* languages in the system. For a learner of a language without extrametricality and iterative constituents, like Selkup, it remains to show how to select the appropriate edge marking and headedness values for Line 0. I'll review this logic here and then lay out which regions of the parameter space still need to be handled. The following steps *are* dependent on what has gone before. I leave it as an exercise for the reader to determine whether a clever generalization to more independent cues can be provided. I have not attempted to do so.

Step H: Assign Line 0 headedness

Again, a string of light syllables plays an important role. Since such strings can't possibly receive a bracket because of quantity sensitivity, they make the quantity sensitivity parameters irrelevant. If it has already been determined that the language builds no iterative constituents and does not allow the extrametrical edge marks, then a string of light syllables will contain at most one constituent. This constituent will be created by the insertion of an edge

mark. In the Halle and Idsardi system, there are four possibilities relevant to monomorphemic words, shown in (3.6.7).

(3.6.7) a. LLL

```
# [*   *    *    *    *    *   # Line 0
# L    L    L    L    L    L   #
```

b. RRR

```
# *    *    *    *    *    *]  # Line 0
# L    L    L    L    L    L   #
```

c. RRL

```
# *]   *    *    *    *    *   # Line 0
# L    L    L    L    L    L   #
```

d. LLR

```
#   *   *    *    *    *    [*  # Line 0
# L    L    L    L    L    L   #
```

Dresher, noting that either (3.6.7.a) or (3.6.7.c) lends itself to an analysis of Selkup in this system, states that he is omitting the RRL and LLR options from the system, which create single syllable constituents at the edge of the word. Of course, this move restricts the expressive power of the system because there are distinct languages in the system that only receive a proper analysis by adopting one of these *edge-trapping* options. This done, however, it is clear that a string of light syllables will form a single long constituent in a language without "extrametricality" and iterative constituents. If the leftmost syllable in the constituent receives stress, then the language is left-headed on Line 0. If the rightmost syllable in the constituent receives stress, the language is right-headed on Line 0. The words actually need not be as long as they are in (3.6.7); two syllables would suffice.

With slightly strings of light syllables, Dresher's cue can be generalized to accommodate languages with "extrametricality" and without bounded constituents. For the constituency shown in (3.6.8.a), stress will fall on either the second or the final syllable. If it falls on the second syllable, the language is left-headed one Line 0. If it falls on the final syllable it is right headed. In the mirror image case in (3.6.8.b), stress on the first syllable diagnoses left-headedness, while stress on the penultimate syllable diagnoses right-headedness.

(3.6.8) a. LRL

```
# *[  *   *   *   *   *  # Line 0
# L   L   L   L   L   L  #
```

b. RLR

```
# *   *   *   *   *  ]*  # Line 0
# L   L   L   L   L   L  #
```

The learner simply needs to determine whether stress falls in the leftmost two syllable window, or in the rightmost two syllable window. Again, adopting this more general cue would incur a cost in terms of the amount of evidence available; now four syllable words are required. I make this generalization for the purposes of filling out the learning path. It may not, of course, prove optimal.

Both values of the parameter are positively cued, in either Dresher's or my scheme. Selkup is left-headed on Line 0.

Step J: Fix value for Line 0 edge marking

The final step in setting the parameters for Selkup involves simply trying out both edge-marking parameters to see which accommodates the data. (Dresher assumes that there is always an edge mark.) In this case, the two possibilities are weakly equivalent, so a tie is broken in favor of the LLL edge mark because it is, in some sense, homogeneous with the left-headedness of Line 0. Note that since the orientation of quantity sensitive brackets is correlated with headedness, we now know that these brackets must be left brackets.

Selkup is now complete, as is any other language that does not build binary constituents and does not make use of the "extrametrical" edge marks. As I have generalized the cues, no further decisions hinge on "extrametricality" for languages without binary constituents. So, it remains to be shown a) that there is a cue for definitively adopting an "extrametrical" edge mark and b) that the direction of iterative constituent construction, the type of edge marking—including the possibility of "extrametrical" edge marks—deployed on Line 0 and the headedness of Line 0 can *all* be determined for languages that adopt iterative constituents.

Dresher does not provide the first of these cues. In cases where there are no iterative constituents constructed, however, there does not seem to be any compelling reason to not simply add these edge marks to the final competition conducted in Step J, which is currently used to select among the other edge marking options. As I have shown it is possible to generalize the cues downstream of the "extrametricality" decision in a way that makes this "extrametricality" decision irrelevant. The cues that follow it can still act appropriately. It only remains to be seen whether the language uses an extrametrical mark. Instantiating grammars for all the competing edge mark possibilities, ruling out those that don't accommodate the data and resolving ties arbitrarily will certainly lead to success in a space like this one, where there are no subset/superset relations between languages. Moreover, it does

not seem to violate Dresher's intentions in this system, since he already proposes a slightly smaller competition. Of course, it is does seem to violate the top-level design philosophy that Dresher and Kaye (1990) and Dresher (1994) advocate.

Similarly, in discussion of the languages Maranungku and Garawa, Dresher proposes a final problematic *generate-and-test* approach to fixing the direction of iterative constituent construction and Line 0 headedness. In keeping with the contract that he establishes on destressing, a stressed syllable never arises from "extra-parametric" manipulation of brackets. Therefore, Dresher proposes simply to instantiate all candidate combinations, with their other parameters fixed, ruling out those grammars that predict the absence of a stress in a position where a stress actually occurs, and resolving ties in an essentially arbitrary manner. In the cases he considers, there is no Line 0 edge marking, but clearly, the set of candidates could simply be extended to accommodate the five edge-marking possibilities that Dresher allows from the Halle and Idsardi system: 1) no edge marking, 2) LLL, 3) RRR, 4) "extrametrical" LRL and 5) "extrametrical" RLR.

These competitions rule out some grammars on the basis of positive evidence, and ties between weak equivalents are broken arbitrarily. If the learner does not explicitly have a way of computing which grammars are equivalent, they are essentially assuming that no further evidence will serve to distinguish the remaining competitors. Therefore, these final competition steps rely, in part, on indirect negative evidence and could fail if insufficient data are gathered.

Extended in this way, the algorithm now succeeds in the space, given the additional limitations on the space that Dresher imposes along the way.

3.7 How does the Instruction Manual Satisfy the Design Philosophy of the Cue-based Approach?

As I have presented and extended it, the cue-based algorithm for the acquisition of a fragment of the Halle and Idsardi system really need not make much use of the increasing inferential leverage that fixing the values of certain parameters might provide. It proved quite straightforward to eliminate some dependencies in the original system. Decisions about the presence or absence of conflation, type of quantity sensitivity (if any), the presence or absence of iterative constituent construction and headedness of Line 1 can all be made in the absence of any knowledge of the other values of the parameters. This has the effect of completely obviating questions about the growth of the initial segment of the decision tree. For these independently cued decisions, there really is only one cue per decision.

Of course, my efforts at generalizing these cues to ensure learnability are hardly without cost. In each case, generalization of the cues limited the available evidence for parameter-setting. As a result, the learner must expect to wait longer for the information it needs to set parameters. Most of these generalizations were made so as to cover the learning of languages in the system that were not covered in Dresher's (1994) discussion. (One parametric cue in the generalized system is dependent on earlier parameter settings. For languages that don't build iterative constituents, an explicit cue is provided for determining Line 0 headedness that does depend on earlier parameter settings. Several non-parametric eliminations are also potentially performed to rule out "extrametricality", which seems to run counter to the top-level strategy that Dresher and Kaye (1990) and Dresher (1994) advocate.)

The alternative to this process of cue generalization would be to provide more order-dependent cues for the relevant decisions. If these could be written as meta-cues (again, these are abstract cues that, on the basis of prior parameter setting, could be easily instantiated to provide the learner with a concrete set of properties of the input stream to monitor for), then the ordering requirement is not necessarily prohibitive. Where this cannot be done, of course, additional, more highly specialized cues must be directly encoded in UG.

It is likely that an unwillingness to use such "unnatural" cues in tandem with an inability to see what else to do that motivates the generate-and-test approach that Dresher (1994) advocates to fix the final parameters of the system. The decisions that complete the various paths in the decision tree, both in Dresher's original proposal and my extension, get made by simply instantiating the remaining possibilities and seeing which ones fit the data; this is essentially a parallel, exhaustive search strategy. Of course, the cue-based algorithm that I have presented is sufficient for acquisition of the restricted space, but it is hardly necessary. Again, I take it, however, that Dresher moved to a generate-and-test strategy to fix the final parameters because plausible cue stories seemed to simply require dependent cues that could not be shared across the tree in a reasonable fashion.

The key question for the cue-based approach, given our concerns about resource requirements, is where the line can be drawn between the part of the decision tree that relies on independent or easily reconfigurable meta-cues and the final part of the decision tree that essentially requires unique cue paths, or, equally bad, a generate-and-test strategy which will not succeed in exponentially large regions of the decision tree.

3.8 The Impact of Extending the Phonological Fragment on the Cue-based approach

The fragment of the Halle and Idsardi system that Dresher (1994) analyzes is not the real parametric system for stress. The question to be addressed in this section is the question of how the cue-based algorithm is likely to scale up when the system is required to both accommodate more data from languages that already exist in the system and more extensive parametric variation. I will focus discussion on the impact of system extensions on parametric cue-based algorithms.

Clearly, the danger for a particular parametric cue-based algorithm is that alterations of the system will invalidate the deductive logic embodied in the algorithm's decision tree.

One way in which a parametric system could be altered is by modification of the data sets of existing languages in the system. When new forms get added to existing languages in the system, extended languages have the potential to more readily "trip" cues that are based on positive evidence. Form observations or property observations (of single forms or of sets) might no longer have the same unique explanations in the same background contexts of fixed parameters, and a learner who relies on these is potentially subject to inappropriately setting parameter values. On the other hand, adding forms in this way makes cues based on indirect negative evidence more difficult to satisfy. This could prevent a learner from ever setting certain parameter values.

In the domain of phonology, the contribution of the lexicon provides a particularly pernicious source of potential cue invalidation. Lexical exceptions that cause a language to generate data that mimics "exceptionless" data from other languages has a clear potential to

lead the learner astray. If these lexical exceptions are few and far between, the learner might be able to get by if evidence must pass some sort of frequency threshold to trigger parameter setting. This strategy might also prove workable in the face of noise. Clearly, this issue requires further consideration.

Removing forms from existing languages in the system has a mirror image effect. Cues based on positive evidence might fail to apply for certain target languages because crucial data are no longer available. Cues based on indirect negative evidence might inappropriately be triggered because evidence that would have blocked them from applying is no longer available.

If a new parametric system modifies the set of languages in the original space and their parametric labels—say by adding a new parameter—matters are more complicated. Clearly, new parametric cues for any new parameters must be added to the system at an appropriate point in the decision tree. Moreover, it is clearly possible for the old parameters to interact with new parameter values in such a way that would invalidate previously developed cues. Consider the case where a new dimension of parametric variation is added. Suppose that all grammars in the original space implicitly had one particular value of the new parameter. All cues for parameter setting in the original space, from the point of view of the larger space, implicitly now rely on a the new parameter having a particular value. They may no longer work once this assumption is relaxed.[22]

The paragraphs above discuss some possible effects of systemic modifications on particular, existing, cue-based algorithms. Clearly since cue-based algorithms are "hand-

---

[22] If the new parameter value could be set independently, then its cue could be placed towards the root of the decision tree and the deductive logic of the old decision tree could be preserved as a valid subtree. Of course, the other subtrees stemming from this level would need to be developed.

crafted" to accommodate particular spaces, the effects on the deductive logic of the system can only be disruptive. What can be said about the more general impact of extending the parameter space on the ease of developing parametric cue-based algorithms? I will defer this discussion until Chapter 6, where I will address it, in part, by analyzing which spaces an OPL learner can be expected to succeed in.

3.9 Cue-based Algorithms for the Gibson and Wexler space

Gibson and Wexler's (1994) presentation of the TLA centers around an analysis of its behavior in a space defined by three well-motivated syntactic parameters that govern phrase structure (see Bach 1962, den Besten 1983, Bierwisch 1963, Haider and Prinzhorn 1986, Hoekstra 1984, Thiersch 1978, Travis 1984 among many others). The first parameter fixes the relative base order of an XP's head and complement, with the exception of the top-level CP projection, which always has its head precede its complement. For example, in a grammar with this parameter set to *comp-final*, as in English, the verb would precede its object in the base order. The second parameter fixes the relative base order of an XP's head and specifier positions, with the exception of the CP, which, in the Gibson and Wexler system always has its specifier precede its head. In a language with the value of this parameter set to *spec-first*, as in English, a subject's base order position would precede that of the IP, and, thus, the verb. The final parameter captures the so-called V2 phenomenon exhibited by many Germanic languages. In V2 languages, the finite verbal item in a sentence—the auxiliary if there is one, otherwise the tensed verb—moves from its base position to the C position. In addition, an XP such as the subject, an object or an adverb must move from its base position to the specifier of CP. Since the

CP is the top-level category in a phrase structure, and since, in the context of Gibson and Wexler's system, the CP is always configured as shown in 3.9.1, the moved XP occupies the first position, while the moved verbal material occupies the second position—thus, the term V2, for verb-second.

(3.9.1) [$_{CP}$ Spec-CP [$_{C'}$ C ...

For the eight languages in this three parameter system, Gibson and Wexler generated the data sets for simple matrix clauses shown in (3.9.2). 'Adv' stands for adverb. This adverb occupies a position immediately to the left of the auxiliary's base position. 'Aux' stands for auxiliary. 'O' stands for the object of a simple transitive. 'O1' stands for the first object of a double object construction. 'O2' stands for the second object of a double object construction. 'S'stands for subject. Note the explicit assumption that learners, on the basis of semantic context, can distinguish between noun phrases acting as subjects, noun phrases acting as objects of simple transitives, noun phrases acting as the first object in a double object construction and noun phrases acting as the second object in a double object construction. Internal details of these NP categories are suppressed in the representation that Gibson and Wexler adopt. See Gibson and Wexler (1994) for further motivation of the system. From the point of view of the present section, which will develop a cue-based algorithm for the acquisition of this particular system, we can simply inspect the contents of the data set, and not worry so much about their origins.

Table 3.9.2
Gibson and Wexler's Parameter Space

Parameter Settings	Data in Defined Grammar
spec-final, comp-final, -V2:	(V S) (V O S) (V O1 O2 S) (Aux V S) (Aux V O S) (Aux V O1 O2 S)(Adv V S) (Adv V O S) (Adv V O1 O2 S) (Adv Aux V S) (Adv Aux V O S) (Adv Aux V O1 O2 S)
spec-final, comp-final, +V2:	(S V) (S V O) (O V S) (S V O1 O2) (O1 V O2 S) (O2 V O1 S)(S Aux V) (S Aux V O) (O Aux V S) (S Aux V O1 O2) (O1 Aux V O2 S) (O2 Aux V O1 S) (Adv V S) (Adv V O S) (Adv V O1 O2 S) (Adv Aux V S) (Adv Aux V O S) (Adv Aux V O1 O2 S)
spec-final, comp-first, -V2:	(V S) (O V S) (O2 O1 V S) (V Aux S) (O V Aux S) (O2 O1 V Aux S) (Adv V S) (Adv O V S) (Adv O2 O1 V S) (Adv V Aux S) (Adv O V Aux S) (Adv O2 O1 V Aux S)
spec-final, comp-first, +V2:	(S V) (O V S) (S V O) (S V O2 O1) (O1 V O2 S) (O2 V O1 S) (S Aux V) (S Aux O V) (O Aux V S) (S Aux O2 O1 V) (O1 Aux O2 V S) (O2 Aux O1 V S) (Adv V S) (Adv V O S) (Adv V O2 O1 S) (Adv Aux V S) (Adv Aux O V S) (Adv Aux O2 O1 V S)
spec-first, comp-final, -V2:	(S V) (S V O) (S V O1 O2) (S Aux V) (S Aux V O) (S Aux V O1 O2) (Adv S V) (Adv S V O) (Adv S V O1 O2) (Adv S Aux V) (Adv S Aux V O) (Adv S Aux V O1 O2)
spec-first, comp-final, +V2:	(S V) (S V O) (O V S) (S V O1 O2) (O1 V S O2) (O2 V S O1) (S Aux V) (S Aux V O) (O Aux S V) (S Aux V O1 O2) (O1 Aux S V O2) (O2 Aux S V O1) (Adv V S) (Adv V S O) (Adv V S O1 O2) (Adv Aux S V) (Adv Aux S V O) (Adv Aux S V O1 O2)
spec-first, comp-first, -V2:	(S V) (S O V) (S O2 O1 V) (S V Aux) (S O V Aux) (S O2 O1 V Aux) (Adv S V) (Adv S O V) (Adv S O2 O1 V) (Adv S V Aux) (Adv S O V Aux) (Adv S O2 O1 V Aux)
spec-first, comp-first, +V2:	(S V) (S V O) (O V S) (S V O2 O1) (O1 V S O2) (O2 V S O1) (S Aux V) (S Aux O V) (O Aux S V) (S Aux O2 O1 V) (O1 Aux S O2 V) (O2 Aux S O1 V) (Adv V S) (Adv V S O) (Adv V S O2 O1) (Adv Aux S V) (Adv Aux S O V) (Adv Aux S O2 O1 V)

Given Gibson and Wexler's work in generating all the forms in the languages, it is a fairly simple matter to provide a cue-based algorithm for the acquisition of this system. Note that the names of the comp-first/comp-final and spec-first/spec-final *almost* directly indicate what properties of the input forms to attend to. To set the comp-first/comp-final parameter look and see whether complements precede their heads in the base word order. If so, the language is comp-first; if not, the language is comp-final. Similarly, to set the spec-first/spec-final parameter, inspect the order of specifiers and heads in the base order. Of course, the problem with this simple scheme, of course, is that the base order is only transparent for languages where no movements transform the base word order.

In this "single-transformation" system, however, the base word order is entirely unobscured when V2 movement is not allowed. Therefore, if a learner could establish that V2 movement was disallowed, the cues described in the above paragraph would work directly. Here too, the name of the parameter suggests a cue for fixing the value of the V2 parameter. Every sentence in all of the +V2 grammars has an auxiliary or a verb in the second position. Each –V2 grammar has sentences that do not have this property. The spec-final, comp-final, –V2 grammar has, for example, many sentences where the verb or auxiliary occurs in initial position. In fact, the only way for a "V2" sentence to occur in this grammar is for an adverb to appear. Inspection of the data for the spec-final, comp-first, –V2 grammar indicates that only four of its twelve sentence types have a verb or auxiliary in second position. Similar considerations of the data in the spec-first, comp-final, –V2 and spec-first, comp-first, --V2 cells show that there is always ample evidence to set the V2 parameter to its negative value. Any sentence which does not have a verb or auxiliary in second position can serve as a cue for setting the V2 parameter to its negative to value. The internal instruction that the cue-based learner

needs to encode is clear: look to see if the verb or auxiliary is in second position. Failure to find a non-V2 sentence, of course, leads to the conclusion that the language is +V2. Setting the V2 parameter to its positive value, on this particular cue-based scheme, uses indirect negative evidence.

While the +V2 setting of the V2 parameter obscures the base order of the sentences that a language generates, it does not do so completely and in all cases. The comp-first/comp-final parameter can be set by noticing that certain V2 patterns leave the relative base ordering of a verb and its complements intact. In sentences that contain an auxiliary, it is the auxiliary, not the verb, that moves from its base position. Moreover, once it has been established that a language is +V2, it is quite simple to determine what XP has moved from its base position to the specifier of CP; the XP in first position has been moved. In the present system, if the moved XP is an adverb, the subject, or one of the objects of a double object verb, then the base position of the verb's complements, relative to the verb, is directly visible in the surface string. For example, in the spec-final, comp-final, +V2 language, the sentence 'S Aux V O' could be used to fix the comp-first/comp-final parameter to comp-final because the object, which I have argued must occupy its base position, follows the verb.

Similarly, in sentences that contain an auxiliary and a moved XP other than the subject, the relative base ordering of the subject and verb provides a direct indication of the value of the spec-first/spec-final parameter. For example, in the spec-final, comp-final, +V2 language, the sentence 'O Aux V S' cues the spec-final value of the spec-first/spec-final parameter because the subject follows the verb.

The learner can take any form with an unobscured context for the base order regulated by these parameters and use it as a cue for the appropriate setting.

The cue for setting the word order parameters that I have developed clearly depends on the decision about the V2 status of the target. The word order cues might be stated separately, with one version of each for the +V2 value, and one version of each for the –V2 value. Alternatively, the cues might stated as follows: if the base order of the verb and the complement (or the specifier and the verb, as the case may be) if the language is not V2, or if the base order is not obscured for one of the reasons listed above, then the order of the verb and complement in the string reflects the base order.

This first cue system for the space, while relatively natural, relies on indirect negative evidence. The question might be raised the question of whether it is possible to construct any cue system for the space that has a single cue per parameter value, but which does not rely on indirect negative evidence. I leave this as an open question, but note that a fair answer would not do something tricky like allow disjunctive cues that allowed the learner to look for either form A, or form B, etc.

Alternatively, as another example of a cue-based system for the acquisition of the 3-parameter space modeled after an observation in Fodor (1995), the following set of eight independent form cues—one for each language—suffices to fix all three parameter values at once. This system has the virtue of relying only on positive evidence. The cues in 3.9.3 are simply forms from the data set that are uniquely generated by the language that they appear in. A complete list of such forms appears in Table 3.9.4. Within the confines of the system, the unique explanation of any of these forms requires a fixing of all unfixed parameters.

(3.9.3)

spec-final, comp-final, -V2:	'Aux V O1 O2 S'
spec-final, comp-final, +V2:	'O2 Aux V O1 S'
spec-final, comp-first, -V2:	'Adv O2 O1 V Aux S'
spec-final, comp-first, +V2:	'Adv Aux O2 O1 V S'
spec-first, comp-final, -V2:	'Adv S Aux V O1 O2'
spec-first, comp-final, +V2:	'Adv Aux S V O1 O2'
spec-first, comp-first, -V2:	'Adv S O2 O1 V Aux'
spec-first, comp-first, +V2:	'Adv Aux S O2 O1 V'

Table 3.9.4
Unique Data in Gibson and Wexler's Parameter Space

Parameter Settings	Unique Data in Defined Grammar
spec-final, comp-final, -V2:	(AUX V O1 O2 S) (AUX V O S) (AUX V S) (V O1 O2 S) (V O S)
spec-final, comp-final, +V2:	(O2 AUX V O1 S) (O1 AUX V O2 S)
spec-final, comp-first, -V2:	(ADV O2 O1 V AUX S) (ADV O V AUX S) (ADV V AUX S) (ADV O2 O1 V S) (ADV O V S) (O2 O1 V AUX S) (O V AUX S) (V AUX S) (O2 O1 V S)
spec-final, comp-first, +V2:	(ADV AUX O2 O1 V S) (ADV AUX O V S) (ADV V O2 O1 S) (O2 AUX O1 V S) (O1 AUX O2 V S)
spec-first, comp-final, -V2:	(ADV S AUX V O1 O2) (ADV S AUX V O) (ADV S AUX V) (ADV S V O1 O2) (ADV S V O)
spec-first, comp-final, +V2:	(ADV AUX S V O1 O2) (ADV AUX S V O) (ADV V S O1 O2) (O2 AUX S V O1) (O1 AUX S V O2)
spec-first, comp-first, -V2:	(ADV S O2 O1 V AUX) (ADV S O V AUX) (ADV S V AUX) (ADV S O2 O1 V) (ADV S O V) (S O2 O1 V AUX) (S O V AUX) (S V AUX) (S O2 O1 V) (S O V)
spec-first, comp-first, +V2:	(ADV AUX S O2 O1 V) (ADV AUX S O V) (ADV V S O2 O1) (O2 AUX S O1 V) (O1 AUX S O2 V)

Less arbitrarily and more quickly, the learner might simply wait until they had witnessed an instance of any one of the forms in an appropriate cell of Table 3.9.4. This, of course, would impose an additional memory requirement on the learner: they would have to explicitly store more forms in their UG-provided instruction manual. Of course, even the more succinct system for the "instantaneous" acquisition of the system will be unable to scale up the storage of the necessary cues to larger systems. It will not do to explicitly record even a single sentence for each possible language. Moreover, in the succinct system, the cues may appear rather sparsely in natural language input.

Although, logically these "rapid acquisition" cues could be used to fix several parameters at a time, this runs counter to Dresher and Kaye's design philosophy since there is not a single cue per parameter value. A major reason for viewing this, single-cue-per-parameter value as a desirable feature of cue-based learning systems is that it eliminates some concerns about exponential growth in the number of cues. As noted in discussion of the Dresher (1994) system, it was indicated that the single-cue-per-parameter value approach was not entirely satisfied there either. If this approach largely breaks down and the learner needs to find other ways of coping with large regions of the hypothesis space, then we have an argument against a cue-based approach. The key issue here is what will happen in the extension to larger systems.

As noted in discussion of Dresher's system of cues for the Halle and Idsardi system, addition of any data or languages that robs a candidate parameter assignment (to a candidate extension in the background of a fixed context of set parameters) of its ability to uniquely explain some aspect of the data robs the system of a potential parametric cue. For example, if the V2 data in the Gibson and Wexler system were expanded to include non-V2 data from special structures in V2 languages, then the first cue-based algorithm that I presented would not extend easily to the new system. Similarly, expansion of the parameter set could, for example, result in a non-V2 language capable of generating 'O2 Aux V O1 S'—currently a dead giveaway for the spec-final, comp-final, +V2 language. If this happened, obviously, this cue would no longer work. This suggests that things will only become more difficult for a cue-based learner as the parametric system becomes more complex. This is certainly the case as more parameters are added.

Fodor's (1995) proposal, discussed in Chapter 6 suggests a way of possibly exploiting this same type information without explicit storage of the cues. If this were really true, then

the tractability motivation for the single-cue-per-parameter value property would be undermined.

Chapter 4
The Triggering Learning Algorithm in the Limit

The main purpose of this chapter is to present identification in the limit results obtained from an application of the TLA, our paradigm IGMS algorithm, to several phonological subspaces of the Halle and Idsardi model. In Section 4.1, I will introduce the TLA and discuss its key features. In Section 4.2, I will briefly remind the reader of the difficulties that Gibson and Wexler encountered in ensuring the learnability of the class of languages contained in their three parameter syntactic space. I will also discuss several remedies that they proposed. In Section 4.3, I will present results from the application of the TLA to several parametric spaces derived from the Halle and Idsardi system. Here, as in the Gibson and Wexler system, the TLA cannot straightforwardly guarantee identification in the limit for the class of grammars in the fragments without some additional, perhaps not unnatural, assumptions. Following the lead of Gibson and Wexler, I then present one possible set of modifications to the learning algorithm and the phonological fragment that leads to a workable system. As was also probably true with the cue-based systems discussed in Chapter 3, Section 4.3 will almost certainly leave the reader with questions about the robustness and extendibility of the results obtained. In Section 4.4, I will do what I can to address these concerns. I will also address the question of whether there are grounds for hope that learnability results obtained with the TLA might somehow provide an independent source of constraint on linguistic theory. It would be much more difficult for the cue-based approach to do so because of its relative lack of constraint.

4.1 The Triggering Learning Algorithm

In this section, I will first describe how the TLA works. Next, I will draw attention to what, from my point of view, are its key features. Finally, I will discuss some potential difficulties that natural language parametric systems could pose for the TLA. This discussion will receive much elaboration in the following chapter.

Gibson and Wexler's (1994) TLA is presented in (4.1.1).

(4.1.1) a. Randomly select an initial hypothesis grammar from the set of all possible parameter settings.

b. Analyze an input from the target language:

i. If the current grammar provides an analysis of the current input, then keep the current grammar and start again at (b).

ii. If the current grammar cannot provide an analysis of the current input, then randomly select a single parameter to change, change it, and then:

1. If the new grammar provides an analysis of the current input, then keep the new parameter settings and start again at (b).

2. If not, then revert to the previous grammar by unflipping the changed parameter and start again at (b).

The algorithm is essentially an infinite loop. There is never any explicit decision that learning has terminated; ideally, in the absence of noise, the learner will eventually stop encountering inputs that they cannot analyze because they have succeeded in finding the right settings for the parameters.[23]

The TLA algorithm embodies the following set of assumptions about parametric learning:

---

[23] Of course, it is easy to see that this aspect of the algorithm could be straightforwardly modified by, for example, deciding to stop after a fixed number of inputs if no changes occurred.

(4.1.2) a. Single Hypothesis Search:

The learner maintains a single hypothesis grammar at all times.

b. Error-Driven Learning:

The learner only changes hypotheses when it makes an "error", that is to say when it can't correctly analyze an input because its parameter settings are incorrect.

c. Single Value Constraint:

The learner can change at most one parameter value per input.

d. Greediness:

The learner only changes hypotheses if doing so will allow it to analyze an input that was, otherwise, unanalyzable.

The first of these assumptions is simply an extreme version of the belief that the resources that the learner has to devote to parameter setting are limited. As discussion of the cue-based approach indicates, it is hardly a necessary assumption. The parametric cue-based learner, for example, does not commit to a complete setting of parameter values until it encounters the very last cue in a decision path. Indeed, cue-based algorithms do not commit to the setting of *any* particular parameter until a body of evidence has been gathered in support of it. It seems to argue the case for either approach on representational grounds: the amount of information it takes to represent a hypothesis, under either assumption, is quite minimal. There may, of course, be other reasons of psychological plausibility for requiring the learner to have a fully specified grammar at every point in development.

The second TLA assumption fits naturally, although not necessarily, with the observation that children only receive positive evidence about the forms that their target language generates.

Note also the restrictiveness of the evidence that the TLA is allowed to use. The notion of a cue, both as abstractly developed in Chapter 3, and as concretely instantiated in Dresher's system, is quite broad. The cue-based learner can make parametric changes on the basis of narrow form observations or quite broad property observations and conjectures. The cue-based learner can also require a large amount of evidence to be gathered, categorized and tabulated before any parametric decisions get made, and these decisions. In the language of cue-based algorithms, the TLA can only consider form observations. Gibson and Wexler argue that restricting the learner to making simple use of the parser is motivated on the grounds of psychological plausibility. Indeed, the machinery that the TLA learner is almost conceptually minimal so there can be no argument against it based on the amount of cognitive resources that it demands. As noted in the introduction to the thesis, adoption of the TLA involves making a strong hypothesis. Namely, that the constraints provided by UG are so strong that a simple, constrained learner can learn parametric systems quickly and accurately. There are clear scientific reasons for wanting to begin with such a hypothesis, and adopt additional cognitive machinery only as forced to by investigation of the hypothesis' empirical success.

Once these first assumptions of Single Hypothesis Search and Error-Driven Learning have been made, something like the Single Value Constraint and Greediness are necessary to prevent the learner from degenerating into a random walk through the space. To see why something like the Single Value Constraint is crucial, consider the following: if an Error-Driven learner engaged in Single Hypothesis Search through a binary space with n parameters

were allowed to freely select the next hypothesis to consider, the chance of correctly selecting the target grammar would be only $1/2^n$. On average, then, the learner must expect to make 2^{n-1} such selections before correctly selecting the target grammar—an unacceptable result given our assumptions about the size of the space. At the very least, then, something must limit the range of new hypotheses available to the learner. The Single Value Constraint does exactly that. It also imposes a meaningful notion of distance on the space; grammars that differ by m parameter values are at least m TLA steps apart. However, although the Single Value Constraint cuts down the number of choices and imposes a sort of spatial framework on the learning problem, this, in itself, does not guarantee that the learner will not perform a random walk through the parametric space. There is still nothing that forces a net motion in the direction of the target. Since the TLA freely chooses its next move from the immediately neighboring grammars, an Ungreedy learner with half the parameters set correctly has no bias to move either towards or away from the target grammar. Worse, a learner who has $n - 1$ parameters set correctly and 1 parameter set incorrectly is much more likely to attempt a move away from the target, rather than towards the target. Given that there is no useful bias in the criterion used to propose new hypothesis grammars, there must be some useful bias in the criterion used to decide whether to adopt a new hypothesis grammar. Otherwise, even with the addition of the Single Value Constraint, the learner is still performing a random walk, albeit one with considerably smaller steps. The hope is that Greediness provides an appropriate bias. Of course, variants on this particular notion of Greed might turn out to be preferable. It is possible, for example, to imagine relaxing the all-or-nothing aspect of greed. For example, changing a parameter value might lead to a better set of partial parses.

Again the strong claim that the TLA embodies is that the set relations between target grammars, source grammars and their neighbors are such that, under a realistic distribution of inputs from target languages, learners will tend to take TLA steps in the direction of the target grammar, rather than away from the target grammar; there must be some target-directional biases.[24] The simplest way to imagine this happening would be for the overlap between grammars to be a sharply decreasing function of their distance in parameter space. If this were the case, and there were no other systematic relations among the forms in the space's languages, then the expectation is that if the learner opted to attempt a TLA step towards the grammar it would be more likely to succeed than if it opted to attempt a TLA step away from the grammar. This issue of *target-directional biases* will be a focus of Chapter 5. In the present chapter, our emphasis is on identification in the limit.

Gibson and Wexler (1994) demonstrated that even from the point of view of identification in the limit in a very simple system, the strong assumptions of Greediness and the Single Value Constraint can result in the learner being unable to converge in the limit, without some additional assumptions. In their syntactic system, Gibson and Wexler (1994) discovered that for a number of targets there were a number of local maxima—states that were not on any TLA path to the target grammar. A learner in one of these local maxima would routinely encounter sentences that it was unable to analyze; since no neighboring grammar was

---

[24] It may be acceptable for this bias to be absent at a great distance, as long as there is not an active bias away from the target. For example, if the learner simply performed a random walk through a binary space, then the expectation is that, on average, half of the parameters would be set correctly. Thus, even without a target directional bias, the learner can expect to get halfway to the target "for free".

capable of analyzing any of the "error-generating" sentences either, the learner became permanently committed to a non-target grammar.[25]

In fact, it is surprisingly easy to find possible parametric systems which cause the TLA approach to break down. Consider a binary two-parameter space with four grammars, each of which generates a single unique sentence. A TLA learner who starts life with two parameters set incorrectly will never be able to acquire the target; neither of the two one-parameter changes that they might make would enable them to parse the input from the target grammar. The parametric structures provided by UG must preclude such possibilities if the TLA is to succeed.

The discussion above indicates that, in addition to certain considerations of simplicity and psychological plausibility, the TLA's properties must also be crucially motivated by a belief in their feasibility in real parametric spaces. The learner, by hypothesis, does not have the time or computational resources to engage in a full exploration of the space; variations on a random walk through the space are similarly prohibitive. However, discussion in Gibson and Wexler (1994) focuses primarily on the "prior" question of identification in the limit: Can the TLA acquire the target grammar given an unbounded amount of time and input data? As noted above, some fine-tuning of the algorithm are necessary at this stage to produce in-the-limit modification, even before more difficult questions of computational feasibility are addressed. The present chapter is also pitched at the level of identification in the limit. In the metrical phonological system, the standard TLA encounters some initial difficulties. I will present a

[25] It is also possible to have local maxima cycles where the learner interminably loops through a set of non-target hypotheses. This situation does not arise in any of the systems discussed here.

partial solution to the problem of local maxima in the implementation of the Halle and Idsardi system that depends on a both a slight modification of the parametric system and the imposition of a maturational solution of the sort proposed in Gibson and Wexler (1994). Our criterion of in-the-limit acquisition with high probability can be guaranteed given this set of additional assumptions. In the next chapter, however, I will suggest that this line of analysis, with its narrow emphasis on identification in the limit, might be premature until some more basic results are established about the general tendency for a learner to move towards the target. Gibson (1995), Broihier (1995a) and Broihier (1995b) all provide discussion of the importance of such target-directional biases.[26] Gibson (1995) presents some calculations documenting such biases in the Gibson and Wexler (1994) three parameter space.

4.2 Gibson and Wexler's (1994) results

Gibson and Wexler applied the TLA to the 3-parameter syntactic space developed above in Section 3.9. They found six *local maximum* pairs—source and target grammars such that a learner that proceeds through the parameter space via TLA-licensed steps will never be able to make a transition from the source to the target. There were 56 possible pairs to consider. These local maximum pairs are listed in Table 4.2.1.

[26] James Thomas played an important role in the development of these ideas.

Table 4.2.1
Gibson and Wexler's Maxima

Source grammar	Target grammar
Spec-final, Comp-final, +V2	Spec-first, Comp-final, –V2
Spec-final, Comp-first, +V2	Spec-first, Comp-final, –V2
Spec-final, Comp-final, +V2	Spec-first, Comp-first, –V2
Spec-final, Comp-first, +V2	Spec-first, Comp-first, –V2
Spec-first, Comp-final, +V2	Spec-final, Comp-first, –V2
Spec-first, Comp-first, +V2	Spec-final, Comp-first, –V2

Given that the algorithm allows the learner to begin in the source states of these local maximum pairs, it is impossible to ensure identification in the limit for all of the grammars in the space. If a learner begins in a local maximum for the target, or enters into a local maximum from the target, it follows from the definition of a local maximum that the learner will never attain the target. The three parameter syntactic space cannot be identified in the limit.

This negative learnability result, of course, follows from the adoption of both a particular parametric space and a particular learning algorithm—the TLA as stated in (4.1.1). In response, Gibson and Wexler consider a range of possible responses that involve modifying the assumptions that led to the problematic maxima. The responses fall into the categories indicated in (4.2.2):

(4.2.2) a. Accept the failure of identification in the limit for certain targets; perhaps the space contains natural, but unattainable languages.

b. The parametric system is incorrect; modify it.

c. The TLA is incorrect as a description of learning; abandon it.

d. The TLA is incorrect as a description of learning; modify it.

Obviously, the line between (4.2.2.c) and (4.2.2.d) is not sharp. There is a lot of room between (4.2.2.c) and (4.2.2.d) for adjustments to the TLA.

Perhaps most interestingly, in a variant on (4.1.2.d) Gibson and Wexler propose a maturational restriction on the hypotheses that the learner is able to adopt. This move is inspired by the observation that all of the source grammars in the local maxima pairs in Table 4.1.1. are +V2, and that all of the target grammars in the local maxima pairs in Table 4.1.1 are –V2. If the learner is initially prevented from adopting a +V2 grammar, then they will not start out in a local maximum for any target grammar. It also turns out that there are always TLA-licensed paths between any –V2 source and –V2 target. As a result, if the target grammar were –V2, the learner could successfully acquire the target even with the maturational restriction imposed. If, on the other hand, the target is +V2, there is no problem at all. Since there are no maxima for +V2 grammars, acquisition of the +V2 grammars is guaranteed once the maturational barrier is relaxed. There is one risk in this scheme. By the maturational hypothesis, the learner begins with a –V2 grammar, if the target is also a –V2 grammar and the maturational barrier relaxes before the target is acquired, then, it turns out, there are cases where a sequence of input could drive the learner to a +V2 grammar that is a local maximum for the target. Obviously, the maturational barrier needs to be maintained long enough to make

this possibility reasonably unlikely. Here, as in discussion of cue-based algorithms that make use of conjectures, the learning criterion has switched from straightforward identification in the limit to identification in the limit with high probability. Gibson and Wexler also consider allowing values in the parameter space to be initially unset, and, thereby, derive an alternative solution which does strictly satisfy the requirements of in-the-limit identification. I will be focusing on the maturational approach in this thesis because I believe it will have greater applicability in large parameter spaces.

Bertolo (1995) provides detailed discussion of the use of maturational barriers as a tool for circumventing local maxima. In applications reported in this thesis, however, I will not make use of anything more complicated then a simple one-parameter maturational barrier of the sort deployed by Gibson and Wexler.

At present, it is difficult to evaluate any predictions that might follow from Gibson and Wexler's maturational solution because the empirical record (see, for example, Wexler 1996 for a summary) suggests that acquisition of the basic word order parameters proceeds so rapidly that learners have fixed their values by the time that they begin to utter sentences containing enough words to provide evidence about those values.

4.3 Application of the TLA to the phonological fragment

Given example words generated from the fragment of Halle and Idsardi's system presented in Section 3.5, how successful is the TLA, in the limit, at determining the target parameter settings?

Even with the sort of thorough formulation of the system that Halle and Idsardi (1994) provide, some implementation decisions are left open. In particular, it becomes critical to be completely explicit about which grammars are neighbors. There is a fairly straightforward way to cast the system described above in terms of binary parameters. This is illustrated in (4.3.1).

(4.3.1) The Test System: Binary Formulation

a. Parameter 1: Quantity Sensitivity (YES/NO)

b. Parameter 2: Quantity Sensitivity Bracket Type (L/R)

c. Parameter 3: Edge Marking Line 0 (YES/NO)

d. Parameter 4: Edge Marking Line 0: Bracket Type (L/R)

e. Parameter 5: Edge Marking Line 0: Which Side of Selected Syllable? (L/R)

f. Parameter 6: Edge Marking Line 0: Selected Syllable (Leftmost/Rightmost)

g. Parameter 7: Iterative Constituent Construction (YES/NO)

h. Parameter 8: Iterative Constituent Construction: Direction of Insertion (L-R/R-L)

i. Parameter 9: Iterative Constituent Construction: Orientation of Bracket (L/R)

j. Parameter 10: Line 0 Headedness (L/R)

k-n Parameters 11-14 for Line 1 Edge Marking see c-f.

o. Parameter 6: Line 1 Headedness (L/R)

Initial considerations, however, led to the following set of multi-valued parameters shown in (4.3.2). In this system, languages make one choice "per process".

(4.3.2) The Test System: Two Levels

a. Parameter 1: Quantity Sensitivity

0: No bracket.

1: Place a bracket to the LEFT of the grid mark associated with a heavy syllable.

2: Place a bracket to the RIGHT of the grid mark associated with a heavy syllable.

b. Parameter 2: Edge Marking Line 0[27]

0: No edge marking.

1: Place a LEFT bracket to the LEFT of the LEFTMOST grid mark.

2: Place a LEFT bracket to the LEFT of the RIGHTMOST grid mark.

3: Place a LEFT bracket to the RIGHT of the LEFTMOST grid mark.

4: Place a RIGHT bracket to the LEFT of the RIGHTMOST grid mark.

5: Place a RIGHT bracket to the RIGHT of the LEFTMOST grid mark.

6: Place a RIGHT bracket to the RIGHT of the RIGHTMOST grid mark.

c. Parameter 3: Iterative Constituent Construction (ICC)

0: No iterative constituent construction.

1: ICC from the LEFT with LEFT brackets.

2: ICC from the LEFT with RIGHT brackets.

3: ICC from the RIGHT with LEFT brackets.

4: ICC from the RIGHT with RIGHT brackets

---

[27]Why aren't there nine options here? In principle, there are, but the missing two options—left brackets to the right of the rightmost grid mark and right brackets to the left of the leftmost grid mark—can't create new constituents in the monomorphemic words considered here. Therefore, they can't affect stress.

d. Parameter 4: Headedness Line 0

0: LEFT.

1: RIGHT.

e. Parameter 5: Edge Marking Line 1

see b.

f. Parameter 6: Headedness Line 1

see d.

The problem with a purely binary formulation—for example: +/– has iterative constituent construction, from the left/right, with a left/right bracket, etc.—is that there is really a sort of hierarchical arrangement here. If the "top" parameter is set to a particular value, then the values of the other parameters have no effect, until the learner attempts to flip the first parameter. All of the "hidden" parameter values need to be set in an appropriate way or else such a change will fail. Moving to these multi-valued parameters avoids this difficulty. Of course, it also can't help but make satisfying the identification in the limit criterion easier for the TLA since every grammar now has an expanded set of neighbors and, therefore, more potential paths to the target. Collapsing parameters in this way could make a TLA-unlearnable space TLA-learnable. On the other hand, as a result of this increased connectivity, if any local maxima *do* remain in the space, there are more ways for the learner to fall into them.

With these considerations in mind, the first space I chose to investigate is the one described in (4.2.2). Note that the conflation parameter is not included. This was done primarily for reasons of the time and space available for my investigation; my overall implementation of Halle and Idsardi's system is capable of generating conflation structures, but I do not allow this option in the spaces described here. The system is, minus conflation,

obviously very similar to the fragment analyzed in Dresher (1994), although I have not undertaken a cue-based analysis of this particular variant.

For each of the 2940 (i.e. 3 x 7 x 5 x 2 x 7 x 2) possible parameter settings in the space, I computationally generated all possible 1-6 syllable monomorphemic words composed of light and heavy syllables. Six syllables is enough to tell apart all the languages in the space that can, in principle, be distinguished. Each grammar, in turn, was considered as the target grammar. Using a version of Gibson and Wexler's (1994) search algorithm, a local maxima search was performed for each target to identify source grammars that would never converge to the target under the TLA. A new complication, not encountered by Gibson and Wexler, arises here. In this space, there are a number of sets of grammar that are weakly equivalent. That is to say, they produce exactly the same syllable-to-stress mapping, despite the fact that they have different parameter settings. These maxima are eliminated from the results reported here. If there really is no way for the learner to distinguish between weakly equivalent grammars, converging to a given grammar should be as good as converging to one of its equivalents. (It is possible that in a more complete system there would not be such weak equivalents. For example, multimorphemic words might expose some otherwise undetectable differences between grammars.) Similarly, if a grammar cannot reach the target grammar, but it can reach a grammar that generates the same language as the designated target, this was counted as a success. There are 838 distinct languages in the system.

The search discovered an abundance of local maxima. In fact, every target language had local maxima. The number of maxima per target ranged from 8 to 1008. The mean number of maxima per target language was 116. The median number of maxima per target was 66. In terms of the targets, it is appropriate to group by languages because the learner will not be sensitive

to which of several variant grammars that generate the target language speakers actually happen to use.[28]

Of the 2940 grammars in the space, 2016 were maxima for some target pattern of data—again, there were 838 distinct target patterns. For an unmodified TLA learner to successfully acquire any grammar in the space, the learner would certainly have to begin in one of the remaining 924 states. Of course, starting in a state that is not a local maximum does not guarantee that data from some target grammar in the space could not eventually drive the learner into a local maximum for the target. It turns out in this space that for each of the "non-maxima" initial states there are target grammars such that a learner driven by the target grammar could move to a local maximum for the target. In terms of the learner's internally represented mental hypotheses, it is appropriate to count by grammars.

Analysis of the space indicated that there is also no simple maturational solution of the sort that Gibson and Wexler discovered in their 3-parameter syntactic space. Every parameter value is instantiated in at least one source grammar in a local maximum pair. It will not do to simply restrict the initial range of parameter values, because this will not prevent a learner from initially adopting a grammar that is a local maximum for some target.

As mentioned above, Bertolo (1995) outlines a program for exhaustively searching through all possible sets of maturational sequences for those, if any, that would incrementally steer the learner successfully from some subset of the 924 candidate starting points to

---

[28] This is true unless speakers with distinct grammars that speak weakly equivalent languages have tendencies to produce different distributions of forms as a consequence of grammatical differences. These differences might, for example, affect the ease of producing or comprehending certain forms.

appropriate target grammars. I have not carried out this search for the present "maximum-ridden" space.

Instead, in the interests of attaining a better concrete understanding of the problematic interactions between parameters in the system—something that proved difficult to do in the space presented above—I next moved to a smaller subspace. The parameters shown in (4.3.3) are those involved in the mapping from Line 0 to Line 1. In languages that do not conflate, this mapping is, in some sense, independent of the mapping from Line 1 to Line 2. The parameters that map from Line 0 to Line 1 make a contribution to the overall stress contour by virtue of the grid marks that they project to Line 1—the secondary stresses. In languages that do not conflate, then, their contribution is plainly visible. It is conceivable that learners that have somehow previously established that their language does not conflate might employ a TLA learner to set the Line 0 parameters. At this point, though, it is good to keep in mind the question of whether the remaining subspace could, perhaps, prove small enough to make more exhaustive search techniques seem tractable.

(4.3.3) The Test System: One Level

a. Parameter 1: Quantity Sensitivity

0: No bracket.

1: Place a bracket to the LEFT of the grid mark associated with a heavy syllable.

2: Place a bracket to the RIGHT of the grid mark associated with a heavy syllable.

b. Parameter 2: Edge Marking Line 0

0: No edge marking.

1: Place a LEFT bracket to the LEFT of the LEFTMOST grid mark.

2: Place a LEFT bracket to the LEFT of the RIGHTMOST grid mark.

3: Place a LEFT bracket to the RIGHT of the LEFTMOST grid mark.

4: Place a RIGHT bracket to the LEFT of the RIGHTMOST grid mark.

5: Place a RIGHT bracket to the RIGHT of the LEFTMOST grid mark.

6: Place a RIGHT bracket to the RIGHT of the RIGHTMOST grid mark.

c. Parameter 3: Iterative Constituent Construction (ICC)

0: No iterative constituent construction.

1: ICC from the LEFT with LEFT brackets.

2: ICC from the LEFT with RIGHT brackets.

3: ICC from the RIGHT with LEFT brackets.

4: ICC from the RIGHT with RIGHT brackets

d. Parameter 4: Headedness Line 0

0: LEFT.

1: RIGHT.

In all, 126 of the 210 languages in this space are distinct.

Identical analytical steps were followed here. The results of the search indicated that 51 of the target languages (corresponding to 80 of the grammars) had local maxima. Again, in a

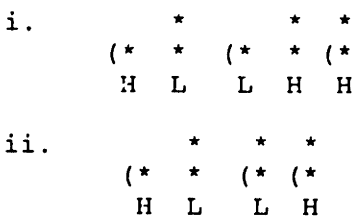
number of cases, there were quite a few maxima for a given target. For this space, the pattern of local maxima more readily suggested a solution. Note, however, that this is one solution among a space of other, still unexplored maturational, possibilities.

The solution that I will present here involves a combination of parameter space modification and maturation. A modification of the parameter space that eliminates certain options is made. With this done, a one-parameter maturational solution is readily available, at the cost of a slight expansion of UG. Of course, this also involves a slight modification of the IGMS program because the maturational restriction is, by definition, a modification of the learning algorithm that is sensitive to the particular details of the parametric space.

A particularly bad parameter interaction accounts for most of the maxima in the system. A typical result of iterative constituent construction is the formation of new binary constituents. However, two settings of the ICC parameter can also result in the formation of single syllable constituents at the end of a word. If the ICC mechanism proceeds from left to right inserting left brackets, it can result in the creation of a single syllable constituent at the end of the word. The mirror image situation obtains when the ICC goes from right to left inserting right brackets. Effectively, these *forward-grouping* settings force the end syllable to be included in a metrical constituent. These ICC settings, in combination with certain other parameter settings, as shown in (4.3.4.a) for example, can result in grammars that have what I will call an *unavoidable* final stress. For example, all words generated by the grammar in (4.3.4.a) have a final stress. Moreover, the same holds for all of (4.3.4.a)'s neighbors. If all words in the target grammar lack a final stress, as in (4.3.4.b), a learner who adopts (4.3.4.a) will be irretrievably stuck; there is no one-step way out. If the forward-grouping option is turned off, other options step up to ensure that the end syllable receives stress. More than one parameter must be changed at a time to eliminate this stress.

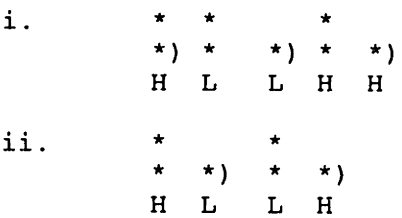
(4.3.4) a. Source Grammar with "unavoidable" final stress.

Quantity Sensitivity:	0—Quantity Insensitive
Edge Marking:	2—LEFT Bracket to the LEFT of the RIGHTMOST grid mark
ICC:	1—ICC from the LEFT with LEFT brackets
Headedness:	1—RIGHT



b. Target Grammar with no final stresses.

Quantity Sensitivity:	0—Quantity Insensitive
Edge Marking:	6—RIGHT Bracket to the RIGHT of the RIGHTMOST grid mark
ICC:	4—ICC from the RIGHT with RIGHT brackets
Headedness:	0—LEFT



As mentioned above in Chapter 3, the two problematic ICC options depicted in (4.3.4) are among the least well-motivated aspects of the system; they were tentatively suggested as part of a possible analysis for Ojibwa. Therefore, it seems reasonable to consider how learnability changes in the system if they are not included. When this was done, the resulting 126 grammar (3 x 7 x 3 x 2) space has the simple set of maxima shown in Table 4.3.5, which I will show straight away has a simple maturational solution. (Interestingly, the space still generates 104 distinct languages. Only 22 patterns of data were eliminated despite the fact

there are 84 fewer grammars in the system.) In principle, this might be an argument for modifying linguistic theory to accommodate learning results, although there are reasons to be skeptical. I will have more to say about this below.

As discussed above, in certain parametric spaces it is possible to use maturational barriers to prevent a TLA learner from entering into an inescapable local maxima. First, of course, it is necessary to restrict the range of initial starting states, so that the learner never begins in a state that is a local maximum for some target grammar. (If there happened to be states that weren't on a path to any local maxima for any target grammar, it would be enough to simply start in one of these states. As determined, by a computational search of the system, there are no such states in the system at hand. Every start state is on a path to a local maxima for some target in the system when the learner is driven by data from that target.) Second, it is necessary to block any paths to local maxima. Again, maturational restrictions on the range of parameter values aim to perform this task in the following way: initially, when the learner could potentially enter into a local maximum, the maturational restriction prevents them from doing so. Later after some innately fixed amount of time has passed, the hope is that the structure of the space is such that the learner will have reached a point in the space where they are no longer on a possible path to any local maxima. Then, it is safe to remove the maturational restriction on the learner.

Table 4.3.5
Local Maxima without the forward-grouping ICC options:

	QS	Edge Mark	ICC	Head
Target:	0	4	From R	1
Source:	0	0	From R	0
Target:	0	4	None	1
Source:	0	0	From R	0
Target:	0	4	None	0
Source:	0	0	From L	0
Target:	0	3	From L	0
Source:	0	0	From L	1
Target:	0	3	None	1
Source:	0	0	From R	1
Target:	0	3	None	0
Source:	0	0	From L	1

* The two stressless grammars that result from forming no constituents are also maxima for many targets. I will assume that UG recognizes that these are unacceptable stopping states and will restart the learning process if it gets caught up here.

The modified system allows just such a solution. Notice, as indicated in the boxed column under *Edge Mark*, that all of the grammars displayed in the table that were maxima for any targets had no edge mark. A computational analysis indicates that initially requiring an edge mark, with two notable exceptions, allows the learner to avoid all maxima in the space. These two exceptions are *stressless* grammars that form no constituents at all because they do not adopt any bracket-inserting options. These grammars may actually be linguistically motivated (see Idsardi 1992 for discussion), but they might, not implausibly, be viewed as

something of a special case. If these grammars are eliminated from consideration, then, at least tentatively, the modified TLA remains a candidate learning algorithm for metrical phonology.

As in the syntactic domain, what we know about the acquisition of phonology does not seem to bear on the claim that children begin life with the edge marking option turned off. Fikkert (1994), for example, in one of the most extensive studies in this domain, provides a detailed discussion of the acquisition of Dutch metrical phonology by a number of children. The general result of the study, as in the syntactic case of word order parameters, is that very little evidence can be found in this domain suggesting a parametric missetting of the sort that would distinguish two adult grammars. The development that is displayed seems to be along a non-parametric dimension governing how much of a word that the child realizes in its utterances. This difficulty in testing the predictions of the TLA is not particular to this approach; if children, empirically, set parameters quickly, before they are producing complex linguistic structures, it will be difficult to catch children in the act of parameter change. This does not constitute an argument for or against the approach.

4.4 The Impact of Extensions to the Parametric System on the Identification in the Limit Approach

Above, I suggested tentatively that we had an example of a learnability result possibly putting some constraint on linguistic theory. Indeed, an important question for learnability research is whether formal analysis of the interaction between learning algorithms and linguistic models will significantly inform and guide research in linguistic theory and developmental psycholinguistics, or whether learnability research will primarily

play the role of a check on the adequacy of theory in these areas; if no reasonable learning algorithm consistent with what we know about the brain can be provided that leads from a parametric system to an explanation of the course of acquisition, we will know that something has gone wrong. Linguistic theory has undergone very dramatic transformations over the course of the past several decades, and it is safe to say that it continues to do so today, so it is important to ask what can be learned from attempts to model the learnability of a "moving target". Learnability research could certainly result in a body of theoretical results detailing the connections between particular instances or classes of learning algorithms, particular instances or classes of parameter spaces and their associated generative probability distributions. The worry is that results will not fall somewhere in the useful middle ground between overly particular and overly abstract and general.

Of course, there have been instances in which learnability considerations led to quite concrete proposals about linguistic constraints. For example, Wexler and Culicover's (1980) learnability results pinpointed the need, given plausible psychological desiderata, for a restriction on the scope of transformational operations to ensure the identification in the limit of the infinite class of transformational grammars that they studied. The question is whether such results can be derived in the more restrictive Principles and Parameters framework when identification in the limit is much more easily attained.

Given a complete specification of (1) the topology of the hypothesis space provided by Universal Grammar, (2) a model of the distribution of linguistic patterns produced by a speaker at any given point in parametric space and (3) a learning algorithm that maintains grammatical hypotheses that it changes as a result of its exposure to linguistic data, it is possible, as we have seen, to compute the trajectories that the learning algorithm can and

cannot follow through the hypothesis space. The TLA's hypothesis space can be identified with the parametric space that it is navigating. In the more general case, this need not be true. For example, the hypothesis states maintained by a cue-based learner essentially consist of lists of parameters that either have definitive values or are currently unset. Since the unset option is not available in the parametric system, the space of hypothesis states and the space of grammars cannot be matched up one to one. Nonetheless, it would still be possible to consider the cue-based learner's dynamics in its hypothesis space.

If linguists, aided perhaps by a corps of statistical corpora analysts, eventually become quite confident that they have (1) and (2) right, and if developmental psycholinguists can provide a good description of (4) the trajectory of linguistic hypotheses that children go through on the way to adult competence, then, in principle, it should be quite straightforward to determine whether the learning algorithm specified in (3) fits the bill.

Similarly, if the field eventually became quite certain about the specification of (2), (3) and (4) it might be possible to rule out particular instances, or even entire classes of parametric spaces. For a particularly simple example of how this might play out, consider the following highly implausible, but possible example of how the world could have turned out. Assume that neurological investigations reveal that the standard TLA is correct in all its details, and that developmental psycholinguists discover that all children, everywhere, are entirely successful in acquiring their target languages in the limit. In such a world, any linguistic parameter space which had the property that all the data generated by any particular grammar was uniquely generated by that grammar would be immediately recognizable as a disastrous failure. Any learner who happened to start off more than one step away from the target grammar would never move because, by hypothesis, the only state

capable of generating data in the target language is the target language itself. A learner in any state that did not border the target grammar would constantly receive error data, but since, by hypothesis, only the target grammar is capable of providing an analysis of this error data, learners in the nether regions of the space would remain fixed in place. If we knew that the TLA were correct, there would be a straightforward argument for rejecting such a parameter space.

Experience to date suggests that the first scenario described above is considerably more plausible than the second. Linguistic theory appears to have made a considerable amount of progress in the description of UG, in the absence of much detailed and concrete evidence about the nature of the learning algorithms employed by humans learning natural language. It has done so largely through the pursuit of its primary methodology of analyzing the range of linguistic patterns and interpretations of linguistic patterns that speakers of a language find to be acceptable, unacceptable, or somewhere in between. Given the current state of our psychological knowledge, it seems harder to imagine independently establishing anything beyond the most general properties of human natural language acquisition algorithms—for example, motivating Principles and Parameters theory. It seems most likely that a linguistic theory would be dropped on learnability grounds only in those cases where it was impossible to provide *any* plausible learning algorithm.

A related concern is whether usefully general learnability results for a particular algorithm, such as the TLA, can be attained in the absence of a full and correct specification of the parametric system. As one pressing example of this concern, the reader of Section 4.3 is probably left with the feeling that the analysis carried out above and the demonstration of a positive learnability result for the maturationally extended TLA in the amended phonological

space are tenuous. I do not wish to dispel this feeling. The results presented in Section 4.3 are highly, indeed unavoidably, dependent on the fine details of the parametric space. Instead, I hope to discuss the conditions which govern whether or not a learning result for the TLA or MTLA will carry over in any recognizable form when the parametric system is modified. In particular, let us see how far we can pursue the optimistic possibility that our success in providing a maturational solution in the final fragment discussed above will carry over to larger systems.

The TLA essentially induces a graph structure (actually a set of related graph structures—one per target grammar) on the learner's hypothesis space. Grammars in the parametric space correspond to the vertices of this graph structure. Directed edges run from a first grammar to a second if it is possible to make a TLA transition from the first to the second. Modifications to the hypothesis space can affect identification in the limit results for the TLA or the corresponding identification in the limit with high probability results for the MTLA only to the extent that they affect these graph structures.

An extension of the system could take several forms. Perhaps most simply, the sets of linguistic forms contained in existing languages in the model might be altered, while the set of languages and the overall neighborhood structure remains the same. Such an alteration of the system would adding or deleting linguistic forms from the languages in the space. It seems quite natural to expect such alterations as the modeling enterprise becomes more ambitious. For example, Gibson and Wexler (1994) confine themselves to matrix clauses although clearly all the languages in the space that they present also really generate multi-clause data. Similarly, in both Dresher's (1994) and my own efforts with the Halle and Idsardi system, the focus has

been entirely on monomorphemic words. Deletions, on the other hand, would only seem to arise if the action of the parameters was initially incorrect.

For simplicity, I will consider the addition or deletion of a single piece of data to a single grammar in the space. The overall effects of the addition of data throughout the space are the same whether the new data are added incrementally or *en masse*.

What consequences can the addition of a single piece of data have for the TLA-connectivity of the space? Most simply put, the addition of data can result in the addition or removal of edges in the graph structures reflecting TLA-connectivity for a particular target grammar. The consequences are potentially most dramatic in the graph structure which has the extended language as the target. Adding a new pattern to a designated target grammar provides the target with a larger arsenal of potential error-inducers that could cause a learner that is stuck in a maximum to attempt to change hypotheses. In order to have any effect, on the graph structure, of course, a previously inaccessible neighbor of a state in the space must also generate the form that is added to the target grammar. In this case a link from the state to its neighbor is added. If a state that receives a new *outgoing* link was a maximum in the original space, then the addition of this new link may make it a non-maximum in the new space. In order for the maximum to be eliminated, at least one newly accessible neighbor must, itself, not be a maximum.

Clearly, adding a form to the target grammar and not to other grammars in the space can *only* result in the addition of new edges in the corresponding graph structure. Therefore, it can only eliminate—never create—maxima. Note also that adding a form to the target grammar has global effects on the connectivity of the space; it can potentially affect the

connections between any two neighboring grammars, regardless of their distance from the target.

Adding a new linguistic form to a non-target grammar can both add and delete links between states, although it can only have a much more local effect on the connectivity of the graph structure corresponding to that target. It can only affect the links between the extended grammar and its one-parameter neighbors.

The set of outgoing links from the extended grammar can actually only be reduced. This is the case because extension of the grammar must either leave the set of error-signals the same or reduce its size by one. If the neighbors of the extended grammar are unchanged, then the ability of these potential triggers to actually induce change is unaltered. If the set of outgoing links that gets eliminated disrupts all previously existing paths to the target, then at least one new local maximum is created. In addition, any other state in the space that crucially relied on a path through the new local maximum also becomes a local maximum.

The set of *incoming* links to the extended grammar can only be expanded. A new incoming link will be formed if the newly added form is generated by the target grammar, but not by a neighbor that previously lacked an incoming link. If the extended grammar is not itself a local maximum then any neighbor that manages to link into it by virtue of the extension is no longer a local maximum. Similarly, any local maximum in the space that is capable of reaching a newly linked neighbor of the extended grammar will shed its maximum status.

This exhausts the possibilities for edge modification that can be induced by the addition of a single form to a state in the space. The effects of a larger set of additions can be incrementally computed by adding a single new form at a time. However, some interesting observations can be made here about the effect of adding a set of forms *en masse*. If the forms

that get added to the space do not occur anywhere in the previous space, then they can only result in increased connectivity. Connectivity only decreases when the addition of a form to a grammar prevents that form from generating an error for a source grammar in the space. If all the forms that are added do not occur in the original space, then previously existing triggers will not be disrupted and new ones may be created.

The effect of removing a single form from a grammar is a sort of "mirror image" of the effect of adding a single form. Removing a single form from the target grammar can result in the elimination of edges throughout the space since it robs the target grammar of a potential error-inducer. Removing a single form from a non-target grammar potentially increases its set of outgoing links because the modified grammar generates less data, and, therefore, is potentially more prone to encounter errors on data from the target. The set of incoming links is potentially reduced since the modified grammar now generates fewer sentences for a neighbor grammar, seeking to make a hypothesis change, to latch onto in accordance with the principle of Greediness.

Obviously, learnability results are sensitive to these changes in connectivity. The addition of edges can turn a space that was previously unlearnable by the TLA into one that is learnable by the TLA. If a space was previously TLA-learnable, the addition of edges will not change this. On the other hand, the elimination of edges potentially changes a TLA-learnable space into a TLA-unlearnable. If a space was previously TLA-unlearnable, then the removal of edges will not change this. Things are obviously more complicated if edges are both added and eliminated.

Bertolo (1995) provides a sufficient condition on the extension of the data sets for existing languages in the space not to disrupt previously obtained learnability results with the

TLA. Essentially, if the data that gets added to a language does not occur in any of the other unmodified languages, and if the data does not get added to any other language, then the data will be inert from the point of view of the TLA learner. As discussed above, connectivity will not decrease in virtue of the fact that the newly added forms do not occur anywhere in the old space. Connectivity also will not increase, however, because none of the new patterns for any target grammar in the space are generated elsewhere in the space. A greedy learner would not be able to take advantage of the new patterns. Data that is unique to a particular language cannot lead to the modification of the grammar. Clearly, this is far from a necessary condition for ensuring the continued success of the TLA.

Bertolo's interest is in specifying the conditions under which a learnability result, either positive or negative, will be preserved under extensions of the system. One could also imagine taking a more partisan, pro-TLA stance and focus on a search for conditions that would preserve positive learnability results and eliminate negative learnability results. This would clearly allow for a relaxation of the conservative criterion above.

The effects of adding and subtracting forms from the grammars in the space are somewhat more complicated when the TLA is supplemented with a maturational timetable. The effects on the *mature* connectivity graphs that result when all barriers have been dropped are, of course, exactly the same as those described above for the original TLA. For the TLA, of course, increased connectivity can only be beneficial for identification in the limit. If the original space had no local maxima, then none will be introduced by increased connectivity. If the original space had local maxima, increased connectivity could potentially eliminate some of these maxima with no risk of creating new ones. Similarly, decreased connectivity can only be detrimental.

Whereas a successful TLA solution is robus. in the face of increased connectivity, a maturational TLA solution may not be. With the maturational TLA, there is a tension between two potential effects of increased connectivity. Increased connectivity could eliminate local maxima in the mature space. If all the maxima in the mature space are eliminated by changes in connectivity induced by the addition or subtraction of forms, then a maturational solution for the original space will trivially extend to the new space. Of course, in the new space, the maturational solution will not play any useful role. Instead, it will simply temporarily impede the learner's progress through parameter space.

The addition of new edges, however, clearly does not guarantee the elimination of *all* the original maxima. Moreover, the elimination of edges has the potential to create new local maxima that did not exist in the original space. If all maxima are not eliminated, then increased connectivity can actually have a detrimental effect on a maturational solution. I will consider how this could happen for both residual maxima from the original space and newly created maxima.

Consider, first the case where residual maxima remain, but no new maxima are created.

Recall, as analyzed by Bertolo (1995), that maturational restrictions function to steer the learner around local maxima in a space. When a successful maturational solution's final restriction on a parameter value gets released, the learner is, with high probability, no longer on track to a local maximum. None of the terminal states that the learner can reach in the presence of the maturational barrier is on a path to the maturational barrier. This is true regardless of which target grammar the learner is exposed to. If there is more than one "relaxation step" in the removal of maturational barriers, then the penultimate set of parameters that are unlocked might have played an important role in guaranteeing that the

set of terminal states that the learner reaches before the final restriction drops are "safe". This penultimate maturational barrier might, in turn, depend on a prior maturational barrier ensuring that the learner ended up in an appropriate terminal state for the penultimate barrier. And, so on. Each maturational barrier in the increasingly less restrictive sequence of barriers has a set of terminal states that the learner will enter into given enough time, and given the successful operation of previous maturational barriers.[29] For all but the last set of terminal states, a set of terminal states is safe if the set of states that can be reached from the terminal set in between the current relaxation of the maturational barrier and the next relaxation of a maturational barrier is itself safe. The last set of terminal states is safe if they are not on a path to a local maximum in the fully mature space.

A new edge can disrupt a maturational solution by altering the set of terminal states for a maturational barrier to include an unsafe state. By allowing the learner to pursue new paths—when those paths are not blocked by the maturational barrier, of course—the addition of a new edge has the potential to lead the learner to an unsafe state that it could not have reached previously.

The loss of an edge also has the potential to disrupt a maturational solution even if it does not create new local maxima. Instead of reaching the original set of terminal states for a maturational barrier, the learner could be stopped short on the path to the original terminal states in a new, unsafe terminal state that is on a "maturationally licensed" path to a local maxima.

---

[29] The learner might not necessarily reach a single steady state, because there could be cycles between safe states.

If the loss of an edge does create new local maxima, of course, then a maturational solution could be disrupted if it does not steer the learner clear of newly created maxima.

The set of languages in the space might also be increased by the addition of a new parameter to the system. Perhaps, the simplest case of parameter extension would be the case in which all the grammars in the original system can be directly interpreted as implicitly adopting a single value of the new parameter. For each new alternative value of the parameter, there would be a separate subspace whose learnability might be investigated. The simplest sub-case, here, occurs when there is only a single alternative value. In Bertolo's (1995) terminology, the question of interest is whether learning results obtained from *sections* of parametric systems hold of the parametric system as a whole. A section, relative, to a target, corresponds to the subgraph induced when states that do not have a subset of their parameters set to certain designated values are removed from the system.

Consider first the consequences for the simple TLA.

If each section of a larger space corresponding to one of the values of a particular parameter has no maxima for targets within it, then the larger space will have no maxima *if and only if* every state in the larger space is always on a path to some state with the alternative parameter value when driven by data from a target in a different section. In other words, if the learner can always manage to set the new parameter appropriately, the rest of the required learning paths are guaranteed by the hypothesis that there are not *within-section* maxima.

Note that this is a global condition that, as stated, can only be evaluated by implementing the larger system.

If sections of a larger space do happen to have local maxima, the larger space itself may or may not have local maxima. The situation for within-section maxima can only improve, since the merger of the sections essentially gives every state in the space more neighbors. Old within-section links are preserved, and new *between-section* links are potentially added. A learner may be able to make transitions between states that were previously impossible by leaving the original section, following a path in the new section, and then returning to a state in the original section that is on track to the target. A "maximum-voiding" path might even require the learner to jump back and forth between sections numerous times. Of course, the trade off is that any of the between-section source/target pairs could themselves turn out to involve local maxima.

What consequences does section-wise extension of the space have on a maturational solution that was discovered for one of the sections? Again, increased connectivity can have mixed effects. Of course, if the merger of the sections eliminates all maxima in the mature space navigated by the unmodified TLA, then a maturational solution is trivially preserved. The barriers, of course, no longer play any useful role. For those within-section maxima that remain, however, any increased connectivity via the other sections can only lead to trouble. The maturational barriers discovered for a section are designed to restrict the set of paths that a learner can pursue so as to prevent them from entering into a local maximum (with high probability). Increased connectivity alters the sets of terminal states for maturational barrier. Since the maturational solution ensured that all sets were safe in the section that it was developed for, if the safety of any of these sets changes it can, obviously, only be for the worse. Obviously also, the maturational barrier may fail to prevent within-section maxima in sections of the space other than the one it was designed for. And, again, merger of the spaces raises the

possibility of between-section maxima that are not accommodated by the current maturational solution and that cannot be remedied by maturationally restricting the learner to begin in a particular section. [30]

Of course, it is possible that while a merger of sections might void particular maturational solutions developed for particular sections, it might lead to a space that lends itself to different, simpler maturational solutions. Note, however, that if every parameter value is instantiated in some local maximum for the system, no maturational solution will be available, unless, perhaps, some further non-parametric restrictions is made on the set of possible start states.

More generally, changing the system both the basic neighborhood structure and the contents of the states that populate the structure may change. In summary, it seems that there is reason to be skeptical about the ability of particular maturational results to extend to larger spaces.

In the domain of phonology, as was also discussed with the cue-based approach, certain other domain-specific questions about the extendibility of results arise.

For example, in many empirically attested stress systems there are a considerable number of lexical exceptions that stand out against the background of parametrically predicted forms. Lexical exceptionality often allows the stress to surface in a variety of positions that would not be predicted parametrically. To the extent that these lexical exceptions mimic forms that are parametrically generated by other languages in the system, the connectivity of the space that confronts a TLA learner that is naïve about lexical exceptionality can only increase.

---

[30] I have not treated the case where there are additional restrictions on the set of starting states that are not reflected in the maturational locking of parameters.

The target language is essentially armed with an extra set of "error-inducers". As we've seen above, increased connectivity is typically beneficial for a simple TLA learner. For the maturational TLA learner, there is a tradeoff between increased connectivity's potential to eliminate local maxima and its ability to thwart maturational barriers. In this case, however, there is an additional wrinkle: a TLA learner now faces the prospect of encountering errors even when they have converged to an appropriate grammar in the system. If these error-generating patterns are generated by neighbors of this target grammar, then the learner is free to wander back off into parameter space.

It is not clear how to accommodate lexical exceptionality in the TLA framework. Perhaps lexical entries should themselves be parametrized to allow for a description of the possible range of lexical exceptionality. A learner who found themselves unable to generate a stress pattern for a particular form would, then, be confronted with the options of changing the parametric system as a whole or specifying that the lexical item is exceptional, perhaps in some UG-restricted way.

A related problem arises from language-particular, but general exceptions in the application of the bracketing processes. For example, Idsardi (1992) argues that Latin prohibits the placement of a left bracket to the right of the final syllable of a word. This prohibition is capable of blocking the placement of a quantity sensitive bracket and, in interaction with the other settings for Latin, ensures that the final syllable of a multi-syllabic word does not receive stress. In (4.4.1), the settings for Latin are shown.

(4.4.1) Latin: Final Extrametricality

Line 0: Project L for Heavy Syllables; Edge Mark RLR; Iterate RL Binary; Head L; Avoid *(*#.

Line 1: Edge Mark RRR; Head R.

The structure in (4.4.2), shows the result of generating the Latin stress pattern for the word *reprimuntur*. The 'X' illustrates a position where a bracket would have been placed as a result of Latin's quantity sensitivity.

(4.4.2)

```
#                       *                   #
#           *           * ]                 #
#[          *     * [   * ] X   * [         #
#           L     L     H       H           #
```

Without the Latin-specific restriction, the stress pattern would be as in (4.4.3).

(4.4.3)

```
#                                 *         #
#           *           *       * ]       #
#[          *     * [   * [     * [       #
#           L     L     H       H         #
```

It turns out that it is not possible for the TLA to set parameters and then learn, in some unspecified way, this language-specific pattern of exceptionality. A learner attempting to accommodate data from actual Latin and converge to the parameter settings for Latin minus the bracket disruption will not be able to do so. Since there is no mechanism in the TLA for detecting that the basic parameters are correctly, set, then if any neighbor of Latin minus bracket disruption can accommodate data from actual Latin that Latin minus bracket disruption cannot, then even if the learner manages to reach Latin minus bracket disruption, there is no guarantee

that they will remain there. For example, consider the stress pattern for the fictitious word *muntur*. In Latin, it would receive the stress in (4.4.4). In Latin minus bracket disruption, however, it would receive the stress in (4.4.5). Moreover, there are neighbors of Latin minus bracket disruption that generate the actual Latin pattern, as in (4.4.6). Therefore, there is always a trigger for the learner to abandon the desired parameter settings.

(4.4.4)

```
#          *                #
#          * ]              #
#[         * ]      * [     #
#          H        H       #
```

(4.4.5)

```
#                   *       #
#          *        * ]     #
#[         * [      * [     #
#          H        H       #
```

(4.4.6)

```
#          *                #
#          * ]              #
#          * ]      * [     #
#          H        H       #
```

Chapter 5
Markov Chain Analysis of the Triggering Learning Algorithm

The strong hypothesis that goes hand in hand with the use of either the TLA or MTLA is that neighboring states in a parametric space are connected in such a way that grammars that are close in parametric space will at least *tend* to overlap in data space in a particular way. In other words, UG must strongly constrain parametric variation so that the learner tends to follow trajectories to the target grammar. The basic requirement necessitated by the combination of Greed and the Single Value Constraint is that neighbors of incorrectly hypothesized grammars tend to accommodate more error data when those neighbors are closer to target, rather than when they are farther away. If this is not the case for a particular incorrect hypothesis, then the learner will not be any more likely to accept a proposed hypothesis that is closer to the target grammar than one that is further away.[31] An important possibility, not so extensively considered in the literature, is that this hypothesis could generally turn out to be the right move to make even though the precise details of the TLA turn out to be wrong from the point of view of identification in the limit.

The details could turn out to be wrong, for example, if every grammar turned out to be a local maximum for a small, but non-negligible, number of targets in the space. In this case, there would be no potential starting states for a TLA learner, even if most regions in the space turned out to be beautifully suited to a learner guided by the combination of the Single Value Constraint and Greed.

---

[31] In situations where several weakly equivalent grammars generate the target language, their is more than one grammar that it is appropriate to tend towards.

This chapter is an initial effort to examine whether the basic pattern of "flow" that the TLA pursues in parametric spaces will lead to tractable acquisition. Section 5.1, following Niyogi and Berwick (1993), will suggest that not all maxima are created equal when it comes to the sorts of problems that they raise for learners. It is possible for a maxima to be relatively inert, in the sense that they are primarily only problems for learners who begin life inside of them. It is also, however, possible for there to be maxima that attract Greedy learners from throughout the hypothesis space. Section 5.2 argues for an emphasis on time course of acquisition rather than the presence or absence of local maxima in initial investigation of the feasibility of the guiding insight behind the TLA. Section 5.3 suggests a simple modification of the TLA, involving a probabilistic violation of Greed, which ensures identification of the limit with probability *1*. This section also places Niyogi and Berwick's (1993) comparison of the TLA and random walk algorithms in an appropriate context. Section 5.4 provides some informal motivation for believing that TLA-appropriate biasing conditions might exist in both Gibson and Wexler's three-parameter space and a variant of the metrical phonological space of Halle and Idsardi that was considered in Chapter 4. Section 5.5 indicates how it is possible to improve upon the sort of coarse considerations presented in Section 5.4. The rules governing the TLA's operation are such that analysis of the probability of TLA convergence to the target and the expected number of steps to convergence can be carried out exactly within the context of a well-developed branch of mathematics dealing with Markov Chains. More on this below.

5.1 A Finer-grained Look at Local Maxima

The TLA imposes a geometrical pattern of connectivity between the states in parametric space. Some transitions between neighboring states are possible; some transitions between neighboring states are not. Of course, given the Single Value Constraint, no transitions at all can be made between non-neighboring states. As discussed above, Gibson and Wexler provide a detailed case study that shows how the geometry of possible transitions that the TLA imposes can lead to certain types of trouble for the learner. These *geometrically* discernible difficulties with the TLA come in the form of local maxima; in certain parameter spaces, the TLA learner can either start in or, during the course of acquisition, find its way into states which are not on any path to the target grammar (or a suitable equivalent grammar). More formally, a local maximum set for a parametric system P , with respect to a target T, can be defined as a set of states M which does not contain the target T, such that if a state A is in M and a state B is not in M, then there is no sequence of TLA transitions from A to B. Equivalently with respect to the set of the states that occupy some maximum set for a given target, there is no sequence of transitions that begins in M and ends in T. Call the states that fall within some local maximum set M, for a parametric space P, local maxima. Clearly, once a learner enters a local maxima for a given target, failure for the TLA is inevitable. If the purpose is to ensure identification in the limit, then maxima must be avoided at all cost.

There are several ways that a learner might find themselves in a local maximum for the target.

First, the learning algorithm could either begin with or, during the course of acquisition, adopt a grammar that generates a superset of the target language. Since such a superset grammar generates every linguistic pattern generated by the target grammar's implemented in the minds of the members of the learner's linguistic community, the parser will never report any errors, so no hypothesis changes will be attempted. If such a hypothesized grammar does not generate a *proper* superset of the languages generated by these target grammars, there is, in principle, no way that a learner could be expected to distinguish data from the hypothesized grammar and the target grammar. Either the parametric model is not yet rich enough to distinguish between the operations of two different sets of parameters that would have different effects in interaction with additional parameters, or, perhaps, learners simply converge to different parameter settings that generate the same data. In this second case, it is perhaps possible that clever psycholinguistic experimentation would be able to tell which of several weak equivalents a learner has settled on. If, on the other hand, the hypothesized grammar is a superset grammar for the target, the consequences for the TLA are clear: since the TLA has no mechanism for abandoning overly general hypotheses, a learner who adopts a superset grammar will fail to acquire the target.

These subset/superset problems, however, are ones that would confront any learning algorithm that was strictly error-driven. If a learner adopts a grammar that generates all the sentences generated by the target language (and then some, in the case of proper supersets), then a learner who depends entirely on parsing failures to initiate grammatical experimentation will (in the absence of grammatically analyzable noise data that is not part of the target language) never change its hypothesis. The TLA (depending of course on what natural language parametric systems actually look like) is potentially subject to additional difficulties. It may

be possible for the learner to find itself in points in parametric space where plenty of errors are encountered, but, nor.etheless, there is no sequence of TLA transitions that will lead to a grammar capable of generating the target language. Clearly, a learner in such a state will never successfully acquire the target.

It is this second type of maxima—the consequence of pursuing the joint policies of Greed and the Single Value Constraint—that I intend to focus on here. Here, too, I would like to make several distinctions.

By definition, there are no transitions *out of* a local maximum set *M*. It is possible that, for a given target, there could also be no transitions *into* local maximum set *M* from states outside of *M*. If this is the case, then the local maximum set *M* is *isolated*. The only way that a TLA learner can find itself in *M* is to start in *M*. If the TLA learner does not begin life in an isolated maxima set, it will find that it can't get there from here. Figure 5.1 represents a case where there are no paths from the target *T* to grammars in the local maximum set *M*. This is graphically represented by the absence of any connecting edges. There are also no links from the grammars *B1* and *B2* to grammars in *M*.

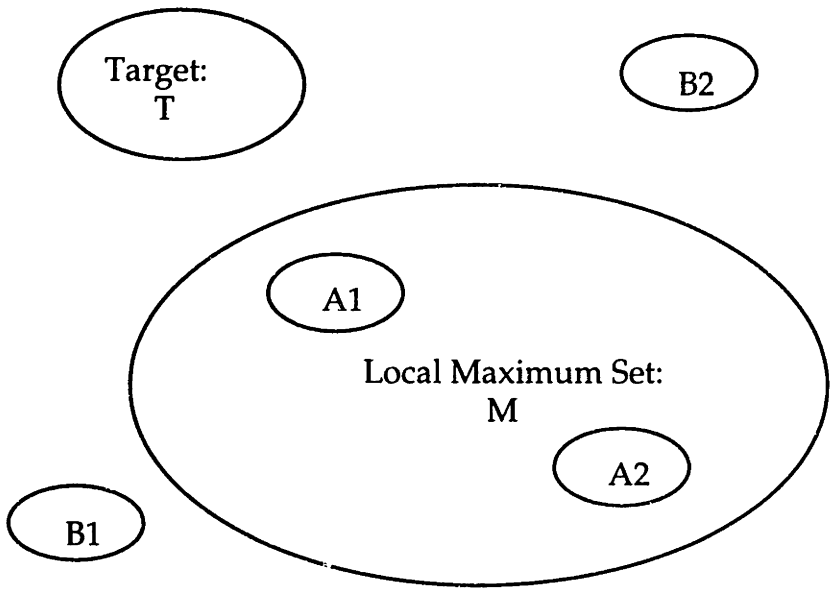


Figure 5.1: In isolated maxima, there are no transitions into the maximum set from without.

Conversely, if some sort of *deus ex machina* chose to intervene on behalf of the TLA or MTLA learner and move it to a hypothesis that fell outside of a local maximum set *M*, it would

never return there via TLA-licensed steps. For the TLA, then, isolated local maximum sets are only a danger for learners with the misfortune to begin life inside of them.

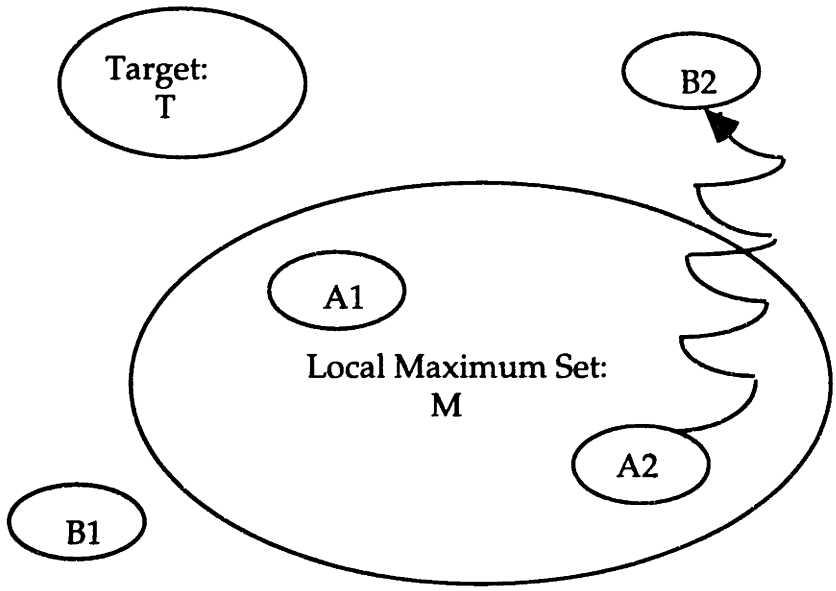


Figure 5.2: A learner somehow displaced from an isolated maximum set, would never return.

Maxima sets with the geometry schematized in Figure 3, on the other hand, pose risks for more than just their original occupants.

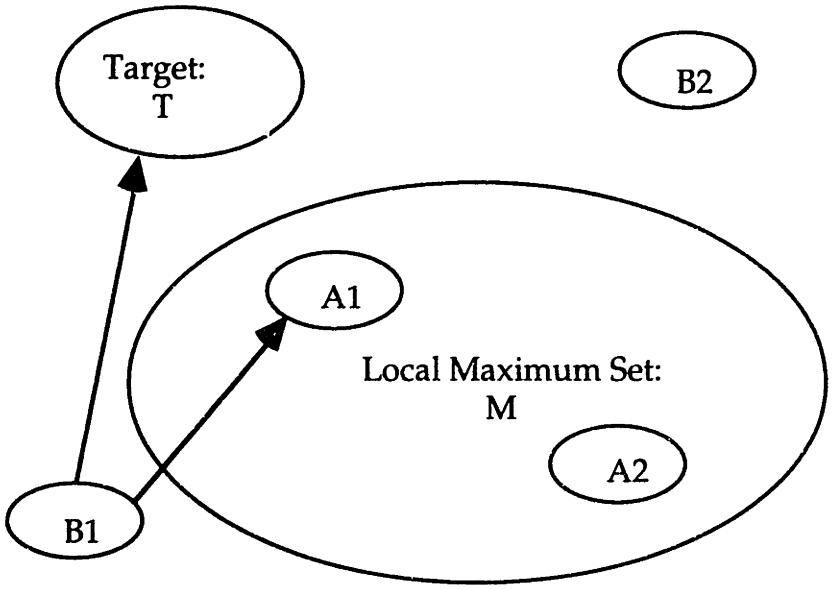


Figure 5.3: In the general case there can be transitions from grammars that are not in any local maximum set to grammars that are in a local maximum set

By definition, isolated local maxima cannot be reached from grammars that fall outside of the isolated local maximum set. However, in the more general case, exemplified in Figure 3, it is possible for grammars that are not in any maximum set to, nonetheless, have connections to grammars that are in a maximum set. A maximum set with incoming connections has a sort of *basin of attraction*. All the learners that begin life in the maximum set *M* end life in the maximum set *M*. Moreover, a certain proportion of learners that begin life outside the local maximum set will, at some point, in the course of acquisition, elect to pursue a path that leads them into *M*. The probability of this happening depends on the set of triggers for hypothesis change that are available and their distribution in the input.

What does subsequent grammatical development look like for a learner who either begins development with a hypothesis that falls in a local maximum set for their target grammar, or adopts such as hypothesis somewhere along the way? The answer to that question depends entirely on the number and pattern of connectivity of the attainable set of states associated with the starting state.

For example, it could be the case that, for a learner driven by the target grammar, there are no transitions at all out of the starting state; in this case, the prediction from the TLA is that the learner's final grammatical system is entirely determined by the initial random choice that landed it in the local maximum. The amount of "frustration" that such a learner experiences, measured in terms of the number of failed efforts to escape the local maximum, will depend on the extent of the overlap between the probabilistic distribution of linguistic patterns generated by speakers of the target language and the grammar that the learner starts with. If, for example, by some fluke, the American dialect of English syntax turned out to be a TLA local maximum for a more British variety, the learner would not necessarily encounter great difficulties. If, on the other, hand the target were a "free word order" language, such as the native Australian language Warlpiri, and the learner again began life with an unshakable assumption that it was learning English, this initial wrong choice would have more drastic consequences.[32] In this case, the learner would be ignoring a lot of information—in the form of

---

[32] This discussion may sound a little ridiculous, but, for the TLA and the MTLA, unless additional work is done to populate the parameter space with more "childlike" grammars that are somehow marked as unacceptable final states, these hypotheses seem to make the strong claim that the learner, at all points, hypothesizes a possible adult grammar. Of course, as I have pointed out earlier, there may be little evidence for "childlike" parameter setting. (Gibson and Wexler's unmarked proposal setting looks more like a parametric cue-based learner with respect to the class of hypotheses that the learner entertains.) In fact, in a large, finite parametric space, there is always a small, but non-zero, probability that the learner begins life

repeated failures to parse—that indicates that its current grammatical hypothesis should be abandoned.

It could also be the case that the learner was able to make several transitions among grammars before settling into one of several non-target absorbing states—states with no outward-bound transitions. After this absorption, the learner would be in the same situation as a learner who began life in an absorbing state. Finally, states in the local maximum set could be connected in such a way that the learner cycled indefinitely among a set of states—perpetually seeming to be making progress, but never leaving the confines of the local maximum set.

5.2 Shifting the Emphasis to Time to Convergence

If human learners use the TLA or MTLA to set parameters for natural language systems, then, in order to successfully acquire a target grammar with the same high probability consistent with what we empirically observe, the learner must obviously avoid local maxima with that same high probability. As Niyogi and Berwick (1993) point out, this requires that the learner also, with high probability, avoid states that are likely to lead to maxima. Whether or not it is easy to do so will depend on the number of local maxima sets for the different target grammars in a parametric system, the size of these local maxima sets and the attractive influence that these local maxima sets exert on the grammars that border on them.

---

with the grammatical system intact and need only acquire the lexicon of their target language. Of course confusions in grammatical analyses that arose from difficulties in lexical acquisition could potentially lead the learner to incorrectly abandon the correct grammar.

The emphasis in Gibson and Wexler (1994) and Bertolo (1995) on strategies for ensuring that the learner *never* enters into local maxima is, from my own point of view, somewhat too narrow. As I have presented the issues in this thesis, I believe that a top-level question that learnability research should address at this point, in the absence of detailed independent information about the workings of the human language acquisition device, is whether or not there are computationally feasible algorithms for human natural language acquisition that fall within the class of IGMS algorithms that do not draw on (or implicitly and redundantly encode) any sort of rich meta-knowledge about the relationship between parameter settings and linguistic data. The discovery of *any* such algorithms would be a major discovery and would contribute importantly to the ongoing debate about the need for explicit instruction manuals for parameter setting.

Indeed, it is easy to see that guaranteeing identification in the limit with high probability does not guarantee feasibility. A guarantee of identification in the limit is not a guarantee of success in a reasonable amount of time with a reasonable amount of input data.

As we have seen, it is absolutely crucial for a TLA or MTLA learner to avoid falling into local maxima. Once a learner falls into a maximum, the chances of recovery are exactly zero. Imagine, however, that the learner were equipped with a mechanism that probabilistically allowed the possibility of escaping from local maxima. The learnability situation could change quite dramatically.

Most obviously, identification-in-the-limit type learnability can be guaranteed for a finite parameter space by the addition of any mechanism that would guarantee a non-zero transition probability between any non-target state and each of its neighbors. In the most general case, where grammars can be in subset/superset relations to one another, some questions

arise as to what this mechanism might be. In the actual cases that concern Gibson and Wexler, and that arise in the metrical phonological space considered here, however, it is trivial to construct algorithms that will succeed in the limit. Most simply, a learner who randomly jumped to a neighboring grammar every time it encountered an unanalyzable input would eventually converge to the target grammar—where it would find that all inputs were analyzable.

Since the problem of guaranteeing in-the-limit convergence of the TLA (*modulo* subset/superset relationships between grammars), can be trivially solved by adding a probabilistic component to the learning algorithm, I take this as a further indication, that, in some sense, identification in the limit is only a small component of the real problem of interest. Of course, if there were some overwhelming argument for adopting a TLA type algorithm without any probabilistic component, then there would be strong argument for focusing research on maturation and other techniques for restricting the paths that the learner pursues as ways of avoiding local maxima. In the absence of such an argument, it seems valuable to explore other strategies for dealing with local maxima within the context of a system that still respects the TLA's basic hunch about the general picture of transition probabilities between states.

5.3 The Triggering Learning Algorithm and Random Walk

The TLA avoids changing its hypothesis when it receives no information at all that would help it decide whether it was, at least potentially, worth proceeding in a given direction. It is somewhat difficult to see the virtue of patiently adhering to the requirements of the Single Value Constraint and Greed in a space as small as the one that Gibson and Wexler

(1994) consider, so this space does not really provide a fair arena for testing the hypothesis. The TLA is essentially making a bet that the space will tend to produce useful pushes in the right direction if it waits for them. In a small parametric space, where all states in the space can be quickly explored by a learner willing to change hypotheses every time that an error is encountered, patience may not be a virtue. Indeed, Niyogi and Berwick (1993) point out that, given certain not-so-controversial assumptions about the probabilistic distribution of inputs generated by languages in Gibson and Wexler's 3-parameter syntactic space, the TLA is outperformed by several different variants on a random walk through the space. However, Niyogi and Berwick (1993) do not go on to make the crucial observation that random walk algorithms are doomed if natural language parameter spaces are large. As discussed in Chapter 2, the running time of a random walk algorithm will grow as $\Omega(2^n)$, where n indicates the number of parameters. If natural language parameter spaces are large, then random walk cannot be the right approach.

The important question is whether the TLA, or other IGMS algorithms, will overtake random walk algorithms as the parameter spaces that linguists construct in their efforts to model reality continue to grow, not whether the TLA can surpass random walk in an exceedingly small space.[33] It may, however, be possible to begin an initial assessment of the plausibility of this approach even within smaller parameter spaces by comparing the convergence times of the TLA to the convergence times for a suitably "penalized" random walk algorithm. As noted above, intuition suggests that the TLA pays a cost for the "deliberate" nature of its exploration

[33] Clearly, they need not do so. In fact, in certain spaces we have a guarantee that the learner cannot hope to do better than a random walk. The example presented in (6.1.1) provides one such case.

of the parameter space. It pays a certain premium in terms of the quantity of state changes that it makes in a fixed window of time in the hopes of profiting greatly from the increased quality of decisions that it makes. One can imagine a random walk "algorithm" that was somehow constrained to change states at roughly the same average rate as the TLA. The TLA had best outperform such an algorithm even in smaller spaces. If Greed did not provide the learner with any useful (or harmful!) directional guidance, then the race between the TLA and random walk should be decided simply by who gets to take the most steps per input. If the TLA could not outperform such a random walk process that was *yoked* to it in this way, hope that the TLA's deliberation will eventually pay off in larger spaces seems unwarranted, although not impossible. One way of attempting to simulate the effects of *yoking* the average speed of random walk to that of the TLA is to adjust the probabilities that the random walk algorithm sees in the following way: for each state, set the probability that the random walk algorithm encounters an error after examining one input equal to the probability that a TLA learner would change hypotheses after examining one input. When the yoked random walk algorithm does make a move, of course, it will distribute its moves equally among the current state's neighbors. The TLA, in accordance with Greed, will continue to distribute its moves to neighboring states to reflect the proportion of the error signals that these neighboring states are capable of analyzing. Comparison with a *yoked random walk* , thus, provides one analytic approach for assessing the potential of the TLA approach.

A stronger response to this implied criticism of the TLA, of course, would be to scale up to a larger space and demonstrate that the TLA begins to surpass even an unencumbered random walk. Ultimately, of course, the TLA will have to beat random walk quite convincingly. If parameter spaces are quite large it will not be enough, say, for the TLA to do twice as well as

random walk, since, as is now familiar, cutting a problem that is $\Omega(2^n)$ in half does not constitute much of an improvement.

Of course, remember, random walk had the virtue that it would eventually reach the target grammar, modulo superset grammars. The TLA and the MTLA cannot provide such guarantees for all such spaces because of their strict adherence to Greed. However, the slightest relaxation of Greed can lead to a guarantee that the TLA is also capable of learning a finite parametric system in the limit, modulo superset grammars. Consider the following modification of the TLA:

(5.3.1) a. Randomly select an initial hypothesis grammar from the set of all possible grammars.

b. Analyze an input from the target language:

i. If the current grammar provides an analysis of the current input, then keep the current grammar and start again at (b).

ii. If the current grammar cannot provide an analysis of the current input, then randomly select a single parameter to change, change it, and then:

1. If the new grammar provides an analysis of the current input, then keep the new parameter settings and start again at (b).

2. If not, then with small positive probability e retain the current settings. Otherwise, revert to

the previous grammar by unflipping the changed parameter. Start again at (b).

In this version of the TLA, the learner decides with probability *e* to change states in response to an error, even though this change does not allow the offending input to be successfully analyzed.

To demonstrate that this guarantees learning in the limit with probability 1, modulo superset relations, consider the following. If the hypothesis grammar is not capable of generating the target, an error signal will almost certainly occur eventually.[34] When this error signal occurs, each neighbor of the hypothesis grammar will be considered as a candidate to become the new hypothesis grammar. The TLA variant in (5.3.1) has been modified so that a new candidate grammar will be adopted with a probability that is greater than or equal to *e*. If, under the standard TLA, no triggering data existed that would drive a transition to the candidate grammar, this probability will be exactly *e*. If the probability of making a transition under the standard TLA was greater than zero, then the new transition probability will be strictly greater than *e*. With the addition of the stochastic component, then, the probability of any one-parameter transition from non-target grammar *a* to non-target grammar *b* is non-zero. The probability of making a transition out of the target grammar, of course, remains

---

[34] The identification in the limit framework actually does not require a text to repeat patterns from the target language. With this in mind, it is clear to see how to make life difficult for a TLA learner. For example, if two languages were identical except that each contained a single form that was unique to them, a text for one of the languages could conceivably present the unique form first and never thereafter. If a learner failed to exploit that occurrence, it is conceivable that they could incorrectly adopt the other highly similar language. I will assume that a learner who entertains an incorrect hypothesis that is not a subset of the target grammar will receive a steady flow of error signals.

zero in the absence of noise.[35] Now, for every non-target state, there are paths with a finite number of steps to the target grammar that the learner has some finite probability of following. In fact, if there are more than two grammars in the space, there will be many such paths from each non-target grammar in the space.

For our purposes, however, one such path from each grammar in the space is sufficient to establish identification in the limit with probability *1*; call this path the state's absorbing path. Now consider what it would mean for the learner not to converge. This would require the learner to never follow any of absorbing paths. Consider a learner beginning in a state S_1 whose absorbing path is length l_1, and whose absorption probability is p_1. After l_1 steps, the learner will be have reached the target with probability p_1. With probability $1 - p_1$, the learner will be in another non-target state S_2, whose absorbing path length is *l2*, and whose absorption probability is *p2*. And, so on. Clearly, since there are a finite number of states, there is a maximal absorbing path length l_{max} and a minimal, but non-zero, absorption probability p_{min}. The probability of remaining in the non-target states for l_{max} steps is a most $1 - p_{min}$. After $n * l_{max}$ steps, the probability of remaining in the non-target states will be at most $(1 - p_{min})^n$. As n goes to infinity this probability goes to zero. The probabilistically modified TLA algorithm will converge.

Of course, I have suggested that identification in the limit should, perhaps, not be the focus of research in this area. In fact it is clear that if the e parameter in the modified TLA

---

[35] With respect to noise, adoption of this modified TLA comes at a cost. Whereas, the standard TLA is only sensitive to noise that can be analyzed in accordance with Greed, the modified TLA is potentially sensitive to *any* noise. To avoid repeated wandering off from the target, it seems likely that the modified TLA would actually need to observe that failed standard TLA steps out of a state occurred at a certain frequency that made it unlikely they were due to noise alone.

algorithm is set low enough (say one in a billion), the behavior of the algorithm, for all intents and purposes, would be identical to that of the TLA learning algorithm, although local maximum states would have undergone a sort of metaphysical transformation into definitely-not-but-might-as-well-be local maximum states. If *e* is set to one, we have a random walk algorithm. If such a learner cannot analyze an error-generating form in a neighboring grammar, it invariably decides to adopt the neighboring grammar anyway; it may as well not have bothered checking for satisfaction of Greed.

Interesting possibilities, however, arise when *e* takes on a value somewhere between negligible and certain. If *e* is chosen large enough, then the chance of a learner, at least temporarily, escaping from standard TLA local maxima that do not involve proper supersets of the target language improves. If *e* is chosen small enough, it might not eliminate the standard TLA's overall bias to move in the direction of the target—if, of course, the standard TLA had such a bias in the first place. If the standard TLA is not biased to seek the target from the certain regions of parameter space, but is instead biased to seek a local maximum, then the effect of *e* may or may not be strong enough to make a practical difference. The learner might continually escape from the maximum set with probability at least *e*, only to find itself—with, for all intents and purposes, overwhelmingly high probability—revisiting the same maximum set again and again. States that, under the probabilistically modified TLA's acquisition regime, are "definitely-not-but-might-as-well-be" local maximum sets that have a strong pull and a large basin of attraction will make the modified TLA practically unworkable despite any guarantees of identification in the limit. Of course, the problem of local maximum states with a basin of attraction confronts the standard TLA as well. If the problematic states in a given parameter space have substantial basins of attraction, it may become necessary to turn

once again to devices such as restrictions on the initial starting states and maturational barriers that aim to keep learners out of particular regions of parameter space. Maturation could resurface as a valuable piece of machinery, if there are regions in the space that do not contain the target, and that are easy to fall into and hard to escape from. As Bertolo (1995) discusses in detail, maturational restrictions basically provide barriers that prevent entry into certain parts of parameter space during critical phases of development. They could clearly be deployed as such here as well, although it would be less easy to guarantee their safety and effectiveness due to the increased connectivity induced by the *e* parameter.

What do states that were local maxima under the standard TLA look like when neighboring grammars are fully connected in this way, through the addition of a stochastic escape hatch?

For purposes of developing some intuitions about the usefulness of the probabilistic modification, it is convenient to focus first on the now-familiar example of Gibson and Wexler's (1994) eight-grammar space. Gibson and Wexler's exhaustive analysis of this space identified six source/target maxima pairs. As shown in Table 5.3.1, a learner who ended up in a local maxima for a target would have ample opportunity to detect its plight. The table reports the probabilities that an error will be registered on any particular input, given a uniform distribution over forms in a target grammar. One virtue of an implementation of a system is that such results are calculable.

I will work throughout with uniform distributions in this chapter, although any results are obviously distribution-sensitive. In part, I am doing this because I believe this work will be more interesting for the general issues that it raises rather than the more particular results

reported. Clearly, study of natural language production data could help in the development of more realistic models of the distribution of input forms.

In the very worst case, 50% of the forms from a target grammar generate errors for a TLA learner trapped in a maximum in this space. There is, therefore, ample opportunity for a learner to detect that a maximum grammar is in error.

Table 5.3.1
Lost Opportunities to Escape from Local Maxima in Gibson and Wexler's (1994) 3-Parameter Syntactic Space Assuming Uniform Distribution of Inputs

Max Grammar	Target Grammar	Probability of Error
(SPEC-FIRST COMP-FINAL +V2)	(SPEC-FINAL COMP-FIRST -V2)	5/6
(SPEC-FIRST COMP-FIRST +V2)	(SPEC-FINAL COMP-FIRST -V2)	5/6
(SPEC-FINAL COMP-FINAL +V2)	(SPEC-FIRST COMP-FINAL -V2)	1/2
(SPEC-FINAL COMP-FIRST +V2)	(SPEC-FIRST COMP-FINAL -V2)	3/4
(SPEC-FINAL COMP-FINAL +V2)	(SPEC-FIRST COMP-FIRST -V2)	11/12
(SPEC-FINAL COMP-FIRST +V2)	(SPEC-FIRST COMP-FIRST -V2)	11/12

Of course, once the standard TLA has fallen into a local maximum, it is completely unable to learn anything more, despite the fact that its parser is quite consistently unable to analyze a large fraction of the sentences that it receives as input.

Table 5.3.2 shows the probabilities that, under the same uniform distributions from the target grammars, a learner will be drawn into a local maximum from a neighbor of the maximum. Consider a learner with a given *e* value who has been trapped in a local maximum. The probability that the learner will abandon the local maximum hypothesis on a given input is at least $e * p(error)$ for the given source/target pair. The probability is strictly greater than this if there are triggers that lead to other local maximum grammars. The probability that a given learner will return to the maximum in the next step after being *ejected* is simply the sum over the probabilities of returning to the maximum from a neighbor weighted by the probability of ejection to that neighbor. If there are no triggers at all out of the local maximum

to other local maxima, then the learner is ejected to a neighbor uniformly at random, so the return probabilities can be simply summed. This return probability has two components: one due to the triggering process, one due to the e parameter. Table 5.3.2 presents the component due to the triggering process for the Gibson and Wexler space.

Table 5.3.2
Probability of Attraction into Local Maxima for Grammars Neighboring on Local Maxima in Gibson and Wexler's (1994) 3-Parameter Syntactic Space Assuming Uniform Distribution of Inputs

Source Grammar	Target Grammar	Neighbors	P-max
(SPEC-FIRST COMP-FINAL +V2)	(SPEC-FINAL COMP-FIRST -V2)	(SPEC-FIRST COMP-FINAL -V2)	1/18
		(SPEC-FIRST COMP-FIRST +V2)	0
		(SPEC-FINAL COMP-FINAL +V2)	0
(SPEC-FIRST COMP-FIRST +V2)	(SPEC-FINAL COMP-FIRST -V2)	(SPEC-FIRST COMP-FIRST -V2)	1/18
		(SPEC-FIRST COMP-FINAL +V2)	0
		(SPEC-FINAL COMP-FIRST +V2)	0
(SPEC-FINAL COMP-FINAL +V2)	(SPEC-FIRST COMP-FINAL -V2)	(SPEC-FINAL COMP-FINAL -V2)	1/6
		(SPEC-FINAL COMP-FIRST +V2)	1/12
		(SPEC-FIRST COMP-FINAL +V2)	0
(SPEC-FINAL COMP-FIRST +V2)	(SPEC-FIRST COMP-FINAL -V2)	(SPEC-FINAL COMP-FIRST -V2)	1/12
		(SPEC-FINAL COMP-FINAL +V2)	0
		(SPEC-FIRST COMP-FIRST +V2)	0
(SPEC-FINAL COMP-FINAL +V2)	(SPEC-FIRST COMP-FIRST -V2)	(SPEC-FINAL COMP-FINAL -V2)	1/36
		(SPEC-FINAL COMP-FIRST +V2)	0
		(SPEC-FIRST COMP-FINAL +V2)	0
(SPEC-FINAL COMP-FIRST +V2)	(SPEC-FIRST COMP-FIRST -V2)	(SPEC-FINAL COMP-FIRST -V2)	1/36
		(SPEC-FINAL COMP-FINAL +V2)	0
		(SPEC-FIRST COMP-FIRST +V2)	0

The most strongly attractive tug from a maximum in the space pulls a grammar in from a neighbor in one out of six inputs, or 16.7% of the time. Since the probability of encountering an error in a maximum is at least 50%, setting *e* to .33 will ensure that the learner is ejected from the maximum at least 16.7% of the time. Therefore, inward movement due to the triggering process will be countered by the ejection process. Given an equal amount of traffic into a maximum and traffic out of a maximum, the maximum can no longer act as a basin of attraction. In order to do this, *e* will have to be strictly greater than .33, so as to counter the "pull" exerted by the random component.

Of course, there may not have been an appropriate bias towards the target even in the original space; the probabilistic jolt provided by the *e* parameter certainly can't provide one. Even if a learner equipped with a modified version of the TLA travels with a guarantee of identification in the limit, they may not be any more inclined to seek the target than a random walk algorithm. Moreover, upping the value of the *e* parameter, so as to escape from weakly attractive maxima, could disrupt an existing general bias toward the target.

5.4 Target Directional Biases

The discussion of the Gibson and Wexler space above was intended to informally prime intuitions about the possible impact of introducing a probabilistic, non-Greedy mechanism for effecting hypothesis change that would minimize concerns about local maxima in spaces where such maxima have limited attractiveness. However, it does not otherwise address concerns about the time course of acquisition in large parametric spaces. In order for acquisition to

succeed, it is not only necessary that the learner avoid becoming more or less stuck in regions of the space that do not contain the target grammar. The space must also provide a positive bias in the direction of the target that the learner can exploit. A learner must be more likely to take steps in the right direction rather than the wrong direction.

If such target directional biases exist uniformly throughout the space, it is easy to show that the TLA can be quite successful even in large parametric spaces. Consider, for example, the following artificial *n*-parameter space modeled on discussion in Clark (1992). In this parameter space, data takes the form of *n*-length strings. Each position in the string directly corresponds to a parameter in the system. Each position in the string can be filled by a symbol that indicates the value of that parameter in the language that generated it, or an *unknown* symbol that does not give any indication of the value. Each string, then, gives the learner a direct, but usually partial, look at the setting of parameters in the system. Let the data sets for languages in the space consist of all possible strings formed by obscuring from 0 to *n* parameters in this way.

In this space, a TLA learner will *only* be able to move in the direction of the target. The learner will encounter errors when exposed parameter values do not match the parameter values of its current hypothesis, and the learner will only move to a new state if it can accommodate the exposed parameter values. Given the Single Value Constraint, the triggers for movement in the system will be strings with a single parameter value that the learner cannot accommodate with its present hypothesis. The learner will *always* be able to move in the direction of the target because there are always appropriate forms that expose a single new parameter value from the target. The direction of motion is completely, and exceptionlessly biased towards the target. The details of how rapidly the learner moves toward the target will depend on the distribution of forms in the input the learner receives.

Of course, in this space the learner would be much better off with a different learning algorithm that simply recorded all of the exposed parameter values.

More generally, if the assumption of uniform biasing held true, it would be straightforward to calculate the effects that they would have on steps to convergence. The effects can be quite dramatic. Small uniform biases can lead to rapid progress through the space. This is suggested in Table 5.4.1, which displays the expected number of TLA steps to convergence given a particular uniform pattern of biasing towards the target. In particular, the biasing metric is the ratio of the probability that a "forward" TLA step in the direction in the direction of the target satisfies Greed to the probability that a "backward" TLA step away from the target satisfies Greed. As can be seen, even somewhat moderate biases can change the expected time to convergence by several orders of magnitude.

Table 5.4.1

Expected Steps to Convergence

	Number of Binary Parameters			
Probability Ratio	10	20	30	40
1	1.18e+03	1.11e+06	1.11e+09	1.13e+12
2	104	5.08e+03	2.76e+05	1.56e+07
3	41	600	9.59e+03	1.63+05
4	25	197	1.58e+03	1.37e+04
5	18	102	526	2.93e+03
6	15	66	255	1.04e+03
7	13	49	154	502
8	12	39	107	294
9	11	33	82	197
10	10	29	66	144

We cannot expect naturally parameter spaces to be so homogeneous, but, for the TLA hypothesis to work some set of appropriate biases must exist.

Gibson (1995) has presented some global results for the 3-parameter syntactic space that suggests that the biases there are in the right direction.

A similar chart for the small phonological space from Chapter 4 that existed before modification for TLA learning is presented in Table 5.4.2. (With our new algorithm, there is not an issue of identification in the limit, so the considerations that led us to the solution proposed there are not operative here.) Here, with the multi-valued parameters, the learner has the option of making moves that do change its distance from the target. Changing an incorrect parameter value to another incorrect parameter value results in no net progress towards the target.

Table 5.4.2
Target Directional Bias Averaged over Target Grammars

	Distance from target			
	1	2	3	4
P (no change)	.89	.92	.94	.96
P ("sideways")	.04	.03	.03	.03
P (forward)	.05	.03	.02	.02
P (backward)	.02	.01	.008	.00

Several features of this table are worth noting. The TLA's overwhelming bias in the space is to do nothing. Even when the learner is a single step away from the target, the chance of revising the grammar on any given input is only .89. However, at every distance from the target there is a net bias to take steps towards the target rather than away from it. This provides some preliminary, but extremely coarse, evidence that the TLA might find the type of bias that it

requires in such a space. Although the net bias looks promising, though, it may not hold for particular targets, and even if it holds for a particular target, it may not hold for particular regions in the parameter space for that target.

The TLA might be modified by allowing the algorithm to check *every* neighbor of the current hypothesis grammar when an error occurs; such a Super-Greedy TLA would never miss out on an opportunity for hypothesis change, and could choose simply choose randomly among its Greed-satisfying neighbors. Such an alteration, would not shape the overall bias of the system to move in one direction or other, but could affect the absolute speed at which it does so. This would be reflected in Table 5.4.2 by a reduction of the probability of no change, and a rescaling of the other entries in each column. As long as the absolute probability of making a move does not become too small, it is more appropriate to focus on the biases. Note, again, that an error-driven random walk algorithm, which maximally exploits the opportunity to make state transitions, will certainly be intractable in large spaces.

In light of the discussion about escaping maxima by adding a probabilistic escape hatch, a calculation of the amount of error data available to a learner is also of considerable interest. In the phonological space that has been considered here, each language generates a single stress contour for each distinct string of syllables. The worst case for the probabilistically modified TLA, then, would be if the syllable-to-stress mappings computed by source and target grammars in a local maximum pair were highly overlapping. As shown in (5.4.3), this is clearly not the case. In the worst case, 69% of the syllables receive the same syllable-to-stress mapping. The summary figures are based on the same type of calculation used to compute the results in Table 5.3.1, by computing the ratio of the size of the set difference between the target grammar and the maximum grammar to the size of the target grammar.

(5.4.3) Detectability of errors

Greatest overlap for a source/target local maximum pair: .69

Average overlap for a source/target local maximum pair: .05

Under a uniform distribution of inputs, the learner stuck in even the most deceptive of maxima in this space would still be unable to analyze almost one third of the inputs that it received. Even with a relatively small value for e, the probabilistic TLA would find ample opportunity to, at least temporarily, escape.

5.5 Markov Chain Analysis of the Triggering Learning Algorithm in Metrical Phonological Space

Given a particular space, one does not have to make do with such coarse indications of the time course of acquisition of a TLA-type strategy (where by TLA-type now I include the probabilistically non-Greedy case). Rather, given a transition matrix indicating the probability of transitions between the finite set of states in the parameter space, it is possible to compute a closed form solution for both the expected probabilities of convergence, and expected times to convergence for a given target from any source grammar.[36] The key properties of this transition matrix from the point of view of Markov chain theory are that: (1) entries are

[36] It is also straightforward to compute standard deviations of convergence times. These could also be quite useful in assessing the plausibility of a proposed learning algorithm, but I will not carry out such calculations here. See Niyogi and Berwick (1993) for further discussion.

greater than or equal to zero and less than or equal to one and (2) the sum of the entries in any row is one. This ensures that for any transition matrix A, $A[ij]$ can be interpreted as the probability that an input from the target language will cause a transition from state i to state j. Note that the probability of a transition between states, given a matrix for a target grammar, depends only on the state that the learner is currently in—that is to say, the algorithm is memoryless.[37] Problems of this sort fall squarely within the realm of a well-developed branch of mathematics dealing with Markov chains See, for example, Isaacson and Madsen (1976) for a textbook introduction. Pursuing the implications of a footnote in Gibson and Wexler (1994), Niyogi and Berwick (1993) provide the first paper to explicitly draw upon this literature and apply it to the problem of analyzing the behavior of algorithms for parametric natural language acquisition. In particular, they analyze the Gibson and Wexler space in considerable detail.

The entries in a transition matrix representing the movement of a TLA type algorithm depend, obviously, on both the set of triggers that generate movement between grammars and their distribution in the input data received from different target languages. Where the matrix calculations involved in the computation of this closed form solution proves to onerous (as it might in larger spaces), numerical estimates of these properties of the system might be developed via simulation of the learning algorithm. All results reported in this section some from such closed form computations.

---

[37] Of course, if memory can only take on a finite set of states, the learning process can still be represented as a finite Markov chain process whose states reflect both the current hypothesis and the contents of memory.

This section will report a number of such results for the metrical phonological system. As noted above, a certain amount of sophistication is required in order to use such results to obtain any kind of estimate about what might happen in a "real" parametric space. For example, our coarse study of the target directional biases in the metrical phonological space is hopeful for the TLA; there is an overall tendency to move towards the target, rather than away from it. However, the fact that doing nothing is far and away the dominant response of the TLA, despite the ready availability of error signals for a random walk algorithm to exploit, makes it seems quite likely that in this 4-parameter, 210 grammar space, random walk will continue to beat out variants on the TLA.[38] We may not yet have a large enough space for the "deliberative" virtues of the TLA to be displayed.

---

[38] Since the probabilistically Greedy TLA may void some concerns about local maxima, it may be interesting to consider computing similar target directional bias and Markov chain convergence results for a purely binary space.

The move to a binary space would certainly eliminate one category of wasted motion. It would no longer be possible for a learner to make a step that did not either move it closer to the target or further away. The overall effects of such a move, however, are potentially rather complicated. For example, in the phonological space, the move would have the effect of rescaling the distances of parameters. Consider, example, the edge marking options: 1) have an edge mark (yes/no), 2) place a (left/right) edge-mark , 3) to the (left/right), (4) of the (leftmost/rightmost) grid mark. In the present system, these options are treated as nine values of a single parameter—seven of which have an effect in monomorphemic words. Therefore, every grammar has eight neighbors that can be reached by a single flip of this parameter. In a strictly binary formulation, only a proper subset of these neighbors will be a single step away. (This is somewhat more complicated because the "no edge mark" grammar might be multiply instantiated in the new space. Alternatively, the interpretation could be that obscured values did not actually count in the computation of distance.) The transition probabilities between grammars that are no longer neighbors obviously goes to zero, while the transition probabilities between neighboring grammars simply needs to be reweighted to reflect the smaller size of neighborhoods. If any forward motion in the original system involved transitions to neighbors that were inaccessible in the new system, then the forward biasing of the space would be correspondingly reduced. On the other hand, if any reverse motion in the system involved transitions that were inaccessible in the new system, then the reverse biasing of the space would be correspondingly reduced.

In order to consider the effectiveness of Greed as a bias on the TLA's motion through the space, I will calculate and compare the probabilities of convergence and time to convergence for a number of different processes that move through the parameter space: (1) the TLA, (2) a yoked random walk "algorithm" whose net transition probabilities out of any given state have been scaled to match those of the TLA, (3) a straightforward single-step, error-driven random walk algorithm and (4) our probabilistic variant of the TLA, which chooses to violate Greed and make a move in some fraction, *e*, of the times that the normal TLA procedure fails to produce a move in response to an error. [39] The process in (2) deserves some further commentary. It is not intended to be understood as a realistically implementable algorithm. Rather, as suggested earlier in this section, it is to be understood as a simulation intended to answer the question of what work Greed is doing in the system. Comparing (1) and (2) will address the question of whether Greed is playing a useful role in guiding the learner around the maxima in the system. If so, then worries about basins of attraction would be lessened considerably. If the standard TLA does fairly well in the space as is, then a small boost in *e* might be enough to overcome any weakly attractive maxima in the system. Comparing (2) and (4) would allow for another measure of the target directional bias of the system. This comparison will be sensitive to the fact that the space might be too small for the virtue of Greed to be directly visible. Without the "deliberation" penalty, does the TLA surpass random walk? Of course, since there are known local maxima in the space variants (1) and (2) will not always converge to the target

[39] I will not be considering maturational variants of the TLA. One reason for not doing so is my optimism that maturational barriers might not be vital for a probabilistically enhanced TLA. Another reason, of course, is that doing so is more complicated since the transition matrix is not "stationary" throughout the learning process. Rather, there are several distinct shifts in the transition matrix that reflect the sudden availability of previously forbidden transitions.

with probability 1. Comparing (3) and (4) will allow me to determine whether the actual space that we have adopted is large enough that a TLA variant that can match random walk's guarantee of in-the-limit acquisition will begin to surpass it in this larger space.

Assuming a uniform probability distribution over the inputs and a uniform probability distribution over initial starting states, I calculated the probabilities that the TLA learner and its yoked random counterpart would converge to the target, given that it began in a state other than a target or a maximum.[40] This was done for each target grammar in the space. Our criterion for convergence, allows the learner to end up in any grammar that generates the target. In cases where there are weak equivalents of the target grammar, these weak equivalents were simply connected to the target grammar with a link of probability 1. Of course, the convergence results should be the same for any grammar that generates the target language, and, in fact, they are. From the learner's point of view, learning the trajectories that a learner follows cannot be sensitive to undetectable differences in the grammars of the speakers around them.

Table 5.5.1
Probability of Convergence to Target Grammars

Target	TLA P	Yoked P	Target	TLA P	Yoked P	Target	TLA P	Yoked P
(2 6 4 1)	100	100	(1 6 4 1)	100	100	(0 6 4 1)	94.2	70.4
(2 6 4 0)	100	100	(1 6 4 0)	100	100	(0 6 4 0)	100	46.6
(2 6 3 1)	100	100	(1 6 3 1)	100	100	(0 6 3 1)	99.8	16.2
(2 6 3 0)	100	100	(1 6 3 0)	89.5	50	(0 6 3 0)	91.5	11.9
(2 6 2 1)	100	100	(1 6 2 1)	99.9	50	(0 6 2 1)	100	46.6
(2 6 2 0)	100	100	(1 6 2 0)	100	100	(0 6 2 0)	94.2	70.4
(2 6 1 1)	100	100	(1 6 1 1)	100	100	(0 6 1 1)	100	46.6
(2 6 1 0)	100	100	(1 6 1 0)	100	100	(0 6 1 0)	94.2	70.4
(2 6 0 1)	100	100	(1 6 0 1)	99.9	50	(0 6 0 1)	99.6	34
(2 6 0 0)	100	100	(1 6 0 0)	100	100	(0 6 0 0)	99.6	34
(2 5 4 1)	100	100	(1 5 4 1)	100	100	(0 5 4 1)	100	100
(2 5 4 0)	100	100	(1 5 4 0)	100	100	(0 5 4 0)	95.4	24.8

[40] The stressless grammars are problematic here also. The transition matrices that arise when these two grammars are the target are so close to singular, that the numerical calculations involved are so subject to round-off error that MATLAB, the matrix mathematics package that I used, cannot reliably compute them.

(2 5 3 1)	100	100	(1 5 3 1)	100	100	(0 5 3 1)	46.9	19.1
(2 5 3 0)	99.3	50	(1 5 3 0)	100	100	(0 5 3 0)	100	46.6
(2 5 2 1)	100	100	(1 5 2 1)	100	100	(0 5 2 1)	94.2	70.4
(2 5 2 0)	99	50	(1 5 2 0)	100	100	(0 5 2 0)	71.7	10
(2 5 1 1)	100	100	(1 5 1 1)	100	100	(0 5 1 1)	100	100
(2 5 1 0)	100	100	(1 5 1 0)	100	100	(0 5 1 0)	98.9	50.2
(2 5 0 1)	100	100	(1 5 0 1)	100	100	(0 5 0 1)	99.6	34
(2 5 0 0)	98.2	36.5	(1 5 0 0)	100	100	(0 5 0 0)	99.6	34
(2 4 4 1)	100	100	(1 4 4 1)	100	100	(0 4 4 1)	98.9	50.2
(2 4 4 0)	100	100	(1 4 4 0)	100	100	(0 4 4 0)	100	100
(2 4 3 1)	100	100	(1 4 3 1)	100	100	(0 4 3 1)	66.6	11.3
(2 4 3 0)	100	100	(1 4 3 0)	100	100	(0 4 3 0)	100	19.5
(2 4 2 1)	89.2	50	(1 4 2 1)	100	100	(0 4 2 1)	96.9	13.1
(2 4 2 0)	100	100	(1 4 2 0)	100	100	(0 4 2 0)	100	29.7
(2 4 1 1)	100	100	(1 4 1 1)	100	100	(0 4 1 1)	95.4	24.8
(2 4 1 0)	100	100	(1 4 1 0)	100	100	(0 4 1 0)	100	100
(2 4 0 1)	100	100	(1 4 0 1)	100	100	(0 4 0 1)	69.6	11.5
(2 4 0 0)	100	100	(1 4 0 0)	100	100	(0 4 0 0)	92.1	15.7
(2 3 4 1)	100	100	(1 3 4 1)	100	100	(0 3 4 1)	100	100
(2 3 4 0)	100	100	(1 3 4 0)	100	100	(0 3 4 0)	95.4	24.8
(2 3 3 1)	100	100	(1 3 3 1)	100	100	(0 3 3 1)	100	29.7
(2 3 3 0)	100	100	(1 3 3 0)	89.2	50	(0 3 3 0)	96.9	13.1
(2 3 2 1)	100	100	(1 3 2 1)	100	100	(0 3 2 1)	100	19.5
(2 3 2 0)	100	100	(1 3 2 0)	100	100	(0 3 2 0)	66.6	11.3
(2 3 1 1)	100	100	(1 3 1 1)	100	100	(0 3 1 1)	100	100
(2 3 1 0)	100	100	(1 3 1 0)	100	100	(0 3 1 0)	98.9	50.2
(2 3 0 1)	100	100	(1 3 0 1)	100	100	(0 3 0 1)	92.1	15.7
(2 3 0 0)	100	100	(1 3 0 0)	100	100	(0 3 0 0)	69.6	11.5
(2 2 4 1)	100	100	(1 2 4 1)	100	100	(0 2 4 1)	98.9	50.2
(2 2 4 0)	100	100	(1 2 4 0)	100	100	(0 2 4 0)	100	100
(2 2 3 1)	100	100	(1 2 3 1)	99	50	(0 2 3 1)	71.7	10
(2 2 3 0)	100	100	(1 2 3 0)	100	100	(0 2 3 0)	94.2	70.4
(2 2 2 1)	100	100	(1 2 2 1)	99.3	50	(0 2 2 1)	100	46.6
(2 2 2 0)	100	100	(1 2 2 0)	100	100	(0 2 2 0)	46.9	19.1
(2 2 1 1)	100	100	(1 2 1 1)	100	100	(0 2 1 1)	95.4	24.8
(2 2 1 0)	100	100	(1 2 1 0)	100	100	(0 2 1 0)	100	100
(2 2 0 1)	100	100	(1 2 0 1)	98.2	36.5	(0 2 0 1)	99.6	34
(2 2 0 0)	100	100	(1 2 0 0)	100	100	(0 2 0 0)	99.6	34
(2 1 4 1)	100	100	(1 1 4 1)	100	100	(0 1 4 1)	94.2	70.4
(2 1 4 0)	100	100	(1 1 4 0)	100	100	(0 1 4 0)	100	46.6
(2 1 3 1)	100	100	(1 1 3 1)	100	100	(0 1 3 1)	94.2	70.4
(2 1 3 0)	99.9	50	(1 1 3 0)	100	100	(0 1 3 0)	100	46.6
(2 1 2 1)	89.5	50	(1 1 2 1)	100	100	(0 1 2 1)	91.5	11.9
(2 1 2 0)	100	100	(1 1 2 0)	100	100	(0 1 2 0)	99.8	16.2
(2 1 1 1)	100	100	(1 1 1 1)	100	100	(0 1 1 1)	100	46.6
(2 1 1 0)	100	100	(1 1 1 0)	100	100	(0 1 1 0)	94.2	70.4
(2 1 0 1)	100	100	(1 1 0 1)	100	100	(0 1 0 1)	99.6	34
(2 1 0 0)	99.9	50	(1 1 0 0)	100	100	(0 1 0 0)	99.6	34
(2 0 4 1)	100	100	(1 0 4 1)	100	100	(0 0 4 1)	94.2	70.4
(2 0 4 0)	100	100	(1 0 4 0)	100	100	(0 0 4 0)	100	46.6
(2 0 3 1)	100	100	(1 0 3 1)	100	100	(0 0 3 1)	100	29.7
(2 0 3 0)	100	100	(1 0 3 0)	90.8	50	(0 0 3 0)	93.1	13.7
(2 0 2 1)	90.8	50	(1 0 2 1)	100	100	(0 0 2 1)	93.1	13.7
(2 0 2 0)	100	100	(1 0 2 0)	100	100	(0 0 2 0)	100	29.7

(2 0 1 1)	100	100	(1 0 1 1)	100	100	(0 0 1 1)	100	46.6
(2 0 1 0)	100	100	(1 0 1 0)	100	100	(0 0 1 0)	94.2	70.4
(2 0 0 1)	100	100	(1 0 0 1)	99.9	50	(0 0 0 1)	*	*
(2 0 0 0)	99.9	50	(1 0 0 0)	100	100	(0 0 0 0)	*	*

The data in Table 5.5.1 have several interesting properties. First, a perusal of the table indicates that, all in all, the TLA's probability of convergence is quite high—although certainly not high enough that there is not room for improvement. This, in and of itself, suggests that the space may not be so ridden with attractive local maxima, so as to require maturational machinery once Greed is probabilistically relaxed in the way that I have suggested above. More importantly, though, note that in every case that the probabilities of convergence to the target language are different, the probability actually declines for the yoked random walk "algorithm". Rather than leading to basins of attraction that lure the learner away from the target language, Greediness actually helps the learner in this space to avoid being attracted into local maxima.

Table 5.5.2 indicates the expected number of inputs it takes for a learner to be absorbed into either the target, a weak equivalent or a local maximum, given that it did not start in the target or a local maximum.[41]

Table 5.5.2
Expected Inputs to Convergence

Target	TLA	Yoked	Target	TLA	Yoked	Target	TLA	Yoked
(2 6 4 1)	80	802	(1 6 4 1)	106	1400	(0 6 4 1)	79	762
(2 6 4 0)	131	2724	(1 6 4 0)	124	874	(0 6 4 0)	112	793
(2 6 3 1)	664	3571	(1 6 3 1)	471	5828	**(0 6 3 1)**	**5870**	**1315**

[41] Here, the implementation of the convergence of weak equivalent grammars to the target results in very slight discrepancies for weakly equivalent alternative targets. The learner is penalized a single step if they find the designated target via one of its weak equivalents because the scorekeeping here requires them to pursue the extra link to the designated target grammar.

(2 6 3 0)	207	4753	(1 6 3 0)	494	1701	(0 6 3 0)	360	779
(2 6 2 1)	124	874	(1 6 2 1)	189	3354	(0 6 2 1)	112	793
(2 6 2 0)	106	1400	(1 6 2 0)	480	2865	(0 6 2 0)	79	762
(2 6 1 1)	124	874	(1 6 1 1)	131	2724	(0 6 1 1)	112	793
(2 6 1 0)	106	1400	(1 6 1 0)	80	802	(0 6 1 0)	79	762
(2 6 0 1)	81	1434	(1 6 0 1)	218	5251	(0 6 0 1)	309	2731
(2 6 0 0)	265	7642	(1 6 0 0)	1618	4749	(0 6 0 0)	309	2731
(2 5 4 1)	74	859	(1 5 4 1)	147	2506	(0 5 4 1)	116	1162
(2 5 4 0)	176	4716	(1 5 4 0)	144	3028	(0 5 4 0)	192	1285
(2 5 3 1)	1401	3269	(1 5 3 1)	399	4157	(0 5 3 1)	422	1493
(2 5 3 0)	174	3224	(1 5 3 0)	124	874	(0 5 3 0)	112	792
(2 5 2 1)	79	802	(1 5 2 1)	303	3851	(0 5 2 1)	79	762
(2 5 2 0)	177	3286	(1 5 2 0)	349	3896	(0 5 2 0)	359	1242
(2 5 1 1)	74	859	(1 5 1 1)	141	2910	(0 5 1 1)	116	1162
(2 5 1 0)	163	3656	(1 5 1 0)	184	2540	(0 5 1 0)	219	1681
(2 5 0 1)	81	1434	(1 5 0 1)	195	5902	(0 5 0 1)	309	2731
(2 5 0 0)	142	2962	(1 5 0 0)	81	1434	(0 5 0 0)	309	2731
(2 4 4 1)	184	2540	(1 4 4 1)	163	3656	(0 4 4 1)	219	1681
(2 4 4 0)	141	2910	(1 4 4 0)	74	859	(0 4 4 0)	117	1162
(2 4 3 1)	320	4166	(1 4 3 1)	197	5269	(0 4 3 1)	312	2409
(2 4 3 0)	184	4785	(1 4 3 0)	854	5133	(0 4 3 0)	496	4859
(2 4 2 1)	235	2079	(1 4 2 1)	166	5952	(0 4 2 1)	1262	2664
(2 4 2 0)	179	4643	(1 4 2 0)	397	1839	(0 4 2 0)	387	1446
(2 4 1 1)	144	3028	(1 4 1 1)	176	4716	(0 4 1 1)	192	1285
(2 4 1 0)	147	2506	(1 4 1 0)	74	859	(0 4 1 0)	117	1162
(2 4 0 1)	400	4666	(1 4 0 1)	169	7898	(0 4 0 1)	471	3365
(2 4 0 0)	177	7072	(1 4 0 0)	2845	5294	(0 4 0 0)	799	4449
(2 3 4 1)	74	859	(1 3 4 1)	147	2506	(0 3 4 1)	117	1162
(2 3 4 0)	176	4716	(1 3 4 0)	144	3028	(0 3 4 0)	192	1285
(2 3 3 1)	397	1839	(1 3 3 1)	179	4643	(0 3 3 1)	387	1446
(2 3 3 0)	166	5952	(1 3 3 0)	235	2079	(0 3 3 0)	1262	2664
(2 3 2 1)	854	5133	(1 3 2 1)	184	4785	(0 3 2 1)	496	4859
(2 3 2 0)	197	5269	(1 3 2 0)	320	4166	(0 3 2 0)	312	2409
(2 3 1 1)	74	859	(1 3 1 1)	141	2910	(0 3 1 1)	117	1162
(2 3 1 0)	163	3656	(1 3 1 0)	184	2540	(0 3 1 0)	219	1681
(2 3 0 1)	2845	5294	(1 3 0 1)	177	7072	(0 3 0 1)	799	4449
(2 3 0 0)	169	7898	(1 3 0 0)	400	4666	(0 3 0 0)	471	3365
(2 2 4 1)	184	2540	(1 2 4 1)	163	3656	(0 2 4 1)	219	1681
(2 2 4 0)	141	2910	(1 2 4 0)	74	859	(0 2 4 0)	116	1162
(2 2 3 1)	349	3896	(1 2 3 1)	177	3286	(0 2 3 1)	359	1242
(2 2 3 0)	303	3851	(1 2 3 0)	79	802	(0 2 3 0)	79	762
(2 2 2 1)	124	874	(1 2 2 1)	174	3224	(0 2 2 1)	112	792
(2 2 2 0)	399	4157	(1 2 2 0)	1401	3269	(0 2 2 0)	422	1493
(2 2 1 1)	144	3028	(1 2 1 1)	176	4716	(0 2 1 1)	192	1285
(2 2 1 0)	147	2506	(1 2 1 0)	74	859	(0 2 1 0)	116	1162
(2 2 0 1)	81	1434	(1 2 0 1)	142	2962	(0 2 0 1)	309	2731
(2 2 0 0)	195	5902	(1 2 0 0)	81	1434	(0 2 0 0)	309	2731
(2 1 4 1)	80	802	(1 1 4 1)	106	1400	(0 1 4 1)	79	762
(2 1 4 0)	131	2724	(1 1 4 0)	124	874	(0 1 4 0)	112	793
(2 1 3 1)	480	2865	(1 1 3 1)	106	1400	(0 1 3 1)	79	762
(2 1 3 0)	189	3354	(1 1 3 0)	124	874	(0 1 3 0)	112	793
(2 1 2 1)	494	1701	(1 1 2 1)	207	4753	(0 1 2 1)	360	779
(2 1 2 0)	471	5828	(1 1 2 0)	664	3571	**(0 1 2 0)**	**5870**	**1315**
(2 1 1 1)	124	874	(1 1 1 1)	131	2724	(0 1 1 1)	112	793

(2 1 1 0)	106	1400	(1 1 1 0)	80	802	(0 1 1 0)	79	762
(2 1 0 1)	1618	4749	(1 1 0 1)	265	7642	(0 1 0 1)	309	2731
(2 1 0 0)	218	5251	(1 1 0 0)	81	1434	(0 1 0 0)	309	2731
(2 0 4 1)	80	802	(1 0 4 1)	106	1400	(0 0 4 1)	79	762
(2 0 4 0)	131	2724	(1 0 4 0)	124	874	(0 0 4 0)	112	793
(2 0 3 1)	397	1839	(1 0 3 1)	572	6086	(0 0 3 1)	387	1446
(2 0 3 0)	171	7345	(1 0 3 0)	581	1832	(0 0 3 0)	439	991
(2 0 2 1)	581	1832	(1 0 2 1)	171	7345	(0 0 2 1)	439	991
(2 0 2 0)	572	6086	(1 0 2 0)	397	1839	(0 0 2 0)	387	1446
(2 0 1 1)	124	874	(1 0 1 1)	131	2724	(0 0 1 1)	112	793
(2 0 1 0)	106	1400	(1 0 1 0)	80	802	(0 0 1 0)	79	762
(2 0 0 1)	250	2550	(1 0 0 1)	259	5625	(0 0 0 1)	•	•
(2 0 0 0)	259	5625	(1 0 0 0)	250	2550	(0 0 0 0)	•	•

The first thing to note, perhaps, is that even the worst cases of the random walk algorithm do not take anything like an impossible number of inputs. Clearly, we are not operating in a space that is large enough to definitively rule out even this particularly slow random walk solely on grounds of speed. However, the TLA is, in some sense, deriving a benefit from its policy of Greed. The TLA is much quicker to reach a final state than the yoked random walk algorithm, and we know already that the TLA is not trading off accuracy for speed. Indeed, for all but the two highlighted grammars (and the ever-problematic stressless grammars where the convergence time could not be reliably calculated because of the near-singularity of the transition matrices), the TLA is anywhere from 2 to 47 times faster than its yoked counterpart.

We have, then, I believe, at least a preliminary example of what it might look like for the TLA's "leap of faith" about biases towards the target to land on solid ground.

Of course, the price that the TLA pays for the guidance that Greed provides is a reduction in the number of states that it explores for a given amount of input data. The simple error-driven learner, whose properties were described in (2), makes a transition every time an error signal occurs. These same signals also provide the impetus for the TLA. However, the TLA

only makes a move if the neighbor that it happens to select as its new candidate hypothesis is capable of generating the previously unanalyzable form.

To get an initial indication of how Greed fared against an unencumbered random walk in the present space, I computed the expected times to convergence for the random walk algorithm, a version of the probabilistically modified TLA algorithm with an *e* value of .1, and a version of the probabilistically modified TLA algorithm with an *e* value of .01. (Obviously, more could be done to explore the consequences of modifying the *e* parameter. Intuition suggested, however, that at the .01 level the learner's behavior would look very much like the standard TLA's, although convergence is, of course, guaranteed. At the .1 level, given the low overlap between grammars in the space, the effect of the non-Greedy steps might become more visible. Tables 5.5.3.a–c report the expected convergence times from the best and worst start states for a target grammar and the overall expected convergence times for each of these TLA variants. Table 5.5.4 provides summary information.

Table 5.5.3.a
Inputs to Convergence

Target	Rand Best Start	Rand Avg	Rand Worst Start	.1 Best Start	.1 Avg	.1 Worst Start	.01 Best Start	.01 Avg	.01 Worst Start
(2 6 4 1)	66	91	100	26	79	106	17	79	128
(2 6 4 0)	85	122	131	22	100	152	12	117	336
(2 6 3 1)	285	328	345	124	283	412	60	525	1072
(2 6 3 0)	257	295	305	97	193	243	44	200	378
(2 6 2 1)	51	83	96	21	99	159	13	120	234
(2 6 2 0)	66	96	106	23	90	136	15	101	257
(2 6 1 1)	51	83	96	21	99	159	13	120	234
(2 6 1 0)	66	96	106	23	90	136	15	101	257
(2 6 0 1)	84	106	113	31	85	111	14	81	131
(2 6 0 0)	239	276	285	85	185	240	33	210	466
(2 5 4 1)	68	85	92	28	74	96	18	74	119
(2 5 4 0)	122	158	168	34	115	169	18	138	367
(2 5 3 1)	282	326	344	148	385	538	75	976	1705
(2 5 3 0)	250	288	298	88	180	227	34	168	370
(2 5 2 1)	66	91	100	26	79	106	18	79	128
(2 5 2 0)	249	288	298	89	188	233	38	178	378
(2 5 1 1)	68	85	92	28	74	96	18	74	119
(2 5 1 0)	126	164	174	46	141	189	28	151	320

(2 5 0 1)	84	106	113	31	85	111	13	81	131
(2 5 0 0)	234	270	280	75	167	217	26	150	378
(2 4 4 1)	127	166	177	50	155	209	33	177	305
(2 4 4 0)	124	161	172	47	130	179	32	137	268
(2 4 3 1)	259	300	312	115	246	315	63	296	481
(2 4 3 0)	248	285	295	98	193	238	49	183	290
(2 4 2 1)	256	296	308	94	223	287	54	275	439
(2 4 2 0)	259	297	307	94	189	232	51	178	278
(2 4 1 1)	127	165	176	36	130	180	18	139	267
(2 4 1 0)	128	166	177	42	134	181	24	143	248
(2 4 0 1)	254	294	306	108	256	330	59	350	585
(2 4 0 0)	238	274	284	88	186	235	35	173	325
(2 3 4 1)	68	85	92	28	74	96	18	74	119
(2 3 4 0)	122	158	168	34	115	169	18	138	367
(2 3 3 1)	141	183	198	52	198	296	29	336	654
(2 3 3 0)	242	279	289	81	178	226	38	163	311
(2 3 2 1)	277	318	331	133	294	377	99	473	867
(2 3 2 0)	245	283	294	90	204	249	35	193	322
(2 3 1 1)	68	85	92	28	74	96	18	74	119
(2 3 1 0)	126	164	174	46	141	189	28	151	320
(2 3 0 1)	458	516	585	393	1100	1652	101	2390	4434
(2 3 0 0)	233	269	279	76	180	229	27	162	313
(2 2 4 1)	127	166	177	50	155	209	33	177	304
(2 2 4 0)	124	161	172	47	130	179	32	137	268
(2 2 3 1)	274	316	328	145	265	337	96	324	527
(2 2 3 0)	255	295	306	125	252	302	42	282	381
(2 2 2 1)	51	83	96	21	99	158	13	120	233
(2 2 2 0)	260	300	311	140	265	311	77	343	476
(2 2 1 1)	127	165	176	36	129	180	18	139	266
(2 2 1 0)	128	166	177	42	134	181	24	143	248
(2 2 0 1)	84	106	113	31	85	111	13	81	131
(2 2 0 0)	237	274	284	96	208	255	28	190	299
(2 1 4 1)	66	91	100	26	79	106	17	79	128
(2 1 4 0)	85	122	131	22	100	152	12	117	337
(2 1 3 1)	298	343	362	145	328	452	77	445	741
(2 1 3 0)	257	295	305	92	184	232	31	182	391
(2 1 2 1)	400	447	526	250	491	619	152	588	864
(2 1 2 0)	378	421	504	243	425	597	125	456	654
(2 1 1 1)	51	83	96	21	99	159	13	120	234
(2 1 1 0)	66	96	106	23	90	136	15	101	257
(2 1 0 1)	451	508	578	364	930	1339	138	1485	2584
(2 1 0 0)	253	288	298	91	200	252	29	202	444
(2 0 4 1)	66	91	100	26	79	107	17	79	128
(2 0 4 0)	85	122	131	22	100	152	12	117	337
(2 0 3 1)	141	183	198	52	198	296	29	336	654
(2 0 3 0)	255	292	302	83	171	219	31	162	348
(2 0 2 1)	399	446	524	288	556	777	171	652	931
(2 0 2 0)	376	420	502	288	511	764	132	559	842
(2 0 1 1)	51	83	96	21	99	159	13	120	234
(2 0 1 0)	66	96	106	23	90	136	15	101	257
(2 0 0 1)	157	186	195	86	221	273	42	246	350
(2 0 0 0)	250	285	295	103	227	278	30	241	475

Table 5.5.3.b
Inputs to Convergence

Target	Rand Best Start	Rand Avg	Rand Worst Start	.1 Best Start	.1 Avg	.1 Worst Start	.01 Best Start	.01 Avg	.01 Worst Start
(1 6 4 1)	66	96	106	23	90	136	15	101	257
(1 6 4 0)	51	83	96	21	99	159	13	120	234
(1 6 3 1)	378	421	504	243	425	597	125	456	654
(1 6 3 0)	400	447	526	250	491	619	152	588	864
(1 6 2 1)	257	295	305	92	184	232	31	182	391
(1 6 2 0)	298	343	362	145	328	452	77	445	741
(1 6 1 1)	85	122	131	22	100	152	12	117	337
(1 6 1 0)	66	91	100	26	79	106	17	79	128
(1 6 0 1)	253	288	298	91	200	252	29	202	444
(1 6 0 0)	451	508	578	364	930	1339	138	1485	2584
(1 5 4 1)	128	166	177	42	134	181	24	143	248
(1 5 4 0)	127	165	176	36	129	180	18	139	266
(1 5 3 1)	260	300	311	140	265	311	77	343	476
(1 5 3 0)	51	83	96	21	99	158	13	120	233
(1 5 2 1)	255	295	306	125	252	302	42	282	381
(1 5 2 0)	274	316	328	145	265	337	96	324	527
(1 5 1 1)	124	161	172	47	130	179	32	137	268
(1 5 1 0)	127	166	177	50	155	209	33	177	304
(1 5 0 1)	237	274	284	96	208	255	28	190	299
(1 5 0 0)	84	106	113	31	85	111	13	81	131
(1 4 4 1)	126	164	174	46	141	189	28	151	320
(1 4 4 0)	68	85	92	28	74	96	18	74	119
(1 4 3 1)	245	283	294	90	204	249	35	193	322
(1 4 3 0)	277	318	331	133	294	377	99	473	867
(1 4 2 1)	242	279	289	81	178	226	38	163	311
(1 4 2 0)	141	183	198	52	198	296	29	336	654
(1 4 1 1)	122	158	168	34	115	169	18	138	367
(1 4 1 0)	68	85	92	28	74	96	18	74	119
(1 4 0 1)	233	269	279	76	180	229	27	162	313
(1 4 0 0)	458	516	585	393	1100	1652	101	2390	4434
(1 3 4 1)	128	166	177	42	134	181	24	143	248
(1 3 4 0)	127	165	176	36	130	180	18	139	267
(1 3 3 1)	259	297	307	94	189	232	51	178	278
(1 3 3 0)	256	296	308	94	223	287	54	275	439
(1 3 2 1)	248	285	295	98	193	238	49	183	290
(1 3 2 0)	259	300	312	115	246	315	63	296	481
(1 3 1 1)	124	161	172	47	130	179	32	137	268
(1 3 1 0)	127	166	177	50	155	209	33	177	305
(1 3 0 1)	238	274	284	88	186	235	35	173	325
(1 3 0 0)	254	294	306	108	256	330	59	350	585
(1 2 4 1)	126	164	174	46	141	189	28	151	320
(1 2 4 0)	68	85	92	28	74	96	18	74	119
(1 2 3 1)	249	288	298	89	188	233	38	178	378
(1 2 3 0)	66	91	100	26	79	106	18	79	128
(1 2 2 1)	250	288	298	88	180	227	34	168	370

(1 2 2 0)	282	326	344	148	385	538	75	976	1705
(1 2 1 1)	122	158	168	34	115	169	18	138	367
(1 2 1 0)	68	85	92	28	74	96	18	74	119
(1 2 0 1)	234	270	280	75	167	217	26	150	378
(1 2 0 0)	84	106	113	31	85	111	13	81	131
(1 1 4 1)	66	96	106	23	90	136	15	101	257
(1 1 4 0)	51	83	96	21	99	159	13	120	234
(1 1 3 1)	66	96	106	23	90	136	15	101	257
(1 1 3 0)	51	83	96	21	99	159	13	120	234
(1 1 2 1)	257	295	305	97	193	243	44	200	378
(1 1 2 0)	285	328	345	124	283	412	60	525	1072
(1 1 1 1)	85	122	131	22	100	152	12	117	336
(1 1 1 0)	66	91	100	26	79	106	17	79	128
(1 1 0 1)	239	276	285	85	185	240	33	210	466
(1 1 0 0)	84	106	113	31	85	111	14	81	131
(1 0 4 1)	66	96	106	23	90	136	15	101	257
(1 0 4 0)	51	83	96	21	99	159	13	120	234
(1 0 3 1)	376	420	502	288	511	764	132	559	842
(1 0 3 0)	399	446	524	288	556	777	171	652	931
(1 0 2 1)	255	292	302	83	171	219	31	162	348
(1 0 2 0)	141	183	198	52	198	296	29	336	654
(1 0 1 1)	85	122	131	22	100	152	12	117	337
(1 0 1 0)	66	91	100	26	79	107	17	79	128
(1 0 0 1)	250	285	295	103	227	278	30	241	475
(1 0 0 0)	157	186	195	86	221	273	42	246	350

Table 5.5.3.c
Inputs to Convergence

Target	Rand Best Start	Rand Avg	Rand Worst Start	.1 Best Start	.1 Avg	.1 Worst Start	.01 Best Start	.01 Avg	.01 Worst Start
(0 6 4 1)	46	69	77	18	80	118	9	99	307
(0 6 4 0)	44	73	83	14	89	140	7	111	313
(0 6 3 1)	355	397	421	193	402	456	78	635	1230
(0 6 3 0)	304	342	365	166	344	399	49	458	811
(0 6 2 1)	44	73	83	14	89	140	7	111	313
(0 6 2 0)	46	68	76	18	80	118	9	99	307
(0 6 1 1)	44	73	83	14	89	140	7	111	313
(0 6 1 0)	46	69	77	18	80	118	9	99	307
(0 6 0 1)	68	90	129	46	153	201	17	267	560
(0 6 0 0)	68	90	129	46	153	201	17	267	560
(0 5 4 1)	56	81	88	23	102	132	10	109	219
(0 5 4 0)	113	148	157	41	161	215	15	195	419
(0 5 3 1)	227	264	274	144	348	412	81	968	1521
(0 5 3 0)	44	73	83	14	89	140	7	111	313
(0 5 2 1)	46	68	76	18	80	118	9	99	307
(0 5 2 0)	219	254	263	121	290	347	53	504	767
(0 5 1 1)	56	81	88	23	102	131	10	109	219
(0 5 1 0)	119	157	167	75	202	246	49	216	353

(0 5 0 1)	69	90	129	46	153	201	17	267	560
(0 5 0 0)	68	90	129	46	153	201	17	267	560
(0 4 4 1)	119	157	167	75	202	246	48	216	354
(0 4 4 0)	56	81	88	23	102	132	10	109	219
(0 4 3 1)	214	248	257	136	333	395	49	540	881
(0 4 3 0)	212	245	254	130	327	391	38	422	756
(0 4 2 1)	213	247	256	117	307	374	42	663	1081
(0 4 2 0)	148	185	209	116	241	293	42	328	571
(0 4 1 1)	113	148	157	41	161	215	15	195	419
(0 4 1 0)	56	81	88	23	102	132	10	109	219
(0 4 0 1)	208	241	250	138	364	434	41	666	1045
(0 4 0 0)	438	492	566	302	719	1020	56	877	1429
(0 3 4 1)	56	81	88	23	102	132	10	109	219
(0 3 4 0)	113	148	157	41	161	215	15	195	419
(0 3 3 1)	148	185	209	116	241	293	42	328	571
(0 3 3 0)	213	247	256	117	307	374	42	663	1081
(0 3 2 1)	212	245	254	130	327	391	38	422	756
(0 3 2 0)	214	248	257	136	333	395	49	540	881
(0 3 1 1)	56	81	88	23	102	132	10	109	219
(0 3 1 0)	119	157	167	75	202	246	48	216	354
(0 3 0 1)	438	492	566	302	719	1020	56	877	1429
(0 3 0 0)	208	241	250	138	364	434	41	666	1045
(0 2 4 1)	119	157	167	75	202	246	49	216	353
(0 2 4 0)	56	81	88	23	102	131	10	109	219
(0 2 3 1)	219	254	263	121	290	347	53	504	767
(0 2 3 0)	46	68	76	18	80	118	9	99	307
(0 2 2 1)	44	73	83	14	89	140	7	111	313
(0 2 2 0)	227	264	274	144	348	412	81	968	1521
(0 2 1 1)	113	148	157	41	161	215	15	195	419
(0 2 1 0)	56	81	88	23	102	132	10	109	219
(0 2 0 1)	68	90	129	46	153	201	17	267	560
(0 2 0 0)	69	90	129	46	153	201	17	267	560
(0 1 4 1)	46	69	77	18	80	118	9	99	307
(0 1 4 0)	44	73	83	14	89	140	7	111	313
(0 1 3 1)	46	68	76	18	80	118	9	99	307
(0 1 3 0)	44	73	83	14	89	140	7	111	313
(0 1 2 1)	304	342	365	166	344	399	49	458	811
(0 1 2 0)	355	397	421	193	402	456	78	635	1230
(0 1 1 1)	44	73	83	14	89	140	7	111	313
(0 1 1 0)	46	69	77	18	80	118	9	99	307
(0 1 0 1)	68	90	129	46	153	201	17	267	560
(0 1 0 0)	68	90	129	46	153	201	17	267	560
(0 0 4 1)	46	69	77	18	80	118	9	99	307
(0 0 4 0)	44	73	83	14	89	140	7	111	313
(0 0 3 1)	148	185	209	116	241	293	41	328	571
(0 0 3 0)	301	338	362	178	366	425	48	468	809
(0 0 2 1)	301	338	362	178	366	425	48	468	809
(0 0 2 0)	148	185	209	116	241	293	41	328	571
(0 0 1 1)	44	73	83	14	89	140	7	111	313
(0 0 1 0)	46	69	77	18	80	118	9	99	307
(0 0 0 1)	111	127	133	92	233	276	51	745	1029
(0 0 0 0)	111	127	133	92	233	276	51	745	1029

The summary data in Table 5.5.4 suggests that there are cases where the target directional bias is so strong that upping the value of the *e* parameter impedes learning. The time from the best start state decline as *e* increases. These, however, are likely to be cases where the learner starts in a state adjacent to the target grammar. As the parameter space grows, the chance of beginning this close to the target will become less and less likely. Increasing the *e* parameter appears to have a beneficial effect on the worst cases in the parameter spaces; in cases where Greed fails to pull the learner towards the target, a random jolt appears to help. However, neither of these two values of the *e* parameter lead to a performance that is better than random walk overall, although the differences between the overall expected convergence times are not particularly striking. In fact, the probabilistic TLA variant with an *e* value of .01 does better, on average, than the random walk algorithm many of target grammars in the system. I am averaging here over languages. In general, one must be concerned about the worst cases for learnable languages in the system. Here, random walk appears to still be preferable, although the absolute number of inputs required for convergence is small in all cases.

Table 5.5.4
Inputs to Convergence

	Rand Best Start	Rand Avg	Rand Worst Start	.1 Best Start	.1 Avg	.1 Worst Start	.01 Best Start	.01 Avg	.01 Worst Start
Overall Average over Languages	219	256	275	111	259	342	49	365	643

Overall, then, for this system, the picture that emerges with respect to the TLA's prospects is mixed. Comparison of the TLA (or, indeed, the probabilistic variant of the TLA with an e value of .01) with the yoked, random walk algorithm indicates that Greed is providing useful guidance. Not enough useful guidance in this still small space, however, that the learner is better off—at least with the standard TLA or either of the probabilistic variants considered here. Consider, however, using the super-Greedy variant of the TLA, discussed above, that exhaustively considered Greedy steps to its neighbors, and only made a random step when no neighbor could satisfy Greed. Such an algorithm is virtually certain to push the TLA over the top of random walk, given the slight differences that exist between the expected running times presented in Table 5.5.4.

5.6 The Impact of Extending the Parametric System on the Probabilistic Triggering Learning Algorithm Approach

Some hopeful examples have emerged for the general TLA approach in this chapter. With the probabilistic modification we have a proof of convergence in the limit. We also have evidence that useful biases are provided by Greed in the phonological space discussed in 5.5.

What can be said next? It seems that we are left with the key question of whether some version of Greed, perhaps not an immutable and absolute one, will be good as linguists and learnability modelers extend and modify parametric systems.

With respect to slight modifications of the data sets in a fixed parameter space topology, at least, the hope is that convergence time results with the probabilistically modified TLA will prove more robust than in-the-limit convergence results with the TLA or

MTLA. Since most steps that this algorithm takes will be governed by either high frequency forms or the random component, once the distribution of input forms is approximately correct one would expect learning results to be reasonably robust. This is unlike the case of identification-in-the-limit results with the TLA where the slightest alteration in the set of forms in a language has the potential to dramatically alter the learning landscape.

With respect to extensions of parametric systems to encompass more and more parameters, I have found little else to say other than the obvious. In the general vicinity of the target, Greed must lead the learner to take steps in the direction of the target. There need not be strong biases in the furthest reaches of parameter space, since a random walk algorithm, on average, expects to be half way to the target (even more than half way when there are weak equivalents.) If strong pressures exist, in general, and if the cost of these pressures is not long periods of stasis due to an inability to satisfy Greed, then the approach will work well in large parametric spaces. Although I have identified some artificial scenarios with appropriate biases, it has proven difficult to translate this basic intuition into a set of reasonably general, locally identifiable properties that would allow a researcher, without simply implementing and testing the system, to identify extensions that will or will not satisfy this criterion.

Chapter 6
The Parser as a Tool for Parameter Setting

If the parametric, cue-based approach succeeds in the limit for a space, then its deductive winnowing of the parameter space results directly in an algorithm that only goes through a number of hypothesis changes that is linear in the number of parameters. Of course, for such a scheme to be tractable, exponential growth must not be hidden in the individual steps that apply the cues, or in the storage of the cues in UG. The challenge for such an approach is establishing that UG can provide a compact set of cues that ensure learning by paring down the space a parameter or parameter value at a time. As more and more parameters of linguistic systems interact it is unclear whether it will be possible to implement an easily applicable, explicitly coded set of cues in UG that perform this function. Indeed, it can only become more difficult. Dresher and Kaye (1990) and Dresher (1994) specifically intend for there to be a single cue for each parameter, although, as we have seen from consideration of Dresher (1994), this requirement may be difficult to satisfy in practice. Moreover, urges in the direction of parsimony are hardly satisfied by the introduction of a complicated, domain-specific device whose only function is to enable language acquisition.

The TLA and its ilk are quite parsimonious. Since, however, they do not pursue a policy which deductively divides the space at each step, but rather rely on the Single Value Constraint and Greed for their guidance, they are keenly reliant on the hypothesis that *distance* in parameter space will translate into particular patterns of overlap among languages. It seems that there is reason to believe that reliance on these type of properties is a general

property of search algorithms that have only the TLA's most limited access to UG. As shown in Chapter 5, the TLA is easily modified to ensure identification in the limit with probability 1 for systems without subset/superset relations among languages, while maintaining many of the potential benefits provided by Greed. On the other hand, whether or not the TLA would find appropriate biases in the direction of the target, so as to vastly outperform an error-driven random walk in a large parametric system, remains an open question, although promising results were obtained in the simplified phonological system. A randomly constructed transition matrix that approximately respected the TLA's Single Value Constraint would not be expected to provide the critical biasing guidance that the TLA needs. To generally ensure the feasibility of the TLA in the parametric spaces that underlie human linguistic competence it seems necessary to either demonstrate its application to the "real thing"—an event that is unlikely to occur in the immediate future—or to provide principled arguments to the effect that we can expect the overlap relations in the space to be appropriate. It would be interesting if such arguments could be provided in a way that made specific predictions about the nature of parameters. It would be compelling finding if there turned out to be appropriate biases in the actual spaces that proponents of the Principles and Parameters approach believe actually exist. Such a finding would provide convincing evidence in favor of a TLA-type approach.

A recent proposal made by Fodor (1995) promises a solution to the problem of language acquisition (at least in certain domains) that combines the virtues of deductively warranted beliefs with the ability to do without an explicit, domain-specific set of instructions for acquisition. Acknowledging the fact that the complex pattern of interaction between parameters makes it unlikely that a general-purpose inferential system would be able to successfully navigate a large parameter space, and questioning the existence of appropriate

biases for the TLA, Fodor proposes instead to deploy a special purpose inferential device that, at least in some form, has substantial independent motivation—the human parser. (The references providing general motivation for such a device are really are too numerous to cite; an entire branch of psycholinguistics is devoted to the study of the properties of this device.) However, as will be seen, her proposal involves deploying the parser in a mode not normally posited for adult processing. In particular, Fodor requires an unlimited parallel parser. (See Gibson (1991) and the references there for an argument for a limited parallelism). Care must be taken that operation in this special mode does not lead to tractability problems in large spaces.

I refer to the parser as a special-purpose inference device because, given a parametric specification of a language, it is capable of deducing what structures, couched in the internal mental vocabulary of the grammar, should be assigned to the strings of a language. Fodor essentially proposes that the parser provides a way of bridging the epistemological gap assumed by Dresher (1994) and Dresher and Kaye (1990).[42] On her proposal, not an unreasonable one, a speaker of a +V2 language, like German, automatically constructs a representation for 'S V O' sentences that directly reveals the parameter value as it is encoded in UG—whether that be in terms of overt material in C, a strong feature on C, etc. Of course, a speaker of a -V2 language, like English, forms a very different syntactic percept of the same string. Therein lies the rub for a language learner.

In this chapter I will present and generalize Fodor's proposals about how, unlike adults, learners, with no parametric commitments, might be able to initially perceive this sort

---

[42] In actuality, some of Dresher and Kaye's (1990) and Dresher's (1994) cues bridge the epistemological gap in the sense that they require an instantiation and testing of competing grammars.

of "parametric ambiguity" for sentences like 'S V O', and more, importantly, how certain parametrically unambiguous sentences might push the learner to adopt particular values for their initially open parameters. In Section 6.1, I will present Fodor's (1995) proposal and suggest two natural generalizations of the proposal. As indicated in Chapter 1, I will refer to this class of learners as On-line Parsing Logic (OPL) learners, since they draw conclusions about parameter setting by a simple inspection of the set of structures inferred by the parser. Generalizations to Fodor's proposal are motivated, in part, by difficulties that arise in application to an extended parametric space, and, perhaps, even in the application to the particular syntactic space—the familiar Gibson and Wexler three-parameter space—that provides Fodor's leading example. I will point out these difficulties throughout. In Section 6.2, I will discuss the set-theoretic properties that must hold in order for these OPL learners to be able to acquire a parametric space. In Section 6.3, I will describe an extended version of the original Gibson and Wexler syntactic space due to Wexler, Gibson, Bertolo and myself. In Section 6.4, I will describe the extent to which the OPL proposal is applicable to the current space and point to several problems inherent in the set-theoretic relations among grammars in the space. In Section 6.5, I will briefly describe a set of phrase structure rules for this space. This discussion is intended to motivate discussion of the feasibility of the superparsing approach. This set of rules, list in the Appendix, is capable of parsing all and only strings in the data sets described in Section 6.3. and is suitable for use by an OPL learner. Section 6.5 will also discuss the need for a defense of the scalability of this approach. It is important to ensure that exponential growth is not pushed off into the resource requirements of the parsing process.

The terminology developed in Chapter 3 to discuss the effect of cues on the set of the learner's hypotheses carries over directly to discussions of hypothesis change for the learner

that Fodor proposes, as well as several less restrictive variants that I will discuss here. As indicated in Chapter 1, I will call the class of learners inspired by Fodor's proposal On-line Parsing Logic (OPL) learners. OPL learners behave essentially like parametric cue-based systems in their deductive narrowing of the space of possible languages. As we will see, in fact, given a parametric space and a member of the OPL family, it is possible, in principle, to construct a cue-based learner (possibly an infinite one) that would exhibit an equivalent course of acquisition. In my interpretation of the basic proposal, in fact, the system relies entirely on what are, in some sense, the simplest possible cues—form observations. There are no special-purpose devices outside of the parser that monitor for special properties of the input stream, or store and analyze larger sets of data. Crucially though, unlike cue-based systems, OPL learners do not explicitly encode this set of form observation cues in UG. Rather the learner determines that forms provide cues on-line by a simple, mechanistic bit of reasoning about a set of parses that it constructs.

6.1 Fodor's (1995) Proposal

Fodor's proposal is elegant and, conceptually at least, quite straightforward. Her leading example is the Gibson and Wexler 3-parameter syntactic space. In this space, she notes that a learner's problems would largely be over if they could only somehow hear the syntactic *structures* of their native language, rather than just the corresponding *strings* of syntactic categories. The following example captures this insight. The 'S V O' sentence "John saw Bill" could come from a grammar that was either +V2 or -V2. The structure [$_{CP}$ John [$_{C'}$ saw [IP ... Bill...]]], on the other hand could only have come from a +V2 grammar because the verb has

moved into the C position, and this can only happen in +V2 grammars.[43] In current conceptions of syntax, parameter values tend to correspond to sets (often singleton sets) of discretely identifiable pieces of the phrase structure used to assemble syntactic trees. For example, a +V2 grammar, in the spaces considered in this paper, can be identified by the presence of a matrix C filled with finite verbal material. The presence of an XP in the matrix spec-CP positions also diagnoses +V2. More abstractly, it has been proposed (see, for example, Chomsky 1992) that the C position in +V2 languages is marked with a particular feature which has the effect of drawing verbal material to it. Alternatively, empty matrix C's and spec CP's indicate that a grammar is –V2.

A learner that was able, somehow, to produce all and only the possible phrase structure analyses allowed by UG that corresponded to strings they heard from the target language would be in a position to confidently adopt certain parameter values and reject others. If every parse that resulted from a string made use of a piece of structure corresponding to a particular parameter value, then that parameter value should be adopted; alternative values for the parameter, on the other hand, should be rejected. If this *superparsing* process also returned humanly impossible parses that expressed inconsistent parameter values, further caution would be required.

As earlier discussion has made clear, for this proposal to be tractable in large parametric spaces, the learner must be able to perform this search through possible parses in a way that does not lead to a run-in with the problem of exponential growth. Clearly, it will not do to simply reconfigure one version of the parser for each of the possible grammars in the

---

[43] Recall that Gibson and Wexler assume that context indicates that John is the subject and Bill is the object.

space. Fodor proposes, instead, that the learner parse with a single *supergrammar* that contains all the universal elements of phrase structure, as well as all the components of phrase structure that correspond to particular parameter values. If parameters are binary, then the supergrammar will be on the same scale as an individual adult grammar. Among the set of $2n$ parameter values for n parameters, there will be some parameter value which requires a maximal number of phrase structure components. Any language that adopts this parameter value will require at least this maximal number of phrase structure components. The entire supergrammar, on the other hand, cannot require more than $2n$ times this maximal amount.

In particular, Fodor proposes that the learner uses the following algorithm: First, the learner attempts to parse the input that it receives with the set of phrase structure components that it has either assumed innately or has already been forced to adopt. If this initial parse succeeds, then no modification needs to be made to the grammar and the learner can move on to the next input. On the other hand, if the initial attempt at parsing fails, the learner, in the absence of noise, is clearly missing some necessary structural component.[44] The next step is to attempt a new parse—this time, with the more extensive supergrammar providing additional raw material for parse construction. Fodor's particular proposal dictates that if this second parsing effort with the *full* supergrammar, containing all parametrically available structures that human language provides, yields a unique parse, then any parameter values that contribute structural components to this parse are adopted by the learner. Fodor also claims that the learner is not constrained in any way to be parametrically consistent within a parse; the superparser might, then, build a structure that required, say, both a comp-final and a comp-

[44] The learner must also be able to identify the relevant syntactic categories and features of the words in the string, as must Gibson and Wexler's TLA.

initial piece of phrase structure. Clearly, in the absence of noise, the learner will not *misset* parameters. If there really is a unique parse compatible with the input, then the language that the learner is acquiring must require the parameter values expressed in this unique parse for its generation. The only potential logical problem of learnability for a strictly Fodorian learner, or indeed any of the OPL learners that I will present, arises if parameters fail to get set. This is true because they rely only on positive evidence.

In its details, Fodor's proposal is more stringent than the logic of cue-based parameter setting requires in at least two ways. First, it fails to allow the learner to exploit previously established parameter settings. If a learner has already rejected certain parameter values, then there is no reason to keep the corresponding rejected structures in the supergrammar. A sentence that would not receive a unique parse if the learner had not yet set any parameters may well receive a unique parse in the context of a set of previously fixed parameters. Second, even if the parsing effort yields multiple parses, in some cases there is enough information in the set of successful parses to narrow the range of possible values for parameters. If all the structures that the parser returns provide evidence for a particular parameter value, it is plainly safe to adopt that parameter value and reject alternative values. Similarly, if none of the structures makes use of a particular parameter value's structures, it is safe to reject that parameter value. This is a natural extension of Fodor's proposal, and, in fact, one that she mentions in passing. With strictly binary parameters, of course rejecting one value and accepting another go hand in hand. It is only with multi-valued parameters that the possibility of rejecting a parameter value without definitively setting a parameter arises.

This OPL scheme requires the ability to parse and the ability to recognize which parameter value a piece of syntactic structure corresponds to. Some form of the first ability is

well-established in theories of human psycholinguistics. The second ability follows without much effort from UG on the structural conception of parameters that Fodor argues for. The learner must simply be able inspect a syntactic structure for the component pieces that express particular parameter values. Note, however, that this scheme requires the learner to parse in an *exhaustive mode* that may be quite different than the mode that the adult processor normally uses. In the adult sentence processing literature, there is considerable debate between proponents of serial processing views, who maintain that the human sentence processing mechanism only pursues a single parse at a time, and proponents of limited parallelism, who maintain that this mechanism is capable of holding several alternative partial parses in mind. However, there does not seem to be any serious argument for the claim that the parser maintains *all* alternative partial parses. Of course, these discussions have to do with *within-language* ambiguity, so it is not clear that they bear directly on considerations of *between-language* ambiguity of the sort that could confront and OPL learner. Nonetheless, it is clearly crucial that the OPL learner constructs *all* parses that are humanly compatible with the sentences that it processes. Otherwise, the absence of a possible parse from the set returned by the parser could erroneously lead to a parameter value being eliminated.

Holding off on the question of whether such parsing is possible, notice what the OPL proposal would accomplish in spaces for which there is no logical barrier to its application—a key caveat. In earlier discussion of cue-based algorithms and in Fodor (1995), it was noted that it was easy to provide form-based cues for the Gibson and Wexler space since every grammar contained strings that it alone generated. The concern that I raised was that the need to explicitly store these uniquely generated forms for each possible grammar in the space would prove overwhelming in large parametric spaces. An OPL learner, however, can be viewed as a

cue-based learner, driven by form observations, that does not need to explicitly know in advance what forms it is looking for. In particular, any OPL learner is isomorphic to a parametric cue-based learner with fully explicit form cues. Given a particular OPL learner and a parametric space, it would be possible to deduce all the possible learning paths that the learner might follow and to determine all the form observations that would drive the learner along those paths. This done, it would be possible to explicitly encode this information in a (potentially infinite) cue-based algorithm. The discussion of the form-based cues for the Gibson and Wexler space focused on the dramatic possibility of setting all parameters at once, but this is clearly not necessary.

In the following section, I will lay out the properties that must hold of binary parameter spaces in order for OPL learners to be successful. I will do so for two variants: (1) the particular learner that Fodor proposes, which requires a unique parse with the full supergrammar in order to set parameters, (2) a learner that may inspect the entire set of parses for the occurrence of a parametrically licensed structure, and that can eliminate structures corresponding to rejected parameter values from the supergrammar. It should be straightforward to generalize this discussion to the case of a learner that eliminates parameter values for multi-valued parameters, but I will not do so here. The requirements for an intermediate OPL learner which requires a unique parse, given certain previously fixed parameter settings should be reasonably clear once the requirements for (1) are described.

Before I proceed, however, a side note. The potential problem of noise has not been addressed extensively elsewhere in this thesis, but it seems worthwhile to suggest that OPL learners might be extended in a way that could prove robust in the face of noise. Clearly, an OPL learner will not change its hypothesis when it encounters a form that is not parsable given

the parameters that it has already fixed. The OPL learner could, however, be misled if it encountered consistent with languages other than the target. This is also, of course, true with the TLA and other IGMS algorithms. When the TLA changes a parameter value, there is no presumption that the form that caused the change requires that parameter value. With OPL learners, there clearly is. An OPL learner, suitably extended, would be able to simply keep count of the number of input forms that motivated a particular parameter setting and, perhaps, refrain from setting a parameter until its confidence that such a setting was motivated crossed some crucial threshold.

6.2 The Logical Requirements on Parametric Spaces for OPL learning

Unique Parse, Full Supergrammar learner (UPFS)

The requirements on a binary parameter space for Fodor's OPL learner—call this a unique parse, full supergrammar (UPFS) learner are clearly stated in terms of the vocabulary developed in Chapter 3. Since the learner does not restrict the membership of the supergrammar during acquisition, any cues that act to set parameters must be independent. That is to say, they must be capable of acting when there are no fixed parameters. As alluded to above, there is plainly a sufficient condition on parameter spaces that will guarantee success. If a grammar uniquely generates a particular string, then, in order to parse that string, the parser must make use of some structural component from each of the grammars parameter values. Otherwise, an alternative parse that did not require one of the grammar's parameter values would be possible. Note, however, that this would contradict the claim that the grammar

uniquely generated the string. If each grammar in a parametric space generates unique strings, then "one-trial" acquisition of a grammar emerges as a possibility. Upon encountering such a string, the learner could successfully set all of its parameter values at once. This striking possibility would be consistent with an extreme version of Wexler's (1996) Very Early Parameter Setting hypothesis, which is motivated by the empirical finding that there is very little evidence to suggest that children have misset parameters during the course of acquisition.

Of course, this striking possibility is hardly necessary. While Fodor's learner does require a unique supergrammar parse for a string in order to set any unset parameters, that unique parse need not shed light on the value of every parameter in the space. It is possible, and almost certainly true, that certain syntactic structures will not make use of any of the structures associated with any of the values of a particular parameter. For example, as Fodor notes, in the Gibson and Wexler system, the string "V S" receives a unique, –V2 parse. Within the context of the system, the structure is clearly spec-final. Its value for the comp-final parameter, however, is indeterminate. It is impossible to tell the relative ordering of heads and their complements since the structure contains no complements. The string is uniquely explained by setting a subset of the system's parameter values, and it works even though no other parameters have been fixed. More concretely, on hearing "VS" a learner in the Gibson and Wexler space can set its grammar to –V2, spec-final. In order for a target language to be acquired, every parameter value for the grammar that generates the target must participate in a unique explanation for some string in the language. In particular, there must be independent, sufficient form-observation type cues for each parameter, for each language in the space. The pressure to provide *generally available* cues that would work for all languages is reduced by the elimination of the need to explicitly encode the machinery for using cues.

However, the existence of such forms is a necessary, but not quite sufficient, condition for the UPFS learner. The fact that the UPFS learner requires a unique parse with a supergrammar that, on Fodor's assumptions, could potentially return parses that are incompatible with a single human grammar requires an additional condition to be satisfied.

Obviously, any "impossible" parses with incompatible parameter values could block the applicability of crucial cues that would work if such incompatible parses were ruled out. This must not be the case.

On-line form observer (OLFO)

Relaxing the requirement on unique parses, forcing the supergrammar parser to respect previously fixed parameters and disallowing impossible parses puts us squarely in the domain of parametric, cue-based learners that restrict themselves to form observations. I would like to attend most closely to the logical requirements of this OPL variant, which I will call an On-line Form Observer (OLFO). This variant can be placed in exact correspondence with the class of parametric, cue-based learners that restricts itself to form observations. Given OLFO and a parametric space, it is possible to exhaustively determine all possible learning paths and the form observations that drives learners along them. Of course, the big difference here is that the cues on the decision trees implicit in both types of algorithms do not receive explicit mental representation for an OPL learner. Since there is no burden of explicit representation, very large and, even, highly redundant decision trees with multiple paths to the same grammars become reasonable if intractable complexity is not hidden in the "black box" of the parser.

The general conditions under which such learners succeed are laid out in Chapter 3, but these conditions are not stated directly in terms of the set-theoretic relations among the languages in a parametric space. Doing so, I believe, leads to a clearer insight about problems that can arise for the OLFO. It is easy to show that there are binary parametric spaces for which it is not possible to provide a parametric, cue-based learner that restricts itself to form observations. The artificial parameter space in (6.1.1) provides such an example. The data generated by the grammars in the space consist of the names of all *other* grammars in the space. (The "strikethrough" is intended to draw attention to the fact that a form is not in the language.) Here, no form is a sufficient indepent cue for any parameter, so the parameter setting process can never begin.[45]

(6.1.1)

Grammar 0 0 0: {~~0 0 0~~, 0 0 1, 0 1 0, 0 1 1, 1 0 0, 1 0 1, 1 1 0, 1 1 1}

Grammar 0 0 1: {0 0 0, ~~0 0 1~~, 0 1 0, 0 1 1, 1 0 0, 1 0 1, 1 1 0, 1 1 1}

Grammar 0 1 0: {0 0 0, 0 0 1, ~~0 1 0~~, 0 1 1, 1 0 0, 1 0 1, 1 1 0, 1 1 1}

Grammar 0 1 1: {0 0 0, 0 0 1, 0 1 0, ~~0 1 1~~, 1 0 0, 1 0 1, 1 1 0, 1 1 1}

Grammar 1 0 0: {0 0 0, 0 0 1, 0 1 0, 0 1 1, ~~1 0 0~~, 1 0 1, 1 1 0, 1 1 1}

Grammar 1 0 1: {0 0 0, 0 0 1, 0 1 0, 0 1 1, 1 0 0, ~~1 0 1~~, 1 1 0, 1 1 1}

---

[45] An unrestricted cue-based learner could, in the limit, acquire such a space. A particular language can be cued, for example, by the observation of all the sentences that it generates. Of course, this is a space for which no learner can hope for a good outcome in an exponentially growing space. The best that a learner can do here is to rule out a single grammar at every step; hearing the name of a grammar indicates only that that particular grammar is not the target. I have yet to construct an example that would thwart a parametric, cue-based learner, but be amenable to the TLA in a large space.

Grammar 1 1 0: {0 0 0, 0 0 1, 0 1 0, 0 1 1, 1 0 0, 1 0 1, ~~1 1 0~~, 1 1 1}

Grammar 1 1 1: {0 0 0, 0 0 1, 0 1 0, 0 1 1, 1 0 0, 1 0 1, 1 1 0, ~~1 1 1~~}

When generally will the OLFO succeed and fail in binary parameter spaces? At each point during acquisition, the OPL algorithm has fixed the settings of certain parameters, while certain other parameters remain open. In effect, the OLFO learner has more closely pinpointed the location of the target grammar: it falls within the subspace of grammars defined by all possible settings of the open parameters. What property must hold of this subspace so that it will be possible to continue setting parameters for any target grammars that fall within it?

Further OLFO parameter setting will be guaranteed if and only if each grammar in the subspace generates forms that are only generated by a natural subclass of the subspace. My use of natural class here is the typical one from linguistic theory. A class is said to be natural if it can be specified by indicating the value of certain features (here parameters) and leaving the value of others unspecified. If a particular target fails to generate such forms, then it will be impossible for the learner to make further progress in parameter setting for the language generated by this target. By hypothesis, no form that the language generates is consistent with only a natural subclass of the sort that would result from further parameter setting, so the OLFO cannot proceed for the particular target. If, on the other hand, a target does generate such forms, then those forms will provide an appropriate parsing cue for the OLFO to fix further parameters—in particular, those parameters that pick out the appropriate natural subclass. By hypothesis, the forms are only generated by a natural subclass of the subspace. All parses of such forms, then, must involve the parsing structures that define the parameters that pick out the subclass. Otherwise it would be possible to construct some alternative parse for a

language in the subspace that did not require them. This would mean that they were generated by some grammar that fell outside of the natural class—a contradiction. If all grammars in the subspace generate such forms, then OLFO parameter setting can always proceed.

Inductively, then, it is possible to see what must hold for the space as a whole to be learned by this variant of the OLFO algorithm. There must be no non-trivial (i.e., single member) subspaces of the parameter space that do not allow further parameter setting to take place. In other words, there must be no non-trivial subspaces that contain languages without any forms that, within the confines of the subspace, can only be generated by a natural subclass of the subspace.

What can be said for parameter spaces that do contain such problematic subspaces? If all languages in a problematic subspace are weakly equivalent, it would be possible to postulate a final, additional "clean up" step that simply selected one of the weak equivalents. If this is not the case, then this won't help. Arbitrarily selecting a language in the space could cause convergence to an incorrect grammar. An example of this most problematic of cases is shown in (6.2.1). Clearly none of the languages are weakly equivalent; all the sets of sentences are different. Nonetheless, no form is generated by a natural class in the system. Arbitrary choice, here, stands a 3/4 chance of leading to failure to converge.

(6.2.1) Parameter A+, Parameter B+: {A, C, D}

Parameter A+, Parameter B+: {A, B, D}

Parameter A+, Parameter B+: {A, B, C}

Parameter A+, Parameter B+: {B, C, D}

6.3 A Syntactic Test Space

In order for the OPL proposal to apply directly to a space, it must be possible to decompose the structures in a linguistic space into independent, parametrically available or universal components that "snap together", tinker-toy style. This is definitely not the case in the Halle and Idsardi system of parameters for metrical phonology. The constituent and metrical structure that arises in that system is, in some sense, an epiphenomenon of the parameter settings that a language adopts. A number of parametrized bracket insertion and grid extension processes contribute to these structures in a highly interactive fashion. It is not so straightforward to zero in on a modular piece of the structure and identify it as the reflex of a single, particular parameter value. Perhaps other more "component-based" alternatives to the Halle and Idsardi systems would lend themselves more directly to Fodor's scheme.

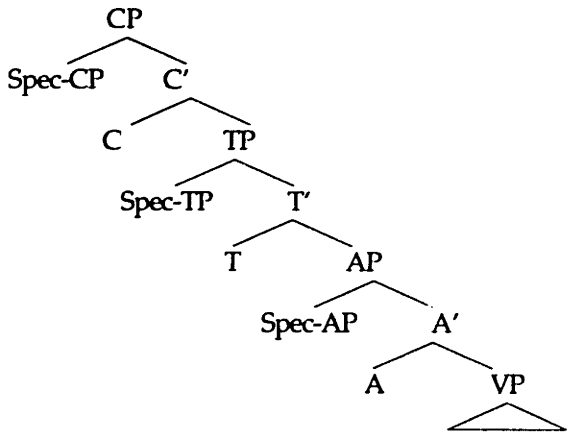
From here on out the focus will be on syntax. The space presented in this section takes the Gibson and Wexler parameter space as its starting point.

In the Gibson and Wexler space, remember, Fodor claimed that the UPFS learner could clearly succeed. A simple demonstration of this pointed to the existence of unique strings for every language. This *would* be true if the learner were simply a cue-based learner not subject to additional requirements on the output of the parser. Note, however, that for the UPFS this claim depends crucially on certain further assumptions about the superparser's production of impossible parses. Enforcing consistency of parameter values in a parse would eliminate this possible concern, but, as will be shown, leads to complication of the grammar. As a reminder, that system contained three parameters : a) a specifier-head parameter with values spec-first

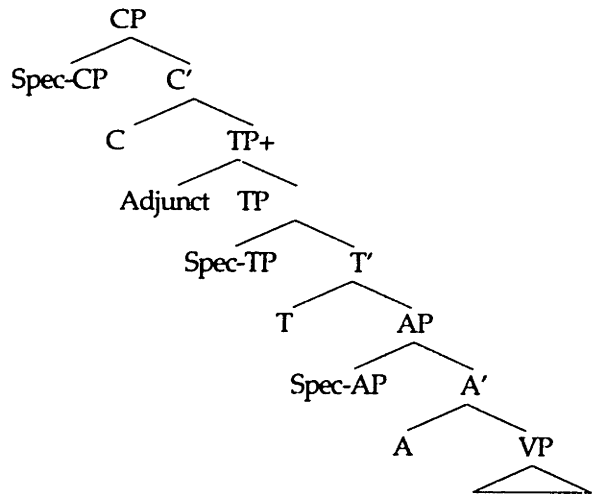
and spec-final, b) a complement-head parameter with values comp-first and comp-final and c) a V2 parameter with values +V2 and -V2, which indicates that a finite verb moves from its base position to the second position in root declarative clauses.

The space described in this section expands the base X-bar structure of a root clause in keeping with the split-Infl hypothesis (see, for example, Emonds 1978, Pollock 1989). The basic clausal architectures, ordered somewhat arbitrarily to reflect the base word order of English, is shown in (6.3.1) and (6.3.2). Instead of a single IP level, there are now separate TP (Tense Phrase) and AP (Agreement Phrase) levels.

(6.3.1)



(6.3.2)



First, I will explicitly describe the base structures formed by "populating" this clausal architecture with syntactic categories. Then, I will review the effect of the original parameters and introduce the three new parameters indicated below in (6.3.4).

The following list exhaustively describes which categories occupy which base positions before the application of any movement transformations.

- In every sentence, a subject occupies Spec-AP.
- A sentence may contain either an auxiliary and a verb, or a verb alone. It must contain one of these two combinations. Auxiliaries originate in T. Verbs originate in VP. A fuller specification of the contents of VP is given below.
- If the sentence does not contain an adjunct, the basic clausal architecture is as in (6.3.1). If the sentence does contain an adjunct, the basic clausal architecture is as in (6.3.2) and the adjunct occupies the slot labeled "Adjunct".
- The following VPs are universally available in the base of all grammars:

V, V Adverb, Adverb V.

- The following VPs, in combination with the universally available VPs exhaust the possibilities for comp-final grammars:

V Object, V Object Adv, Adv V Object, V Object-1 Object-2, V Object-1 Object-2 Adv, Adv V Object-1 Object-2.

- The following VPs, in combination with the universally available VPs exhaust the possibilities for comp-first grammars:

Object V, Object V Adv, Adv Object V, Object-2 Object-1 V, Object-2 Object-1 V Adv, Adv Object-2 Object-1 V.

To briefly summarize the VP possibilities. An adverb can optionally appear at the left or the right edge of the VP. A verb can be (1) intransitive, (2) simply transitive with a single object immediately adjacent to it on the side appropriate to the setting of the comp-final parameter or (3) ditransitive with the first object immediately adjacent to the verb and the second object immediately adjacent to the first object, again in accordance with the setting of the comp-final parameter. The "flat" structure of the VP simply reflects the current implementation in our system and should not be construed as making any strong claims about VP internal phrase structure.

The original spec-first/spec-final parameter, as applied to the extended clausal architecture determines the relative base order of Spec-AP and A'. Spec-CP, as in Gibson and Wexler's original system, still falls invariantly to the left of C'. It turns out that Spec-TP is never occupied in the present parametric system, so its order relative to T' is not so important.

In the original system, the comp-first/comp-final parameter had the effect of specifying a set of possible base VPs. It still has that effect here, as noted above. In addition, it fixes the relative ordering of T and AP, and of A and VP. The relative order of C and TP is independently parametrized, as will be discussed below.

In the original system, one transformation was possible if the V2 parameter was set to its positive value. In +V2 languages in the current system, the rule in (6.3.3) is used to transform the base structure.

(6.3.3) If the sentence contains an auxiliary, move the auxiliary to C. In addition, select a single category from the tree that falls in the set {Subject, Object, Object-1, Object-2, Adjunct} and move it to Spec-CP.

If the sentence does not contain an auxiliary, move the verb to C. In addition, select a single category from the tree that falls in the set {Subject, Object, Object-1, Object-2, Adjunct} and move it to Spec-CP.

The -V2 value of the parameter does not specify any motion.

The extended system provides the additional parametric options listed in (6.3.1)

(6.3.4) a. a parameter governing the relative order of C and TP.

b. a V-to-Agr parameter

c. a V-fronting parameter

Recall that in the Gibson and Wexler (1994) system the relative ordering of C and the unarticulated IP was fixed. The parameter in (6.3.4.a) relaxes this restriction in order to take a step in the direction of capturing data from languages like Japanese and Korean that are generally assumed to have C on the right. Note that this switch makes the usual name for this parameter—V2—a misnomer. In a "+V2" language, with TP before C, the moved verbal material—auxiliary or verb—ends up in final position.

The V-to-Agr and V-fronting (again a misnomer in certain cases, it might be better to call it V-to-T) parameters have the effect of extending the possible range of landing sites for moved verbal material. The V-to-Agr rule is stated in (6.3.5), while the V-fronting rule appears in (6.3.6).

(6.3.5) +V-to-Agr:

If there is no auxiliary, move the verb to A. Otherwise, move the auxiliary to A.

– V-to-Agr:

No movement is specified.

(6.3.6) +V-fronting:

If there is no auxiliary, move the verb to T.

–V-fronting:

If there is an auxiliary, move it to A.

The transformation rules are understood to apply in the following order: (1) V-to-Agr, (2) V-fronting, (3) V2.

The addition of the V-to-Agr parameter is intended to allow the system to begin to capture languages like French where the verb is argued to move to a category intermediate

between C and V. For example, following Emonds (1978) and Pollock (1989), we have the following alternation:

(6.3.7) a. Le chat ne chasse pas le chien.
"The cat isn't chasing the dog."

b. Ne pas aimer ses parents est une mauvaise chose.
"Not to love one's parents is a bad thing."

In (6.3.7.a), the finite verb has raised to the left of negation *pas*. Since *pas* is argued to be a marker of the left edge of VP, the verb *chasse* must have moved from its base position. In the present system the adverbial category plays the role of delineating an edge of the VP. Note that we do not explicitly indicate which edge an adverb delineates. If adverbs bore this information on their faces, if, for example, there were separate classes of "left-edge" and "right-edge" adverbs, things would only become simpler for an OPL learner. Unlike the TLA, which in some cases must exploit ambiguity to make progress, an OPL learner thrives on this sort of disambiguating information. There are a number of different theories characterizing what we have described as the V-to-Agr parameter (see, for example, Chomsky 1992, Bobaljik 1994). Our implementation is not sufficiently rich to involve us in detailed choices among particular theories of this phenomenon.

The addition of the V-fronting parameter is intended to allow the system to begin to capture "VSO" languages like Irish.

An additional parameter governing the presence of V2 movement in embedded clauses has also been implemented for the system. This parameter is not incorporated into discussion in this chapter which will focus entirely on data in root clauses.

Linguists might ask a lot of questions about the particular decisions that went into the implementation of the system at hand. These, however, will not be addressed here. For the rest of this chapter, I will simply take the system as a given and investigate the implications for OPL learners.

In principle, there are 64 (2^6) grammars defined by these 6 binary parameters. However, the 3 verb movements described by the V2, V-fronting and V-to-Agr parameters shadow one another in the following sense. If a grammar is +V2, the finite verbal element—auxiliary if there is one, otherwise the verb—ultimately moves to C. The fact that, on its way, it might have stopped off in either Agr or T cannot be detected from the surface string. In this case, the values of the V-fronting and V-to-Agr parameters are effectively obscured. Similarly, if a -V2 grammar is +V-fronting, its value for the V-to-Agr parameter is obscured. As a result, the number of distinct grammars in the space is reduced. Moreover, in matrix clauses—the source of our data for this section—the positioning of C relative to TP can only potentially be observed in +V2 languages. In -V2 languages, nothing occupies the C position.

There are at most 8 distinct +V2 grammars—one for each possible setting of the unobscured spec-head, comp-head and C position parameters. There are at most 4 distinct –V2, +V-fronting grammars because the C parameter has no effect for –V2 languages. Similarly, there are 4 distinct -V2, -V-fronting, +V-to-Agr parameters. Finally, there are at most 4 distinct -V2, -V-fronting, -V-to-Agr grammars. From these considerations, then we can see that there are at most 20 possible distinct languages in the space. We will see below that, among these 20 groupings into possibly distinct languages, two additional pairs of groups also turn out to be equivalent, at least with respect to the fragment of the grammar that has been implemented so far.

As we have seen above, the existence of such structurally distinct, but string-equivalent grammars in a parametric system weak can cause a problem for OPL learners. If the target is one of the weak equivalents, then no OPL learner will be able to satisfactorily set any parameters that the set of weak equivalents of the target differ on. Clearly, no string could provide definitive evidence for the setting of one of these parameters. If the weak equivalents constitute a natural class, then the situation might be resolved by adding an additional final step to the learning process which arbitrarily fixes unset parameters. If the weak equivalents do not constitute a natural class, a learner who insists on a unique parse involving a piece of structure corresponding to a parameter value in order to set a parameter will not be able to do so under these circumstances, and the learner will be left with several parameters unset.

6.4 Form Cues for the Parameters in the Test Space

The system at hand has a number of groups of weakly equivalent grammars that will quickly derail the UPFS learner, which requires unique parses as evidence for parameter setting. For example, consider that every sentence in the grammar requires a C position, but only +V2 grammars are capable of providing explicit evidence about the order of C. Therefore, there will always be at least two parses for sentences from –V2 grammars—one with C on the left, one with C on the right. The UPFS, therefore, will not be able to make any progress at all in learning –V2 grammars. Moreover, it is quite easy to construct impossible parses for every possible parse in the space, as the following example shows. Any given grammar can only place overt material in a subset of the heads in the system: +V2 grammars use only C and V, grammars that are –V2 and +V-fronting use only T and V, grammars with no movement only use

A and V. The relative ordering of A and T and their complements is governed by a single parameter, but since only one of these heads can be filled in any language, only one position is "detectable" in the string. An unconstrained superparser, then, would always be free to adopt any value for the comp-first parameter when it builds the level of phrase structure containing such a head. In particular, the superparser can choose to make the ordering of the undetectable head either compatible or incompatible with the ordering of other heads in the system. If the UPFS is correct, then, clearly, these cases must prove to be an artifact of this system. Language architecture must make it such that in order to build syntactic structures it is necessary to adopt one parameter or another if there is not always evidence as to which should be adopted.

Let us consider, then, how the more powerful OLFO learner fares in the space. This is the least restrictive of the OLFO variants we have considered, so any problem for the OLFO will be a problem for the OPL approach in general, and for any cue-based system that relies entirely on form observations. Since the space contains sets of weakly equivalent grammars, there is already a problem for the OLFO learner, but the OLFO learner, unlike Fodor's OPL, is at least capable of setting parameters in this system up until the point where this weak equivalence makes it logically impossible to definitively set a parameter. Table 6.4.1 groups languages together into the weakly equivalent classes that were identified above from an initial inspection of the mechanics of the parameter system. These groupings all constitute natural classes, so if there were always a sufficient set of cues to allow the learner to narrow its hypotheses to one of these natural classes, then a suitably extended OLFO could arbitrarily choose one of the grammars in the subspace. This, of course, would represent a deviation from OLFO's otherwise straightforward reliance on positive evidence. Presumably, the learner

would only do so after a substantial period in which it was unable to modify its hypothesis any further.

For 16 out of the 20 natural classes of weak equivalents that were identified at the end of Section 6.3, there are strings that are generated by all and only the grammars in that natural class. These strings are listed in Table 6.4.1. Upon parsing one of these strings, the OLFO learner would discover that all the parses that it was able to construct required the parameter settings that defined the natural class. For example a sentence of the form, "Adjunct0 Adv Obj0 Subj0 Verb0" would lead the OLFO learner to fix its parameters to spec-final, comp-final, c-final, +V2, while leaving the V-to-Agr and V-fronting parameters open. Table (6.4.1) exhaustively lists the uniquely generated strings for the 20 natural subclasses that were picked out at the end of Section 6.3.

It is important to note that while unique strings can allow many parameters to be set at once, this is not a requirement of the OPL approach. Here, I am simply trying to establish the logical limits of the OPL approach as applied to this space. Since there are unique strings found in many of the weakly equivalent natural classes established above, I am focusing my attention there.[46]

[46] Preliminary evidence from simulation of OPL learning for the space indicates, in fact, that unique strings are not required for the acquisition of certain of the weakly equivalent languages.

Table 6.4.1
Unique Strings for Natural Classes in the System
1. Spec-final 2. Comp-final 3. C-final 4. V-to-Agr 5. V-fronting 6. V2

Group	1	2	3	4	5	6	Data
1	yes	yes	yes	y/n	y/n	yes	(ADJUNCT0 ADV OBJ0 SUBJ0 VERB0)
							(ADJUNCT0 OBJ1-0 OBJ2-0 SUBJ0 VERB0)
							(OBJ1-0 ADJUNCT0 ADV OBJ2-0 SUBJ0 VERB0)
							(OBJ2-0 ADJUNCT0 ADV OBJ1-0 SUBJ0 VERB0)
							(ADJUNCT0 ADV OBJ1-0 OBJ2-0 SUBJ0 VERB0)
							(ADJUNCT0 VERB0 OBJ0 SUBJ0 AUX0)
							(OBJ0 ADJUNCT0 ADV VERB0 SUBJ0 AUX0)
							(ADJUNCT0 ADV VERB0 OBJ0 SUBJ0 AUX0)
							(OBJ1-0 ADJUNCT0 VERB0 OBJ2-0 SUBJ0 AUX0)
							(OBJ2-0 ADJUNCT0 VERB0 OBJ1-0 SUBJ0 AUX0)
							(ADJUNCT0 VERB0 OBJ1-0 OBJ2-0 SUBJ0 AUX0)
							(OBJ1-0 ADJUNCT0 ADV VERB0 OBJ2-0 SUBJ0 AUX0)
							(OBJ2-0 ADJUNCT0 ADV VERB0 OBJ1-0 SUBJ0 AUX0)
							(ADJUNCT0 ADV VERB0 OBJ1-0 OBJ2-0 SUBJ0 AUX0)
							(OBJ1-0 ADV OBJ2-0 SUBJ0 VERB0)
							(OBJ2-0 ADV OBJ1-0 SUBJ0 VERB0)
							(OBJ0 ADV VERB0 SUBJ0 AUX0)
							(OBJ1-0 VERB0 OBJ2-0 SUBJ0 AUX0)
							(OBJ2-0 VERB0 OBJ1-0 SUBJ0 AUX0)
							(OBJ1-0 ADV VERB0 OBJ2-0 SUBJ0 AUX0)
							(OBJ2-0 ADV VERB0 OBJ1-0 SUBJ0 AUX0)
2	yes	yes	y/n	yes	no	no	NONE
3	yes	yes	y/n	y/n	yes	no	NONE
4	yes	yes	y/n	no	no	no	(ADJUNCT0 ADV VERB0 OBJ0 SUBJ0)
							(ADJUNCT0 ADV VERB0 OBJ1-0 OBJ2-0 SUBJ0)
							(ADV VERB0 OBJ0 SUBJ0)
							(ADV VERB0 OBJ1-0 OBJ2-0 SUBJ0)
5	yes	yes	no	y/n	y/n	yes	(OBJ1-0 VERB0 ADJUNCT0 ADV OBJ2-0 SUBJ0)
							(OBJ2-0 VERB0 ADJUNCT0 ADV OBJ1-0 SUBJ0)
							(OBJ0 AUX0 ADJUNCT0 ADV VERB0 SUBJ0)
							(OBJ1-0 AUX0 ADJUNCT0 VERB0 OBJ2-0 SUBJ0)
							(OBJ2-0 AUX0 ADJUNCT0 VERB0 OBJ1-0 SUBJ0)
							(OBJ1-0 AUX0 ADJUNCT0 ADV VERB0 OBJ2-0 SUBJ0)
							(OBJ2-0 AUX0 ADJUNCT0 ADV VERB0 OBJ1-0 SUBJ0)
							(OBJ1-0 VERB0 ADV OBJ2-0 SUBJ0)
							(OBJ2-0 VERB0 ADV OBJ1-0 SUBJ0)
							(OBJ0 AUX0 ADV VERB0 SUBJ0)
							(OBJ1-0 AUX0 VERB0 OBJ2-0 SUBJ0)
							(OBJ2-0 AUX0 VERB0 OBJ1-0 SUBJ0)
							(OBJ1-0 AUX0 ADV VERB0 OBJ2-0 SUBJ0)
							(OBJ2-0 AUX0 ADV VERB0 OBJ1-0 SUBJ0)

6	yes	no	yes	y/n	y/n	yes	(OBJ1-0 ADJUNCT0 OBJ2-0 ADV SUBJ0 VERB0) (OBJ2-0 ADJUNCT0 OBJ1-0 ADV SUBJ0 VERB0) (OBJ0 ADJUNCT0 VERB0 ADV SUBJ0 AUX0) (OBJ1-0 ADJUNCT0 OBJ2-0 VERB0 SUBJ0 AUX0) (OBJ2-0 ADJUNCT0 OBJ1-0 VERB0 SUBJ0 AUX0) (OBJ1-0 ADJUNCT0 OBJ2-0 VERB0 ADV SUBJ0 AUX0) (OBJ2-0 ADJUNCT0 OBJ1-0 VERB0 ADV SUBJ0 AUX0) (OBJ1-0 OBJ2-0 ADV SUBJ0 VERB0) (OBJ1-0 OBJ2-0 VERB0 SUBJ0 AUX0) (OBJ1-0 OBJ2-0 VERB0 ADV SUBJ0 AUX0)
7	yes	no	y/n	yes	no	no	(ADJUNCT0 OBJ0 ADV VERB0 SUBJ0) (ADJUNCT0 OBJ2-0 OBJ1-0 ADV VERB0 SUBJ0) (OBJ0 ADV VERB0 SUBJ0) (OBJ2-0 OBJ1-0 ADV VERB0 SUBJ0)
8	yes	no	y/n	y/n	yes	no	(ADV SUBJ0 VERB0) (ADV SUBJ0 VERB0) (VERB0 SUBJ0 AUX0) (ADV VERB0 SUBJ0 AUX0) (VERB0 ADV SUBJ0 AUX0)
9	yes	no	y/n	no	no	no	(ADJUNCT0 OBJ0 VERB0 ADV SUBJ0) (ADJUNCT0 OBJ2-0 OBJ1-0 VERB0 ADV SUBJ0) (OBJ2-0 OBJ1-0 VERB0 ADV SUBJ0)
10	yes	no	no	y/n	y/n	yes	(ADJUNCT0 VERB0 OBJ0 ADV SUBJ0) (ADJUNCT0 VERB0 OBJ2-0 OBJ1-0 SUBJ0) (OBJ1-0 VERB0 ADJUNCT0 OBJ2-0 ADV SUBJ0) (OBJ2-0 VERB0 ADJUNCT0 OBJ1-0 ADV SUBJ0) (ADJUNCT0 VERB0 OBJ2-0 OBJ1-0 ADV SUBJ0) (ADJUNCT0 AUX0 OBJ0 VERB0 SUBJ0) (OBJ0 AUX0 ADJUNCT0 VERB0 ADV SUBJ0) (ADJUNCT0 AUX0 OBJ0 VERB0 ADV SUBJ0) (OBJ1-0 AUX0 ADJUNCT0 OBJ2-0 VERB0 SUBJ0) (OBJ2-0 AUX0 ADJUNCT0 OBJ1-0 VERB0 SUBJ0) (ADJUNCT0 AUX0 OBJ2-0 OBJ1-0 VERB0 SUBJ0) (OBJ1-0 AUX0 ADJUNCT0 CBJ2-0 VERB0 ADV SUBJ0) (OBJ2-0 AUX0 ADJUNCT0 OBJ1-0 VERB0 ADV SUBJ0) (ADJUNCT0 AUX0 OBJ2-0 OBJ1-0 VERB0 ADV SUBJ0) (OBJ1-0 VERB0 OBJ2-0 ADV SUBJ0) (OBJ2-0 VERB0 OBJ1-0 ADV SUBJ0) (OBJ0 AUX0 VERB0 ADV SUBJ0) (OBJ1-0 AUX0 OBJ2-0 VERB0 SUBJ0) (OBJ2-0 AUX0 OBJ1-0 VERB0 SUBJ0) (OBJ1-0 AUX0 OBJ2-0 VERB0 ADV SUBJ0) (OBJ2-0 AUX0 OBJ1-0 VERB0 ADV SUBJ0)

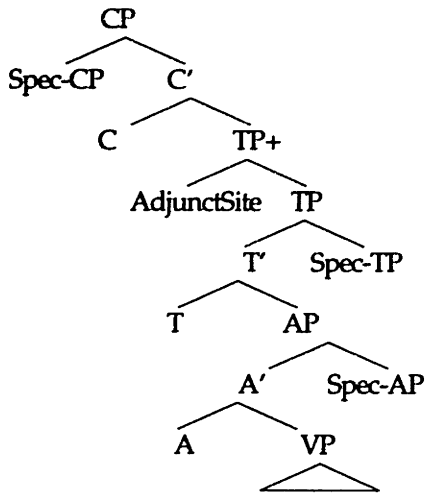
11	no	yes	yes	y/n	y/n	yes	(ADJUNCT0 SUBJ0 ADV OBJ0 VERB0)
							(ADJUNCT0 SUBJ0 OBJ1-0 OBJ2-0 VERB0)
							(OBJ1-0 ADJUNCT0 SUBJ0 ADV OBJ2-0 VERB0)
							(OBJ2-0 ADJUNCT0 SUBJ0 ADV OBJ1-0 VERB0)
							(ADJUNCT0 SUBJ0 ADV OBJ1-0 OBJ2-0 VERB0)
							(ADJUNCT0 SUBJ0 VERB0 OBJ0 AUX0)
							(OBJ0 ADJUNCT0 SUBJ0 ADV VERB0 AUX0)
							(ADJUNCT0 SUBJ0 ADV VERB0 OBJ0 AUX0)
							(OBJ1-0 ADJUNCT0 SUBJ0 VERB0 OBJ2-0 AUX0)
							(OBJ2-0 ADJUNCT0 SUBJ0 VERB0 OBJ1-0 AUX0)
							(ADJUNCT0 SUBJ0 VERB0 OBJ1-0 OBJ2-0 AUX0)
							(OBJ1-0 ADJUNCT0 SUBJ0 ADV VERB0 OBJ2-0 AUX0)
							(OBJ2-0 ADJUNCT0 SUBJ0 ADV VERB0 OBJ1-0 AUX0)
							(ADJUNCT0 SUBJ0 ADV VERB0 OBJ1-0 OBJ2-0 AUX0)
							(OBJ1-0 SUBJ0 ADV OBJ2-0 VERB0)
							(OBJ2-0 SUBJ0 ADV OBJ1-0 VERB0)
							(OBJ0 SUBJ0 ADV VERB0 AUX0)
							(OBJ1-0 SUBJ0 VERB0 OBJ2-0 AUX0)
							(OBJ2-0 SUBJ0 VERB0 OBJ1-0 AUX0)
							(OBJ1-0 SUBJ0 ADV VERB0 OBJ2-0 AUX0)
							(OBJ2-0 SUBJ0 ADV VERB0 OBJ1-0 AUX0)
12	no	yes	y/n	y/n	yes	no	(VERB0 SUBJ0 ADV)
							(VERB0 SUBJ0 ADV)
							(VERB0 SUBJ0 OBJ0)
							(VERB0 SUBJ0 ADV OBJ0)
							(VERB0 SUBJ0 OBJ1-0 OBJ2-0)
							(VERB0 SUBJ0 ADV OBJ1-0 OBJ2-0)
							(AUX0 SUBJ0 VERB0)
							(AUX0 SUBJ0 VERB0 ADV)
							(AUX0 SUBJ0 ADV VERB0)
							(AUX0 SUBJ0 VERB0 OBJ0)
							(AUX0 SUBJ0 ADV VERB0 OBJ0)
							(AUX0 SUBJ0 VERB0 OBJ1-0 OBJ2-0)
							(AUX0 SUBJ0 ADV VERB0 OBJ1-0 OBJ2-0)
13	no	yes	y/n	yes	no	no	(ADJUNCT0 SUBJ0 VERB0 ADV OBJ0)
							(ADJUNCT0 SUBJ0 VERB0 ADV OBJ1-0 OBJ2-0)
14	no	yes	y/n	no	no	no	(ADJUNCT0 SUBJ0 ADV VERB0 OBJ0)
							(ADJUNCT0 SUBJ0 ADV VERB0 OBJ1-0 OBJ2-0)
							(SUBJ0 ADV VERB0 OBJ0)
							(SUBJ0 ADV VERB0 OBJ1-0 OBJ2-0)
15	no	yes	no	y/n	y/n	yes	(OBJ1-0 VERB0 ADJUNCT0 SUBJ0 ADV OBJ2-0)
							(OBJ2-0 VERB0 ADJUNCT0 SUBJ0 ADV OBJ1-0)
							(OBJ0 AUX0 ADJUNCT0 SUBJ0 ADV VERB0)
							(OBJ1-0 AUX0 ADJUNCT0 SUBJ0 VERB0 OBJ2-0)
							(OBJ2-0 AUX0 ADJUNCT0 SUBJ0 VERB0 OBJ1-0)
							(OBJ1-0 AUX0 ADJUNCT0 SUBJ0 ADV VERB0 OBJ2-0)
							(OBJ2-0 AUX0 ADJUNCT0 SUBJ0 ADV VERB0 OBJ1-0)
							(OBJ1-0 VERB0 SUBJ0 ADV OBJ2-0)
							(OBJ2-0 VERB0 SUBJ0 ADV OBJ1-0)
							(OBJ0 AUX0 SUBJ0 ADV VERB0)
							(OBJ1-0 AUX0 SUBJ0 VERB0 OBJ2-0)
							(OBJ2-0 AUX0 SUBJ0 VERB0 OBJ1-0)
							(OBJ1-0 AUX0 SUBJ0 ADV VERB0 OBJ2-0)
							(OBJ2-0 AUX0 SUBJ0 ADV VERB0 OBJ1-0)

16	no	no	yes	y/n	y/n	yes	(OBJ1-0 ADJUNCT0 SUBJ0 OBJ2-0 ADV VERB0) (OBJ2-0 ADJUNCT0 SUBJ0 OBJ1-0 ADV VERB0) (OBJ0 ADJUNCT0 SUBJ0 VERB0 ADV AUX0) (OBJ1-0 ADJUNCT0 SUBJ0 OBJ2-0 VERB0 AUX0) (OBJ2-0 ADJUNCT0 SUBJ0 OBJ1-0 VERB0 AUX0) (OBJ1-0 ADJUNCT0 SUBJ0 OBJ2-0 VERB0 ADV AUX0) (OBJ2-0 ADJUNCT0 SUBJ0 OBJ1-0 VERB0 ADV AUX0) (OBJ1-0 SUBJ0 OBJ2-0 ADV VERB0) (OBJ2-0 SUBJ0 OBJ1-0 ADV VERB0) (OBJ0 SUBJ0 VERB0 ADV AUX0) (OBJ1-0 SUBJ0 OBJ2-0 VERB0 AUX0) (OBJ2-0 SUBJ0 OBJ1-0 VERB0 AUX0) (OBJ1-0 SUBJ0 OBJ2-0 VERB0 ADV AUX0) (OBJ2-0 SUBJ0 OBJ1-0 VERB0 ADV AUX0)
17	no	no	y/n	yes	no	no	NONE
18	no	no	y/n	y/n	yes	no	NONE
19	no	no	y/n	no	no	no	(ADJUNCT0 SUBJ0 OBJ0 VERB0 ADV) (ADJUNCT0 SUBJ0 OBJ2-0 OBJ1-0 VERB0 ADV) (SUBJ0 OBJ0 VERB0 ADV) (SUBJ0 OBJ2-0 OBJ1-0 VERB0 ADV)
20	no	no	no	y/n	y/n	yes	(ADJUNCT0 VERB0 SUBJ0 OBJ0 ADV) (ADJUNCT0 VERB0 SUBJ0 OBJ2-0 OBJ1-0) (OBJ1-0 VERB0 ADJUNCT0 SUBJ0 OBJ2-0 ADV) (OBJ2-0 VERB0 ADJUNCT0 SUBJ0 OBJ1-0 ADV) (ADJUNCT0 VERB0 SUBJ0 OBJ2-0 OBJ1-0 ADV) (ADJUNCT0 AUX0 SUBJ0 OBJ0 VERB0) (OBJ0 AUX0 ADJUNCT0 SUBJ0 VERB0 ADV) (ADJUNCT0 AUX0 SUBJ0 OBJ0 VERB0 ADV) (OBJ1-0 AUX0 ADJUNCT0 SUBJ0 OBJ2-0 VERB0) (OBJ2-0 AUX0 ADJUNCT0 SUBJ0 OBJ1-0 VERB0) (ADJUNCT0 AUX0 SUBJ0 OBJ2-0 OBJ1-0 VERB0) (OBJ1-0 AUX0 ADJUNCT0 SUBJ0 OBJ2-0 VERB0 ADV) (OBJ2-0 AUX0 ADJUNCT0 SUBJ0 OBJ1-0 VERB0 ADV) (ADJUNCT0 AUX0 SUBJ0 OBJ2-0 OBJ1-0 VERB0 ADV) (OBJ1-0 VERB0 SUBJ0 OBJ2-0 ADV) (OBJ2-0 VERB0 SUBJ0 OBJ1-0 ADV) (OBJ0 AUX0 SUBJ0 VERB0 ADV) (OBJ1-0 AUX0 SUBJ0 OBJ2-0 VERB0) (OBJ2-0 AUX0 SUBJ0 OBJ1-0 VERB0) (OBJ1-0 AUX0 SUBJ0 OBJ2-0 VERB0 ADV) (OBJ2-0 AUX0 SUBJ0 OBJ1-0 VERB0 ADV)

For groups 2, 3, 17 and 18, however, there is a problem for the OPL learner. It turns out that the languages in groups 2 and 3 are all also weakly equivalent. So too, are the languages in groups 17 and 18. However, neither of these larger groupings constitutes a natural class. The focus on unique strings is clearly not the problem because, by definition of weak equivalence,

there is no evidence at all to tell apart the languages in these unnatural groupings. Syntactically, the reason that these groupings arise is that for certain configurations of the spec-final and comp-final parameters, there is no possible way for any element to intervene between the A and T positions. For example in the base order shown in (6.4.2), nothing intervenes between A and T in the linear order.

(6.4.2)



In these cases if the language is –V2, it will be impossible to generate strings that would indicate whether verbal material had moved to A or, instead, to T. On the other hand, in these same cases, the adverbs on the edge of the VP provide a way detecting whether the verb has, in fact, left the VP. Therefore, languages, of this type, that are +V-fronting and +V-to-Agr, +V-fronting and –V-to-Agr, or –V-fronting and +V-to-Agr will all be weakly equivalent. However, they will be distinct from the corresponding –V-fronting and –V-to-Agr grammar. Such a non-natural grouping of 3 weak equivalents does not allow for any sort of final arbitrary setting of parameters because one of the languages in the remaining subspace is not capable of generating

the target language. The OLFO learner, in principle and independent of any details of the parser, will be able to narrow things down to this point, but no further.

If further development of the space does not eliminate such non-natural groupings of weak equivalents, then the OLFO might be forced to supplement its learning routine by doing something undesirable like what Dresher's cue-based algorithm for the metrical phonological system did at the end of several learning paths—instantiate and test all the remaining possibilities. Alternatively, the learner might perform some sort of error-driven random walk through the subspace. As is clear from previous discussion, neither of these possibilities is attractive if the non-natural groupings of weak equivalents encompass many languages. Extending the range of possible forms in the grammars in the space might eliminate this problem.

Gibson (p.c.) has made an observation that might lead to a device for appropriately breaking the ties in the particular cases seen in this extended syntactic system. The class of weakly equivalent grammars in the residual set are of the form {++, +-, -+}, while the non-weakly equivalent grammar in the residual set is of the form --. Each parse of a sentence from one of the weakly equivalent grammars will lead to two parses supporting the positive value of each parameter and one parse supporting the negative value of each parameter. A suitably modified learner that entered into a tie-breaking mode when it had gone for a considerable length of time without setting any further parameters could interpret this as evidence for setting both parameters to their positive values. This is a acceptable move to make in these cases. Clearly, this proposal could encounter difficulties in more complex residual subspaces.

6.5 Providing a Superparser for the Space

For the most part, this chapter has focused on the logical requirements of the OPL proposal and treated the exhaustive mode, supergrammar parser as a "black box". This section will, at least tentatively, attempt to look inside.

A leading idea behind Fodor's proposal is that it is possible to identify a parameter value with a particular piece or set of pieces of phrase structure. Fodor's presentation emphasizes cases where a single piece of phrase structure is identified with a parameter value, but this is clearly not necessary to her proposal. In fact, as I've argued, logically, it is equally acceptable for the parameter to be identified with a *set* of phrase structure components that come as an indivisible package; if a learner encounters a sentence that forces it to use one of the phrase structure components in a designated set, they are forced to adopt the entire set. The notion of setting a parameter to have one value or another implies that certain sets are designated competitors; if a learner adopts one of these competitor sets, they are prohibited from adopting any other.

Fodor (1995) does not provide any particular implementation but suggests that the ability to superparse with all of the phrase structure components available in UG will not be significantly more difficult than the ability to parse with an individual grammar, so as to make the proposal tractable. The intuition seems to be that the fact that a human parser for an individual language is capable of tractably analyzing primary linguistic data of the sort used to acquire language, which seems to be the case, then a superparser grammar will have no

difficulty operating in superparser mode. No explicit argument is given this effect, so I believe it is important to give a rational reconstruction of what such an argument might look like if the

One, logically correct, but obviously flawed way to fill in Fodor's idea would be to have completely separate sets of phrase structure components for each grammar in the parametric space, and to mark each rule in these sets with a full set of features corresponding to the parameter values for that language. (This is in addition to any features that might be needed otherwise.) These parametric features could act as the necessary "parsing chunks". The learner could then parse with this supergrammar, constraining parses to adopt consistent values of the parametric features. After parsing, the learner could inspect the parses in the usual way to see which parametric features were required of all parses. Clearly, however, this is intractable because the number of rules in the supergrammar is now exponential in the number of parameters in the space. If the smallest number of rules for a language is k, and there are n parameters, then the supergrammar will contain at least $k*(2^n)$ rules, while an individual grammar would only contain at most k rules.

If it were possible to identify each phrase structure rule in the system as either universally available, or as the reflex of a *single* parameter value, then the number of rules in the supergrammar system would be linearly related to the number of rules in an individual grammar. If there are m universally available rules and n binary parameters, and if any given parameter value had at least k_1 and at most k_2 corresponding rules, or "phrase structure chunks", then the number of rules in an individual grammar would be in the range of $m + n*k_1$ and $m + n*k_2$.. For the supergrammar which adopts both values of each binary parameter, the number of rules would be in the range of $2*(m + n*k_1)$ and $2*(m + n*k_2.)$.

Parsing with context-free phrase structure rules is known to be $O(G^{2*}n^3)$, where n is the length of the sentence to be parsed and G is a measure of the size of the grammar that can be identified with the number of rules (see, Barton, Berwick and Ristad (1987) and the references therein, particularly Earley (1968)). If the supergrammar contained only context-free phrase structure rules, then since G is only roughly twice as large for the supergrammar, there would be no differences between supergrammar and regular parsing with respect to this upper bound measure; the constant introduced by doubling the supergrammar does not affect the order of growth. For reasonably small G such that the "constants" of the human mind allowed parsing with an arbitrary context-free grammar to work simply virtue of context-freeness, then, the supergrammar scheme would also be expected to work, since it ony roughly doubles the size of the grammar.

However, it is highly unlikely that, if the grammar is represented in terms of some system of phrase structure rules, it is done so in terms of simple context-free rules. Typical implementations for grammars that aim towards application to real systems use more complicated machinery. For example, GPSG (Gazdar, Klein, Pullum and Sag (1985)) rules, provide, among other things, a mechanism for specifying features on categories in phrase structure rules and enforcing cross-categorial consistency between these features throughout a tree. This allows, for example, facts about subject/verb agreement to be captured without separate VP expansions for each case in a verbal paradigm. Although, they allow for more concise grammatical development, such systems can be shown to be weakly equivalent to systems that use only simple context-free rules. However, the proof of this equivalence depends on explicitly "blowing" up the GPSG system and providing phrase structure rules corresponding

to all possible combinations of the feature values for a given rule. As a result, the G contribution of such a system depends exponentially on the number of features in a grammar. Therefore, the fact that a parsing scheme is dependent only on G^2 is no consolation in a system with a large inventory of features.

One consequence of this line of argument is that parsing with even a single grammar from the class that does this type of feature checking is not likely to be tractable solely in virtue of its adherence to a formalism like GPSG. In order for a grammatical system that employs such a formalism to be successful, it must embody additional constraints that allow it to stay far below the upper bound results established for the grammatical framework as a whole (except, perhaps, for cases where human parsing breaks down). Therefore, in such a system, it is no longer possible to get an argument for the tractability of superparsing solely on the basis of grammar size and the assumption that parsing with an individual grammar is tractable. Moreover, the upper bound on grammar size that applies in supergrammar mode could be even worse if some constant number of additional features were required per parameter solely in order to get rules to interact appropriately for purposes of superparsing. This could happen if, as was necessary in our test system, some constant number of additional features were required per parameter solely to block incompatible parameter values, or otherwise get structural components to interact appropriately for the purposes of superparsing. This would mean that the number of features, which contributes to exponential growth in the equivalent context-free phrase structure grammar, would grow with the number of parameters in the space. This is clearly undesirable if an upper bound of $O(G^{2*}n^3)$ were meant to provide any assurances.

To see how such a situation might arise consider the set of phase structure rules that Bertolo and I have developed for the syntactic space described in Section 6.2. They work in

conjunction with an extended chart parser like the one that Gazdar and Mellish (1989) implement to allow feature unification, and they aim to satisfy the design requirements of having each rule express at most a single parameter value. Therefore, if a learner is forced to use a rule particular to a parameter in the course of parsing, it has evidence for fixing certain parameters, and thereby eliminating certain phrase structure rules from further consideration. The supergrammar, shown in the Appendix, employs a number of features that the parser, *via* the mechanism of feature checking, uses to keep track of what elements have moved from their base positions and enforce the appropriate cooccurrence restrictions (for example, if the verb has moved to C, it shouldn't also occur in its base position). These would be required within any given grammar that needed to enforce such restrictions. It ›, however, uses several features to enforce the consistency of parameter setting. For example, if a parse uses one phrase structure rule that requires comp-first, all rules in the parse must be consistent with comp-first. The alternative is to allow the learner using the superparser to build impossible parses that no speaker of a given human language would be able to construct. As noted above, Fodor is explicitly willing to entertain the possibility that this does in fact occur; the learner simply would not be able to learn when an impossible parse "competed against" a unique humanly possible parse. In practice, this seemed like an unworkable scheme to us. In the current space, as shown above, it is impossible to get things to work. The need to include these additional mechanisms introduces complexity into the parsing process that is motivated solely on the grounds of making the OPL approach workable.

I am currently viewing the construction of this parser as a sort of test of the concept of OPL learners and do not have a strong commitment to this being the correct way to go in

parsing.[47] It may well be possible to establish that feasibility of superparsing in exhaustive mode is a side effect of the feasibility of parsing with an individual grammar, but I do not believe that this has not been done yet. For the OPL approach to work, however, the "black box" of the parser must be developed in such a way that this entailment holds.

Of course, these possible worst case results for parsing complexity that I have considered do not have anything to say about whether, given the content of the constraints embedded in UG in a formalism such as the one discussed above, parsing or superparsing leads to intractability. Adopting the OPL hypothesis requires a concomitant adoption of the belief that for the particular linguistic systems encoded in UG, a logically sufficient set of cues can be tractably analyzed by a learner operating in superparser mode. Moreover, these cues must be readily available among the types of inputs that children receive. Whether this is true, remains to be seen.

[47] One particular aesthetic improvement this system could use might involve enriching the feature system and recasting the rules in X-bar format, so that there really could be a single parsing structure for all parameter values.

Chapter 7
Conclusion

Clearly, much remains to be discovered about the algorithms that human learners might use for natural language parameter setting, and the final answer to this question may ultimately need to await a more complete specification of the parametric systems that underlie human linguistic competence. This thesis has provided a step towards answering this question by indicating some general properties of parametric systems that will have an impact of the success of a variety of algorithms. Moreover, application of these algorithms to simplified parametric models provides concrete examples of the type of situations that might confront various proposals.

In Chapter 3, analysis of the cue-based algorithms for parameter setting pointed to their desirable property of deductive narrowing. A parametric cue-based learner will make a very limited number of hypothesis changes during the course of acquisition. Potential difficulties arise, however, if either the amount of information that the learner needs to encode in UG or the amount of computation that goes into a learning step depends linearly on the size of the parameter space, or otherwise place empirically unsupportable demands on the learner's resources. Nothing in the general outline of the cue-based approach guarantees that such problems will be avoided. Analysis of Dresher's (1994) system for acquisition of a variant of the Idsardi (1992) and Halle and Idsardi (1994) model of stress assignment indicated one possible way in which large parameter spaces could prove problematic. The final parameter setting steps in the system essentially require to initiate a search through the remaining

candidate grammars. In a small space, such as the one considered, it is hard to argue definitively against such a step. However, if the "residual" subspaces become larger as the cue-based approach is scaled up to larger systems, such exhaustive searches will not be possible.

In Chapter 4, the in-the-limit behavior of the Triggering Learning Algorithm was discussed. The version of the Halle and Idsardi metrical phonological system implemented there provided an interesting example of the type of interaction between parameters that could prove problematic for the TLA approach. Certain clusters of parameter values in the original system were capable of forcing forms to have particular properties even if one of the parameter values changed. In particular, there were candidate grammars in the space that invariantly required stress on one edge of the word. Since, for some of these grammars, all neighboring grammars also enforced the same requirement, the learner was unable of acquiring any target languages where the requirement did not hold. The elimination of one class of parametric options greatly simplified the set of local maxima in the system and allowed for a simple maturational solution.

Several reasonable strategies exist for accommodating local maxima in-the-limit without fundamentally changing the desirable cognitive parsimony of the TLA. This thesis has suggested that a key issue for the TLA is whether or not the learner, when exposed to input from a target language, is biased to move in the direction of a grammar capable of generating the target. Results from Gibson (1995) and the analyses performed in Chapter 5 for the TLA in the metrical phonological space developed in Chapter 4 provide initial examples of spaces where such biases exist.

Chapter 6 investigated the application of OPL learners that, like parametric cue-based learners, deductively narrow the set of possible parameter values with each hypothesis

change. For an extended syntactic space of 6 parameters, certain logical difficulties arose because of clusters of grammars that generated weakly equivalent languages. Whether or not such patterns will arise in larger parameteric spaces and whether or not the superparsing device that this approach requires will be tractable in such spaces remain open questions.

Appendix: Phrase Structure Rules from Chapter 6

The system uses a very simple set of atomic features. It could be made more concise by using complex features. Conceptually, the system passes two types of features down through the tree: (1) features about what movable elements occur above and (2) features designed to enforce parametric consistency in a parse.

XP categories simply pass information about verb and aux movement on to the X-bar categories that they immediately dominate. Similarly, X-bar categories pass simply pass information about NP and adjunct movement on to the XPs that they immediately dominate.

When an XP category receives feature information about NP and adjunct movement, it routes that information to spec-XP, which does two things: (1) uses these features to determine what can be realized in that spec-XP position and (2) passes down appropriate features, which reflect what NPs and adjuncts have been realized either above or in spec-XP, to X-bar.

X-bar categories, similarly, route features to their heads, which do two things: (1) uses these features to determine what can be realized in that head position, (2) passes down appropriate features, which reflect what verbs and auxiliaries have been realized in or above the head.

```
(setq rules
      '(
       ;; CP: Universal
       ((Rule (XP -> spec-X X-bar)
              ;; Category requirements
              (XP cat) = CP
              (spec-X cat) = spec-C
              (X-bar cat) = C-bar
              ;; Enforcing base-order consistency
              (XP spec-head) = initial
              (XP comp-head) = (X-bar comp-head)
              (XP spec-head) = (X-bar spec-head)
              ;; Message passing about XPs: In to spec
              ;;(XP obj-blocked) = (spec-X obj-blocked)
              ;;(XP obj1-blocked) = (spec-X obj1-blocked)
              ;;(XP obj2-blocked) = (spec-X obj2-blocked)
              ;;(XP subj-blocked) = (spec-X subj-blocked)
              ;; Message passing about XPs: Out of spec
              (spec-X block-adjunct) = (X-bar adjunct-blocked)
              (spec-X block-obj) = (X-bar obj-blocked)
              (spec-X block-obj1) = (X-bar obj1-blocked)
              (spec-X block-obj2) = (X-bar obj2-blocked)
              (spec-X block-subj) = (X-bar subj-blocked)
              (spec-X V2) = (X-bar V2)
              ;; Message passing about X0s: Pass through
              (XP higher-aux) = (X-bar higher-aux)
              (XP higher-verb) = (X-bar higher-verb)
             )
       1)
```

```
;; TP: spec-initial
((Rule (XP -> spec-X X-bar)
       (XP adjunct-blocked) = 0
       ;; Category requirements
       (XP cat) = TP
       (spec-X cat) = spec-T
       (X-bar cat) = T-bar
       ;; Enforcing base-order consistency
       (XP spec-head) = initial
       (XP comp-head) = (X-bar comp-head)
       (XP spec-head) = (X-bar spec-head)
       ;; Message passing about XPs: In to spec
       (XP obj-blocked) = (spec-X obj-blocked)
       (XP obj1-blocked) = (spec-X obj1-blocked)
       (XP obj2-blocked) = (spec-X obj2-blocked)
       (XP subj-blocked) = (spec-X subj-blocked)
       ;; Message passing about XPs: Out of spec
       (spec-X block-obj) = (X-bar obj-blocked)
       (spec-X block-obj1) = (X-bar obj1-blocked)
       (spec-X block-obj2) = (X-bar obj2-blocked)
       (spec-X block-subj) = (X-bar subj-blocked)
       ;; Message passing about X0s: Pass through
       (XP higher-aux) = (X-bar higher-aux)
       (XP higher-verb) = (X-bar higher-verb)
       )
 2)

;; TP: spec-final
((Rule (XP -> X-bar spec-X)
       (XP adjunct-blocked) = 0
       ;; Category requirements
       (XP cat) = TP
       (spec-X cat) = spec-T
       (X-bar cat) = T-bar
       ;; Enforcing base-order consistency
       (XP spec-head) = final
       (XP comp-head) = (X-bar comp-head)
       (XP spec-head) = (X-bar spec-head)
       ;; Message passing about XPs: In to spec
       (XP obj-blocked) = (spec-X obj-blocked)
       (XP obj1-blocked) = (spec-X obj1-blocked)
       (XP obj2-blocked) = (spec-X obj2-blocked)
       (XP subj-blocked) = (spec-X subj-blocked)
       ;; Message passing about XPs: Out of spec
       (spec-X block-obj) = (X-bar obj-blocked)
       (spec-X block-obj1) = (X-bar obj1-blocked)
       (spec-X block-obj2) = (X-bar obj2-blocked)
       (spec-X block-subj) = (X-bar subj-blocked)
       ;; Message passing about X0s: Pass through
       (XP higher-aux) = (X-bar higher-aux)
       (XP higher-verb) = (X-bar higher-verb)
       )
 3)
```

```
;; AP: spec-initial
((Rule (XP -> spec-X X-bar)
       ;; Category requirements
       (XP cat) = AP
       (spec-X cat) = spec-A
       (X-bar cat) = A-bar
       ;; Enforcing base-order consistency
       (XP spec-head) = initial
       (XP comp-head) = (X-bar comp-head)
       (XP spec-head) = (X-bar spec-head)
       ;; Message passing about XPs: In to spec
       (XP obj-blocked) = (spec-X obj-blocked)
       (XP obj1-blocked) = (spec-X obj1-blocked)
       (XP obj2-blocked) = (spec-X obj2-blocked)
       (XP subj-blocked) = (spec-X subj-blocked)
       ;; Message passing about XPs: Out of spec
       (spec-X block-obj) = (X-bar obj-blocked)
       (spec-X block-obj1) = (X-bar obj1-blocked)
       (spec-X block-obj2) = (X-bar obj2-blocked)
       (spec-X block-subj) = (X-bar subj-blocked)
       ;; Message passing about X0s: Pass through
       (XP higher-aux) = (X-bar higher-aux)
       (XP higher-verb) = (X-bar higher-verb)
       )
 4)

;; AP: spec-final
((Rule (XP -> X-bar spec-X)
       ;; Category requirements
       (XP cat) = AP
       (spec-X cat) = spec-A
       (X-bar cat) = A-bar
       ;; Enforcing base-order consistency
       (XP spec-head) = final
       (XP comp-head) = (X-bar comp-head)
       (XP spec-head) = (X-bar spec-head)
       ;; Message passing about XPs: In to spec
       (XP obj-blocked) = (spec-X obj-blocked)
       (XP obj1-blocked) = (spec-X obj1-blocked)
       (XP obj2-blocked) = (spec-X obj2-blocked)
       (XP subj-blocked) = (spec-X subj-blocked)
       ;; Message passing about XPs: Out of spec
       (spec-X block-obj) = (X-bar obj-blocked)
       (spec-X block-obj1) = (X-bar obj1-blocked)
       (spec-X block-obj2) = (X-bar obj2-blocked)
       (spec-X block-subj) = (X-bar subj-blocked)
       ;; Message passing about X0s: Pass through
       (XP higher-aux) = (X-bar higher-aux)
       (XP higher-verb) = (X-bar higher-verb)
       )
 5)

;; spec-C: +V2 adjunct
```

```
((Rule (spec -> occupant)
       ;; Category requirements
       (spec cat) = spec-C
       (occupant cat) = adjunct
       ;; Messages sent down the tree
       (spec V2) = 1
       (spec block-adjunct) = 1
       (spec block-obj) = 0
       (spec block-obj1) = 0
       (spec block-obj2) = 0
       (spec block-subj) = 0
       )
 6)

;; spec-C: +V2 obj
((Rule (spec -> occupant)
       ;; Category requirements
       (spec cat) = spec-C
       (occupant cat) = obj
       ;; Messages sent down the tree
       (spec V2) = 1
       (spec block-adjunct) = 0
       (spec block-obj) = 1
       (spec block-obj1) = 0
       (spec block-obj2) = 0
       (spec block-subj) = 0
       )
 7)

;; spec-C: +V2 obj1
((Rule (spec -> occupant)
       ;; Category requirements
       (spec cat) = spec-C
       (occupant cat) = obj1
       ;; Messages sent down the tree
       (spec V2) = 1
       (spec block-adjunct) = 0
       (spec block-obj) = 0
       (spec block-obj1) = 1
       (spec block-obj2) = 0
       (spec block-subj) = 0
       )
 8)

;; spec-C: +V2 obj2
((Rule (spec -> occupant)
       ;; Category requirements
       (spec cat) = spec-C
       (occupant cat) = obj2
       ;; Messages sent down the tree
       (spec V2) = 1
       (spec block-adjunct) = 0
       (spec block-obj) = 0
       (spec block-obj1) = 0
```

```
      (spec block-obj2) = 1
      (spec block-subj) = 0
      )
 9)

;; spec-C: +V2 subj1
((Rule (spec -> occupant)
      ;; Category requirements
      (spec cat) = spec-C
      (occupant cat) = subj
      ;; Messages sent down the tree
      (spec V2) = 1
      (spec block-adjunct) = 0
      (spec block-obj) = 0
      (spec block-obj1) = 0
      (spec block-obj2) = 0
      (spec block-subj) = 1
      )
 10)

;; spec-C: -V2
((Rule (spec ->)
            ;; Category requirements
      (spec cat) = spec-C
      ;; Messages sent down the tree
      (spec V2) = 0
      (spec block-adjunct) = 0
      (spec block-obj) = 0
      (spec block-obj1) = 0
      (spec block-obj2) = 0
      (spec block-subj) = 0
      )
 11)

;; spec-T: Universal
((Rule (spec ->)
      ;; Category requirements
      (spec cat) = spec-T
      ;; Messages passed along
      (spec obj-blocked) = (spec block-obj)
      (spec obj1-blocked) = (spec block-obj1)
      (spec obj2-blocked) = (spec block-obj2)
      (spec subj-blocked) = (spec block-subj)
      )
 12)

;; spec-A: -V2
((Rule (spec -> occupant)
      ;; Category requirements
      (spec cat) = spec-A
      (occupant cat) = subj
      ;; Messages received
      (spec subj-blocked) = 0
      ;; Messages sent down the tree
```

```
      (spec block-subj) = 1
      ;; Messages passed along
      (spec obj-blocked) = (spec block-obj)
      (spec obj1-blocked) = (spec block-obj1)
      (spec obj2-blocked) = (spec block-obj2)
      )
 13)

;; spec-A: +V2
((Rule (spec ->)
      ;; Category requirements
      (spec cat) = spec-A
      ;; Messages received
      (spec subj-blocked) = 1
      ;; Messages sent down the tree
      (spec block-subj) = 1
      ;; Messages passed-along
      (spec obj-blocked) = (spec block-obj)
      (spec obj1-blocked) = (spec block-obj1)
      (spec obj2-blocked) = (spec block-obj2)
      )
 14)

;; C-bar -> C TP: C-first
((Rule (X-bar -> X YP)
      (X-bar V2) = (X V2)
      ;; Category requirements
      (X-bar cat) = C-bar
      (X cat) = C
      (YP cat) = TP
      ;; Message passing about XPs: Pass through
      (X-bar adjunct-blocked) = (YP adjunct-blocked)
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
      )
 15)

;; C-bar -> C TP: C-final
((Rule (X-bar -> YP X)
      (X-bar V2) = (X V2)
      ;; Category requirements
      (X-bar cat) = C-bar
      (X cat) = C
      (YP cat) = TP
      ;; Message passing about XPs: Pass through
      (X-bar adjunct-blocked) = (YP adjunct-blocked)
```

```
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
      )
 16)

;; C-bar -> C +TP: C-first
((Rule (X-bar -> X YP)
      (X-bar adjunct-blocked) = (YP adjunct-blocked)
      (X-bar V2) = (X V2)
      ;; Category requirements
      (X-bar cat) = C-bar
      (X cat) = C
      (YP cat) = +TP
      ;; Message passing about XPs: Pass through
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
      )
 17)

;; C-bar -> C +TP: C-final
((Rule (X-bar -> YP X)
      (X-bar adjunct-blocked) = (YP adjunct-blocked)
      (X-bar V2) = (X V2)
      ;; Category requirements
      (X-bar cat) = C-bar
      (X cat) = C
      (YP cat) = +TP
      ;; Message passing about XPs: Pass through
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
```

```
      )
 18)

;; +TP: Universal
((Rule (+TP -> AdjunctP TP)
      (+TP adjunct-blocked) = (AdjunctP adjunct-blocked)
      ;; Category requirements
      (+TP cat) = +TP
      (AdjunctP cat) = AdjunctP
      (TP cat) = TP
      ;; Enforcing base-order consistency
      (+TP comp-head) = (TP comp-head)
      (+TP spec-head) = (TP spec-head)
      ;; Message passing about XPs: Pass through
      (+TP obj-blocked) = (TP obj-blocked)
      (+TP obj1-blocked) = (TP obj1-blocked)
      (+TP obj2-blocked) = (TP obj2-blocked)
      (+TP subj-blocked) = (TP subj-blocked)
      ;; Message passing about X0s: In to X
      (+TP higher-aux) = (TP higher-aux)
      (+TP higher-verb) = (TP higher-verb)
      )
 19)

;; AdjunctP -> adjunct: Universal
((Rule (AdjunctP -> adjunct)
      (AdjunctP cat) = AdjunctP
      (adjunct cat) = adjunct
      (AdjunctP adjunct-blocked) = 0
      )
 20)

;; AdjunctP -> : Universal
((Rule (AdjunctP ->)
      (AdjunctP cat) = AdjunctP
      (AdjunctP adjunct-blocked) = 1
      )
 21)

;; T-bar: head-initial
((Rule (X-bar -> X YP)
      ;; Category requirements
      (X-bar cat) = T-bar
      (X cat) = T
      (YP cat) = AP
      ;; Enforcing base-order consistency
      (X-bar comp-head) = initial
      (X-bar comp-head) = (YP comp-head)
      (X-bar spec-head) = (YP spec-head)
      ;; Message passing about XPs: Pass through
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
```

```
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
      )
 22)

;; T-bar: head-final
((Rule (X-bar -> YP X)
      ;; Category requirements
      (X-bar cat) = T-bar
      (X cat) = T
      (YP cat) = AP
      ;; Enforcing base-order consistency
      (X-bar comp-head) = final
      (X-bar comp-head) = (YP comp-head)
      (X-bar spec-head) = (YP spec-head)
      ;; Message passing about XPs: Pass through
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
      )
 23)

;; A-bar: head-initial
((Rule (X-bar -> X YP)
      ;; Category requirements
      (X-bar cat) = A-bar
      (X cat) = A
      (YP cat) = VP
      ;; Enforcing base-order consistency
      (X-bar comp-head) = initial
      (X-bar comp-head) = (YP comp-head)
      (X-bar spec-head) = (YP spec-head)
      ;; Message passing about XPs: Pass through
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
```

```
      )
 24)

;; A-bar: head-final
((Rule (X-bar -> YP X)
      ;; Category requirements
      (X-bar cat) = A-bar
      (X cat) = A
      (YP cat) = VP
      ;; Enforcing base-order consistency
      (X-bar comp-head) = final
      (X-bar comp-head) = (YP comp-head)
      (X-bar spec-head) = (YP spec-head)
      ;; Message passing about XPs: Pass through
      (X-bar obj-blocked) = (YP obj-blocked)
      (X-bar obj1-blocked) = (YP obj1-blocked)
      (X-bar obj2-blocked) = (YP obj2-blocked)
      (X-bar subj-blocked) = (YP subj-blocked)
      ;; Message passing about X0s: In to X
      (X-bar higher-aux) = (X higher-aux)
      (X-bar higher-verb) = (X higher-verb)
      ;; Message passing about X0s: Out of X
      (X indicates-higher-aux) = (YP higher-aux)
      (X indicates-higher-verb) = (YP higher-verb)
      )
 25)

;; C -> aux; +V2
;;                    If a +V2 language has an aux in C,
;;                    no aux can appear below.
;;                    Moreover, the verb must be in V.
((Rule (head -> occupant)
       ;; Category requirements
      (head cat) = C
      (occupant cat) = aux
       ;; Messages required to license
      (head V2) = 1
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 0
      (head indicates-higher-aux) = 1
      )
 26)

;; C -> verb; +V2
;;                    If a +V2 language has a verb in C,
;;                    no aux or verb can appear below.
((Rule (head -> occupant)
      ;; Category requirements
      (head cat) = C
      (occupant cat) = verb
      ;; Messages required to license
      (head V2) = 1
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 1
```

```
      (head indicates-higher-aux) = 0
      .

 27)

;; C -> ; -V2.
;;                    A -V2 language has an empty C.
((Rule (head ->)
      ;; Category requirements
      (head cat) = C
      ;; Messages required to license
      (head V2) = 0
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 0
      (head indicates-higher-aux) = 0
      )
 28)

;; T -> aux; +V-front.
;;                    If neither an aux nor a verb appear
;;                    higher in the tree, then T could
;;                    potentially be realized as an aux.
;;                    for +V-front languages. If so, no
;;                    aux can appear in A.
;;                    Moreover, a verb must appear in V.
;;
((Rule (head -> occupant)
      ;; Category requirements
      (head cat) = T
      (occupant cat) = aux
      ;; Messages required to license
      (head higher-verb) = 0
      (head higher-aux) = 0
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 0
      (head indicates-higher-aux) = 1
      )
 29)

;; T -> verb; +V-front.
;;                    If neither an aux nor a verb appear
;;                    higher in the tree, then T could
;;                    potentially be realized as a verb
;;                    for +V-front languages. If so, no
;;                    verb can appear in A, or V.
;;                    Moreover, no aux can appear in A.
;;
((Rule (head -> occupant)
      ;; Category requirements
      (head cat) = T
      (occupant cat) = verb
      ;; Messages required to license
      (head higher-verb) = 0
      (head higher-aux) = 0
```

```
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 1
      (head indicates-higher-aux) = 0
      )
 30)


;; T -> ; Universal.  If a verb appears higher in the tree
;;                    there is no possible movement to T.
;;                    No verb can appear below in V.
((Rule (head -> )
      ;; Category requirements
      (head cat) = T
      ;; Messages required to license
      (head higher-verb) = 1
      (head higher-aux) = 0 ;; redundant
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 1
      (head indicates-higher-aux) = 0
      )
 31)

;; T -> ; Universal.  If an aux appears higher in the tree
;;                    there is no possible movement to T.
;;                    A verb must appear below in V, an
;;                    aux can not appear below in A.
((Rule (head -> )
      ;; Category requirements
      (head cat) = T
      ;; Messages required to license
      (head higher-verb) = 0 ;; redundant
      (head higher-aux) = 1
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 0
      (head indicates-higher-aux)  = 1
      )
 32)

;; T -> ; -V-front.   This will only get incorporated into
;;                    a grammar when the parse does not
;;                    allow the other two universal
;;                    alternatives.
;;                    Therefore it will only get used by
;;                    languages that are -V2.
;;                    Languages that are +V2 will always
;;                    be able to use one of the
;;                    alternatives.
;;                    If the string contains an aux, it
;;                    still has a chance of being realized
;;                    below in A.
((Rule (head -> )
      ;; Category requirements
      (head cat) = T
      ;; Messages sent down the tree
```

```
      (head indicates-higher-verb) = 0
      (head indicates-higher-aux) = 0
      )
 33)


;; A -> aux; Universal
;;                    If neither an aux nor a verb appear
;;                    higher in the tree, then A could
;;                    potentially be realized as an aux.
;;                    If so, a verb must appear in V.
;;                    This rule can never apply in
;;                    languages that are either +V2 or
;;                    +V-front because there will be
;;                    either a verb or an aux higher in
;;                    the tree.
((Rule (head -> occupant)
      ;; Category requirements
      (head cat) = A
      (occupant cat) = aux
      ;; Messages required to license
      (head higher-verb) = 0
      (head higher-aux) = 0
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 0
      )
 34)

;; A -> verb; +V-to-Agr.
;;                    If neither an aux nor a verb appear
;;                    higher in the tree, then A could
;;                    potentially be realized as a verb
;;                    for +V-to-Agr languages. If so, no
;;                    verb can appear in V.
;;
((Rule (head -> occupant)
      ;; Category requirements
      (head cat) = A
      (occupant cat) = verb
      ;; Messages required to license
      (head higher-verb) = 0
      (head higher-aux) = 0
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 1
      )
 35)


;; A -> ; Universal.  If a verb appears higher in the tree
;;                    there is no possible movement to A.
;;                    No verb can appear below in V.
((Rule (head -> )
      ;; Category requirements
      (head cat) = A
```

```
      ;; Messages required to license
      (head higher-verb) = 1
      (head higher-aux) = 0 ;; redundant
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 1
      )
 36)

;; A -> ; Universal.  If an aux appears higher in the tree
;;                    there is no possible movement to A.
;;                    A verb must appear below in V.
((Rule (head -> )
      ;; Category requirements
      (head cat) = A
      ;; Messages required to license
      (head higher-verb) = 0 ;; redundant
      (head higher-aux) = 1
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 0
      )
 37)

;; A -> ; -V-to-Agr.  This will only get incorporated into
;;                    a grammar when the parse does not
;;                    allow the other two alternatives.
;;                    Therefore it will only get used by
;;                    languages that are -V2 and -V-front.
;;                    Languages that are +V2 or +V-front
;;                    will always be able to use one of
;;                    the alternatives.
;;                    In the case where the sentence
;;                    contains an aux for -V2 and -V-front
;;                    languages, this is the only possible
;;                    slot for an aux, so the rule will be
;;                    blocked by the need to incorporate
;;                    all of the lexical material from a
;;                    string.
((Rule (head -> )
      ;; Category requirements
      (head cat) = A
      ;; Messages sent down the tree
      (head indicates-higher-verb) = 0
      )
 38)


;; VPs: All are either universal of comp-final/comp-
;; initial as indicated.
((Rule (VP -> V)
      (VP cat) = VP
      (V cat) = V
      (VP higher-verb) = (V higher-verb)
      (VP obj-blocked) = 0
      (VP obj1-blocked) = 0
```

```
      (VP obj2-blocked) = 0
      )
 39)

((Rule (VP -> V adv)
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (VP higher-verb) = (V higher-verb)
      (VP obj-blocked) = 0
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      )
 40)

((Rule (VP -> adv V)
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (VP higher-verb) = (V higher-verb)
      (VP obj-blocked) = 0
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      )
 41)

;; VP: head-initial
((Rule (VP -> V NP)
      (VP comp-head) = initial
      (VP cat) = VP
      (V cat) = V
      (NP cat) = NP
      (VP higher-verb) = (V higher-verb)
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      (VP obj-blocked) = (NP obj-blocked)
      )
 42)

;; VP: head-final
((Rule (VP -> NP V)
      (VP comp-head) = final
      (VP cat) = VP
      (V cat) = V
      (NP cat) = NP
      (VP higher-verb) = (V higher-verb)
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      (VP obj-blocked) = (NP obj-blocked)
      )
 43)

;; VP: head-initial
((Rule (VP -> V NP adv)
```

```
      (VP comp-head) = initial
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (NP cat) = NP
      (VP higher-verb) = (V higher-verb)
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      (VP obj-blocked) = (NP obj-blocked)
      )
 44)

;; VP: head-final
((Rule (VP -> NP V adv)
      (VP comp-head) = final
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (NP cat) = NP
      (VP higher-verb) = (V higher-verb)
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      (VP obj-blocked) = (NP obj-blocked)
      )
 45)

;; VP: head-initial
((Rule (VP -> adv V NP)
      (VP comp-head) = initial
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (NP cat) = NP
      (VP higher-verb) = (V higher-verb)
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      (VP obj-blocked) = (NP obj-blocked)
      )
 46)

;; VP: head-final
((Rule (VP -> adv NP V)
      (VP comp-head) = final
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (NP cat) = NP
      (VP higher-verb) = (V higher-verb)
      (VP obj1-blocked) = 0
      (VP obj2-blocked) = 0
      (VP obj-blocked) = (NP obj-blocked)
      )
 47)
```

```
;; VP: head-inital
((Rule (VP -> V NP1 NP2)
       (VP comp-head) = initial
       (VP cat) = VP
       (V cat) = V
       (NP1 cat) = NP1
       (NP2 cat) = NP2
       (VP higher-verb) = (V higher-verb)
       (VP obj-blocked) = 0
       (VP obj1-blocked) = (NP1 obj1-blocked)
       (VP obj2-blocked) = (NP2 obj2-blocked)
       )
 48)

;; VP: head-final
((Rule (VP -> NP2 NP1 V)
       (VP comp-head) = final
       (VP cat) = VP
       (V cat) = V
       (NP1 cat) = NP1
       (NP2 cat) = NP2
       (VP higher-verb) = (V higher-verb)
       (VP obj-blocked) = 0
       (VP obj1-blocked) = (NP1 obj1-blocked)
       (VP obj2-blocked) = (NP2 obj2-blocked)
       )
 49)

;; VP: head-initial
((Rule (VP -> V NP1 NP2 adv)
       (VP comp-head) = initial
       (VP cat) = VP
       (V cat) = V
       (adv cat) = adv
       (NP1 cat) = NP1
       (NP2 cat) = NP2
       (VP higher-verb) = (V higher-verb)
       (VP obj-blocked) = 0
       (VP obj1-blocked) = (NP1 obj1-blocked)
       (VP obj2-blocked) = (NP2 obj2-blocked)
       )
 50)

;; VP: head-final
((Rule (VP -> NP2 NP1 V adv)
       (VP comp-head) = final
       (VP cat) = VP
       (V cat) = V
       (adv cat) = adv
       (NP1 cat) = NP1
       (NP2 cat) = NP2
       (VP higher-verb) = (V higher-verb)
       (VP obj-blocked) = 0
       (VP obj1-blocked) = (NP1 obj1-blocked)
```

```
      (VP obj2-blocked) = (NP2 obj2-blocked)
      )
 51)

;; VP: head-initial
((Rule (VP -> adv V NP1 NP2)
      (VP comp-head) = initial
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (NP1 cat) = NP1
      (NP2 cat) = NP2
      (VP higher-verb) = (V higher-verb)
      (VP obj-blocked) = 0
      (VP obj1-blocked) = (NP1 obj1-blocked)
      (VP obj2-blocked) = (NP2 obj2-blocked)
      )
 52)

;; VP: head-final
((Rule (VP -> adv NP2 NP1 V)
      (VP comp-head) = final
      (VP cat) = VP
      (V cat) = V
      (adv cat) = adv
      (NP1 cat) = NP1
      (NP2 cat) = NP2
      (VP higher-verb) = (V higher-verb)
      (VP obj-blocked) = 0
      (VP obj1-blocked) = (NP1 obj1-blocked)
      (VP obj2-blocked) = (NP2 obj2-blocked)
      )
 53)

((Rule (V ->)
      (V cat) = V
      (V higher-verb) = 1
      )
 54)

((Rule (V -> verb)
      (V cat) = V
      (verb cat) = verb
      (V higher-verb) = 0
     )
 55)

((Rule (NP ->)
      (NP cat) = NP
      (NP obj-blocked) = 1
      )
 56)

((Rule (NP -> obj)
```

```
            (NP cat) = NP
            (obj cat) = obj
            (NP obj-blocked) = 0
            )
     57)

     ((Rule (NP1 ->)
            (NP1 cat) = NP1
            (NP1 obj1-blocked) = 1
            )
     58)

     ((Rule (NP1 -> obj1)
            (NP1 cat) = NP1
            (obj1 cat) = obj1
            (NP1 obj1-blocked) = 0
            )
     59)

     ((Rule (NP2 ->)
            (NP2 cat) = NP2
            (NP2 obj2-blocked) = 1
           )
     60)

     ((Rule (NP2 -> obj2)
            (NP2 cat) = NP2
            (obj2 cat) = obj2
            (NP2 obj2-blocked) = 0
            )
     61)
    ))

(setq lexical_rules
      '((Word (wsubj)
            (cat) = subj)
     (Word (wobj)
            (cat) = obj)
     (Word (wobj1)
            (cat) = obj1)
     (Word (wobj2)
            (cat) = obj2)
     (Word (wverb)
            (cat) = verb)
     (Word (waux)
            (cat) = aux)
     (Word (wadv)
            (cat) = adv)
     (Word (wadjunct)
            (cat) = adjunct)
    ))
```

References

Angluin, D. 1978. Inductive inference of formal languages from positive data. *Information and Control* 45: 117–135.

Bach, E. 1962. The order of elements in a transformational grammar in German. *Language* 38:263–269.

Barton, G.E., Berwick, R. and E.S. Ristad. 1987. *Computational complexity and natural language.* Cambridge, MA: MIT Press.

Bertolo, S. 1995. Learnability properties of parametric models for natural language acquisition. Rutgers University Ph.D. dissertation.

Bertolo, S. 1995. Maturation and learnability in parametric systems. *Language Acquisition* 4:277–318

Berwick, R. 1985. *The acquisition of syntactic knowledge.* Cambridge, MA: MIT Press.

Berwick, R. and A. Weinberg. 1984. *The grammatical basis of linguistic performance.* Cambridge, MA: MIT Press.

den Besten, H. 1983. On the interaction of root transformations and lexical deletive rules. In *On the formal syntax of the Westgermania,* Papers from the 3rd Groningen Grammar Talks, Groningen, 1981, ed. Abraham, W., 47–131. Amsterdam: J. Benjamins.

Bierwisch, M. 1963. *Grammatik des deutschen Verbs.* Berlin: Akademie-Verlag.

Bobaljik, J. 1994. Verb-raising, adjacency and morphology. *MIT Working Papers in Linguistics: Morphology.*

Borer, H. and K. Wexler. 1987. The maturation of syntax. In *Parameter setting,* eds. T. Roeper and E. Williams. Dordrecht: Reidel.

Braine, 1971. On two models of internalization of grammars. In *The ontogenesis of grammar,* ed. D. Slobin. New York: Academic Press.

Broihier, K. 1995a. Phonological triggers. Paper presented at the University of Maryland Mayfest 1995: Formal Approaches to Learnability.

Broihier, K. 1995b. Phonological parameter setting with the Triggering Learning Algorithm. Paper presented at the University of British Columbia International Conference on Phonological Acquisition.

Brown, R. and C. Hanlon. 1970. Derivational complexity and the order of acquisition in child speech. In *Cognition and the development of language*, ed. J. Haynes. New York: Wiley.

Chomsky, N. 1957. *Syntactic structures*. The Hague: Mouton.

Chomsky, N. 1965. *Aspects of the theory of syntax*. Cambridge, MA: MIT Press.

Chomsky, N. 1981. *Lectures on government and binding*. Dordrecht: Foris.

Chomsky, N. 1992. *A minimalist program for linguistic theory*. Cambridge, MA: MITWPL.

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

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

Cormen, T., C. Leiserson and R. Rivest. 1990. *Introduction to algorithms*. Cambridge, MA: MIT Press.

Crain, S. 1991. Language acquisition in the absence of experience. *Behavioral and Brain Sciences* 14.

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

Dresher, B.E. 1994. Acquiring stress systems. In *Language Computations: DIMACS Workshop on Human Language, March 20–22, 1992*, ed. E.S. Ristad, 71–92.

Emonds, J. 1978. The verbal complex V'–V in French. *Linguistic Inquiry* 9.

Fikkert, P. 1994. *On the Acquisition of Prosodic Structure (HIL Dissertations, 6)*. Dordrecht: ICG.

Fodor, J. 1995. Fewer but better triggers. *CUNYForum*(19): 37-63.

Gazdar, G., E. Klein, G. Pullum, and I. Sag. 1985. *Generalized Phrase Structure Grammar*. Oxford, England: Basil Blackwell.

Gazdar, G. and C. Mellish. 1989. *Natural language processing in Lisp*. Wokingham: Addison-Wesley.

Gibson, Edward (1991). A computational theory of human linguistic processing: Memory limitations and processing breakdown. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.

Gibson. 1995. Triggers and Parameter Setting. Paper presented at the University of Maryland Mayfest 95: Formal Approaches to Learnability.

Gibson, E. and K. Wexler. 1994. Triggers. *Linguistic Inquiry* 25: 407–454.

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

Goldsmith, J. 1990. *Autosegmental and metrical phonology*. Oxford: Blackwell Publishers.

Grimshaw, J. 1993. Minimal projections, heads and optimality. mss., Rutgers University. In Press, *Linguistic Inquiry*.

Haider, H. and M. Prinzhorn, eds. 1986. *Verb second phenomena in Germanic languages*, Dordrecht: Foris.

Halle, M. and W. Idsardi. 1994. General properties of stress and metrical structure.In*Language Computations: DIMACS Workshop on Human Language, March 20–22, 1992*, ed. E.S. Ristad.

Hoekstra, T. 1984. *Transitivity*. Dordrecht: Foris.

Idsardi, W. 1992. *The computation of prosody*. MIT Ph.D dissertation.

Isaacson, D. and J. Madsen. *Markov chains: Theory and applications*. New York: John Wiley & Sons.

Joos, M. 1957. *Readings in linguistics*. Washington: American Council of Learned Societies.

Manzini, R. and K. Wexler. 1987. Parameters, binding theory and learnability. *Linguistic Inquiry* 18: 413–444.

Marcus, G. 1993. *Negative evidence in language acquisition*. Cognition 46: 53–85.

McNeill, D. 1966. Developmental psycholinguistics. In *The Genesis of Language*, eds. Smith, F. and G. Miller. Cambridge, MA: MIT Press.

Nyberg, E.H., 3rd. 1991. A limited non-deterministic parameter-setting model. In *Proceedings of NELS* 21, 309–322.

Nyogi, P. and R. Berwick. 1993. Formalizing triggers: a learning model for finite spaces. MIT AI Memo No. 1449.

Osherson, D., M. Stob and S. Weinstein. 1986. Systems that learn. Cambridge, MA: MIT Press.

Pesetsky, D. 1993. Principles of sentence pronunciations. Rutgers Optimality Archive–42.

Pollock, J-Y. 1989. Verb movement, UG and the structure of IP. *Linguistic Inquiry* 20: 365-424,

Prince, A. and P. Smolensky. 1995. *Optimality theory*. Cambridge, MA: MIT Press.

Thiersch, C. 1978. Topics in Germanic syntax. MIT Ph.D. dissertation.

Travis, L. 1984. *Parameters and effects of word order variation*. MIT Ph.D. dissertation.

Wexler, K. 1996. The development of inflection in a biologically based theory of language acquisition. In Toward a genetics of language, ed. Rice, M. L. Mahwah, NJ: Lawrence Erlbaum Assoc.

Wexler, K. and P. Culicover. 1980. *Formal principles of language acquisition*. Cambridge, MA: MIT Press.

Wexler, K. and H. Hamburger. 1973. On the insufficiency of surface data for the learning of transformational languages. In *Approaches to natural language*, ed. Hintikka, J. K., J. M. E. Moravcsik and P. Suppes, 167–179. Dordrecht: Reidel.

Wu, A. 1994. The spell-out parameters: a minimalist approach to syntax. UCLA Ph. D. dissertation.