In: *Proceedings of CoNLL-2000 and LLL-2000*, pages 61–66, Lisbon, Portugal, 2000.

Modeling the Effect of Cross-Language Ambiguity on Human Syntax Acquisition

William Gregory Sakas
Department of Computer Science
Hunter College and The Graduate Center
City University of New York
New York, NY 10021
sakas@hunter.cuny.edu

Abstract

A computational framework is presented which is used to model the process by which human language learners acquire the syntactic component of their native language. The focus is *feasibility — is acquisition possible within a reasonable amount of time and/or with a reasonable amount of work?* The approach abstracts away from specific linguistic descriptions in order to make a 'broad-stroke' prediction of an acquisition model's behavior by formalizing factors that contribute to cross-linguistic ambiguity. Discussion centers around an application to Fodor's *Structural Trigger's Learner (STL)* (1998)[1] and concludes with the proposal that successful computational modeling requires a parallel psycholinguistic investigation of the distribution of ambiguity across the domain of human languages.

1 Principles and Parameters

Chomsky (1981) (and elsewhere) has proposed that all natural languages share the same innate universal *principles* (Universal Grammar — UG) and differ only with respect to the settings of a finite number of *parameters*. The syntactic component of a grammar in the *principles and parameters* (henceforth P&P) framework, is simply a collection of parameter values — one value per parameter. (Standardly, two values are available per parameter.) The set of human grammars is the set of all possible combinations of parameter values (and lexicon).

The P&P framework was motivated to a large degree by psycholinguistic data demonstrating the extreme efficiency of human language acquisition. Children acquire the grammar of their native language at an early age — generally accepted to be in the neighborhood of five years old. In the P&P framework, even if the linguistic theory delineates over a billion possible grammars, a learner need only determine the correct 30 values that correspond to the grammar that generates the sentences of the target language.[2] given that a successful syntactic theory must provide for an efficient acquisition mechanism, and since, prima facie, parameter values seem transparently learnable, it is not surprising that parameters have been incorporated into current generative syntactic theories. However, the exact process of parameter setting has been studied only recently (e.g. Clark (1992), Gibson and Wexler (1994), Yang (1999), Briscoe (2000), among others) and although it has proved linguistically fruitful to construct parametric analyses, it turns out to be surprisingly difficult to construct a workable model of parameter-value acquisition.

1.1 Parametric Ambiguity

A sentence is *parametrically ambiguous* if it is licensed by two or more distinct combinations of parameter values. Ambiguity is a natural enemy of efficient language acquisition. The problem is that, due to ambiguity, there does not exist a one-to-one correspondence between the linear 'word-order' surface strings of the input sample and the correct parameter values that generate the target language. Clearly, if ev-

[1]The STL is an acquisition model in the principles and parameters paradigm. The results presented here are not intended to forward an argument for or against the model, or for that matter, for or against the principles and parameters paradigm. Rather, the results are presented to point out (the possibly not-so- earth-shattering observation) that the acquisition mechanism can be extremely sensitive to ambiguity.

[2]30 binary parameters entails approximately a billion grammars ($2^{30} = 1,073,741,824$.)

ery sentence of the target language triggers one and only one set of parameter values (i.e. every sentence is completely *un*ambiguous) and the learner, upon encountering an input, can determine what those values are, the parameter setting process is truly transparent. Unfortunately, not all natural languages sentences are so distinctively earmarked by their parametric signatures. However, if there exists *some* degree of parametric unambiguity in a learner's input sample, a learner can set parameters by: 1) decoding the parametric signature of an input sentence, 2) determining if ambiguity exists, and 3) using the input to guide parameter setting only in the case that the sentence is parametrically unambiguous. The motto of such a learner is: *Don't learn from ambiguous input* and learning efficiency can be measured by the number of sentences the learner has to wait for usable, unambiguous inputs to occur in the input stream.[3]

2 The Structural Triggers Learner

One recent model of human syntax acquisition, ***The Structural Triggers Learner (STL)*** (Fodor, 1998), employs the human parsing mechanism to determine if an input is parametrically ambiguous. Parameter values are viewed as bits of tree structure (treelets). When the learner's current grammar is insufficient to parse the current input sentence, the treelets may be utilized during the parsing process in the same way as any natural language grammar would be applied; no unusual parsing activity is necessary. The treelets are adopted as part of the learner's current grammar hypothesis when: 1) they are required for a successful parse of the current input sentence and 2) the sentence is unambiguous. The STL thus learns only from fully unambiguous sentences.[4]

[3]Of course, the extent to which such unambiguous sentences exist in the domain of human languages is an empirical issue. This is an important open research question which is the focus of a recent research endeavor here at CUNY. Our approach involves tagging a large, cross-linguistic set of child-directed sentences, drawn from the CHILDES database, with each sentence's parametric signature. By cross-tabulating the shared parameter values against different languages, the study should shed some light as to the shape of ambiguity in input samples typically encountered by children.

[4]This is actually the strategy employed by just one of several different STL variants, some of which are designed to manage domains in which unambiguous sentences are rare or nonexistent.

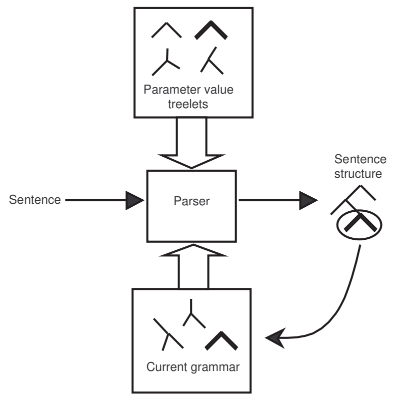

Figure 1: An example of how the STL acquires new parameter values.

See Figure 1.

3 The Feasibility of the STL

The number of input sentences consumed by the STL before convergence on the target grammar can be derived from a relatively straightforward Markov analysis. Importantly, the formulation most useful to analyze performance does not require states which represent the grammars of the parameter space (contra Niyogi and Berwick (1996)). Instead, each state of the system depicts the number of parameters that have been set, t, and the state transitions represent the probability that the STL will adopt some number of new parameter values, w, on the basis of the current state and whatever usable parametric information is revealed by the current input sentence. See Figure 2.

The following factors (described in detail below) determine the transition probabilities:

- the number of parameters that have been set (t)
- the number of relevant parameters (r)
- the expression rate (e)
- the effective expression rate (e')

Not all parameters are relevant parameters. Irrelevant parameters control properties of phenomena not present in the target language, such as clitic order in a language without clitics. For

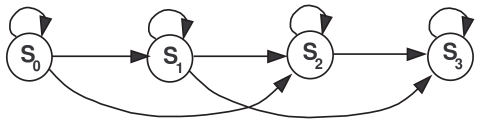

Figure 2: A transition diagram for the STL performing in a parameter space of three parameters. Nodes represent the current number of parameters that have been correctly set. Arcs indicate a change in the number that are correctly set. In this diagram, after each input is consumed, 0, 1 or 2 new parameters may be set. Once the learner enters state 3, it has converged on the target.

our purposes, the number of *relevant parameters*, r, is the total number of parameters that need to be set in order to license all and only the sentences of the target language.

Of the parameters relevant to the target language as a whole, only some will be relevant to any given sentence. A sentence *expresses* those parameters for which a specific value is required in order to build a parse tree, i.e. those parameters which are essential to the sentence's structural description. For instance, if a sentence does not have a relative clause, it will not express parameters that concern only relative clauses; if it is a declarative sentence, it won't express the properties peculiar to questions; and so on. The expression rate, e, for a language, is the average number of parameters expressed by its input sentences. Suppose that each sentence, on average, is ambiguous with respect to a of the parameters it expresses. The *effective expression rate*, e', is the mean proportion of expressed parameters that are expressed unambiguously (i.e. $e' = (e - a)/e$). It will also be useful to consider $a' = (1 - e')$.

3.1 Derivation of a Transition Probability Function

To present the derivation of the probability that the system will change from an arbitrary state S_t to state S_{t+w}, $(0 \leq w \leq e)$ it is useful to set ambiguity aside for a moment. In order to set all r parameters, the STL has to encounter enough batches of e parameter values, possibly overlapping with each other, to make up the full set of r parameter values that have to be established.

Let $H(w|t, r, e)$ be the probability that an arbitrary input sentence expresses w new (i.e. as yet unset) parameters, out of the e parameters expressed, given that the learner has already set t parameters (correctly), for a domain in which there are r total parameters that need to be set.

This is a specification of the hypergeometric distribution and is given in Equation 1.

$$H(w|t,r,e)=\frac{\binom{r-t}{w}\binom{t}{e-w}}{\binom{r}{e}} \tag{1}$$

Now, to deal with ambiguity, the effective rate of expression, e', is brought into play. Recall that e' is the proportion of expressed parameters that are expressed unambiguously. It follows that the probability that any single parameter is expressed unambiguously is also e' and the probability that all of the expressed, but as yet unset parameters are expressed unambiguously is e'^w. That is, the probability that an input is effectively unambiguous and hence usable for learning is equal to e'^w.

$$P(S_t \rightarrow S_{t+w})=\begin{cases} H(w|t,r,e)e'^w, & 0<w\leq e \\ H(0|t,r,e)+\sum\limits_{i=1}^{min(e,r-t)} H(i|t,r,e)(1-e'^i), & w=0 \end{cases} \tag{2}$$

Equation (2) can be used to calculate the probability of any possible transition of the Markov system that models STL performance. One method to determine the number of sentences expected to be consumed by the STL is to sum the number of sentences consumed in each state. Let $E(S_i)$ represent the expected number of sentences that will be consumed in state S_i. E is given by the following recurrence relation:[5]

$$\begin{aligned} E(S_0) &= 1/e'^e \\ E(S_n) &= 1/(1-P(S_n \rightarrow S_n)) \sum_{i=n-e}^{n} P(S_i \rightarrow S_n)(S_i) \end{aligned} \tag{3}$$

The expected total is simply:

$$E_{tot}=E(S_0)+\sum_{i=e}^{r-1} E(S_i) \tag{4}$$

which is equal to the expected number to be consumed before any parameters have been set

[5]The functional E is derived from basic properties of Markov Chains. See Taylor and Karlin (1994) for a general derivation.

e	$a'(\%)$	$r = 15$	$r = 20$	$r = 25$	$r = 30$
1	20	62	90	119	150
	40	83	120	159	200
	60	124	180	238	300
	80	249	360	477	599
5	20	15	22	29	36
	40	34	46	59	73
	60	144	176	210	245
	80	3,300	3,466	3,666	3,891
10	20	14	18	23	28
	40	174	187	203	221
	60	9,560	9,621	9,727	9,878
	80	9,765,731	9,766,375	9,768,376	9,772,740
15	20	28	32	37	41
	40	2,127	2,136	2,153	2,180
	60	931,323	931,352	931,479	931,822
	80	-	... over 10 billion ...		
20	20	-	87	91	95
	40	-	27,351	27,361	27,383
	60	-	90,949,470	90,949,504	90,949,728
	80	-	... in the trillions ...		

Table 1: Average number of inputs consumed by the waiting-STL before convergence. Fixed rate of expression.

($= E(S_0)$) plus the number expected to be consumed after the first successful learning event (at which point the learner will be in state S_e) summed with the number of sentences expected to be consumed in every other state up to the state just before the target is attained (S_{r-1}). E_{tot} can be tractably calculated using dynamic programming.

3.2 Some Results

Table 1 presents numerical results derived by fixing different values of r, e, and e'. In order to make assessments of performance across different situations in terms of increasing rates of ambiguity, a percentage measure of ambiguity, a', is employed which is directly derived from e': $a' = 1 - e'$, and is presented in Table 1 as a percent (the proportion is multiplied by 100).

Notice that the number of parameters to be set (r) has relatively little effect on convergence time. What dominates learning speed is ambiguity and expression rates. When a' and e are both high, the STL is consuming an unreasonable number of input sentences. However, the problem is not intrinsic to the STL model of acquisition. Rather, the problem is due to a too rigid restriction present in the current formulation of the input sample. By relaxing the restriction, the expected performance of the STL improves dramatically. But first, it is informative to discuss why the framework, as presented so far, leads to the prediction that the STL will consume an extremely large number of sentences at rates of ambiguity and expression approaching natural language.

By far the greatest amount of damage inflicted by ambiguity occurs at the very earliest stages of learning. This is because before any learning takes place, the STL must wait for the occurrence of a sentence that is fully unambiguous. Such sentences are bound to be extremely rare if the expression rate and the degree of ambiguity is high. For instance, a sentence with 20 out of 20 parameters unambiguous will virtually never occur if parameters are ambiguous on average 99% of the time (the probability would be $(1/100)^{20}$).

After learning gets underway, STL performance improves tremendously; the generally damaging effect of ambiguity is mitigated. Every successful learning event decreases the number of parameters still to be set. Hence, the expression rate of unset parameters decreases as learning proceeds. And to be usable by the STL, the only parameters that need to be expressed unambiguously are those that have not yet been set. For example, if 19 parameters have already been set and $e = r = 20$ as in the example above, the probability of encountering a usable sentence in the case that parameters are ambiguous on average 99% of the time and the input sample consists of sentences expressing 20 parameters, is only $(1/100)^1 = 1/100$. This can be derived by plugging into Equation (2): $w = 1$, $t = 19$, $e = 20$, and $r = 20$ which is equal to: $H(1|19,20,20)(1/100)^1 = (1)(1/100)$.

Clearly, the probability of seeing usable inputs increases rapidly as the number of parameters that are set increases. All that is needed, therefore, is to get parameter setting started, so that the learner can be quickly be pulled down into more comfortable regions of parametric expression. Once parameter setting is underway, the STL is extremely efficient.

3.3 Distributed Expression Rate

So far e has been conveniently taken to be fixed across all sentences of the target language. In which case, when $e = 10$, the learner will have to wait for a sentence with exactly 10 unambiguously expressed parameters in order to get started on learning, and as discussed above, it can be expected that this will be a very long wait. However, if one takes the value of e to be uniformly distributed (rather than fixed) then the learner will encounter some sentences which express fewer than 10 parameters, and which are correspondingly more likely to be fully unambiguous and hence usable for learning.

In fact, any distribution of e can be incorporated into the framework presented so far. Let $D_I(x)$ denote the probability distribution of expression of the input sample. That is, the probability that an arbitrarily chosen sentence from the input sample I expresses x parameters. For example, if D_I imposes a uniform distribution, then $D_I(x) = 1/e_{max}$ where every sentence expresses at least 1 parameter and e_{max} is the maximum number of parameters expressed by any sentence. Given D_I, a new transition probability $P'(S_t \to S_{t+w}) = P'(w|t, r, emax, e')$ can be formulated as:

$$P'(w|t,r,e_{max},e') = \sum_{i=w}^{e_{max}} D_I(i) P(w|t,r,i,e') \qquad (5)$$

where P is defined in (2) above and e_{max} represents the maximum number of parameters that a sentence may express instead of a fixed number for all sentences.

To see why Equation (5) is valid, consider that to set w new parameters at least w must be expressed in the current input sentence. Also usable, are sentences that express more parameters ($w+1$, $w+2$, $w+3$,..., e_{max}). Thus, the probability of setting w new parameters is simply the sum of the probabilities that a sentence expressing a number of parameters, i, from w to e_{max}, is encountered by the STL ($= D_I(i)$), times the probability that the STL can set w additional parameters given that i are expressed. By replacing P with P' in Equation 3 and modifying the derivation of the base case,[6] the total expected number of sentences that will be consumed by the STL given a distribution of expression $D_I(x)$ can be calculated.

Table 2 presents numerical results derived by fixing r and a' and allowing e to vary uniformly from 0 to e_{max}. As in Table 1, a percentage measure of ambiguity, a', is employed.

The results displayed in the table indicate a striking decrease in the number of sentences that the that the STL can be expected to consume compared to those obtained with a fixed expression rate in place. As a rough comparison, with the ambiguity rate (a') at 80%: when

e_{max}	$a'(\%)$	$r = 15$	$r = 20$	$r = 25$	$r = 30$
1	20	124	180	238	300
	40	166	240	318	399
	60	249	360	477	599
	80	498	720	954	1198
5	20	28	40	53	67
	40	46	65	86	107
	60	89	124	161	199
	80	235	324	417	511
10	20	17	24	32	40
	40	40	55	70	86
	60	102	137	173	209
	80	323	430	538	648
15	20	15	21	27	33
	40	46	62	77	93
	60	134	176	219	262
	80	447	586	726	868
20	20	-	20	26	32
	40	-	74	91	109
	60	-	223	275	327
	80	-	755	931	1108

Table 2: Average number of inputs consumed by the STL before convergence. Uniformly distributed rate of expression.

e varies uniformly from 0 to 10, the STL requires 430 sentences (from Table 2); when e is fixed at 5, the number of sentences required is 3,466 (from Table 1).

4 Discussion

Although presented as a feasibility analysis of parameter-setting — specifically of STL performance, it should be clear that the relevant factors e', e, r, etc. can be applied to shape an abstract input domain for almost any learning strategy. This is important because questions of a model's feasibility have proved difficult to answer in spaces of a linguistically plausible size. Recent attempts necessarily rely on severely small, highly circumscribed language domains (e.g. Gibson and Wexler (1994), among others).

These studies frequently involve the construction of an idealized language sample which is (at best) an accurate subset of sentences that a child might hear. A simulated learner is let loose on the input space and results consist of either the structure of the grammar(s) acquired or the specific circumstances under which the learner succeeds or fails to attain the target. Without question, this research agenda is valuable and can bring to light interesting characteristics of the acquisition process. (cf. Gibson and Wexler's (1994) argument for certain default parameter values based on the potential success or failure of verb-second acquisition in a three-parameter domain. And, for a different perspective, Elman et al.'s (1996) discussions of

[6] $E(S_0) = 1/(1 - P'(S_0 \to S_0))$

English part-of-speech and past-tense morphology acquisition in a connectionist framework.)

I stress that my point here is not to give a full accounting of STL performance. Substantial work has been completed towards this end (Sakas (2000), Sakas and Fodor (In press.)), as well as development of a similar framework to other models (See Sakas and Demner-Fushman (In prep.) for an application to Gibson and Wexler's *Triggering Learning Algorithm*). Rather, I intend to put forth the conjecture that syntax acquisition is extremely sensitive to the distribution of ambiguity, and, given this extreme sensitivity, suggest that simulation studies need to be conducted in conjunction with a broader analysis which abstracts away from whatever linguistic particulars are necessary to bring about the sentences required to build the input sample that feeds the simulated learner.

Ultimately, whether a particular acquisition model is successful is an empirical issue and depends on the exact conditions under which the model performs well and the extent to which those favorable conditions are in line with the facts of human language. Thus, I believe a three-fold approach to validate a computational model of acquisition is warranted. First, an abstract analysis (similar to the one presented here) should be constructed that can be used to uncover a model's *sweet spots* — where the *shape of ambiguity* is favorable to learning performance. Second, a computational psycholinguistic study should be undertaken to see if the model's sweet spots are in line with the distribution of ambiguity in natural language. And finally, a simulation should be carried out.

Obviously, this a huge proposal requiring years of person-hours and coordinated planning among researchers with diverse skills. But if computational modeling is going to eventually lay claim to a model which accurately mirrors the human process of language acquisition, years of fine grinding are necessary.

Acknowledgments. This work was supported in part by PSC-CUNY-30 Research Grant 61595-00-30. The three-fold approach is at the root of a project we have begun at The City University of New York. Much thanks to my collaborators Janet Dean Fodor and Virginia Teller for many useful discussions and input, as well as to two anonymous reviewers for their helpful comments.

References

E.J. Briscoe. 2000. Grammatical Acquisition: Inductive Bias and Coevolution of Language and the Language Acquisition Device. *Language*, 76(2).

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

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

J.L. Elman, E. Bates, M.A. Johnson, A. Karmiloff-Smith, D. Parisi, and K. Plunkett. *Rethinking Innateness: A Connectionist Perspective on Development.* MIT Press, Cambridge, MA.

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

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

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

W.G. Sakas. 2000. *Ambiguity and the Computational Feasibility of Syntax Acquisition.* Unpublished Ph.D. dissertation, City University of New York.

W.G. Sakas and D. Demner-Fushman. In Prep. Simulating Parameter Setting Performance in Domains with a Large Number of Parameters: A Hybrid Approach.

W.G. Sakas and J.D. Fodor. In Press. The Structural Triggers Learner. In Stefano Bertolo, editor, *Parametric Linguistics and Learnability: A Self-contained Tutorial for Linguists.* Cambridge University Press, Cambridge,UK.

H.M. Taylor and S. Karlin. 1994. *An Introduction to Stochastic Modeling.* Academic Press, San Diego, CA.

C.D. Yang. 1999. A selectionist theory of language acquisition. In *Proceedings of the 37th Annual Meeting of the ACL.* Association for Computational Linguistics.