5

The Structural Triggers Learner

W.G. SAKAS and J.D. FODOR

5.1 Introduction[1]

How much work does it take to acquire a human language? For most adults, the acquisition of a new language is a slow and effortful process. But what if one has the right learning equipment, as children evidently do? For first language learners most of the work is done in five or six years. Our research goal is to find out what goes on in those few years. To what extent does it involve the use of special-purpose computational

systems that adults lack? What do the learning routines do, that is so difficult for the human brain to simulate later in life?

We will argue here that very little need be done to acquire a language, over and above the normal processes of comprehension that are involved in all language use. At least, we will argue this for the acquisition of syntax, on the assumption that the syntactic component of a natural language grammar is largely innate and that learning consists exclusively in the setting of parameters. Similar conclusions could apply to any other parameterized domain, such as phonology (see Dresher and Kaye, 1990; Dresher, 1999). Acquisition of the lexicon is likely to be a different and more labor-intensive project. And semantic principles possibly demand no learning at all.

Thus, we assume here the principles and parameters framework for language description (Chomsky, 1981a; 1988; and elsewhere) and consider the process by which syntactic parameters are set. This process is commonly described as triggering, and it is contrasted with more traditional modes of learning such as hypothesis formation and testing. The latter seemed to be the only way in which a grammar could be acquired, back when linguistic theory defined grammars as rule systems differing considerably from one language to another (Chomsky, 1965). Hypothesis

formation clearly carries a very heavy workload, and appears in addition to be too unreliable a mode of learning to model the remarkably uniform achievements of human learners. By contrast, parameter setting is thought to be much less onerous and more uniform across individuals.

The work involved in parameter setting can be broken down into three phases:

I trigger recognition;

II parameter value adoption;

III any necessary error-correction or other relearning (possibly repeating I and II);

The amount of work that each subprocess requires can be measured in terms of the number of computational steps it takes, and/or the number of input sentences consumed before learning is complete. Note that there are both swings and roundabouts here: an obsessively cautious learner would be slow on I and II but would recoup time with respect to III, while a learner that takes chances may coast through I and II but have to put in a lot more work at stage III. Triggering, as originally conceived, was hailed as winning in every direction: it is thought to be fast, accurate, and virtually computation-free.

We will report here a sad conclusion that is a commonplace in com-

putational learning theory but is not widely known in linguistics and psychology: that the classical notion of triggering cannot be effectively implemented for natural languages. This is because trigger recognition is much more difficult than had been recognized. In the following sections we examine stochastic models of a kind recently proposed by Gibson and Wexler (1994) to take the place of classical triggering, and show that they fall far short in terms of efficiency, faring badly with respect to both I and III. We then outline a very different approach to parameter setting, which amounts to just a slight twist on normal sentence parsing and which copes with the difficulty of I without paying heavily on III.

5.2 Triggering

Definitions of *trigger* or *to trigger* are hard to find in the literature.[2] The general understanding appears to be roughly as follows. A trigger is a sentence (a word string) of the target language, perceived by the learner, which 'automatically' flips a parameter switch to the correct value. That triggering is 'automatic' or 'mechanical' is regarded as important but is rarely explicated. We suppose what is intended is that some easily accessible property of the input word sequence is detected by the learning mechanism and causes a change in the value of a parameter without there

being any intervening computation of the linguistic consequences or any evaluation of alternative moves. The property in question need not even have any contentful relation to the parameter it sets (cf. Atkinson, 1987). Imagine, for instance, an artificial language domain in which all and only verbs begin with /w/, and all and only sentences with null subjects are verb-initial. In that domain, a sensor that detects /w/ could reliably trigger the positive value of the null subject parameter.

Unfortunately, this supremely simple triggering mechanism for parameter setting is a workable possibility for artificial languages only. Natural languages are not built for it. The sentence properties correlated with syntactic parameter values in natural languages are often abstract structural properties, not immediately detectable in the word string.[3] Two factors in particular impede superficial triggering. We will call these (i) the depth-of-derivation problem, and (ii) the string-to-structure problem. We consider them in turn.

(i) The depth-of-derivation problem: The criterial property for establishing a parameter value may be obscured by later derivational operations. This does not arise for all parameters. For example, the fact that there is no overt subject in a sentence is normally apparent in the surface form.[4] (But see Exercise 5.5 at the end of this chapter.) However,

some parameters, such as the word order parameters, control underlying properties of sentences. For instance, the head-position parameter for VP determines whether the verb precedes or follows the object in the underlying structure, and this cannot be read off the surface word sequence because movement transformations may have rearranged the constituents.[5] (Underlying order information is preserved in the positions of traces, but these are phonologically empty categories which are not perceptible; see below.) Thus, the information needed to set the parameter is present in the derivation but difficult or impossible for a learner to access.

(ii) The string-to-structure problem: Even surface structure is not overtly registered in the terminal string in all its detail. As a result, even some surface facts about a derivation may be undetectable to a learner. Traces and other empty categories are inaudible, and so are structural nodes unless they have characteristic lexical markers. For instance, in a superficially subject-verb-object (SVO) sentence the verb might be *in situ* within VP, as in English, or moved up to an inflectional head, as in French, or moved up to the C position in a verb-second language such as German (the subject also being moved in the latter case). (See Gibson and Wexler, 1994; Holmberg and Platzack, 1991; and Haegeman, 1994,

or other textbook for details of these derivations.) From the SVO word string it is impossible for the learner to tell which of these structures is present. Other sentences in the language provide more specific cues to structure. For instance, it might be possible for a learner to locate the verb (as linguists do) by reference to fixed-locus constituents such as tense or negation or lexical complementizers (see Bertolo et al., 1997a). But the reasoning involved in these deductions is far from trivial.

These observations make it clear that trigger properties for setting natural language parameters are not always apparent in the linguistic stimuli that learners are exposed to. Hence, for natural languages there can be no simple routing of input sentences towards the right parameter switches by a bank of peripheral sensors or by any kind of simple computation-free sorting procedure. We may still speak of trigger sentences, and even very loosely of triggering; but it must be with the understanding that these are not as in the carefree classical model in which word strings trip parameter switches without any significant linguistic analysis having occurred. We may take a trigger $T_{v_i^m}$ for value v_i^m of parameter p_i to be a sentence which occurs in at least one language and is grammatical in any language only if the grammar for that language has p_i set to v_i^m. Better still, we may take $T_{v_i^m}$ to be the specific structural

property within sentences that the value v_i^m is responsible for licensing. Either way, an encounter with $T_{v_i^m}$ in the target language would constitute reliable evidence for v_i^m in the target grammar (ignoring here the possibility of ungrammatical input). Whether the learning device can recognize this evidence, whether it adopts v_i^m, and if so by what mechanism, are matters deliberately left open by this process-neutral definition. They are what remain to be determined.

All that can be retained from the classical instant-triggering model is the triviality of subprocess II above, i.e., grammar changes consequent on trigger recognition are computation-free. We can assume that once a trigger has been recognized, the relevant parameter value is established and it alters the set of sentences licensed by the learner's grammar without any monitoring of this change by the learning device. For stage II, then, the learning system need not be knowledgeable about the very intricate correlations that hold between languages and the grammars that license them.[6] By retaining the labor-free stage II from the switch-setting metaphor we thus benefit by its considerable simplification over more traditional hypothesis testing procedures, which require a detailed knowledge of the generative consequences of different grammars. However, no general conclusion about this can be secure until we

have established how the learner can recognize triggers, for it remains an open question whether the trigger recognition operations of stage I will demand a comparable amount of metalinguistic sophistication.

To summarize the conclusions of this section: Many natural language parameters control non-perceptible, often deep, structural properties of sentences. What makes a sentence a trigger for some parameter value is that it has the property in question. Trigger recognition is therefore far from trivial, and the metaphor of triggering as instant switch-flipping must be relinquished. The literature on learnability still commonly refers to triggering. This is a convenience which does no harm as long as it is clear that an input sentence could reliably flip a correct parameter switch only after it has undergone a significant amount of linguistic analysis.[7]

5.3 Using the parser to identify triggers

Gibson and Wexler (1994) had the idea of using the sentence parsing mechanism to recognize trigger sentences. This makes excellent sense, since the sentence processing device routinely computes relations between terminal strings and structural representations at all levels of derivation. During normal sentence comprehension, it takes as input a surface word string, and its job is to establish sufficient structure to permit seman-

tic interpretation of the sentence. Plausibly, this involves establishing empty categories and underlying grammatical relations of just the kind needed for natural language parameter setting. Note: the output of the syntactic parse might be a representation of the whole derivation, a *structural description* in the sense of Chomsky (1995); or it might be some more compact representation of significant derivational properties, such as an S-structure as defined by Government Binding theory, which contains movement chains. For present purposes the difference is not important. For concreteness, and greater parity between transformational and non-transformational models, we will assume here that the processor constructs a single-level parse-tree with movement chains, as is common in many current parsing models.

It is not unreasonable to suppose that the human parsing mechanism is innate (see Fodor, 1998b). An infant's parsing routines may perhaps be limited in processing capacity at first, but we assume that the mechanism is ready to operate as soon as it is supplied with a grammar to work with. It would seem, then, that the ideal learning strategy would be for the learner, on encounter with a novel input string, to parse it so that any deep trigger properties it contains will become visible, and then use

these properties to set the relevant parameters. However, this proposal for trigger recognition also faces some practical problems.

One is what we call *the parsing paradox*, first drawn attention to by Valian (1990). The sentence processing mechanism can parse (assign a complete structure to) only those sentences that are licensed by the grammar from which it is drawing its information about the language. Those sentences, however, do not demand that any learning take place. The sentences that should initiate learning are those which the learner's current grammar does *not* yet license. But these sentences the learner's parsing routines cannot parse. In short: the learner cannot parse the very sentences it should learn from.

A second problem, emphasized by Gibson and Wexler (1994) is the existence of parametric ambiguity. This would confound trigger recognition even if the parsing paradox were solved. A sentence is parametrically ambiguous if it is licensed by two or more distinct combinations of parameter values (specifically: values of *relevant* parameters).[8] For example, we have noted that an SVO string is structurally ambiguous; and it is also parametrically ambiguous. Each of the possible structures is licensed by a different set of parameter values. As Gibson and Wexler point out, SVO order can be licensed by the parameter value

for verb-second (V2) structure, with any values for the parameters that control the underlying order of subject, verb and object (in German the underlying order is SOV), or else by the parameter values for underlying subject-before-verb and verb-before-object order without the V2 value, as in English. By contrast, a VOS sentence is not parametrically ambiguous, at least with respect to these three parameters. It can be licensed only by the -V2 value, and the underlying verb-before-subject and verb-before-object values.

Let us consider a little further the fact that sentences that are parametrically ambiguous are commonly structurally ambiguous (perhaps necessarily so; see Fodor, 1998a), i.e., that the different grammars that license the same word string assign it different structural descriptions. As we've seen, the structure of an SVO string in German is very different from that of an SVO string in English (at least on standard analyses); in German, but not in English, the subject and verb are raised into the C projection. Because of this, the learner's parsing mechanism cannot determine what structure to assign to a parametrically ambiguous input string until it knows which grammar to apply to that string. However, for a novel sentence type the learner does not know which grammar is appropriate – that is precisely what it is trying to find out by parsing

the sentence. Caught in this vicious circle, a bold learner might try to guess which structure (which grammar) is correct, while a cautious learner would rather avoid setting any parameters at all when there is danger that the input is ambiguous. Current models of human language learning divide on just this point. Some ignore ambiguity and adopt any parameter settings that succeed, without concern for the fact that the success might turn out to be spurious and the settings incorrect. This is how Gibson and Wexler's model operates, as we explain in Sections 5.4–5.7. By contrast, the model we advocate in Sections 5.8–5.10 attempts always to be aware of ambiguity and to refrain from adopting parameter values unless it has unequivocal evidence for them.

Summary of this section: The sentence parsing mechanism must exist independently of learning, and it is expert at assigning abstract structure to word strings. Putting it to work to identify the triggers for parameter setting is thus an excellent plan. But it can succeed only if two problems can be solved: the problem of how to parse sentences that fall beyond the licensing capacity of the learner's current grammar; and the problem of how to determine the right grammar when an input string can be licensed by two different (combinations of) parameter settings.

5.4 The triggering learning algorithm

Gibson and Wexler's learning procedure is the *Triggering Learning Algorithm* (TLA). It neatly sidesteps the parsing paradox by turning to its own advantage the fact that it cannot parse the sentences from which it needs to learn. It uses this as a stimulus for experimenting with alternative grammars. On receiving an input string s it first tries to parse s with its current grammar G. If this succeeds, no learning is called for; though G may not be fully correct, the learner at least has no specific reason to believe that it is wrong. If the parsing attempt with G fails, the learner tries again with a modified grammar G' that it arrives at by resetting one parameter, chosen at random. (That only one parameter may be reset is the Single Value Constraint; see Section 5.6.) If the parsing attempt with G' also fails, G' is no improvement over G, so the TLA retains G (this follows from the Greediness Constraint, which permits adoption of a grammar only if it licenses the current input; see Section 5.5.1 for discussion). If G' does permit a successful parse of s, the TLA shifts from G to G'. Once again, the grammar that affords a parse is not necessarily correct for the target language, but it has at least the merit of being compatible with the current sentence. This is

a necessary condition, though far from a sufficient one, for being the target grammar.

Gibson and Wexler make no specific assumptions about the nature of the parsing mechanism; they require only that it be capable of delivering a report of success or failure to the learning algorithm. It is not assumed that the parser presents the learning device with a structural description of the sentence in which it could recognize the deep structural properties that can reveal correct parameter values. But importantly, although deep structural properties are not explicitly consulted, they are what drive the outcome of the parsing test. Ambiguity aside, the reason why an attempted parse succeeds is that the sentence does have the properties associated with the parameter values that are being tested. Thus the TLA is able to employ the parser to detect, implicitly, any relevant trigger properties at all, regardless of whether or not they are superficially evident.

The TLA would be maximally effective if there were a perfect correlation between the correctness of a grammar for the target language, and its success or failure in parsing input. In fact the relationship is only partial. If a grammar fails the parse test on some input it must be wrong, since there is at least one sentence of the target language that

it cannot license. But if a grammar passes the parse test on some sentence, it is not necessarily correct for the language.[9] The input string might be parametrically ambiguous, and the TLA might by chance have picked for the parse test a grammar which assigns the input a structure but not the structure it has in the target language. For instance, if the input were an SVO sentence from a -V2 target language, and a +V2 grammar were picked for testing, the parser would report success – even though the verb is analyzed as in C instead of in the VP, and the +V2 parameter value will overgenerate many non-target sentences. The TLA is oblivious to such dangers. It behaves as if there were no such thing as parametric ambiguity. It selects a new parameter value to try out, and adopts it if it succeeds in converting parsing failure to parsing success. Because it picks only one candidate, it never knows whether that is the only one that would succeed, or whether it is merely one among many. And because it selects the candidate at random (respecting the Single Value Constraint), it is a matter of chance whether the one it picks constitutes the correct resolution of an ambiguity. If it chooses wrong there is a penalty: it may unwittingly switch a parameter that was already set correctly, to an incorrect value.

The TLA's blissful ignorance about the danger of ambiguity is not

necessarily a drawback, however, even in face of the considerable parametric ambiguity that exists in natural languages. The TLA is designed as a nondeterministic system which routinely mis-sets parameters and then later sets them again (and perhaps again) until eventually all are correct. In the end, this might be more effective than fussing about which input sentences can be trusted and which cannot. However, a trial-and-error strategy works well only if the setting and resetting process is not too cumbersome or slow. In fact, as we will show below, this kind of learner may take a great many computational steps on average to change even one parameter value. Hence the cost of having to keep repeating the process is considerable. In short: the TLA's trigger recognition process I is imprecise, its parameter adoption process II is unaware of the imprecision and so inherits it, therefore there must be extensive error correction at stage III, but this is a laborious process.

Thus, the TLA solves one of the two problems we identified above, but in such a way that it is forced to give up on the other one. It solves the parsing paradox by testing out alternatives to the current grammar. But testing a grammar is work, and the learning system can only reasonably test one grammar at a time, and so it cannot in principle recognize parametric ambiguity. It could do so only if it were to run the parse test

repeatedly on the same input sentence with different grammars until it either found a second successful grammar or exhausted *all* the possible grammars without having found one. This is presumably not a realistic possibility. But without an exhaustive test, a learner cannot recognize parametric ambiguity, so it cannot defend itself against it. So inevitably it makes mistakes. We see, then, that there is a clear connection between doing the work of trigger recognition by means of the parse test, and the fact that parameter setting is non-deterministic. This relationship is interesting and somewhat unexpected. In the classical instant-triggering model it was assumed that all property detectors were on the job at all times and could function in parallel, so the limitation of having to test grammars one by one did not arise.[10] But there was a lot of wishful thinking about trigger recognition in that model, as we have observed. It is only by taking the process seriously, as the TLA does, that we can see how arduous it actually is, and how much of it must be sacrificed to keep the workload within plausible limits.

Summary of section: The TLA is a significant advance on the classical model because its parse test can detect effects of parameter settings which are not accessible to any superficial property detector. But it purchases this sensitivity at a high price. Though it can achieve trigger

recognition, it is then forced to close its eyes to trigger ambiguity. The TLA's parse test imposes only a necessary condition on correctness of a parameter value, not a sufficient condition. For the TLA to do more would tax its capacity beyond feasibility. But because it cannot do more, it makes errors on ambiguous triggers and so must engage in more work later on to correct those errors.

We turn now to the other significant aspect of the TLA's workload, which arises at Stage I but which interacts with the ambiguity problem as we will show. It is very arduous for the TLA to find a parameter that is worth resetting. The consequence is that the TLA must expend a great deal of effort, and consume a great deal of input, to set even one parameter. To set 20 or 30 is harder still. And every error that is made at stage II due to ambiguity exacerbates this problem because it requires more parameter setting in order to correct the error. In the next section we establish some formal machinery to document these claims.

5.5 Performance of a TLA-like algorithm

Gibson and Wexler demonstrated that the TLA converges on the target grammar under some conditions, though not under all. We will suppose here, since it is not our main concern, that convergence is guaranteed.

Even so, it is unclear that the TLA is a plausible model of the human language learning mechanism. This is because, as we now argue, it is very inefficient at extracting information from the input sentences it encounters. Correspondingly, it has to parse a great many input sentences, on average, before it finds the target grammar. Though its workload per input sentence is not excessive, the accumulated load across the enormous input sample is high.

Our strategy in this section will be to demonstrate the computational inefficiency of the TLA in two steps. For simplicity, we first calculate performance characteristics for an error-driven learning system which we will call TLA- ("TLA minus"). It obeys the Greediness Constraint and is like the TLA in all respects except that it does not obey the Single Value Constraint (SVC). We then estimate the effect on performance of converting this TLA- into the TLA proper, by imposing the SVC. This second stage of evaluation is necessarily less precise, since the effect of the SVC is heavily dependent on the character of the particular language domain, as noted by Berwick and Niyogi (1996). In Berwick and Niyogi's application of the TLA to the simple 8-language domain defined by Gibson and Wexler, the SVC increased the number of trials to convergence; but this doesn't rule out the possibility that there

are circumstances under which it improves performance.[11] However, we do not believe that the SVC substantially affects the results presented here concerning the exponential complexity of trigger recognition in the TLA-.

Niyogi and Berwick (1996) formalized the behavior of the TLA as a Markov process. This is elegant and of some interest (though it can be difficult to manage for very large language domains); see Appendix, and Niyogi and Berwick (1996) for details. For our goals, however, it is not ideal. We wish to establish a foundation for comparing the TLA with other models. We therefore develop a framework that articulates degrees of parametric ambiguity and parametric irrelevance, and that can distinguish among different sources of learning difficulty: grammar sampling problems, memory problems, parsing capacity problems, error rates, and so forth. However, our aims are too ambitious for us to achieve all this in this chapter. As will become clear, we must settle for formalizing simplified versions of the learning models we are interested in. We believe that even this offers some insight into the very different strategies that learning systems may adopt. But conclusions drawn here on the basis of the simplified models will not necessarily extend,

of course, to the richer versions that have been proposed as models of human language learning.[12]

5.5.1 TLA- = TLA without SVC

We focus first on stage I, and later consider its impact on stages II and III. Although stage I serves the function of discovering the triggering information in input sentences, in the TLA- no particular role is played by the association of any sentence or sentence property with a particular parameter value. The TLA- in effect ignores the parameterization of the language domain, except insofar as it offers a finite and orderly array of possible grammars to hypothesize. The TLA- recognizes a sentence s as a trigger in the same way as the TLA does, by trying out grammars against it in hope of finding one that licenses it (see footnote on page 452). If it succeeds, it adopts that grammar. Thus it treats an input sentence, in effect, as a trigger for adopting all the new values which differentiate the successful grammar from the previous one that failed. The work expended per novel input (i.e., an input not licensed by the learner's current grammar) consists in trying to parse the sentence a second time with a new grammar, subsequent to the failed parse with the current grammar. If this second parse succeeds, some information is

gained, and at relatively little cost. But every sentence from which the learner gains no information constitutes a waste of the effort that went into the second parse, as well as a waste of learning time.[13] If few input sentences were wasted, the TLA- would be a quite economical means of language acquisition. In fact it is very expensive.

We illustrate here with some particular numerical estimates, followed by general formulae in each case. We make the following background assumptions throughout these calculations. Their general trend is to impose homogeneity on the learning process, in order to simplify both the mathematics and the exposition. We do not assert that these points are true of natural languages; indeed, we very much doubt that some of them are. But they will allow us to take some initial steps towards what must ultimately be a more sophisticated formalization.

(i) The sample of the target language that a learner is exposed to entails the value of every parameter relevant to the target language (i.e., every parameter that needs to be set);

(ii) The learner's input sample does not necessarily exhaust the target language (e.g., it may be limited to sentences of two clauses or less), but the sample is uniformly distributed (i.e., no particular sentence type within it is systematically withheld or delayed);

(iii) Every language in the domain shares an equal number of sentence types with the target language;

(iv) If a target sentence is licensed by two grammars, then it is also licensed by every grammar which shares the parameter values they have in common;

(v) All sentences within the target language are ambiguous with respect to the same number of parameters. (We assume this to start with, but later we introduce some flexibility in this regard.);

(vi) Grammars tested by the TLA- are selected with equal probability from the set of candidates (= all possible grammars other than the failed current grammar).

Suppose there are 30 binary syntactic parameters. (See Chapter 3, page 170 on how many parameters it is reasonable to assume for natural language syntax). Assuming that there are no constraints limiting which parameter values can co-occur, their combinations amount to 2^{30} (= 1,073,741,824) possible grammars. Suppose that only 25 out of the 30 parameters are relevant to the target language (irrelevant parameters control properties of phenomena not present in the target language, such as clitic order in a language without clitics). Then 5 parameters are irrelevant, and the consequence is that 2^5 (=32) of the billion gram-

mars count as equally correct for the target language. In general, for r relevant parameters out of a total of n parameters in the domain, the number of correct (target) grammars is 2^{n-r}. A useful way of considering irrelevance is that the total class of 2^n grammars is thereby clumped into 2^r equivalence classes, each containing 2^{n-r} grammars; the grammars within any one equivalence class have identical consequences for target language sentences. In the present case, there are 33,554,432 ($= 2^{25}$) equivalence classes each consisting of 32 ($= 2^5$) grammars; one of these equivalence classes constitutes the target.

Now let us consider parametric ambiguity. Suppose that all sentences of the target language are parametrically ambiguous. Specifically, let us suppose here that each is ambiguous with respect to 8 parameters that are relevant to the target (not the same 8 in every case). (Recall that between them the input sentences must provide unambiguous information about each relevant parameter; see assumption (i) above.) Then each input sentence is licensed by $2^8 * 2^5 = 2^{8+5} = 2^{13} = 8,192$ grammars; this follows from assumption (iv).[14] Of these, 2^5 ($= 32$) grammars are correct and 8,192 - 32 = 8,160 are incorrect. In general, a target sentence that is ambiguous with respect to a parameters is assumed to

be licensed by $2^a * 2^{n-r} = 2^{a+n-r}$ grammars, of which 2^{n-r} are correct and $2^{a+n-r} - 2^{n-r} = 2^{n-r} * (2^a - 1)$ are incorrect.[15]

Let us now cut to the point at which the TLA- adopts a grammar, and ask what the probability is that the grammar it adopts is one of the 2^{n-r} correct grammars. Suppose for the moment (though we will return to this below) that the learning algorithm has established already that its currently hypothesized grammar G does not parse the current input sentence s, and that the candidate new grammar G' does parse s. Given that 2^{n-r} grammars out of the 2^{a+n-r} grammars that could parse s are correct, the probability that G' is correct is $2^{n-r}/2^{a+n-r} = 1/2^a$. For our current estimates it is $1/2^8 = 1/256$, or a 0.39% chance of guessing correctly. It follows that on average the number of grammar changes that would be necessary to identify the target grammar is 256, or in general $1/1/2^a = 2^a$ (see Appendix). Note that this value increases exponentially with the degree of ambiguity (i.e., with a, the number of parameters with respect to which an input is ambiguous).

Now let us unpack the two temporary assumptions we made above. We will take them in sequence. First, there is some probability that G will parse the input s so that no grammar change will be attempted. This must be factored into our calculations. Consider the probability

that an arbitrary grammar G will be able to parse a given sentence s from the target language. Recall from above that an input sentence is licensed by 2^{a+n-r} grammars out of the total of 2^n grammars. Therefore, the probability that a grammar G can parse an input s is $2^{a+n-r}/2^n = 2^{a-r}$, or $2^a/2^r$. This is the probability that the current grammar G can parse the input and will not be given up. The probability that G fails to parse the input, so that the learner recognizes the need to switch to a new grammar, is thus $1 - (2^a/2^r)$. In the present case, this is 0.99999237.[16] In other words, with this degree of ambiguity the learner would be motivated to change grammars on almost every trial. Note that as ambiguity increases, the probability of a needed grammar change decreases, other things being equal; hence the number of inputs between attempted grammar changes increases. This is because at higher ambiguity levels, G will parse more sentences for which it is not in fact correct. Nevertheless, despite this increase, the number of inputs between attempted changes remains relatively low except at extremely high levels of ambiguity. For instance, the number of inputs per attempted change is increased only by a factor of approximately 2 (relative to total unambiguity of all inputs) if all sentences are ambiguous for all parameters except one. It is increased by a factor of 20 if 90% of

sentences are fully ambiguous and 10% are ambiguous for all parameters except one (with an average ambiguity of 24.9 when $r = 25$, i.e., 99.6% ambiguity). Clearly, the range of this multiplier reflecting the effect of parametric ambiguity on the success rate of G is quite limited. As shown below, this factor does not have a major impact on how quickly on average a learner will identify the target grammar.

The other matter that needs to be brought into the equation is the probability that G', the grammar that is put through the parse test when G fails, does parse s. The calculation is similar to the general formula given above for the parsing success of G, except that only $2^n - 1$ grammars are under consideration because the current grammar G has disqualified itself by failing its parse test. For $n = 30$ (or for any other plausible size n) this number is very close to 2^n, so to simplify the calculations that follow we will substitute 2^n (or equivalently: we can simplify by assuming that G is included in the pool of grammars from which the learner selects after G has failed). Thus, the general formula gives us that for any G' that the learner tries out when G fails, the probability that G' will parse the input $s = 2^{a-r}$. This is a very low probability except at high levels of ambiguity. Thus, very often when the learner needs to change to a new grammar, its candidate new grammar

will fail the parse test. Because Greediness does not allow the TLA- to change to a grammar that fails the parse test, the rate of *actual* grammar change will be low. Grammar change occurs in the TLA- only when G fails and G' succeeds, and the probability of this is $(1-2^{a-r}) * 2^{a-r}$. Given our current estimates, it is approximately 0.000007629 or on average one actual grammar change every 131,073 input sentences.[17] Between these events, no learning occurs. As calculated above, 256 such changes would have to occur on average before convergence on the target grammar. This is not a great number, but the rarity with which Greediness is satisfied magnifies it considerably. The total number of inputs consumed before convergence is on average $256 * 131{,}073 = 33{,}554{,}688$.[18] In general, the average total number of inputs consumed before convergence is

$$2^a * 1/((1-2^{a-r}) * 2^{a-r}) = 1/((1-2^{a-r}) * 2^{-r}) = 2^r/(1-2^{a-r})$$

.

Table 5.1 presents some figures for other values of a and r.

Note that the number of inputs required on average rises exponentially in r, the number of relevant parameters (= the number of parameters that need to be set for the target language), and is over a billion for 30

	r = 15	r = 20	r = 25	r = 30
a = 0	32,769	1,048,577	33,554,433	1,073,741,825
a = 5	32,800	1,048,608	33,554,464	1,073,741,856
a = 10	33,835	1,049,601	33,555,456	1,073,742,848
a = 15	can't learn	1,082,401	33,587,232	1,073,774,593
a = 20		can't learn	34,636,833	1,074,791,425
a = 25			can't learn	1,108,378,657
a = 30				can't learn

Table 5.1. *Average number of inputs consumed by the TLA- before convergence. Formula:* $2^r/(1-2^{a-r})$

relevant parameters. As an informal rule of thumb, one can estimate the number of sentences to convergence as approximately 2^r (because the denominator in the formula above differs very little from 1 unless a is high relative to r). For 15 parameters the average cost in input is a little over 2,000 sentences per parameter; for 30 parameters it is over 35 million per parameter. Because efficiency is so much better at the lower end of the scale, it is important for this model that the number of parameters for natural language be low. Linguistic research has this as its goal and may ultimately show that it is so, but at present there would probably be broad agreement among linguists that 15 syntactic parameters underestimates the extent of natural language variation. Note that by the rule of thumb, each extra ten parameters multiplies the total number of inputs needed on average by about 1,000. So for 40

parameters, it would be on the order of a trillion, or about 25 billion per parameter.

The learning process is sketched in Figure 5.1, as a probability tree with all three stages shown: the possible success or failure of the current grammar G to start with, then the success or failure in finding a new grammar G' that satisfies Greediness, and then the question of whether the grammar adopted is correct (in which case no more changes will ever be required) or incorrect (so that the process must be repeated on another input). Note that convergence on the current trial corresponds to the path on which G fails the parse test, G' passes the parse test, and G' is indeed correct. (We round the formulae for the probabilities of G' success or failure here, as noted above.)

With the structure of the situation thus outlined, we can elaborate it in various ways to make it capable of approximating real learning situations more closely. Suppose, for example, that sentences are not uniformly ambiguous. Perhaps 10% of the target language sample are parametrically unambiguous, while 90% are each ambiguous with respect to 8 parameters. We'll retain for now the assumption that only 25 of the 30 parameters are relevant to the language. This is sketched in Figure 5.2 (see Exercise 5.3 below for further examples). Note that this

probability tree contains more branches, to represent the richer range of outcomes, but the calculations along each path are of the same kind as we worked through above. For the ambiguous sentences, the situation is exactly as in Figure 5.1.

Fig. 5.1. Probabilities of different outcomes of an encounter between the TLA- and an input sentence. Homogeneous ambiguity: 8 ambiguous parameters per sentence

Fig. 5.2. Probabilities of different outcomes of an encounter between the TLA- and an input sentence. 10% unambiguous sentences, 90% with 8 ambiguous parameters per sentence

5.5.2 Summary of TLA- performance

The amount of work expended by the TLA- in attaining the target grammar is a function of the number of sentences it consumes before convergence. We have seen that, given the general assumptions made above, this increases exponentially with the number of parameters that need to be set (= r, the number of relevant parameters). To a lesser degree, it increases also with the degree of parametric ambiguity. It becomes implausibly high at levels of parameterization and ambiguity that seem to be not unreasonable (or perhaps overly modest) estimates for the human language domain. We will review the roles of these two factors, r and a, in turn. Note in general that in terms of the three-stage analysis of grammar acquisition in Section 5.1, the TLA-'s grammar selection process at Stage I and grammar adoption process at Stage II have two distinct undesirable effects. They inhibit grammar change in many cases where change is necessary, and they permit change to a wrong grammar in response to ambiguous input. The latter necessitates further parameter setting to correct the errors, which inflates the workload at Stage III. The costs of this Stage III repair process are included in the estimates above of the amount of input consumed en route to the target grammar.

The exponential dependence of workload on the number of parameters to be set is due to the TLA-'s blind (i.e., merely error-driven) search through the field of all possible grammars for one that is compatible with the input. Where the classical instant-triggering model tested a sentence to see what its implications were for each parameter, the TLA- tests a grammar to see what its implications are for a given sentence. Without advance knowledge of what would be a good grammar to test, the latter is a slow business. If the learner had an oracle that would tell it which parameters most likely need to be reset to accommodate a given input, it could avoid the TLA-'s high rate of failure in the parse test, and the extreme waste of input sentences which that causes. How to improve on the TLA- in this respect is the topic of Sections 5.8 and 5.9.

The dependence of the TLA-'s workload on the degree of parametric ambiguity is less extreme and more complex in nature. Greater ambiguity decreases the probability that the current grammar will fail, even if it is wrong and does need to be changed. This is not a fact about the TLA- alone, but is true of all error-driven learning systems: the spurious success of wrong hypotheses encourages complacency, and will slow down discovery of the correct hypothesis. We know of no data on

this for human language learning, but at least at present there is no reason to doubt that it is true. As noted above, this consequence of the fact that learning is error-driven is predicted to be a relatively modest effect except at extremely high degrees of parametric ambiguity (approaching 100%); when ambiguity is moderate, it is just not very likely that a wrong grammar will survive many tests against target language sentences.

Once an error has been detected and G is known to be wrong, ambiguity has no further effect on the overall success rate of the TLA-. In our calculations above we examined Stage I and Stage II separately, and found that ambiguity has an effect on each; but these effects cancel each other out. The rate of potential trigger recognition (i.e., of finding a new grammar G' that satisfies Greediness) in stage I increases exponentially with a, the degree of ambiguity. In other words, for higher ambiguity, grammar change will occur more often when grammar change is needed. However, there is no guarantee that the change that occurs will be the one that is needed. In fact, the probability of correct (target) grammar adoption from the candidates delivered to stage II decreases exponentially with the degree of ambiguity. These two effects are equal and opposite and so cancel out. (The respective formulae are: $2^a/2^r$ and

$1/2^a$; see above.) The *only* net effect of parametric ambiguity within the TLA- is thus in the initial testing of the current grammar G.

This considerable indifference of the TLA- to ambiguity levels could be seen as encouraging, but what it amounts to is really that the TLA- succeeds equally rarely whether the language domain is ambiguous or not. This is because it does not actually make use of the fact that a given input is ambiguous or unambiguous; it simply ignores the matter. The TLA-, in effect, just picks randomly out of the 2^n total grammars, in hope of hitting on one of the 2^{n-r} correct grammars for the target. This is a needle-in-a-haystack problem that gets exponentially worse as more parameters need to be set. It is clearly worth considering whether a solution might be found for the 'sampling problem' of stage I (e.g., by means of some kind of pre-test to identify promising parameters worth checking; see discussion of a proposal by Valian, in Fodor, 1998a). Note, however, that the exponential cost of ambiguity would then emerge at stage II. All in all, then, the complexity characteristics of the TLA- do not commend it as a model of human language learning.

5.6 Adding in the SVC

Our remarks in this section are speculations only. The SVC is not our main focus of inquiry and so we can treat it only briefly. Moreover, it is particularly difficult to calculate any general effects of the SVC, since it interacts in an intricate way with Greediness and with the pattern of ambiguity distribution within a particular language domain. (If of interest, however, it would be possible to apply Berwick and Niyogi's Markov model to an algorithm with SVC in a language domain meeting the assumptions we have made here.) In the Structural Triggers model that we propose in Section 5.8 the SVC does not apply (because it is not needed). We therefore leave it to other investigators to develop a more satisfactory theory of SVC effects than we are able to here. See Nyberg, 1992; Clark, 1992; Niyogi and Berwick, 1996, Sakas (in prep.); see also Exercise 5.2 at the end of this chapter.

Gibson and Wexler's TLA differs from the TLA- only in that the SVC forces the TLA to limit its choice of a new grammar to be tested; it must pick from among those that differ by only one parameter setting from the current (failed) grammar G. The SVC thus causes an uneven sampling of the class of possible grammars at stage I. It has the consequence that the learner will tend to perseverate, cycling through the same cluster

of neighboring grammars to a greater extent than it would without the SVC. To the extent that this increases the rate at which the same grammar is tested repeatedly, or the chance of testing a grammar that differs from a failed one only by an irrelevant parameter, this could postpone convergence on the target. But such effects are probably minor compared with the more fundamental question of whether grammars that are similar tend to license languages that are similar, and the related but more pertinent question of whether successful grammars tend to be similar to other successful grammars.

To extrapolate to the TLA the performance measures computed above for the TLA-, we would need to answer two questions concerning the SVC. (1) Does the SVC alter the probability that a grammar selected for testing at stage I licenses the input sentence s? That is, are the n grammars that differ from G with respect to just one parameter (which we will call "1-adjacent" grammars; Nyberg, 1992) more likely to license s than any of the other grammars that differ from G? (2) Does the SVC alter the probability that G', a grammar selected at stage II, is the target grammar? That is, among the grammars that do parse s, is a grammar that is 1-adjacent to G more likely to be the correct grammar for s than a less similar grammar is? The answers to these questions are not simple,

but we can indicate their general outlines. We emphasize again that this is a preliminary exploration only. And we will keep it as manageable as possible by taking advantage here of the point demonstrated in section 5.5.1, that the conceptual distinction between grammar sampling (stage I) and grammar adoption (stage II) has no substantial consequence in the TLA- and does not affect the overall likelihood of adoption of the target grammar. Thus we can conveniently combine the two questions and ask simply: (3) At a point in the learning process at which the learner's current grammar G has just failed the parse test, are grammars that are 1-adjacent to G more likely to be (or lead to) the target grammar than grammars less similar to G? If this is generally so, then adherence to the SVC facilitates convergence.

It is important to bear in mind that the notion of "n-adjacent" is defined over the parameter value combinations that constitute grammars, not over the sets of sentences that constitute languages. Whether and how the two are correlated is what is under consideration. Berwick and Niyogi (1996) refer to "smoothness" of the relation between grammars and the languages they license, when grammar similarity and language similarity are highly correlated.[19] We do not presuppose smoothness here, but regard it as an empirical issue to be evaluated. The consider-

ations raised below suggest that in fact smoothness cannot be relied on by natural language learners.

To answer question (3) we must start with the properties of the failed current grammar G. What is certain is that G licenses at least one sentence of the target language but not all (a default or randomly chosen grammar at the very outset of learning may license none). In some cases the current grammar will have parsed several target sentences in a row. This is not a fact that a TLA learner has access to (since there is no memory for prior learning events), but it nevertheless has an indirect effect. The more target sentences a grammar licenses, the more likely it is in a greedy system to become the current grammar and to stay that way. Thus on average, over the long run, the learner's current grammar at any point is likely to license more target sentences than an arbitrary other grammar does. This did not figure in our calculations in Section 5.5.1 because there we made the homogeneity assumption (iii) such that all grammars except the target grammar license exactly the same number of target sentences. The advantage of the current grammar may be very slight. It depends on the degree and distribution of ambiguity in the target language. For instance, if *all* grammars in the domain license most of the sentences in the target sample, then an

arbitrary incorrect grammar could be the learner's current grammar for many trials; hence, being the current grammar would not be a strong predictor of ultimate success. It is clear, then, that being able to license many target sentences is not the same as being the target grammar, and nor does it entail having many parameter values in common with the target grammar.

For natural languages Chomsky has emphasized that small changes in parameter settings can have considerable effects on the languages generated, due to the rich interactions of principles, parameters and lexical properties in sentence derivations. For example, he writes (Chomsky, 1988, p.63): "there is no simple relation between the value selected for a parameter and the consequence of this choice as it works its way through the intricate system of universal grammar. It may turn out that the change of a few parameters, or even of one, yields a language that seems to be quite different in character from the original." If this is so, the degree of overlap of the parameter settings in two grammars would *not* correlate highly with the degree of overlap of the languages they generate; the grammar/language relationship would not be a smooth one. But regardless of whether Chomsky's point establishes non-smoothness in general, it is certainly the case that parametric ambiguity can intro-

duce some significant bumps into the grammar/language correlation. In an ambiguous domain there may be several grammars dotted distantly around the grammar space all of which can license many sentences of the target language. Only one of them is G_t, the target grammar. It is very clear that a grammar that is successful for one target sentence, or even for many, need not in this case resemble G_t closely at all. For example, in Gibson and Wexler's small domain of three word order parameters, the sentence pattern SVO is licensed by grammars that differ from each other maximally: every one of their parameters is set differently. SVO order is licensed by the grammar SV, VO and -V2, and by the grammar VS, OV and +V2 (among others). In such a case, the SVC could do the learner more harm than good. For instance, if the starting state were VS, OV, -V2 and the target were SV, VO, -V2, then it would obviously *not* be most profitable for the learner to favor grammars 1-adjacent to G; in response to an SVO input, that would force a step in the wrong direction, to +V2.

In sum: adherence to the SVC gives the TLA a tendency to hover in one area of the total space of grammars, typically (because of Greediness) a promising area where some parsing success has occurred. If this area includes the target grammar, convergence is likely to be more

rapid with the SVC than without it. But in a language space with parametric ambiguity, the SVC may unhelpfully focus the search on an area around a competing grammar, which functions as an attractor because it can parse many of the target sentences but which is nevertheless an incorrect (non-target) grammar. In this case the SVC would retard convergence on the target: the learner needs to make a substantial shift to a very different grammar but it is hindered from doing so by the SVC (and Greediness). In the most extreme case, a TLA learner that has gone astray can get permanently stuck, if it finds itself at a local maximum, i.e., a wrong grammar from which there is no possible route to the target grammar via a sequence of successful one-parameter changes as required by Greediness and the SVC. (See Gibson and Wexler (1994), for discussion, and Winston (1992) on local maxima in other types of learning systems.) Interestingly, this becomes more likely if the degree of parametric ambiguity is low, though in general a reduction in ambiguity would be expected to facilitate learning. In a fully unambiguous domain, the TLA is unable to change grammars at all, unless the target is exactly one parameter away from the starting grammar.

Translated into implications for human learning, a major consequence of the SVC would appear to be a considerable variability in acquisition

outcomes. This would especially be so if individual learners can differ with respect to their starting state (e.g., if there are no innate defaults), and if the order of information received can vary substantially across children (i.e., if the frequency distribution of construction types in input to children is not very dependable). Some children could by good fortune arrive at the target grammar very rapidly, while other children would toil through many blind alleys before attaining the target, and some might never get there at all (unless the local maximum problem can be solved in some way that Gibson and Wexler suggest). Though we cannot quantify the discrepancy, this seems to us to be out of keeping with the remarkably uniform success of human language learners. Thus, even if the distribution of ambiguity in natural language should turn out to be such that the net effect of the SVC is a considerable improvement in average acquisition rates relative to the TLA-, it is arguable that the SVC does not improve the psychological fidelity of the learning algorithm.

In Section 5.7 we will turn to issues concerning the isolation of individual parameters so that their contributions can be evaluated independently of the grammars in which they are embedded. In this connection it may appear that the SVC would be an unqualified blessing. For the

TLA-, which lacks the SVC, a trigger sentence is a trigger for a collection of parameter values, even a whole grammar, as noted above. If a new grammar succeeds in parsing an input sentence, there is no way for the TLA- to tell which of its parameter settings was/were responsible for the success, and which were simply irrelevant (to this sentence, or to the whole language). So the TLA- must accept or reject the collection of parameter settings as a whole, without having any idea as to which of them are now correct. But in the TLA, the SVC makes it possible to attribute a parsing success to the one parameter whose value differs from the prior (failed) grammar. As we will see in Section 5.7, however, the TLA is unable to take advantage of this ability to focus on specific parameters, because of a lack of certainty caused by ambiguity. The new parameter setting is the one that made the parse successful but still it might or might not be correct, because that parse might or might not be the correct parse. The TLA adopts the successful parameter setting, but it cannot do so with confidence; and as we will show, confidence has an important effect on efficiency. Thus in this respect, also, the SVC is not as helpful as might have been expected.

5.6.1 Evaluation of the TLA

Our assessment of the TLA in comparison with the TLA- has been that the TLA may well prove to be more efficient overall, but it seems likely to prove less plausible psychologically than the TLA- because of the greater variability of its routes to convergence in an unevenly ambiguous domain. In this respect the TLA may be more sensitive to parametric ambiguity than the TLA-. For the TLA- there is no sense of the learner making gradual progress toward the correct grammar; each hypothesis is randomly pulled from the total pool. By contrast, for the TLA, the SVC in combination with Greediness causes consistent shifting in the direction of locally more successful grammars. To the extent that smoothness reigns, this is beneficial. But the gradualness entailed by the SVC can be damaging if the learner is steadily converging on a wrong grammar because the domain is skewed due to ambiguity, and the SVC does not allow it an easy escape. In respects other than the SVC, the TLA is like the TLA- and thus it can be expected to exhibit the properties documented in Sections 5.5.1 and 5.5.2. That is, on a high proportion of trials on which it knows it must switch to a new grammar it may fail to find a promising grammar to switch to. When it does switch, its blindness to parametric ambiguity can result in errors. Together,

these problems appear to create an implausibly heavy workload for any reasonable number of natural language parameters. We would stress, however, that it would be valuable to check general remarks such as these by means of Markov modeling or simulations of particular cases (see Exercises 5.2 below).

Note: For convenience in the following discussion we will not always distinguish now between the TLA and the TLA-, despite their partially different characteristics noted here. Our focus shifts now to the comparison between this class of non-deterministic grammar-testing systems and a class of systems which can detect parametric ambiguity and avoid errors.

5.7 The Parametric Principle

The fact that the workload of a learning device is exponential in the number of parameters betrays the fact that it is not really a parameter setting device. The TLA does assign values to parameters, but it does not incorporate the central insight of parameter theory. Parameterization has several advantages for learnability. The set of hypotheses for learners to consider is finite and orderly (while still allowing for a great variety of languages); there isn't an open-ended set of hypotheses that

learners must devise from scratch. The testing of hypotheses does not require the learner to engage in a hunt for generalizations or a laborious comparison of minimal pairs of examples. Though we have seen that the learning machinery cannot be just 'automatic' triggering, it is still true that once the relevant triggers have been identified in the input, the work is done. The TLA takes full advantage of both of these benefits of parametric theory. However, it does not abide by what we will call the *Parametric Principle: The value of each parameter is established independently of the values of all others.* This is what distinguishes a true parameter setting device from learning systems of other kinds, and it is the source of the enormous simplification of the learning task for which the principles-and-parameters model is renowned.

The essential point is familiar: if there are n binary parameters and the learning procedure is able to establish the value of each one independently of the others, then only n bits of information need to be extracted from the input sample for convergence on any one of 2^n grammars. In other words, given the Parametric Principle, the extent of the learning task depends only linearly on the number of parameters. It may take a little or a lot of work to set each one, but at least the workload per parameter is roughly constant. By contrast, any learning device that e-

valuates grammars rather than individual parameter values faces a task that expands linearly with the number of grammars, hence exponentially with the number of parameters. The workload per grammar would have to be vanishingly slight in order for the total labor not to spiral out of hand rapidly for any reasonable number of parameters. Gibson and Wexler tested the TLA on an artificially small domain of eight languages defined by three binary parameters, where the extreme difference in workload between setting parameters and selecting grammars did not become apparent. But if the estimate of 30 syntactic parameters for natural language is more realistic, then the disparity is between setting 30 parameters and checking more than a billion grammars.

A useful way to look at the difference between testing grammars and testing parameters is to see convergence as the elimination of all grammars other than the target, and to consider how effectively different procedures manage to eliminate grammars. A grammar-by-grammar test can eliminate them one at a time at best, so even if the learner kept track of the fate of every one, in the worst case it could take up to $2^n - 1$ eliminative steps to rule out all but the target. In fact, the TLA never eliminates any possible grammars at all, because it does not take the trouble to record negative outcomes of its parse tests. We assume

that this is because the slight reduction in the search space that results from eliminating individual grammars does not compensate for the cost of record-keeping on such a vast scale.[20] Therefore all grammars remain in the pool from which the TLA selects a grammar to test, with the consequence that some incorrect grammars may be tried out many times. More importantly, the pool does not get any smaller as learning proceeds, so the stage I probability of finding a grammar that can parse the input does not improve (except for whatever contribution the SVC makes). This, as we saw, is a major source of the TLA's inefficiency.[21] Because it does not obey the Parametric Principle, the TLA gropes just as randomly later on in the learning process as it does at the beginning.

By contrast, true parameter setting permits a very rapid reduction of the pool of possible grammars. Each time a parameter is set, one parameter value is eliminated. And since half of all grammars have that parameter value, that eliminates from consideration half of the candidate grammars remaining.[22] In a domain of 30 parameters, setting one parameter rules out roughly 500 million grammars; setting the next one excludes another 250 million; setting five reduces the pool to roughly 3% of its original size. This is how the Parametric Principle makes such a great difference to the scale of the learning problem. Chomsky's insight

was that if grammar acquisition is a selective rather than a creative process, its complexity need be no more than linear in the number of ways in which grammars can differ from each other.[23]

Nevertheless, very few existing learning models abide by the Parametric Principle. Statistical weighting systems such as those proposed by Valian (1994) and Kapur (1994) postpone setting a parameter for some time while evaluating the evidence, but do eventually settle on a value for each and set it permanently. The Structural Triggers Learner that we describe here in Sections 5.8-5.10 also is designed to obey the Parametric Principle. It seems astonishing that a parameter-based learning model would *not* take advantage of the powerful reduction of the acquisition problem that the Parametric Principle makes possible. Why would this be so? There are no compensating advantages to be gained by searching through the vast space of grammars. At best, clever search strategies may make it less punishing. Such strategies are being sought in current research in frameworks such as genetic algorithms and neural networks. It remains to be seen, of course, but at present it seems unlikely that any improvement would rival that due to the Parametric Principle. The sole reason for violating the Parametric Principle, it appears, is that obeying it is too difficult. The literature on language learnability does

not make this clear. The point is rarely addressed explicitly. As far as we know, only Clark (1994) considers it, and he judges that the computational costs of respecting what we are calling the Parametric Principle "are too great to be acceptable." If true, this is a very consequential fact. Being forced to give up the idea of 'instant' triggering does some damage to Chomsky's original elegant conception of parameter setting, but to give up the Parametric Principle would be to abandon its whole essence. If it cannot be avoided, then it must be accepted and we must reconcile it as best we can with the efficiency with which children learn language. But the stakes are high enough that it is worth some further thought before we give in. In the remainder of this chapter we will argue for an approach that permits true parameter setting in accord with the Parametric Principle.

In order to obey the Parametric Principle, a learner must be able to establish a parameter value with sufficient confidence to be prepared to rule out forever all grammars in which that parameter takes the opposite value.[24] To simplify here, we continue to set aside the possibility of errors due to faulty input or performance slips. We also make the standard assumption that the two values of a parameter are mutually exclusive in a grammar. Then there is no reason not to set a param-

eter permanently, and permanently discard its contrary value, as soon as clear evidence of its value is received. However, a stochastic learning device such as the TLA cannot do this because it does not *know* when it has received clear evidence for a parameter setting, i.e., evidence that some sentence of the target language cannot be licensed (parsed) without that value. The SVC helps it come close to this knowledge, but parametric ambiguity undermines it. The SVC isolates the contribution of an individual parameter value in the parse test, so the TLA (unlike the TLA-) knows exactly which value potentially earns support from the success of the parse. But the support is only potential, not reliable confirmation, because the sentence might have been ambiguous and the parse assigned to it might have been the wrong one. In that case the positive outcome of the parse test provides no evidence at all for that parameter value; for all the learning system can know, the sentence is equally compatible with the opposite value.

Is there any way out of this crippling uncertainty? The uncertainty would not arise if there were no parametric ambiguity in the domain and if the learner knew that were so; but that is not realistic for natural language. Alternatively, the uncertainty could be resolved if the parser could run an exhaustive check of all possible parses of a sentence. If

the parameter value in question were present in every grammar that could parse the sentence, then it could be adopted with full confidence. However, as we noted in Section 5.4, an exhaustive search through a billion grammars is hopelessly impractical if the parser can try out only one of them at a time; and attempting a billion parses simultaneously is presumably no more feasible than *seriatim*.[25] Finally, the uncertainty due to ambiguity could be avoided by the learner if it could establish which inputs were parametrically ambiguous and refrain from setting parameters in response to them. Learning would be based solely on unambiguous triggers. However, this too demands parsing with multiple grammars. Parametric ambiguity can be established by parsing with enough grammars to find two that parse the input; non-ambiguity can be established only by parsing with all possible grammars and finding no more than one that parses the input. The consensus in sentence processing research is that even adults are capable of only limited parallel parsing if any (see Gibson, 1991), even when the alternative analyses all involve the same grammar. It does not seem plausible to suppose that a two-year old can apply a billion grammars to each passing sentence.

Summary of Section 5.7: We have considered *why* the efficiency of learning procedures such as the TLA is so low, and have found that it is

not a trait that can easily be altered. It stems from the idea of putting the parser to work to identify triggers, which seemed like an essential breakthrough but now appears to cramp optimum performance. The inability of the human parser to cope with ambiguity on a large scale has a serious negative consequence for acquisition. It creates uncertainty, which entails indeterminacy of parameter evaluation, which precludes definitive setting of any parameters at all, which leaves the whole pool of grammars to be considered at every point. In other words, the search problem remains exponential because the Parametric Principle cannot be implemented.

Alternatively: We must find some way of imposing the Parametric Principle or else parameter setting is of little interest.

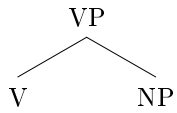
5.8 Structural triggers

Fodor (1998a) argues that parallel grammar testing would be feasible, however large the pool of grammars, if triggers and parameter values were the kinds of things that could be ingredients of grammars and ingredients of trees. If they were suitable ingredients of grammars, they could all be combined into one large grammar (termed a 'supergrammar') which the parser could apply to the input in exactly the same way

as any other grammar. No unusual parsing activity would be needed, yet all parameter values would be evaluated at once. If parameter values were suitable ingredients of trees, they could be detected in the parse trees output by the parser, so that the learning device would be able to see which of them had contributed to parsing an input sentence and would know which to adopt. What would fit the bill for both purposes is a subtree consisting of just a few nodes and/or feature specifications. A trigger and the parameter value it triggers could then be identical, so that only one innate specification would be needed, rather than linked specifications of parameter values and their triggers (as in cue-based learners; see Lightfoot, 1991). UG would provide a pool of these schematic treelets, one for each parameter value, and each natural language could choose to employ some subset of them. As trigger, a treelet would be detected in the structure of input sentences (i.e., 'trigger sentences'). As parameter value, it would then be adopted into the learner's current grammar, and would be available for licensing new sentences.

Consider some examples. For the Complement-final value of the word order parameter for VP, the structural trigger/parameter value might be (178), i.e., a VP subtree with the verb preceding the object. For the Complement-initial value, the treelet would be the mirror image of (178).

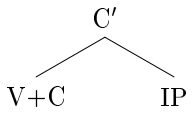
(178)



In order to cope with the depth-of-derivation problem (Section 5.2), the treelet has to reflect underlying order in any language with that value. Assuming for convenience here that the parser's output is an S-structure tree with movement chains (see note in Section 5.3), (178) will reflect underlying order as long as its terminals are not constrained in any way; when the underlying structure has been transformed, either N-P or V or both could be traces. Thus (178) would contribute to parsing not only *I despise decaf* but also *Decaf, I despise* with the O moved out of VP, and not only *I have some change* but also *Have you any change?* with the V moved out. More appropriate than (178) as a parameter value is its ultimate source, i.e., whatever is responsible for the presence of (178) in derivations, according to the linguistic theory that is assumed to be correct. In a TAG framework it might be (178) itself; but in H-PSG it might be a schematic version of (178) underspecified in terms of syntactic features; in a GB framework it might be a government direction feature of the verb; in the Minimalist Program (though the details are different, on the assumption that the deepest order is universal; see footnote on page 449) it might be a weak $\mathrm{Agr}_{\mathrm{O}}$ feature that does not

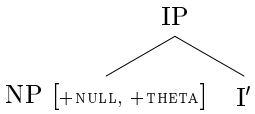
attract the object forward for checking. For purposes of learning, all that is required is that the trigger/parameter value be a piece of a tree; in other respects it is up to linguistic research to determine its properties. We assume that the structural triggers employed by the learner are exactly those elements, whatever they are, that UG specifies as the sources of possible cross-language variation. Some further examples are given for illustrative purposes in (179) and (180). Again, depending on the linguistic theory adopted, the mother-and-daughters configuration might be the real trigger, or it might be an intermediate representation, the true trigger being some ultimately responsible property such as a strong or weak feature value.

(179) The +V2 trigger/parameter value



or just: C[+FIN] or C[+strong] or ...

(180) The null subject trigger/parameter value



or just: I[+PRONOMINAL] or ...

These structural triggers make it possible to attain the goal of efficient error-free learning. We now consider more carefully how this is so. We have observed that to set parameters accurately the learner must be able to conduct an exhaustive parse test of every grammar, against each input sentence. To run an exhaustive test the parser must try out all grammars simultaneously and yet be able to attribute success to individual parameters. This could seem to be patently unachievable, but in fact it is straightforward for the Structural Triggers Learner (STL). Suppose that the STL attempts (like the TLA) to parse each input with the current grammar G first. If that fails, it tries again with the supergrammar consisting of G with all UG-provided triggers/parameter values folded into it (or more precisely: all those not yet definitively disconfirmed by the input). Unlike the TLA model, the STL makes crucial use of the structural analyses assigned to strings by the parsing routines. To say that a parse with G fails is not to say that there is no structural output at all. The parser will build as much of the tree as G licenses before it is forced to a halt at the point at which it needs a new parameter value in order to proceed. It will then draw on the pool of treelets provided by UG to find one (or more) to patch the hole in the parse tree. Thus

the learning device does not attempt to *spot* the trigger treelets in input sentences. Rather, it *contributes* the triggers to the input, when parsing cannot proceed without them. It knows that a treelet must be in the target grammar if it finds that that treelet and no other can enable an otherwise blocked parse.

But what if an input sentence is parametrically ambiguous? If it is, the supergrammar will define more than one parse tree for it. At some point in the parsing process, therefore, the parser will be faced with a choice between two (or more) analyses. So to detect parametric ambiguity, the parser needs to note when a choice point arises in parsing with the supergrammar, a point at which two (or more) analyses present themselves. If there is no such choice point, the input has just one supergrammar parse. It is parametrically unambiguous, and every parameter value present in the parse-tree is correct; the learner should adopt all of them that are not already in G. The same applies to any parameter values involved in the analysis of the sentence prior to a choice point in the parse, since the sentence is unambiguous up to that point.[26] If and when a choice between alternative analyses does arise, there are two strategies the learner might adopt; we will call them the strong and weak strategies. If the parser is capable of pursuing all the possible analyses

that present themselves, that can provide useful information. Specifically, if all the parses involve the *same* parameter values, then the sentence is structurally but not parametrically ambiguous, and those parameter values can safely be adopted. Or the sentence may be parametrically ambiguous but one parameter value contributes to all of the possible parses; if so, that value is correct (it is essential to the licensing of that sentence) and can safely be adopted. Thus the strong strategy looks for the common denominator across all possible parses: treelets that are present in them all. In this way it extracts the maximum secure information out of the mix of reliable and unreliable parametric cues that natural language sentences typically present.

However, the assumption that the human parsing mechanism is capable of full parallel parsing, for sentences that could be multiply ambiguous, does not square with the empirical evidence on adult sentence processing. As noted above, it seems that even the adult parsing mechanism has little or no capacity for parallel parsing; and children, we presume, have no greater capacity in this respect than adults. So this strong learning strategy is not feasible. The weak strategy employs what is essentially a serial parser and is more realistic. When the parser notes a choice point in a sentence, it selects one analysis to pursue for pur-

poses of comprehension and it ignores all other analyses. But it reports the presence of ambiguity to the learning mechanism, and the learner thereafter adopts no new parameter values on the basis of that sentence. Since it cannot know what parameter values might have been involved in the other parses, had it pursued them, it cannot be certain which values, if any, would be common to all analyses of the string, and so it cannot safely acquire any of them.

Note that in this weak version of the STL, alternative grammars are still tested in parallel, in order to detect and avoid ambiguity, even though the system *does not conduct a parallel parse* of the sentence. There is some parallelism but it is only momentary, as the parser registers the existence of more than one way of attaching the next input word into the tree structure it is building for the sentence. This kind of incipient parallelism is presupposed by any parsing system that selects between alternative analyses on the basis of preference strategies (such as Minimal Attachment). Most importantly, since the alternatives are not pursued in full, there is no consumption of exponential space or time resources even in the worst case; processing load for an ambiguous sentence is little or no greater than that for an unambiguous one. Of course, this parser delivers, in consequence, less information than a

fully parallel one does, but we will show that in many circumstances it delivers sufficient for the needs of learning.

In Table 5.2 some examples are given of the actions of the STL, strong and weak versions, in response to various outcomes of the supergrammar parse. Only the three parameters studied by Gibson and Wexler are considered in this illustration, so this is an artificially simple domain.

The weak STL does not find out as much about each sentence as the strong STL does, so there are sentences the strong STL can learn from that the weak STL can only discard. But what is important is that the weak STL is able to detect parametric ambiguity reliably, and can therefore avoid being misled by ambiguous inputs. So the weak STL is just as safe as the strong one.[27] What they have in common, and gives them their ability to cope with parametric ambiguity, is that they approach an input sentence with all UG-permitted grammars at once, each one ready to parse the sentence if called on. This is feasible because these millions of grammars have been compacted into a single grammar, and the parser can call on the individual parameter values separately as and when they are needed to make the parse successful.

Table 5.2. *Examples of STL responses to outcomes of supergrammar parse*

The parser the weak STL employs is very modest; it is hard to imagine that the human sentence parsing mechanism could do much less. Nevertheless, the weak STL (henceforth WSTL) can do two things that the TLA- can't do. It can *always* find a new grammar to parse an input for which the current grammar has failed. So it does not waste numerous inputs on a hunt for a successful grammar, as the TLA- does. In Section 5.5.1 we wished for an oracle that would whisper to the parser which grammar is worth trying out in the parse test. The supergrammar parse of the STL acts like that oracle. The sentence parsing device is specialized in finding an analysis for a given sentence on the basis of structural resources available to it in the form of a grammar; this is the parser's normal job. Given the supergrammar, which makes all possible resources available, the parser exercises its usual skill, and establishes a parse for every sentence that has one (i.e., every sentence compatible with UG), except only for sentences which outstrip the parser's ability, such as multiple center-embeddings.[28] Second, because the STL knows which inputs are ambiguous it can *avoid guessing*, and so avoid errors, and so avoid having to keep resetting parameters until they are all correct at the same time, as the TLA- must do.

Most importantly, the WSTL obeys the Parametric Principle and

reaps all the benefits thereof outlined above. Because the WSTL does not take chances it can set parameters with confidence, so it can be confident enough to discard incorrect values, and cut the size of the subsequent learning problem in half. Note that the WSTL is a true parameter setting device which obtains separate evidence (in the output parse tree) for each parameter involved in the derivation of a sentence. It does not test grammars as wholes. The successive halving of the learning problem manifests itself in this system as a reduction in the number of alternative parses subsequent sentences will have. This will be high to start with, because all the UG parameter values are available to parse with and offer many alternative sentence structures, but the number will shrink as more and more parameters are set. Thus there is progressive disambiguation of the input as learning proceeds. For the strong STL this means fewer analyses to compute per sentence. Eventually, at the point of convergence on the adult (target) grammar, all sentences will be parametrically unambiguous; the only ambiguity that remains will be any structural ambiguity inherent in the target grammar itself. For the WSTL, the advantage of eliminating wrong parameter values is that the proportion of sentences that are fully unambiguous parametrically will increase as learning proceeds. A sentence that was unusable for acqui-

sition due to ambiguity early on in the learning process could become usable some days or weeks or months later.[29]

We argued in Section 5.7 that the Parametric Principle is the only way to defeat the potential exponential complexity of the learning task. Now that we have a true parameter-setting device which respects the Parametric Principle, we can put this to the test. In the next section we check to see whether the WSTL does indeed reduce the learning problem to one whose workload is linear in the number of parameters, as Chomsky envisaged.

First, we make one more distinction among models of the STL variety. The weak STL that we have described is very weak. It throws away entirely any sentence that it suspects even might have an ambiguity in it. But the ambiguity might be only temporary. And in any case it does not impugn any part of the sentence preceding the ambiguity point. Even after the ambiguity point, the parser might be able to pick up the thread again and establish that there is only one parse of the final portion regardless of how an ambiguity in the middle of the sentence should have been resolved. What all this means is that there is more information in sentences that a parser (even a serial parser) could dig out if it were less rigidly programmed than we have described for the WSTL.

It is this more flexible and opportunistic – though still conservative – system that we believe really models the human learner. We assume the parser finds unambiguous information about trigger treelets in the sentence's structure wherever and however it can, and when it does it informs the learning system. The learner adopts a treelet just in case the parser has solid information that it is necessary for parsing the language. However, this flexible system is more difficult to model mathematically and we will not attempt to do so here. The basic sorts of computations relevant to assessing to the performance of the STL will be illustrated here with the weakest model as described above.

5.9 Performance of a STL-like learner

5.9.1 Number of inputs to convergence for STL

We are interested in calculating performance measures for the STL. For comparability to the TLA-, we employ the same framework of factors for representing ambiguity and relevance as we did in Section 5.5.1, and we make some of the same simplifying background assumptions (Section 5.5.1). We continue to assume (i), (ii) and (iv); (iii) and (vi) are unneeded; and (v) will be revised below. We consider here primarily the performance of a simple inflexible weak STL, as described above. We

will comment briefly on the strong STL, which is of theoretical interest but which we do not regard as a plausible model for human learning. Of much more practical interest is the flexible weak variant of the STL which looks for unambiguous trigger properties even in a sentence which is partially ambiguous. This can be expected to learn faster than the simple weak model examined below, which does not even exploit available parametric information prior to an ambiguity in a sentence. Thus, like the relation between the TLA and the TLA- above, our formalization does not do full justice to the system actually proposed as a model of human learning. A more serious simplification in what follows is that we do not attempt to model dynamic aspects of the STL (all variants of). First, since the STL sets parameters with certainty, the input becomes progressively less parametrically ambiguous as learning proceeds. This is just the Parametric Principle at work. Without it, the STL is but a poor shadow of itself. The computations for the dynamically disambiguating system, though along the same lines as those below, would be more complex because of the need to capture the influence of each learning event on the size of the task that remains. A second point to note is that the complexity of a learner's input is likely to increase as learning proceeds (as the child develops). This will affect the rate of expression of param-

eters (see below), giving typically lower expression rates at the outset of learning where this would be most advantageous. This variable expression rate adds considerable complication to the calculations; it may prove more practical to study this aspect of the STL's performance by means of computer simulation techniques than by mathematical modeling. In the meantime, we establish performance estimates here only for the non-dynamic variant, thus underestimating the dynamic STL's chance of encountering an unambiguous trigger, and hence the overall efficiency of STL learning. We will use the term STL- ("STL minus") in what follows to refer to the inflexible, weak version of the model without progressive disambiguation or variable expression in the input.

We now set out some basic calculations relevant to establishing how many sentences the STL- consumes on average before identifying the target grammar. A sentence that is ambiguous with respect to any parameter is discarded by the STL- for learning purposes. We need, therefore, to distinguish now between parametric ambiguity and parametric irrelevance. This was not important in Section 5.5 because results for the TLA- are unaffected by it. But it makes a big difference to the STL-. This is because ambiguity with respect to a parameter increases the number of analyses a sentence has, thereby complicating the pars-

er's task and disqualifying the sentence for learning, while irrelevance of a parameter to a sentence adds nothing to parsing complexity and does not impinge on what can be learned from the sentence about other parameter settings.

Consider, to start with, how learning will proceed if every parameter is relevant to every sentence (equivalently: every sentence expresses every parameter). Then all parameter values for the language will be established by a single sentence, the first unambiguous one encountered by the STL-. If every input sentence in the language is ambiguous with respect to at least a parameters, for $a > 0$, then learning is impossible. If, on the other hand, even one sentence in the input sample is parametrically *un*ambiguous, it will set all of the parameters and learning will be complete; no further inputs will be needed. The probability of encountering an unambiguous trigger in the input sample is thus the only factor of interest. For this we must define what we will call a degree of unambiguity, u, which is the proportion of sentences in the language that are fully unambiguous parametrically.

For comparability to the calculations for the TLA-, we will set u to the same value as in our second example at the end of Section 5.5.1. We supposed there that 10% of inputs are parametrically unambiguous, i.e.,

that $u = 1/10$. Note that the degree of parametric ambiguity of the other 90% of sentences is not pertinent to the outcome for the STL- under the assumptions in place here (i.e., no learning takes place in response to ambiguous inputs; all parameters are expressed by every sentence). The only influence the ambiguity level could have, under present assumptions, is on the failure rate of the current grammar G, as noted in Section 5.5.1 above; but this is masked here by the fact that grammar change is limited to a single shift to the correct grammar. Below, we show that the degree of ambiguity does enter in more interesting ways into the efficiency profile of the STL- under other more natural conditions.[30]

Given $u = 1/10$, and a representative sample of the language (as assumed throughout), the probability of encountering an unambiguous trigger is 1/10, and the average number of sentences to convergence is therefore $1/u = 1/(1/10) = 10/1 = 10$ (regardless of how many parameters need to be set). For smaller u this rises proportionately; for example, if there are only 15 unambiguous triggers on average in every 10,000 input sentences, then $u = 15/10,000$, and so $10,000/15 = 667$ inputs are needed on average for convergence.

Of course, this is not a psychologically realistic situation: as we have described it, nothing happens until suddenly the whole language is

learned in one event. It is also not a linguistically realistic situation, since it is not usual for natural language sentences to express every parameter relevant to the language. For example, whatever parameter governs the acceptability or nonacceptability of multiple (overt) A′-movement (e.g., WH-fronting) in a clause will be irrelevant to any sentence without overt A′-movement, even if the target language as a whole exhibits overt A′-movement.[31] Up to now our calculations have not distinguished the number of parameters relevant to a sentence from the number of parameters relevant to the language. For example, in Section 5.5.1. we assumed that $r = 25$ parameters were relevant to the target language, out of the $n = 30$ total parameters; and the value of a included not only parameters expressed ambiguously by a sentence, but also parameters not expressed by the sentence though relevant to the language as a whole (see footnote on page 450). Hence, an independent measure of parameters relevant to individual sentences was not needed. Now, however, we identify what we will call the expression rate for the target language, e, which is the number of parameters expressed by an input sentence. For simplicity, we will assume here (not realistically) that e is the same for all target sentences. The expression rate e contrasts with r, defined

above, which is the number of parameters that are relevant to the whole target language.

To illustrate the effect of e, let us temporarily set $a = 0$ (i.e., no ambiguously expressed parameters), and let us set $e = 6$, with $n = 30$ and $r = 25$ as before. Now, a sentence expresses only 6 of the 25 parameters that need to be set, and it expresses them all unambiguously. The learner has to encounter enough batches of six parameter values, possibly (in fact, probably) overlapping with each other, to make up the full set of 25 parameter values that have to be established. So convergence is gradual, not a one-step process as above. Let $P(w|t, r, e)$ be the probability that an input sentence provides unambiguous evidence concerning w new (i.e., as yet unset) parameters, given that the learner has already set t parameters (correctly), for some r and e as defined above. In what follows, we will take r and e as given, and refer to this simply as $P(w|t)$. Note that $P(w|t)$ is in effect the probability that the STL- will add w new parameter values (treelets) to its current grammar on a given input, given that it had previously adopted t parameters. We present the formula for $P(w|t)$ in the Appendix; the details are not important here. Note that w will vary between 0 and e, and t will increase over time from 0 to r. $P(0|t)$ corresponds to success of the current grammar

	$r = 15$	20	25	30
e				
1	50	72	95	120
5	9	13	18	23
10	4	6	8	11
15	1	3	5	6
20	-	1	3	4

Table 5.3. *Average number of inputs consumed by STL- before convergence. All input sentences unambiguous.*

G on the current input, so this does not need to be separately calculated here (see Appendix). The STL- is a memoryless system, like the TLA-, and so it can be modelled by means of a Markov chain (see Section 5.5), which in turn can be cast as a probability transition matrix consisting of the values of $P(w|t)$ as learning progresses. In the Appendix we work through the required matrix operations, which yield average numbers of inputs to convergence. Here, we present outcomes only.

Given a fully unambiguous target language with the values above, the necessary size of the input sample is 15. Table 5.3 gives expected sample sizes for other values of r, for different values of e, with no ambiguity. As expected, the fewer parameters expressed per sentence (the lower the value of e), the more input is needed to set them all. However, even for low expression rates the sample sizes are all quite small, and they are not

exponential in r. If we consider the mean cost per parameter (in terms of number of inputs needed) we see that it increases very slightly for higher numbers of parameters; thus, it is almost (though not quite!) linear in the number of parameters needing to be set (see Section 5.8). Note that even at its worst here, it is 4 inputs per parameter. (This contrasts with the exponential rise in Table 5.1, where for unambiguous input the cost per parameter increases many thousandfold from 15 parameters to 30 parameters.) This performance reflects the fact that the STL-, with its 'supergrammar' parsing ability, has no trouble identifying a grammar that licenses an input. Thus, unlike the TLA-, it does not have to discard potentially useful inputs just for inability to find a grammar that satisfies Greediness.

On the other hand, the STL- (unlike the TLA-) does discard sentences that fail its nonambiguity criterion. So now we must factor in the cost of ambiguity. Recall that for the TLA- we observed a potentially exponential cost of parametric ambiguity at Stage II which was offset by improvement at Stage I with respect to satisfaction of Greediness. For the STL- also we will see that there is a significant cost of ambiguity – not because ambiguity breeds errors in this case, but because the *avoid-*

ance of ambiguity effectively limits a conservative learner like the STL- to just a portion of the input.

It is unimportant for the STL- *how* ambiguous an ambiguous sentence is. All that matters is how many unambiguous sentences there are in the learner's sample, and how much information each one provides. Let us retain $e = 6$, and let us suppose, as we did earlier, that 90% of sentences contain some ambiguity, by which we now mean that for each sentence in that 90%, at least one of the $e = 6$ parameters it expresses is expressed ambiguously. Then $u = 1/10$, and only the unambiguous 10% of the input sentences are usable for learning. So: on average, an unambiguous input occurs once every 10 inputs and brings information about 6 parameters. Calculating on these assumptions (see $P'(w|t)$ in Appendix) we obtain an expected sample size of 150 (= ten times larger than with $u = 1$, all sentences unambiguous). If only 1% of sentences were unambiguous, the learner would need 1,500 inputs for convergence. Thus it's clear that as u declines, the number of sentences needed mounts in proportion. Let us consider this further. We know that if $u = 0$, learning is impossible for the STL-. Now we want to consider some other levels of unambiguity that could reasonably be expected for natural language, and compute the input sample sizes they call for.

In fact it is possible, given an average degree of parametric ambiguity and an expression rate, to calculate the probable distribution of unambiguous sentences in the target sample, rather than just stipulating a value for u as we have so far. What we establish is the chance that all the parameters expressed by some sentence happen to be expressed unambiguously, given the incidence of ambiguity in general. Recall that this is what really matters to STL- performance. Once the pattern of unambiguity has been established for the language in this way, we can determine from it (via an adaptation of $P(w|t)$; see Appendix) what the total sample size must be in order for all parameters to be correctly set. For brevity we will not present these calculations here. Table 5.4 relates expected sample size directly to the ambiguity and expression rates. Note that in Table 5.4, a denotes the average number of ambiguously expressed parameters in a sentence as a percentage of the number of parameters expressed by that sentence (in Section 5.5, a represented a constant number of ambiguous parameters in a sentence).

As before, the number of parameters to be set has relatively little effect. For this learner, the factors that dominate learning speed are the degree of ambiguity and the expression rate. When both ambiguity and expression rate are high, unambiguous inputs are very scarce. This is

		$r = 15$	20	25	30
e	a (%)				
1	20	62	90	119	150
	40	83	120	159	200
	60	124	180	238	300
	80	249	360	477	599
5	20	27	40	55	69
	40	115	171	230	292
	60	871	1,296	1,747	2,218
	80	27,885	41,472	55,895	70,983
10	20	34	54	76	98
	40	604	964	1,342	1,738
	60	34,848	55,578	77,397	100,193
	80	35,683,968	56,912,149	79,254,943	102,597,823
15	20	28	91	135	181
	40	2,127	6,794	10,136	13,545
	60	931,323	2,975,115	4,438,464	5,931,148
	80	over 30 billion	...	...	almost 200 billion
20	20		87	256	366
	40		27,351	80,601	115,415
	60		90,949,470	268,017,383	383,783,455
	80			... in the trillions ...	

Table 5.4. *Average number of inputs consumed by STL- before convergence. Ambiguous input.*

to be expected. Clearly, there is little chance of encountering a fully unambiguous sentence if every sentence expresses 24 parameters and the probability that each parameter is ambiguous is 99% (the probability would be $(1/100)^{24}$). As a result, for high e and a there are very few sentences that the learner can make use of, so the expected sample size is enormous, as can be seen in Table 5.4. The effect of e is interesting.

For ambiguous input it differs from the case of zero (or very low) ambiguity as in Table 5.3. As e increases (holding the degree of ambiguity constant), it is rarer for the learner to encounter fully unambiguous triggers. Also, as e increases, the average payoff per unambiguous sentence improves: more parameter settings are acquired. The results here make clear, however, that the improved yield per sentence does not compensate for the longer wait between usable sentences.

The impact of ambiguity is also very sharp. Increasing ambiguity raises the sample size needed into several hundreds of thousands of sentences, and then into billions at the top end of the numerical scales considered here. This certainly does not look promising as an improvement on TLA-type models. Evidently, it can be even less efficient to wait for an unambiguous trigger than to cope with the errors that result from guessing on ambiguous ones. However, that is not true across the board. Fortunately, the generally severe effect of ambiguity is absent at lower expression rates. We see that humble sentences which reveal only a few parameter values are the most useful for a learner seeking reliable information. This is important because expression rate is the one factor that might plausibly be low in real life learning. That a high degree of parametric ambiguity is characteristic of natural languages seems unde-

niable. And, though linguistic research might prove otherwise, it seems vain to hope that the number of syntactic parameters will be reduced to less than a dozen. So there's not much prospect of a breakthrough in learning efficiency due to a reduction of either a or r. But it does seem within the realms of possibility that the expression rate for natural languages is as low as half a dozen parameters per sentence, particularly at the early stages of learning where the threat of parametric ambiguity is probably at its greatest.[32] It is encouraging, therefore, to find that in the Structural Triggers framework, a reduction in the expression rate has a beneficial effect on learning speed.

5.9.2 General assessment of the STL

Our calculations have illustrated some important facts about conservative learning that relies on unambiguous structural triggers, at least for the rather undernourished version of the model that we have been able to formalize here. We see that the problem of finding a grammar that satisfies Greediness has dissolved. We see that performance is fairly constant even for large language domains with many parameters to be set. Of course, we have not demonstrated here that the needed sample size would not explode for domains with more than 30 parameters, but

there is nothing in the mathematics to suggest that it will. (Concerned readers may check for themselves.) On the other hand, the problem of extracting trustworthy information from partially ambiguous input looms even larger than it did in Section 5.5. We have found that to rely exclusively on unambiguous triggers is simply not feasible except at low expression rates. But at least it does appear to be feasible there. Though the worst case for the STL- is very bad indeed, there is also a more favorable region in which performance is efficient even for a sizeable number of parameters to be set. As long as there are sentences for which only a few parameters are relevant, the learner will have a good chance of encountering unambiguous triggers and converging rapidly on the target.

Whether this is so in real-life learning must be determined by empirical research. But as we have noted, the prospects seem tolerably good. It seems reasonable to suppose that learners, especially early learners, do not mostly encounter sentences that exhibit every syntactic phenomenon in the language, packed into 3 or 4 words or so. There are early child-directed sentences that contain negation, or overt WH-movement, or a subordinate clause, but probably few that involve them all, and those few the learner might ignore. We note for the record that the assumption

that early input expresses fewer parameters per sentence than input to older children or adults constitutes a weakening of assumption (ii) above, which posits homogenous input over time; that assumption has been convenient but is surely oversimplistic. In any case, even if the sentences directed to the child (or audible by the child) were independent of the child's stage of development, what the child is able to grasp and make use of almost certainly does increase with age.

These interesting possibilities are not captured by the computational results presented here, which have ignored completely the dynamic aspects of the STL. The latter will assist in pulling the learner down into the favorable zone in which the expression rate is low and the supply of unambiguous trigger sentences improves. Because the STL actually sets parameters, in accord with the Parametric Principle, every successful learning event decreases the number of parameters still to be set. To set up the mathematics for this we would need to change r to a variable whose value reflects the reduction, over time, in the number of parameters remaining to be set. Likewise, for a dynamic treatment we would need to replace e with a variable reflecting how many of the *parameters that remain to be set* are expressed by a sentence; only these parameters need to be expressed unambiguously in order for the input to qualify as

unambiguous and usable as a trigger. Thus if the input were uniform, the probability of encountering an unambiguous trigger would rise as learning proceeds. The major concern for the STL is therefore to establish that there are sufficient unambiguous triggers to get parameter setting started, so that the Parametric Principle can then begin to shift the learner down into more comfortable regions of parametric expression where unambiguous triggers are more plentiful. Note that an ideal environment for the STL- would be one in which the number of novel (= previously unset) parameters expressed per sentence is roughly constant (and quite low) across the learning period. This means that the expression rate e, as it has been defined here (including parameters already set and not yet set), would ideally start low, but can increase without detriment to learning if it keeps pace with the proportion of parameters the learner has already mastered. An interesting speculation that might be empirically investigated is that one of the reasons why second language learning is apparently more arduous for adults than for children is that adults may be exposed early on to complex sentences that express too many novel parameter values.

The working out of this and many other aspects of STL performance must await further research. The STL offers two potential advantages

for a learner, both due to its use of parameter value treelets to parse with. One is a virtually waste-free means of decoding sentences into the parameter values that license them, at stage I. The other is a test for parametric ambiguity. The value of the first seems unassailable. The value of the second is less evident, because recognizing ambiguity at stage I is only useful for purposes of discarding ambiguous stimuli before they engender errors at stage II, yet it appears that discarding them might be hardly more efficient than guessing at random. It is imaginable, then, that the optimal model would parse inputs as the STL does but respond to ambiguity as the TLA does (see Fodor, 1998c). On the other hand, the power of the treelets parsing procedure gives considerable scope for shaping up the STL system, in ways we have suggested and perhaps others too. Our best conjecture at present is that a more substantial analysis of it than we have been able to present here will show that it has considerable resilience to ambiguity. As we have noted, the main points yet to be undertaken are the formalization of the progressive disambiguation of triggers as parameters are set, an estimate of expression rates at the onset of learning, and an assessment of the parser's ability to extract as much reliable parametric information as possible from partially ambiguous inputs. Also, for the STL as for

the TLA, it is important to keep an eye on the uniformity of learning success, by considering worst-case outcomes rather than just the average outcomes studied here.

5.10 Implications for linguistic research

How much work does it take to acquire a human language? Less than if grammars must be composed from scratch by hypothesis formation. But more, it seems, than was anticipated in early conceptions of parameter setting. The recent research that has developed explicit models of the parameter setting process has exposed some serious threats to the central idea, which was that acquiring one of 2^n grammars can be reduced to acquiring the values of n parameters. When the process of setting the n parameters is implemented, the workload shows a strong tendency toward an exponential *expansion* back to a cost proportional to the 2^n grammars. Efforts to beat this reexpansion can be seen as the impetus behind a number of interesting proposals for learning algorithms, such as the genetic algorithm of Clark (1992) and the cue-based models of Lightfoot (1991) and Dresher (1999). The Structural Triggers model aims to achieve this while respecting plausible capacity limits on the psychological mechanisms involved: no complex linguistic reasoning

required (in contrast to some versions of cue-based learning), no multiple parsing of each stimulus (in contrast to genetic algorithms), storage of outcomes only by parameter not by grammar (also unlike genetic algorithms). Like many, but not all, other models it is also intended to be compatible with the absence of systematic negative evidence (direct or indirect), at most degree-1 input, and the severe restriction to learning from individual sentences without cross-sentence comparisons. Time will tell whether this can actually be pulled off. If so, language acquisition will be nearly effortless, as Chomsky proposed, though no longer a matter of just flipping switches.[33]

For linguistics there are tasks and conclusions, the most welcome conclusion being that – if our optimism is justified – a principles-and-parameters theory of Universal Grammar does indeed fit nicely within a psychologically plausible performance model. The tasks for linguistics are of a kind that have been being given increasing attention in recent work: constant rethinking of the actual set of parameters and the way it organizes the space of language phenomena (see, for example, Frank and Kapur, 1996); and identification of unambiguous triggers for setting them. The STL puts especially heavy demands on the latter, since a trigger for parameter p_i must be unambiguous not only with respec-

t to p_i but with respect to all other as-yet-unset parameters that are expressed by it. On the other hand, the STL does not demand that there be one simple, superficially identifiable cue property associated with each parameter value. The observable consequences of a parameter value may be extremely varied, as it interacts with other parameter values in derivations. As long as there is, somewhere in the derivation, a distinctive contribution from that parameter value, the STL will find it.

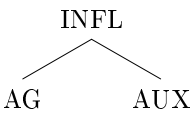
The one essential condition is that each parameter value be definable as (or inherently associated with) an ingredient of tree structures. Only this permits the efficient use of the parsing mechanism to decode sentences into their contributing parameter values. This is the most eccentric aspect of the STL among learning models, but it is very much in keeping with current linguistic theories, both transformational and monostratal.[34] In theories emphasizing phrase structure mechanisms, such as HPSG and TAG theory, there is little explicit talk of parameters, but it has always been natural to think of what differentiates one language from another as being a type of subtree, made available by UG, which is used in generating sentences in one language but not in another. How large a chunk of tree is involved, and whether it is underspecified

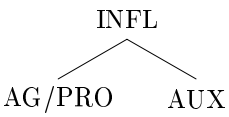
or defined in full detail, differs between theories. The elementary trees of a TAG grammar are quite large and richly endowed (Joshi, 1987); the rule schemata that define treelets in HPSG specify just a small handful of syntactic feature values (Pollard and Sag, 1994).

Government Binding theory was the original locus of parameters as switches, but has undergone an interesting transition from what we will call *freestanding* parameters to parameters as syntactic features, the limiting case of treelets. The earlier view is described by Clark and Roberts (1993). They write "A parameter can be thought of as a descriptive statement that may be either true or false of a given grammatical system" and give as example the statement "IP is a bounding node for Subjacency". This example also qualifies under a similar but more restricted characterization of parameters as points of variation in UG principles, or in the definitions that feed the principles. Another example is the proposal by Wexler and Manzini (1987) of five possible values for a variable in the definition of *governing category*, which feeds Binding Principles A and B. Another descriptive parametric statement is "Wh-movement occurs at Logical Form (and at S-structure)" in which the parentheses mark a parametric option (see Huang, 1981; Lasnik and Saito, 1984). This also counts as a prime example of a phase in which parameters of-

fered choices as to the linguistic level at which certain constraints must be met, which led into the current concept of strong and weak syntactic features in the Minimalism framework (see below).

Constraints, definitions, and levels of derivation are not building blocks of trees, so these early characterizations of parameters do not permit STL learning. Or at least, in order for them to do so there would need to be, in addition to an innate formulation of a parameter value, an innate specification of a treelet guaranteed to be present in sentence structures when and only when the parameter value in question is instantiated. But at about the same time, a conception of parameterization as directly concerned with the properties of elements in tree structures was emerging in the GB framework in work such as that by Rizzi (1982) and Hyams (1986, 1989) on the null subject parameter. It was proposed that a null subject is licensed by a pronominal feature of Infl. This clearly does lend itself to STL parameter setting. The positive value of the null subject parameter could be the treelet I^0[+pron], and the negative value would be I^0[-pron]. Hyams (1989) shows (in an older notation) the treelets in (181); (181)a is for English and (181)b is for Italian. In (181)b the agreement features (AG) of INFL constitute the ungoverned empty category PRO.

(181) a. 

b. 

These treelets for the null subject parameter make no mention of subjects, but that is appropriate since the claim is that the acceptability of null subjects is just one of a number of observable consequences of treelet b. as it interacts with UG principles in derivations.[35] This trend towards parameter values as structural elements (in the simplest case, a single feature specification) that are present in sentence derivations is embraced in the Minimalist Program of Chomsky (1995) where strong features of functional heads drive all overt movement operations.

Thus it seems that there is some significant convergence between the needs of efficient language learning and the conclusions of linguistic research. Linguistic theory need not be bent into unnatural forms to suit the learning device. We propose the following general characterization of parameters and triggers as most compatible with both linguistic theory and learnability concerns: A natural language parameter is the option of adopting a structural trigger into a grammar. A structural trigger is a partial tree that is made available by UG and is adopted into a learner's grammar if and only if it proves essential in parsing input sentences.

Exercises

Note: exercises designed for students with some mathematical expertise are marked (M); those which presuppose some knowledge of linguistics are marked (L). Some of the later questions are open-ended and could be the basis for research projects.

5.1 A nondeterministic learning device such as the TLA- may misset a parameter and have to reset it later.

(a) Using the background assumptions and numerical estimates of Section 5.5.1, compute R = how many times the TLA- resets the same parameter on average before convergence. Note: you will need to estimate how many parameters on average the TLA- resets each time it changes grammars.

(b) Give the general formula for R for any number r of relevant parameters and any degree a of average parametric ambiguity.

(c) Graph the value of R relative to a and relative to r.

5.2 The probability tree in Figure 5.1 gives the probability of attaining the target grammar in one step (one input sentence). Assume a learning domain of 4 languages (2 binary parameters).

(a) Construct a Markov state diagram (as in Appendix) depicting the transitions from one non-target state to another (including itself), as well as from a non-target state to the target.

(b) (M) Use the state diagram to construct a transition matrix, and calculate from it the fundamental matrix Q (as in the Appendix) and the average number of inputs needed for convergence by the TLA-.

For a readable presentation of Absorbing Markov Systems, see Waner and Costenoble (1996).

5.3 Assume as in Section 5.5.1 that 25 parameters are relevant to the target language, and that all parameters are expressed (though perhaps ambiguously) by all sentences.

(a) Calculate the average number of sentences required for convergence by the TLA- for some more varied distributions of parametric ambiguity. For example: 10% of sentences unambiguous, 60% of sentences ambiguous with respect to 8 parameters each, 30% of sentences ambiguous with respect to 20 parameters each.

(b) Discuss informally (or calculate if you can, using Markov

modeling; see Exercise 5.2 and Appendix), how these ambiguity distributions would affect the performance of the STL (weak or strong). For manageability, assume a low degree of parametric relevance and a low expression rate, e.g., $r = 5$, $e = 2$.

5.4 Suppose the prosodic contours of sentences provide learners with information about the surface bracketing (phrase structure) of every sentence (though not bracket labels). Assume also, as throughout this chapter, that the learner can recognize subject, verb and object, and other basic grammatical relations.

(a) (L) How much assistance would this be in setting the word order parameters?

(b) (L) Are there other syntactic parameters it would be helpful for?

(c) How much more efficient would word order learning be if learners had access to bracket labels also?

(d) To what extent would word order learning be facilitated if learners could rely on implicational universals (such as "If the pronominal object follows the verb, so does the nominal object" or "Languages with dominant VSO

order are prepositional, not postpositional") as proposed by Greenberg (1966), Hawkins (1983)?

For linguistic background on this question, read Nespor et al. (1996); for mathematical background, read Levy and Joshi (1978).

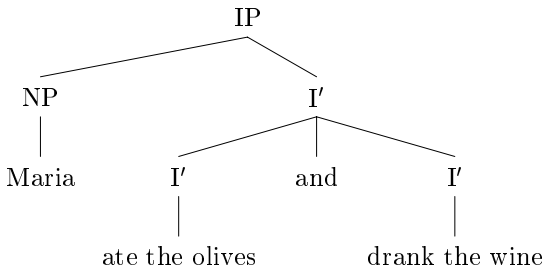
5.5 For the null subject parameter the depth-of-derivation problem does not arise: the fact that there is no overt subject in a sentence is a fact about its surface structure (as well as its underlying structure). But the string-to-structure problem can be seen in examples such as (i) and (ii).

(i) Maria mangiava le olive e beveva il vino.
Maria ate the olives and drank the wine.

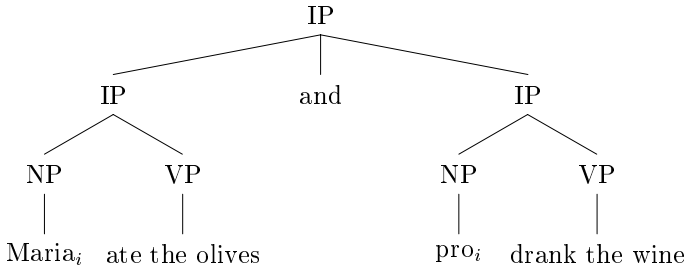
(ii) Maria ate the olives and drank the wine.

The Italian and English sentences mean essentially the same, but there is a structural ambiguity in the Italian (not in the English). The ambiguity in (i) is between a conjunction of verb phrases (technically, I-bars) with only one subject position in the sentence, as shown in (iii), and a conjunction of clauses with a null subject in the second one, as shown in (iv). In the latter case, assume the subjects of the two conjuncts are co-indexed.

(iii)



(iv)



(a) (L)Try to establish which analysis is normally imposed on the word string (i) by native speakers of Italian. What linguistic tests (syntactic, semantic or prosodic) distinguish the two structures?

(b) (L)Would a learner be able to recognize which structure the word string (i) or (ii) has in the target language?

(c) Suppose structure (iii) is what the human parser prefers to compute when it has the choice. If learners imposed this structure on (i) or (ii), what consequences would this have for the learning of Italian? of English?

(d) Suppose instead that structure (iv) is preferred by the

parser. What would the consequences be for the learning of Italian? of English?

(e) (L) Consider comparable questions for the word strings (v) and (vi), or any other examples you can construct of word strings that are ambiguous with respect to the null subject parameter (besides imperatives, and sentences in informal or diary register such as *Went home early. Forgot to buy chicken.*: see Haegeman, 1990).

(v) The policeman warned the woman that didn't have a valid driver's license.

(vi) Is John my friend (punctuation suppressed here)

For references on the null subject parameter, see footnote on page 449.

5.6 Consider a language L which has a variety of surface word orders, in a domain of languages such that each word order in L is the only permitted word order in some other language in the domain.

(a) Under what assumptions about the learning device would L and other languages in the domain be learnable?

(b) Are these assumptions plausible for human language learners?

(c) (L) Is there any reason to believe that the domain of natural languages has this character?

5.7 Natural language sentences are often structurally ambiguous even when generated by a fixed target grammar. For example, *He fed her dog biscuits* is ambiguous within English. For a constant degree of parametric ambiguity, estimate the effect of the degree of structural ambiguity within the target language on the performance of:

(a) the TLA or TLA-;

(b) a structural triggers learner with full parallel parsing capacity (a "strong STL");

(c) the weak STL (with a flagged serial parser).

(d) With *no* structural ambiguity, every target string is associated with exactly one target structure. It is presupposed for (a), (b) and (c), that structural ambiguity within the target is increased by increasing the number of associations between target strings and target structures, not by extending the set of target structures. Hence,

parametric ambiguity is not thereby increased. Reconsider the answers to (a), (b) and (c) without this assumption.

5.8 (M) A strict hill-climbing learner adopts a hypothesis only if it offers improvement over the previous one (e.g., in terms of syntactic parameters: only if the resulting grammar has more parameter values in common with the target grammar than the previous grammar did). It has been shown (Berwick and Niyogi, 1996) that the TLA is not in fact a strict hill-climber for arbitrary language domains.

(a) Are there domains in which the TLA does perform hill-climbing? If so, what are their characteristics? For example: Can they contain parametric ambiguity? Is smoothness essential?

(b) Is the SVC essential? Is the TLA- a hill-climber?

(c) Assume an error-driven learner which is a strict hill-climber (regardless, for now, of how this is implemented), operating in an ambiguous domain. How does the number of input sentences between grammar-changes vary as

a function of how close the learner is to success (i.e., how few parameters remain to be set)?

5.9 Start with the TLA. Add a recording device that keeps a running tally, for each parameter value, of how often a grammar containing it succeeds in parsing an input sentence.

(a) How accurate is this as a guide to whether a given parameter value is in the target grammar?

(b) How does its reliability vary with the degree of parametric ambiguity in the language domain?

(c) Would it be more useful or less useful to keep count, instead, of how often a parameter value being tested (in the sense of its being the one novel value in the grammar that the parser tries out after failure of the current grammar on some input sentence) is adopted?

(d) Given some such ranking of the frequency of success of each parameter value, how could the learner most profitably employ it in deciding which parameter to reset next? By a strategy of always switching to the highest-ranked parameter value not in the current grammar? By switching to a parameter value that is ranked much high-

er than the current value of that same parameter? By switching to higher ranked values with probabilities proportional to their relative advantage over others?

(e) Would any such strategy be useful for a parameter that is expressed only rarely in the target language? For example, the parameter that determines subject-auxiliary inversion in English imperatives when both subject and auxiliary are overt, which is quite rare; e.g., *Don't you touch that!*.

Relevant reading: Clark (1992), Kapur (1994).

5.10 (M) For the TLA-, with the numerical estimates of Section 5.5.1, compute the number of inputs necessary for the learner to identify the target with confidence greater than 50%, 75%, 90%, and 99%. For the relevant mathematics, see Chung (1979).

5.11 Discuss whether average rates of convergence, or convergence at high degrees of confidence, is the more appropriate criterion for evaluating models of natural language learning.

5.11 Appendix

5.11.1 Time to first success

The 'time to first success' of a series of independent success/failure events is distributed geometrically where the expected value of the number of trials (until the first success occurs) is simply the reciprocal of the probability of success.

For example we could ask: on average, how many rolls does it take to get a "6" on a six-sided die? The expected value, E, of the number of die rolls required is:

$$E(\#\text{rolls}) = \sum_{i=1}^{i=\infty} i \times P(\neg 6)^{i-1} \times P(6)$$

Expanding this out we get:

$$E(\#\text{rolls}) = 1P(6) + 2P(\neg 6)P(6) + 3P(\neg 6)P(\neg 6)P(6)\ldots$$

where $P(6)$ is the probability of rolling a 6 and $P(\neg 6)$ is the probability of rolling something other than a 6 Note that $P(\neg 6)P(\neg 6)P(6)$ is the probability that a 6 was rolled on the third roll (preceded by two failed rolls). Solving this series (see Chung, 1979 for a good description), we achieve the compact formula $1/P(6)$. Assuming a "fair" die (all outcomes equally probable) and plugging in, we get:

$$E(\#\text{rolls}) = 1/(1/6) = 6$$

In general,

$$E(\text{time to success}) = 1/P(\text{success})$$

5.11.2 Calculations for the STL-: Expected sample size to convergence

We present here one method for arriving at the expected size of the input sample consumed by the STL-. This approach is related to discussions in the literature by Niyogi and Berwick (1996) and elsewhere. [36]

Assuming all input sentences are unambiguous, $P(w|t)$ can be thought of in terms of the following "urn problem:"

There are 25 balls in an urn, of which t are black and $25-t$ are white. We draw 6 balls. What's the probability that we'll have drawn exactly w white balls? It is equal to the number of ways we can draw w balls from the $25-t$ white balls in the urn, times the number of ways we can draw $6-w$ black balls from the t black ones in the urn, divided by the total number of ways we can draw 6 balls from 25. That is:

$$P(w|t) = \frac{\begin{pmatrix} 25-t \\ w \end{pmatrix} \begin{pmatrix} t \\ 6-w \end{pmatrix}}{\begin{pmatrix} 25 \\ 6 \end{pmatrix}}$$

where

$$\begin{pmatrix} x \\ y \end{pmatrix}$$

denotes the number of ways of choosing y items from a collection of x items. In general:

$$P(w|t) = \frac{\begin{pmatrix} r-t \\ w \end{pmatrix} \begin{pmatrix} t \\ e-w \end{pmatrix}}{\begin{pmatrix} r \\ e \end{pmatrix}}$$

where e is the number chosen at each drawing and r is the number of balls in the urn.

Now we can ask the question that is really of interest. Start out with all r balls in the urn being white (corresponding to "unset" parameters). After drawing e balls, we paint them black ("set them") and return them

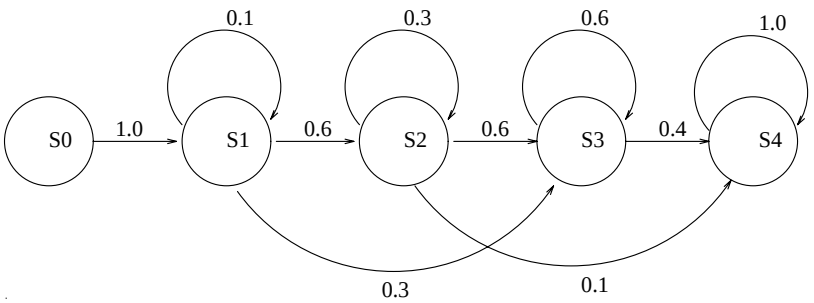


Fig. 5.3. A Markov state diagram for an STL- learner, where $r = 5$, $e = 2$ and no ambiguity.

to the urn. How many times do we need to draw before all the balls are black ("set")?

We can use the states of a Markov system modeling the STL- (or the urn scheme above) to depict the number of parameters that have already been set. The system starts in state S_0 and on the first input (with no ambiguity) moves to state S_e. It may stay in state S_e or move on the next step to state $S_{e+1}, S_{e+2}, S_{e+3} \ldots S_{2e}$. This is diagrammed in Figure 5.3.

The probabilities of making the state transitions are calculated by plugging appropriate values into $P(w|t, e, r)$. The results can also be presented in matrix form, as in Table 5.5.

If we assume that the STL- in Figure 5.3 has already set 3 parameters, then after receiving an input:

		To number of parameters set after an input			
		2	3	4	5
From	0	1.0	0	0	0
number of	2	0.1	0.6	0.3	0
parameters	3	0	0.3	0.6	0.1
currently	4	0	0	0.6	0.4
set	5	0	0	0	1.0

Table 5.5. *A sample transition matrix for an STL- learner, where* $r = 5$, $e = 2$ *and no ambiguity.*

(i) it may not be able to set any additional parameters, or

(ii) it may be able to set one additional parameter, or

(iii) it sets two new parameters.

then, the probability of the STL- changing from having set 3 parameters to 4 is $P(1|3, 2, 5) = 0.6$.

Since the STL- is error-driven, once it has set all relevant parameters (i.e. once it achieves state S_r) it stays in state S_r. Markov chains that have absorbing or "sink" states such as this are referred to as absorbing systems.

A well-known result from Markov chain theory is that the fundamental sub-matrix of a transition matrix yields the waiting time of an absorbing system. The fundamental matrix Q is defined by means of an inverse function applied to the identity matrix, I, (= 1 on the diagonal, 0 else-

1	1.1111	0.9524	2.2619
	1.1111	0.9524	2.2619
		1.4286	2.1429
			2.5000

Table 5.6. *The fundamental matrix Q for an STL learner, where* $r = 5$, $e = 2$ *and no ambiguity*

where) minus the (sub)matrix, N, that gives the transition probabilities of the non-sink state(s). That is:

$$Q = \text{inverse}(I - N)$$

Deriving N from the matrix in Table 5.5 above gives Q as in Table 5.6. The sum of the first row of Q yields the average number of inputs required for the STL- starting in state S_0 (no parameters have been set), to enter the absorbing state S_5 (all parameters have been set). That sum here equals 5.3254.

In order to deal with ambiguity, we will change the definition of $P(w|t, e, r)$. We refer here to the new formula as $P'(w|t, e, e', r)$. Also, we use u' to refer to the probability that *all* e parameters expressed by a sentence are expressed unambiguously, given that on average e' parameters are expressed unambiguously per sentence. The probabil-

ity that any one parameter is unambiguously expressed is e'/e. The probability that e parameters are unambiguously expressed is therefore $(e'/e)^e$. This is u'. We are now in a position to give the definition of $P'(w|t,e,e',r)$.

$$P'(w|t,e,e',r) = \begin{cases} (1-u') + P(w|t,e,r)u' & \text{if } w = 0 \\ P(w|t,e,r)u' & \text{otherwise} \end{cases}$$

For values of w other than 0, the probability of setting w new parameters is simply the probability that the sentence is usable for learning (i.e., all e parameters are unambiguously expressed $(= u')$) times the probability that w of those e parameters were previously unset $(= P(w|t,e,r))$. The probability of setting 0 parameters (i.e., $w = 0$) is the probability that not all e parameters are unambiguously expressed $(= 1 - u')$ plus the probability that even if the e parameters are unambiguously expressed $(= u')$ all of them had already been set $(= P(w = 0|t,e,r))$.

In calculating the transition matrix (as in Figure 5.3) for the STL-operating in an ambiguous domain we substitute P' for P.

Notes

1 We would like to thank Stefano Bertolo, Partha Niyogi and Cullen Schaffer for interesting discussion and technical advice on this work.

2 Chomsky (1981a) introduces the notion of parameters informally. Chomsky (1986) attributes to James Higginbotham the image of setting parametric switches. Atkinson (1987) offers conceptual clarification on the relation between triggering and other modes of learning. Clark (1994) observes: "Our folk-theoretic intuition, then, is that each parameter is associated with a trigger that *automatically* causes the learner to set a parameter to some value *immediately* upon exposure to it" (our emphases). Gibson and Wexler (1994) give a formal definition of *trigger* and of a learning procedure (see below) which they refer to as triggering but which does not reflect the familiar switch-setting metaphor.

3 Prosodic phrasing may be more directly perceptible, and could help learners establish phrase structure. See Mazuka (1996), Nespor et al. (1996), and papers in Morgan and Demuth (1996).

4 The null subject parameter has attracted attention because it presents the opposite problem concerning reliability of evidence: some sentences that clearly lack overt subjects should not trigger the null subject setting. See Hyams (1986, 1994b), Valian (1990), and references given there. Also see Exercise 5.5 at the end of this chapter.

5 In theories such as that of Kayne (1994) there is no variation in underlying word order, but the cross-language differences nevertheless show up at the surface level and must be determined by parameters controlling other aspects of derivations such as movement operations and/or what functional projections are present.

6 The relation between grammars and languages is apparently involved in application of the Subset Principle, which it is generally assumed that learners respect. If the language licensed by one grammar is a superset of the language licensed by another one, the former grammar should not be adopted; this is because of the impossibility of retreat from overgeneration without negative evidence. We will make the simplifying assumption here that learners can compute subset relationships solely by inspection of grammars, without needing to consult the languages they generate.

7 In cue-based models (Dresher and Kaye, 1990; Lightfoot, 1997), it is assumed that there is *another* property which is correlated with the true underlying trigger property, and *is* perceptible (cf. the artificial example of the /w/ cue for null subjects). We do not examine such models here. We doubt that simple superficial cues for syntactic parameters exist in all cases. We believe that cues are inherently related to the parameter values they trigger, and hence are abstract, and that the only feasible way to recognize them is by a mechanism such as the Structural Triggers Learner discussed in Sections 5.8–5.10 below. We also set aside here the argument that natural language learning is not very accurate, as evidenced by the fact that languages change (see Chapter 3 and references there). So we do not take inaccuracy as a goal of our model of how humans learn.

8 A parameter p_i is relevant to a sentence s (alternatively, the sentence expresses the parameter; see Clark, 1992 and Chapter 4, page 264) if and only if there is a combination of values of the other parameters which licenses s if p_i takes one of its values but not if it takes the other. For example, by this definition the parameter controlling underlying word

order in verb phrases is relevant to an SVO sentence (since with -V2, SVO is licensed by underlying VO but not OV) but it is not relevant to an SV sentence (since for every VO grammar that licenses it there is an otherwise identical OV grammar that does). If a sentence is licensed by two grammars due to irrelevance of a parameter, the two grammars assign it the same derivation and both are equally correct (or incorrect) for that sentence. If a parameter is irrelevant to *all* sentences in the target language, then there are two equally correct grammars for the language. (If n parameters are irrelevant, there are 2^n correct grammars.) By contrast, if a sentence is licensed by two grammars due to parametric *ambiguity*, the two grammars will normally assign it different derivations, and one of the two grammars will be wrong for the target language. The TLA does not detect either parametric ambiguity or parametric irrelevance in input sentences, but the distinction nevertheless has an effect on its performance, as will emerge below. Note: We set aside, throughout this chapter, the existence of structural ambiguity of a sentence with respect to a single grammar.

9 There are two ways in which a grammar could afford a parse of an input sentence and nevertheless be wrong for the language. The input might be parametrically ambiguous; or the grammar being tested might contain false values of parameters that are not relevant to this input but are relevant to other sentences in the target language; see footnote on page 450. In either case, a learner that took parse test success as sufficient cause for adopting a parameter value could thereby mis-set a parameter that was previously set correctly; later it would have to relearn the correct value. Typically when the TLA mis-sets parameters it does so

because of ambiguity, not irrelevance, because the SVC provides it with a relevance filter (see Fodor, 1999). The TLA adopts a parameter value only if it is positively helpful in parsing at least one input string. However, ambiguity and irrelevance interact in ways that cannot be controlled even by the SVC and Greediness: a parameter that seems relevant may be so only on the wrong reading of an ambiguity.

10 There is a conceptual shift that we should point out in case it may cause confusion. The shift is from checking the parametric properties of *input sentences* (by running them through a bank of property detectors, in the original instant-triggering model), to testing *grammars* (by seeing whether they succeed or fail on the current input, as in the TLA). The two approaches are presumably intertranslateable, but their different emphases reflect different views of what is a feasible implementation.

11 The Greediness constraint, on the other hand, demonstrably inhibits convergence in the TLA-. See Sakas and Fodor (1997) for discussion.

12 There is a joke in the Mafia genre whose target is not the Mafia but applied mathematics. A Mafia boss kidnaps a mathematician, locks him into a dank cellar, says "I'll be back in six months and you must then give me a formula to predict whether my horse will win at the races. If you don't, I'll shoot you." He leaves. He returns in six months, asks the mathematician for the formula, the mathematician doesn't have it, the Mafia man pulls out his gun. But the mathematician says "No, don't shoot me. I don't have the formula yet but I have made significant progress. I have it worked out for the case of the perfectly spherical horse." We are still at the stage of modeling the perfectly spherical language learner.

13 It may appear to be the Greediness constraint that requires the second parse: a learner without Greediness does not attempt to evaluate the appropriateness of grammars before adopting them, so it does not need to expend any effort on test-parsing candidate grammars. Berwick and Niyogi (1996) comment: "Further, not only does the greedy algorithm take more time, there is also a sense in which it requires more computation at any single step than a nongreedy one. Suppose the learner has received a sentence and is not able to analyze it in its current state. Greediness requires determining whether the new grammar can allow the learner to analyze the input or not" (p.614). However, it is reasonable to suppose that a learner, having picked a new grammar, would in any case then try to use it to parse the input, for the sake of understanding what was said. If so, then a non-greedy learner would also parse each novel sentence twice, once with the current grammar and once with a new one. The workload at each step is thus comparable (though the success rate may differ; see note on page 452). The way to save the labor of these double parses is to converge on the target grammar as soon as possible, so that over the course of learning, fewer sentences are encountered that are beyond the scope of the learner's current grammar. Input consumed and labor expended are thus related measures.

14 We are simplifying the discussion, throughout this section, by not distinguishing between parametric ambiguity of a sentence, and the irrelevance of a parameter to a sentence though it is relevant to the language as a whole. This sentence-level irrelevance (see footnote on page 451) is essentially folded into parametric ambiguity here. It is an

interesting phenomenon, which we address in Section 5.9, but it does not have a major impact on the outcomes for the TLA-.

15 In principle, a sentence that is indeterminate with respect to 8 parameters might be licensed by 256 grammars or by any lesser number between 256 and 2. In the minimum case, the two grammars would have opposite values for each of the indeterminate parameters. But assumption (iv) above entails the maximum, i.e., that every combination of values for those 8 parameters is compatible with the sentence. This imposes a form of 'smoothness' on the relation between languages and grammars (see discussion in Section 5.6 below). It excludes, for example, the possibility that two very divergent grammars, and only they, could license the same sentence. To capture other assumptions formally, it would be necessary to distinguish two measures of ambiguity: a_g, the number of grammars that can license a given input; and a_p, the number of parameters whose values are not determined by a given input, where $a_g = 1$ if $a_p = 0$, and otherwise $2 \leq a_g \leq 2^{a_p}$. Assumption (iv) sets $a_g = 2^{a_p}$. For the TLA-, as analyzed above, a_g is the more useful measure; for the Structural Triggers model we consider below, a_p is more significant. A third measure of ambiguity that is sometimes appropriate is related to assumption (iii). That is: a_l, the number of sentences or sentence types in common between different languages.

16 The probability that G is wrong and *ought* to be changed is $1 - (2^{n-r}/2^n) = 1 - (1/2^r)$, which is of course higher than the *recognized* need to change, which is $1 - 2^{a-r}$. The difference is $((2^a) - 1)/2^r$, and is a measure of the extent to which the learner is lulled into a false sense of achievement due to ambiguity of the input.

17 For simplicity, we have assumed that the outcome of parsing s with G' is independent of the outcome of parsing it with G, which allows the corresponding probabilities to be multiplied together. In fact, the probability of a successful parse by G' is affected by the failure of the prior parse by G. For details, see Sakas (in prep.).

18 For a domain of three parameters, of which two per sentence are ambiguous on average, the average number of inputs to convergence by this calculation is 16. For Gibson and Wexler's three word order parameters, Niyogi and Berwick (1996) showed that their implementation of the TLA needed approximately 100 sentences of child directed adult speech from the CHILDES data base (both English and German) for asymptotic (not average) convergence. However, these modest numbers should not distract attention from the high number of inputs needed for a more likely number of parameters for natural language. Success for small language domains is not a good prognosticator for whether performance will scale up appropriately.

19 Note that smoothness in this sense goes beyond the weaker phenomenon implied by assumption (iv) in Section 5.5.1 above (see footnote on page 454), which concerns a single sentence. Smoothness has to do with the whole language licensed by a grammar and the extent to which it overlaps the languages licensed by neighboring grammars.

20 Nyberg (1992) proposes a model in which the learner's route through the grammar space is recorded, so that it doesn't (normally) retest a failed grammar. The learner tests all grammars one parameter distant from its most recent failed grammar, and performs an evaluation to select one of them to shift to. The learner's path through the grammar space could be

reconstructed from a record of which parameter is reset at each step. Nyberg 's model achieves very rapid learning rates (linear in the number of parameters), but it does so at the cost of an unrealistically heavy parsing load. For 30 parameters, each novel input is parsed by thirty grammars, and the outcomes recorded. This is repeated on successive inputs until a clear winner has been identified to shift to, or a dead heat is declared and a random choice made. Incidentally, we note that Nyberg precodes the input into the set of parameter values each sentence is compatible with. This is more than an expository convenience. It imports assumption (iv) of Section 5.5.1 above. Suppose a sentence is licensed by two grammars, e.g., (for 3 parameters) by grammars 011 and 101. Then the sentence is coded as $**1$. This entails that the sentence is *also* licensed by grammars 001 and 111, i.e. by *all* combinations of parameter values compatible with this coding, of which there will be 2^i for i = the number of asterisks (i.e., $i = a_p$ in the terms of footnote on page 454). This rules out the possibility that the target parameter settings are quite unlike those of other grammars that succeed on some inputs, and it thereby facilitates this learner's systematic search through the grammar domain.

21 Fodor (1998a) notes that irrelevant parameters also inflate the size of the grammar pool; if they could be recognized as irrelevant and discarded, learning would be more efficient. However, Fodor wrongly implies that for the TLA the workload to convergence depends on the total number of parameters, whereas we have seen here in Section 5.5.1 that it depends only on the number of parameters relevant to the target language (because irrelevance of some parameters affects the number of correct

grammars in proportion to the total number of grammars). Thus, irrelevance of some parameters is helpful to the TLA, as it is for the STL model we present below.

22 For simplicity we make the common assumption here that exactly half of the remaining grammars are eliminated by each parameter that is set, but this would not be so if there were co-occurrence restrictions on parameter values or constraints on parameter accessibility such that a parameter does not freely admit of either value in combination with all other parameter values.

23 It is fair to raise the question whether checking through millions of grammars could conceivably be faster or less effortful than setting one parameter. There can be no formal proof here, because it depends in principle on the cost of unit operations of non-comparable types. But it seems very unlikely that the answer could be positive (for any plausible number of parameters) – unless the parameter-setting operation somehow smuggled in some individual grammar checking.

24 This does not rule out careful evaluation of the evidence before the irrevocable decision is made. It might even be combined with some sort of emergency retrieval of a previously discarded value if all else has failed, though we will not explore this possibility. Non-deterministic models like the TLA are of course able to freely return to past parameter values when necessary.

25 The learner cannot even usefully accumulate the results of parse tests over a succession of inputs. In principle it could count how often a given parameter value makes a contribution to licensing an input. But this interacts with how the *other* parameters are set in the grammars tested.

So for decisive evidence in favor of that parameter value the learner would need to store the outcomes of half a billion parse tests, one for each possible combination of values of the other parameters. A learner with statistical capabilities might try to estimate reliability based on partial test data (see Exercise 5.9 below), but even there the values of other parameters could skew the counts so it might take a very long time to distinguish between a parameter value that is correct and one that only seems to be so because of the company it keeps.

26 Future research should consider the possibility that the parser can pick up the analysis of a sentence again following an ambiguous substring. If it can perform accurately on the post-ambiguity fragment, despite not knowing what preceded it, it could provide additional reliable information for setting parameters.

27 Though both the strong and the weak STL are safe in that they do not commit errors, there are some circumstances in which an STL will fail to identify the correct grammar (see Bertolo et al, 1997a,b). For discussion, see Fodor (in press).

28 See Fodor (1998c) on the consequences for acquisition of parser failure due to lack of needed lexical entries, garden paths, and other processing problems.

29 Of course, the input itself (or that portion of it that the child is able to work with) may also be changing during that time. We conjecture that long sentences are typically less parametrically ambiguous than short sentences. So if it is assumed that the length and complexity of the sentences a learner is capable of processing increase with age then this will cause a decrease over time in the degree of input ambiguity. Stefano

Bertolo has kindly checked this hypothesis in the domain of eight parameters for word order and movement which he and his colleagues have established at MIT (see Bertolo et al., 1997a). He reports the following distribution of the number of distinct languages that an input sentence type belongs to, for all the distinct sentence types in the domain under 4 words, and a sample of the sentence types of 5 words.

Sentence length in words	Number of distinct types in length	Number of distinct languages		
		min	max	mean
2	2	10	38	24
3	29	2	24	10.3
4	95	2	16	7.2
5	72 studied	1	9	3.9

Note that in this domain there are no fully unambiguous sentences shorter than 5 words long which the WSTL could use for setting parameters. In the smaller domain studied by Gibson and Wexler, 1994, every language has at least one unambiguous trigger, ranging from 3 to 5 words.

30 The value u (degree of unambiguity) and the value a (degree of ambiguity) are to some extent independent, though they place outer bounds on each other. E.g., if $u > 0$, then $a < n$. Below, we give up the simplification of Section 5.5.1 (assumption (v)) that all sentences are equally ambiguous, and consider a distribution of parametric ambiguity such that a represents the mean (average) number of ambiguous parameters per sentence.

31 This is only true, of course, if multiple A′-movement does not correlate

reliably with some other phenomenon that can show up in non-movement constructions. In general, a parameter that controls phenomenon p can be expressed by a sentence not exhibiting p if the sentence exhibits another phenomenon controlled by the same parameter.

32 The work of Bertolo et al. (1997a) suggests that there might be a linguistic limit on how far e can be reduced. This would be so if some substantial number of parameters are essential to every well-formed sentence. Bertolo et al. note that if these necessarily relevant parameters are ambiguously expressed, a conservative learner could be hung up indefinitely. This is a major reason for the high degree of ambiguity noted in footnote on page 458; see discussion in Bertolo et al., 1997a. We have been taking it for granted here (a) that all parameters have an equal chance of being irrelevant to an input sentence, and (b) that all expressed parameters have an equal chance of being expressed unambiguously. But if relevance and ambiguity are in fact unevenly distributed, then a few troublesome parameters might indeed be expressed ambiguously in many sentences, and depress the incidence of unambiguous triggers. A possible solution to this, if linguistic research supports it, is to revise the presumed parameterization of the language facts, so as to translate ambiguous parameters in some contexts into irrelevant parameters. See Fodor (1998c) for discussion.

33 See Fodor (1998c) for a proposal under which learners merely parse sentences (once each) for comprehension, as adults do but with the supergrammar, and the target grammar emerges as residue, with no additional procedures.

34 Only Optimality Theory seems incompatible with parameter values as

treelets since it rejects parameter values altogether. Cross-language variation is captured in terms of different priority orderings of a UG-provided set of constraints on structure. Constraints are negative, quite unlike the positive ingredients of sentence derivations needed by the STL. See Tesar and Smolensky (1998) on acquisition in an OT framework.

35 If null subjects were characterized directly, in a different linguistic theory, that could be equally compatible with STL learning. For instance, the positive value of the null subject parameter might be a treelet consisting of the feature specification [+NULL] on an XP (NP or DP, depending on the theory) in characteristic subject position (again, at choice of the theory), or perhaps an XP marked [CASE NOM], etc. Or, in a framework in which subjects are generated outside VP, the parametric treelets might offer a choice of VP or S as root categories or selected complements, etc. The learning theory does not dictate these details.

36 There is at least one other approach which can be used for establishing these results. It utilizes dynamic programming to compute the following recurrence relation: that the expected sample size required, on average, to set n parameters can be determined from the size required to set $n - i$, $0 < i < e$ parameters together with the probability of setting i additional parameters given that $n - i$ have been set.

References

Allen, A. O. (1990). *Probability, Statistics and Queueing Theory with Computer Applications*, (Academic Press, Boston).

Atkinson, M. (1987). Mechanisms for language acquisition: Learning, parameter-setting and triggering, *First Language* **7**: 3–30.

Atkinson, M. (1992). *Children's Syntax: An Introduction to Principles and Parameters Theory*, (Blackwell, Oxford).

Atkinson, M. (1996). Now hang on a minute: some reflections on emerging orthodoxies, in H. Clahsen (ed.) *Generative Perspectives in Language Acquisition*, (John Benjamin, Amsterdam).

Baker, M. (1988). *Incorporation; A Theory of Grammatical-Function Changing*, (Chicago University Press, Chicago).

Barber, C. L. (1976). *Early Modern English*, (Andre Deutsch, London).

Benveniste, E. (1968). Mutations of Linguistic Categories, in W. Lehmann and Y. Malkiel (eds.), *Directions for Historical Linguistics: A Symposium*, (University of Texas Press, Austin).

Bertolo, S. (1995a). *Learnability Properties of Parametric Models for Natural Language Acquisition*. Unpublished doctoral dissertation, Rutgers University.

Bertolo, S. (1995b). Maturation and Learnability in Parametric Systems, *Language Acquisition* **4**(4): 277–318.

Bertolo, S, Broihier, K., Gibson, E. and Wexler, K. (1997a) Cue-based learners in parametric language systems: application of general results to a recently proposed learning algorithm based on unambiguous 'superparsing', in M.G. Shafto and P. Langley (eds.), *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*,

(Lawrence Erlbaum Associates, Mahwah NJ).

Bertolo, S, Broihier, K., Gibson, E. and Wexler, K. (1997b) *Characterizing learnability conditions for cue-based learners in parametric language systems* (Technical Report D-97-02 Deutsches Forschungszentrum für Künstliche Intelligenz GmbH).

Berwick, R. and Niyogi, P. (1996). Learning from triggers, *Linguistic Inquiry* **27**: 605–622.

Bickerton, D. (1991). *Language and Species*, (Chicago University Press, Chicago).

Blumer, A., Ehrenfeucht, A., Haussler, D. and Warmuth, M. K. (1987). Occam's razor, *Information Processing Letters* **24**: 377–380.

Bohannon, J. N. and Stanowicz, L. (1988). The issue of negative evidence: adult responses to children's language errors, *Developmental Psychology* **24**: 684–689.

Borer, H. (1984). *Parametric Syntax*, (Foris, Dordrecht).

Borsley, R. D. and Roberts, I. (1996). *The Syntax of the Celtic Languages*, (Cambridge University Press, Cambridge).

Bourciez, E. E. J. (1930). *Elements de linguistique romane*, (Klincksieck, Paris).

Braine, M. D. S. (1971). On two types of models of the internalization of grammar, in D.I. Slobin (ed.), *The Ontogenesis of Grammar*. (Academic Press, New York).

Brill, E. & Kapur, S. (1993). *An information-theoretic solution to parameter setting*. Unpublished manuscript, University of Pennsylvania.

Brown, R. (1977). Introduction, in C.E. Snow and C.A. Ferguson (eds.), *Talking to Children: Language Input and Acquisition*, (Cambridge

University Press, Cambridge).

Brown, R. and Hanlon, C. (1970). Derivational complexity and order of acquisition in child speech, in J.R. Hayes (ed.), *Cognition and the Development of Language*, (Wiley, New York).

Chaitin, G. (1987). *Algorithmic Information Theory*, (Cambridge University Press, Cambridge).

Cheng, L. (1991). *On the Typology of WH Questions.* Unpublished doctoral dissertation, MIT.

Chien, Y.-C. and Wexler, K. (1987). *A comparison between Chinese-speaking and English-speaking children's acquisition of reflexives and pronouns.* Unpublished manuscript, MIT.

Chien, Y.-C. and Wexler, K. (1990). Children's knowledge of locality conditions in binding as evidence for the modularity of syntax and pragmatics, *Language Acquisition* **1**: 225–295.

Chomsky, N. (1956). Three models for the description of language, *I. R. E. Transactions of Information Theory* **IT-2**: 113–124.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*, (MIT Press, Cambridge).

Chomsky, N. (1981a). *Lectures on Government and Binding*, (Foris Publications, Dordrecht).

Chomsky, N. (1981b). Principles and parameters in syntactic theory, in N. Hornstein and D. Lightfoot (eds.), *Explanation in Linguistics: The Logical Problem of Language Acquisition*, (Longman, London).

Chomsky, N. (1986). *Knowledge of Language: Its Nature, Origin and Use*, (Praeger, New York).

Chomsky, N. (1987). *On the nature, use and acquisition of language. Kyoto*

Lectures. Unpublished manuscript, MIT.

Chomsky, N. (1988). *Language and Problems of Knowledge: The Managua Lectures*, (MIT Press, Cambridge, MA).

Chomsky, N. (1995). *The Minimalist Program*, (MIT Press, Cambridge, MA).

Chomsky, N. and Lasnik, H. (1995). The Theory of Principles and Parameters, in Chomsky, N. *The Minimalist Program*, (MIT Press, Cambridge, MA).

Chung, K.L. (1979). *Elementary Probability Theory with Stochastic Processes*, (Springer-Verlag, New York).

Cinque, G. (1989). Parameter setting in 'instantaneous' and real-time acquisition, *Behavioral and Brain Sciences* **12**: 336.

Cinque, G. (1997). *Adverbs and the universal hierarchy of functional projections*. Unpublished manuscript, University of Venice.

Clahsen, H. (ed.) (1996). *Generative Perspectives on Language Acquisition*, (John Benjamin, Amsterdam).

Clark, R. (1990). *Papers on Learnability and Natural Selection* (Technical Reports on Formal and Computational Linguistics 1). University of Geneva.

Clark, R. (1992). The Selection of Syntactic Knowledge, *Language Acquisition* **2**(2): 83–149.

Clark, R. (1993). Finitude, Boundedness and Complexity. Learnability and the Study of First Language Acquisition, in B. Lust et al. (eds.), *Syntactic Theory and First Language Acquisition:Cross Linguistic Perspectives*, (Lawrence Erlbaum Associates, Mahwah, NJ).

Clark, R. (1994). *Kolmogorov complexity and the information content of parameters* (IRCS Report 94-17). University of Pennsylvania.

Clark, R. (1996). *Complexity and the induction of Tree Adjoining Grammars* (IRCS Report 96-14). University of Pennsylvania.

Clark, R. and Roberts, I. (1993). A Computational Model of Language Learnability and Language Change, *Linguistic Inquiry* **24**: 299–345.

Clark, R & I. Roberts (1997) *Complexity is the Engine of Variation*, Unpublished manuscript, University of Pennsylvania and Wales.

Corbett, G. (1983). *Hierarchies, Targets and Controllers: Agreement Patterns in Slavic*, (Croon Helm, London).

Cover, T. M. and Thomas, J. A. (1991). *Elements of Information Theory*, (Wiley, New York).

Croft, W. (1994). *Language Typology*, (Cambridge University Press, Cambridge).

Demetras, M.J., Post, K.N. and Snow, C.E. (1986). Feedback to first language learners: the role of repetitions and clarification questions, *Journal of Child Language* **13**: 275–292.

Denison, D. (1985). The origins of periphrastic *do*: Ellegård and Visser reconsidered, in R. Eaton et al. (eds.), *Papers from the 4th International Conference on English Historical Linguistics*, (John Benjamins, Amsterdam).

Déprez , V. and Pierce, A. (1993). Negation and functional projections in early grammar, *Linguistic Inquiry* **24**: 25–67.

Dresher, E. (1999). Charting the learning path: Cues to parameter setting, *Linguistic Inquiry*. **30**(1): 27–67.

Dresher, E and Kaye, J.D. (1990). A computational learning model for metrical phonology, *Cognition* **34**: 137–195.

Ellegård, A. (1953). *The auxiliary do, the establishment and regulation of its*

use in English, (Almquist and Wiksell, Stockholm).

Fintel, K. von (1995). The Formal Semantics of Grammaticalization, *Proceedings of NELS 25*: 175–189.

Fodor, J. D. (1998a). Unambiguous triggers, *Linguistic Inquiry*. **29**(1): 1–36.

Fodor, J. D. (1998b). Learning to parse?, *Journal of Psycholinguistic Research*. **27**(2): 285–319.

Fodor, J. D. (1998c). Parsing to learn, *Journal of Psycholinguistic Research*. **27**(3): 339–374.

Fodor, J. D. (1999). Learnability theory: Triggers for parsing with, in E.C. Klein and G. Martohardjono (eds.), *The Development of Second Language Grammars: A Generative Approach*.

Fodor, J.D. (in press). Learnability theory: Decoding trigger sentences. In R.C. Schwarz (ed.) *Linguistics, Cognitive Science, and Childhood Language Disorders*, (Lawrence Erlbaum Associates, Hillsdale NJ).

Frank, R. (1992). *Syntactic Locality and Tree Adjoining Grammar: Grammatical, Acquisition and Processing Perspectives* (IRCS Technical Report 92-47). University of Pennsylvania.

Gallistel, C. R. (1990). *The Organization of Learning*, (MIT Press, Cambridge, MA).

Gibson, E. (1991). *A Computational Theory of Human Linguistic Processing: Memory Limitations and Processing Breakdown*. Unpublished doctoral dissertation, Carnegie Mellon University.

Gibson, E. and Wexler, K. (1994). Triggers, *Linguistic Inquiry* **25**(3): 407–454.

Giorgi, A. and F. Pianesi (1998) *The Syntax of Tense*, (Oxford University Press, Oxford).

Gleitman, L. and E. Wanner (1982). The state of the state of the art, in E. Wanner and L. Gleitman (eds.), *Language Acquisition: The State of the Art*, (Cambridge University Press, Cambridge).

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

Gray, D. (1985). *The Oxford book of late medieval verse and prose*, (Oxford University Press, Oxford).

Greenberg, J. (1966). Some Universals of Grammar with Particular Reference to the Order of Meaningful Elements, in J. Greenberg (ed.), *Universals of Language* (2nd edition), (MIT Press, Cambridge, MA).

Grimshaw, J. and Pinker, S. (1989). Positive and negative evidence in language acquisition, *Behavioral and Brain Sciences* **12**: 341–342.

Grimshaw, J. and Rosen, S. (1990). Knowledge and obedience: the developmental status of the Binding Theory, *Linguistic Inquiry* **21**: 187–222.

Grodzinsky, J. and Reinhart, T. (1993). The innateness of binding and coreference, *Linguistic Inquiry* **24**: 69–101.

Haegeman, L. (1990). Understood subjects in English diaries, *Multilingua*, **9**: 157–199.

Haegeman, L. (1994). *Introduction to Government and Binding Theory* (2nd edition), (Blackwell, Oxford).

Halle, M. and Marantz, A. (1993). Distributed Morphology and the pieces of inflection, in K. Hale and S.J. Keyser (eds.), *The view from Building 20: Essays in Honor of Sylvain Bromberger*, (MIT Press, Cambridge, MA).

Hamming, R. W. (1991). *The Art of Probability for Engineers and Scientists*, (Addison Wesley, Redwood City, CA).

Harris, T. and Wexler, K. (1996). The optional-infinitive stage in Child English: evidence from negation, in H. Clahsen (ed.) *Generative Perspectives in Language Acquisition*, (John Benjamin, Amsterdam).

Hawkins, J. A. (1983). *Word Order Universals*, (Academic Press, New York).

Heine, B., U. Claudi and F. Hünnemeyer (1991). *Grammaticalization: A Conceptual Framework*, (Chicago University Press, Chicago).

Hirsh-Pasek, K. Treiman, R. and Schneiderman, M. (1984). Brown and Hanlon revisited: mothers' sensitivity to ungrammatical forms, *Journal of Child Language* **11**: 81–88.

Holmberg, A. and C. Platzack (1991). On the role of inflection in scandinavian syntax, in W. Abraham (ed.), *Issues in Germanic Syntax*, (Mouton de Gruyter, Berlin).

Hopcroft, J. and J. Ullman (1979). *Introduction to Automata Theory, Languages, and Computation*, (Addison Wesley, Reading, MA).

Huang, C.-T. J. (1981). Move *WH* in a language without *WH* movement. *The Linguistic Review* **1**: 369–416.

Huang, C.-T. J. and Tang, C.-C. J. (1991). The local nature of the long-distance reflexive in Chinese, in J. Koster and E. Reuland (eds.), *Long-Distance Anaphora*, (Cambridge University Press, Cambridge).

Hyams, N. (1986). *Language Acquisition and the Theory of Parameters*, (Reidel, Dordrecht).

Hyams, N. (1994a). VP, null arguments and Comp projections, in T. Hoekstra and B. D. Schwartz (eds.), *Language Acquisition: Studies in Generative Grammar*, (John Benjamin, Amsterdam).

Hyams, N. (1994b). Null subjects in child language and the implications of cross-linguistic variation, in B. Lust et al. (eds.), *Syntactic Theory and*

First Language Acquisition:Cross Linguistic Perspectives, (Lawrence Erlbaum Associates, Mahwah, NJ).

Hyams, N. (1996). The underspecification of functional categories in early grammar, in H. Clahsen (ed.), *Generative Perspectives in Language Acquisition*, (John Benjamin, Amsterdam).

Jain, S., D. Osherson, J.S. Royer, and Sharma, A. (1999). *Systems That Learn – 2nd Edition. An Introduction to Learning Theory*, (MIT Press, Cambridge).

Jakubowicz, C. (1984). On markedness and binding principles, *Proceedings of the Northeastern Linguistics Society* **14**: 154–182.

Jespersen, O. (1954). *A Modern English Grammar on Historical Principles*, (George Allen & Unwin, London).

Joshi, A. (1987). An introduction to Tree Adjoining Grammars, in A. Manaster-Ramer (ed.), *Mathematics of Language*, (John Benjamins, Amsterdam)

Kapur, S. (1991). *Computational Learning of Languages* (Computer Science Technical Report 91-1234). Cornell University.

Kapur, S. (1994). Some applications of formal learning theory results to natural language acquisition, in B. Lust et al. (eds.), *Syntactic Theory and First Language Acquisition:Cross Linguistic Perspectives*, (Lawrence Erlbaum Associates, Mahwah, NJ).

Kapur, S. and R. Clark (1996). The automatic construction of a symbolic parser via statistical techniques, in J. Klavans and P. Resnik (eds.), *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, (MIT Press, Cambridge, MA).

Kapur, S., B. Lust, W. Harbert and Martohardjono, G. (1993). Universal

Grammar and Learnability Theory: the case of binding domains and the 'Subset Principle,' in E. Reuland and W. Abraham (eds.), *Knowledge and Language: Volume 1: From Orwell's Problem to Plato's Problem*, (Kluwer, Dordrecht).

Kauffman, S. (1995). *At Home in the Universe*, (Viking, London).

Katada, F. (1991). The LF representation of anaphors, *Linguistic Inquiry* **22**: 287–313.

Kayne, R. (1994). *The Antisymmetry of Syntax*, (MIT Press, Cambridge, MA).

Kemenade, A. van (1987). *Syntactic Case and Morphological Case in the History of English*, (Foris, Dordrecht).

Kiparsky, P. (1994). The Indo-European Origins of Germanic Syntax, in A. Battye and I. Roberts (eds.), *Clause Structure and Change*, (Oxford University Press, Oxford).

Koopman, H. and Sportiche, D. (1991). The position of subjects, *Lingua* **85**: 211–258.

Koster, J. and Reuland, E. (eds), (1991). *Long-Distance Anaphora*, (Cambridge University Press, Cambridge).

Kroch, A. S. (1989). Reflexes of grammar in patterns of language change, *Journal of Language Variation and Change* **1**: 199–244.

Labov, W. (1972). *Language in the Inner City*, (University of Pennsylvania Press, Philadelphia).

Labov, W. (1994). *Principles of linguistic change: Volume One, Internal Factors*, (Blackwell, Oxford).

Lasnik, H. and Saito, M. (1984). On the nature of proper government, *Linguistic Inquiry* **15.2**: 235–289.

Lehmann, C. (1985). Grammaticalization: synchronic variation and diachronic change, *Lingua e Stile* **20.3**: 303–18.

Lema, J. and M.-L- Rivero (1991). Types of Verbal Movement in Old Spanish: Modals, Futures and Perfects, *Probus* **3.3**: 237–78.

Levelt, W. M. (1975). *What Became of LAD?*, (Peter de Ridder, Lisse).

Levy, L. S. and Joshi, A. K. (1978). Skeletal structural descriptions, *Information and Control* **39.2**: 192–211.

Li, M. and Vitanyi, P. (1993). *An introduction to Kolmogorov complexity and its applications*, (Springer-Verlag, New York).

Lightfoot, D. (1979). *Principles of Diachronic Syntax*, (Cambridge University Press, Cambridge).

Lightfoot, D. (1989). The child's trigger experience: Degree-0 learnability, *Behavioral and Brain Sciences* **12**(2): 321–334.

Lightfoot, D. (1991). *How to Set Parameters*, (MIT Press, Cambridge, MA).

Lightfoot, D. (1997). Catastrophic change and learning theory, *Lingua* **100**(2): 171–192.

Lockwood, W. (1964). *An Introduction to modern Faroese*, (Foeroyar Skulabøkgrunnur, Tørshavn).

Lyell, C. (1830-33). *Principles of Geology*, (Murray, London).

Longobardi, G. (1994). Reference and proper names: A theory of N-movement in Syntax and Logical Form, *Linguistic Inquiry* **24**: 299-345.

MacLaughlin, D. (1995). Language acquisition and the Subset Principle, *The Linguistic Review* **12**: 143–191.

McCarthy, J. and A. Prince (1986). *Prosodic Morphology*. Unpublished manuscript, Brandeis University.

McCloskey, J. (1996). On the scope of verb movement in Irish, *Natural Language and Linguistic Theory* **14**: 47-104.

McNeill, D. (1966). Developmental psycholinguistics, in F. Smith and G. A. Miller (eds.), *The Genesis of Language: A Psycholinguistic Approach*, (MIT Press, Cambridge, MA).

Manzini, R. (1995). ¿From Merge and Move to Form Dependency, *UCL Working Papers in Linguistics* **8**: 323–346.

Manzini, R. and Wexler, K. (1987). Parameters, Binding Theory and Learnability, *Linguistic Inquiry* **18**: 413–444.

Marcus, G. (1993). Negative Evidence in Language Acquisition, *Cognition* **46**(1): 53–85.

Matsuoka, K. (1997). Binding conditions in young children's grammar: interpretation of pronouns inside conjoined NPs, *Language Acquisition* **6**: 37–48.

May, R. (1985). *Logical Form: Its Structure and Derivation* (MIT Press, Cambridge, MA).

Mazuka, R. (1996). Can a grammatical parameter be set before the first word? Prosodic contributions to early setting of a grammatical parameter, in J.L. Morgan and K. Demuth (eds.), *Signal to Syntax*, (Lawrence Erlbaum Associates, Hillsdales, NJ).

Morgan, J. L. (1989). Learnability considerations and the nature of trigger experiences in language acquisition, *Behavioral and Brain Sciences* **12**: 352–53.

Morgan, J. L. and Travis, L. (1989). Limits on negative information, *Journal of Child Language* **16**: 531–52.

Nespor, M., Guasti, M.T. and Christophe, A. (1996). Selecting word order:

The rhythmic activation principle, in U. Kleinhenz (ed.), *Interfaces in Phonology*, (Akademie Verlag, Berlin).

Newport, E., L. Gleitman and Gleitman, H. (1977). Mother, I'd rather do it myself: some effects and non-effects of maternal speech style, in C.E. Snow and C.A. Ferguson (eds.), *Talking to Children: Language Input and Acquisition*, (Cambridge University Press, Cambridge).

Newson, M. (1990). *Questions of Form and Learnability in Binding Theory.* Unpublished doctoral dissertation, University of Essex.

Niyogi, P and Berwick, R.C. (1995) The Logical Problem of Language Change, *MIT A.I. Memo no. 1516.*

Niyogi, P and Berwick R.C. (1996) A language learning model for finite parameter spaces, *Cognition* 61: 161–193.

Nyberg, E. (1992). *A non-deterministic success-driven model of parameter setting in language acquisition.* Unpublished doctoral dissertation, Carnegie Mellon University.

Papadimitriou, C. (1994). *Computational Complexity.* (Addison-Wesley, Reading, MA).

Partee, B., A. ter Meulen and R. Wall (1990). *Mathematical Methods in Linguistics*, (Kluwer Academic Publishers, Dordrecht).

Penner, S. (1987). Parental responses to grammatical and ungrammatical child utterances, *Child Development* **58**: 376–384.

Phillips, C. (1996). *Order and Structure.* Unpublished doctoral dissertation, MIT.

Pica, P. (1987). On the nature of the reflexivization cycle, *Proceedings of the Northeastern Linguistics Society* **17**: 483–499.

Pinker, S. (1989). *Learnability and Cognition: The Acquisition of Argument*

Structure, (MIT Press, Cambridge, MA).

Pintzuk, S. (1991). *Phrase Structures in Competition: Variation and Change in Old English Word Order*. Unpublished doctoral dissertation, University of Pennsylvania.

Platzack, C. (1987). The Scandinavian Languages and the Null-Subject Parameter, *Natural Language and Linguistic Theory* **5**: 377–401.

Platzack, C. (1994). The Loss of Verb Second in French and English, in A. Battye and I. Roberts (eds.), *Clause Structure and Change*, (Oxford University Press, Oxford).

Platzack, C. (1996). The initial hypothesis of syntax: a minimalist perspective on language acquisition and attrition, in H. Clahsen (ed.), *Generative Perspectives in Language Acquisition*, (John Benjamin, Amsterdam).

Poeppel, D. and Wexler, K. (1993). The Full Competence Hypothesis of clause structure in early German, *Language* **69**: 1–33.

Pollard, C. and Sag, I.A. (1994). *Head-driven Phrase Structure Grammar*, (University of Chicago Press, Chicago).

Pollock, J. Y. (1989). Verb Movement, UG and the structure of IP, *Linguistic Inquiry* **20**: 365-424.

Radford, A. (1990). *Syntactic Theory and the Acquisition of English Syntax: the Nature of early Child Grammar in English*, (Blackwell, Oxford).

Rambow, O. (1994). *Formal and Computational Aspects of Natural Language Syntax* (IRCS Technical Report 94-08). University of Pennsylvania.

Resnik, P. (1993). *Selection and Information: A Class-Based Approach to Lexical Relationships* (IRCS Technical Report 93-42). University of Pennsylvania.

Rizzi, L. (1982). *Issues in Italian Syntax*, (Foris, Dordrecht).

Rizzi, L. (1994). *Some Notes on Linguistic Theory and Language Development: The Case of Root Infinitives.* Unpublished manuscript, University of Geneva.

Rizzi, L. (1997). *The Fine Structure of the Left Periphery*, in L. Haegeman (ed.) *Elements of Grammar*, (Kluwer, Dordrecht).

Roberts, I. (1985). Agreement Parameters and the Development of English Modal Auxiliaries, *Natural Language and Linguistic Theory* **3**: 21–58.

Roberts, I. (1992). *Verbs and Diachronic Syntax*, (Kluwer, Dordrecht).

Roberts, I. (1998). Verb-movement and markedness, in M. Degraff and A. Pierce (eds.), *Language Acquisition, Creoles and Language Change*, (MIT Press, Cambridge, MA).

Roberts, I, and Roussou, A. (1997). *Interface Interpretation.* Unpublished manuscript, Universities of Wales and Stuttgart.

Rogers, H. (1967). *Theory of Recursive Functions and Effective Computability*, (MIT Press, Cambridge, MA).

Ruhlen, M. (1987). *A Guide to the World's Languages. Volume 1: Classification*, (Edward Arnold, London).

Saffran, J., Aslin, R. and Newport, E. (1996). Statistical learning by 8-month-old infants, *Science* **274**: 1926–1928.

Safir, K. (1987). Comments on Wexler and Manzini, in T. Roeper and E. Williams (eds.), *Parameter Setting*, (Reidel, Dordrecht).

Safir, K. (1996). Semantic atoms of anaphors, *Natural Language and Linguistic Theory* **14**: 545–589.

Sakas, W. G. (in prep.). *Ambiguity and the Computational Feasibility of Syntax Acquisition.* Unpublished manuscript. The Graduate Center of

the City University of New York.

Sakas, W. G. and Fodor, J. D. (1997). *Triggering, hill-climbing and the conservative learner: Can a stochastic trigger-based learner afford Greediness as a constraint?*. Unpublished manuscript, Graduate Center CUNY; paper presented at First Annual Conference on Computational Psycholinguistics (CPL97), Berkeley, CA.

Shlonsky, U. (1997). *Clause structure and word order in Hebrew and Arabic. An essay in comparative Semitic Syntax*, (Oxford University Press, Oxford).

Sigurjónsdóttir, S. and Hyams, N. (1992). Reflexivization and logorophicity: evidence from the acquisition of Icelandic, *Language Acquisition* **2**: 359–413.

Snow, C. E. and Ferguson, C.A. (eds), (1977). *Talking to Children: Language Input and Acquisition*, (Cambridge University Press, Cambridge).

Sportiche, D. (1981). Bounding nodes in French, *The Linguistic Review* **1**: 219–246.

Tekavcic, P. (1980). *Grammatica Storica dell'Italiano*, (Il Mulino, Bologna).

Tesar, B. and P. Smolensky (1998). Learnability in Optimality Theory, *Linguistic Inquiry* **29**: 229–268.

Thráinsson, H. (1991). Long distance reflexives and the typology of NPs, in J. Koster and E. Reuland (eds.), *Long-Distance Anaphora*, (Cambridge University Press, Cambridge).

trau:ag Traugott, E. and B. Heine (1991). *Approaches to Grammaticalization* (Typological Studies in Language 19), (John Benjamins, Amsterdam).

Travis, L. (1984). *Parameters and Effects of Word Order Variation*. Unpublished doctoral dissertation, MIT.

Valian, V. (1990). Logical and psychological constraints on the acquisition of syntax, in L. Frazier and J. de Villiers (eds.), *Language Processing and Language Acquisition*, (Kluwer, Dordrecht).

Valiant, L.G. (1984). A Theory of the Learnable, *Communications of the ACM* **27**: 1134–1142.

Vikner, S. (1995). V-to-I movement and inflection for person in all tenses, *Working Papers in Scandinavian Syntax* **55**: 1-27.

Vincent, N. (1991). Latin and the Romance Languages, in K. Börjars and N. Vincent (eds.), *Complement Structures in the Languages of Europe*, EUROTYP Working Papers III,1.

Visser, F. T. (1963). *An Historical Syntax of the English Language*, (Brill, Leiden).

Waner, S. and Costenoble, S. R. (1996). *Finite Mathematics Applied to the Real World*, (Harper Collins, New York).

Watkins, C. (1963). Preliminaries to the Historical and Comparative Analysis of the Syntax of the Old Irish Verb, *Celtica* **6**: 11–49.

Watkins, C. (1964). Preliminaries to the Reconstruction of the Indo-European Sentence Structure, in H.G. Lunt (ed.), *Proceedings of the Ninth International Congress of Linguistics*, (Mouton, The Hague).

Wexler, K. (1993). The Subset Principle is an intensional principle, in E. Reuland and W. Abraham (eds.), *Knowledge and Language: Volume 1: From Orwell's Problem to Plato's Problem*, (Kluwer, Dordrecht).

Wexler, K. (1994). Finiteness and Head Movement in Early Child Grammars, in D. Lightfoot and N. Hornstein (eds.), *Verb Movement*, (Cambridge University Press, Cambridge).

Wexler, K. and Manzini, R. (1987). Parameters and learnability in Binding

Theory, in T. Roeper and E. Williams (eds.), *Parameter Setting*, (Reidel, Dordrecht).

Wexler, K. and Culicover, P. (1980). *Formal Principles of Language Acquisition*, (MIT Press, Cambridge, MA).

Winston, P. (1992). *Artificial Intelligence* (3rd edition), (Addison Wesley, Reading, MA).

Zwart, J.W. (1994) *Dutch Syntax*. Unpublished doctoral dissertation, University of Groningen.