Implementing the Subset Principle in Syntax Acquisition: Lattice-Based Models

Janet Dean Fodor (jfodor@gc.cuny.edu)
Ph.D. Program in Linguistics, The Graduate Center, City University of New York
365 Fifth Avenue, New York, NY 10016

William Gregory Sakas (sakas@hunter.cuny.edu)
Department of Computer Science, Ph.D. Programs in Linguistics and Computer Science
Hunter College and The Graduate Center, City University of New York
North Building 1008; 695 Park Avenue, New York, NY 10021

Arthur Hoskey (ahoskey1@gc.cuny.edu)
Ph.D. Program in Computer Science, The Graduate Center, City University of New York
365 Fifth Avenue, New York, NY 10016

Abstract

Language learners with insufficient access to negative evidence about what is not in their target language must rely on the Subset Principle (SP), or some other similar conservative learning strategy, in order to avoid overgeneration. Recent attempts to incorporate such a strategy into psychologically realistic models of syntax acquisition have revealed two severe problems: SP application appears to demand computational resources that exceed those of children; and SP causes *under*generation failures if learning is incremental. We present a representational scheme for the domain of grammars which can alleviate both problems, and we report simulation data showing how it can best be employed in a learning model.

Implementation Challenges

Because language learners receive little information about non-sentences of their target language (Marcus, 1993), any model of natural language syntax acquisition must have some means of avoiding or minimizing overgeneration. The learning mechanism (LM) must be conservative: other things being equal, the grammar hypothesis it adopts must be the one that fits the positive input most snugly. This general principle has been cast as the *Subset Principle* in studies of syntax acquisition grounded in generative linguistics (Berwick, 1985; Manzini & Wexler, 1987). It is also a close relation of the domain-general *size principle* of Bayesian learning theory (Tenenbaum & Griffiths, 2001). For convenience here we will refer to this conservative tendency as the Subset Principle (SP) but leaving open the existence of many varied implementations of it. Our concern is a duo of recently uncovered practical problems that must be addressed by *any* such implementation if it is intended as a contribution to a psychological model of how children acquire syntax.

As noted in Fodor & Sakas (2005), one problem is that rigorous application of SP appears to demand an undue share of the on-line computational resources that can reasonably be ascribed to a pre-school child. The second problem is that under some familiar learning regimes, SP becomes over-zealous and prevents convergence on the target grammar: without SP, learners are at risk of *over*generation errors, but with SP they are at risk of *under*generation errors. Thus despite its central importance, it is unclear whether SP (and/or its close relations in other frameworks, including statistical learning models) can be successfully incorporated into psychologically faithful models of language acquisition.

We illustrate these problems below in a specific modeling framework that has served in the past as our basis for simulation experiments comparing the efficiency of various acquisition tactics (Fodor & Sakas, 2004). The targets for learning are parameter-based grammars (Chomsky, 1981 et seq.). In parameter setting ('triggering') models, it is commonly assumed that LM has no memory for prior input sentences or for which grammars it entertained previously. It retains from its past experience only the knowledge that is encapsulated in its current grammar. Thus, in contrast to models that accumulate data and seek regularities in it, parameter setting is *incremental*, in the sense that LM receives target language sentences one at a time and decides, on the basis of each one, either to retain its current grammar hypothesis or to switch to a different one.

Despite these specific properties, we believe that the points we raise here have bearing on a broad range of approaches to syntax acquisition. The implementation of SP is equally challenging, or more so, for other current learning models, and any advances that can be made may therefore benefit those other approaches as well. In this paper we argue that it is essential to augment in some way the severely restricted memory of incremental models, and we propose a novel representational scheme that allows LM to keep track of the domain of grammar hypotheses, and thereby alleviates both the problem of on-line computational resources and the undergeneralization problem.

The Computational Resources Problem

SP is a *comparative* criterion for grammar selection: whether it permits a grammar hypothesis to be adopted depends on what alternative hypotheses are available. Given input sentence *i,* LM should ideally adopt a grammar G such that the language L(G) includes *i* and has no proper subset L(G′) that includes *i,* where G′ is a possible grammar that has not been disconfirmed by prior input (if the model has knowledge of that; see below). But how can LM know which grammar satisfies these criteria? It appears that LM must have the ability to identify grammars that license an arbitrary sentence *i*, and moreover that it must have exhaustive knowledge of *all* (non-disconfirmed) grammars that license *i,* so that it can compare them against each other to ensure that it does not unwittingly adopt one that is prohibited by the existence of a less inclusive one. Thus, when LM's current grammar fails on an input *i* and a new grammar must be adopted, LM has three tasks to do. Task A: Find a new grammar hypothesis G which does license *i*. Task B: Identify all other grammars that license *i* (in order to be able to check for subset relations as in Task C). Task C: Check whether any other grammar that licenses *i* generates a subset of L(G).

Task A has proved to be a cumbersome problem for syntax acquisition models. It is not always obvious by inspection of an input word string what grammar might have generated it. Various strategies which start from the current grammar and amend it (e.g., reset one parameter at a time; reset only incorrect parameters) have been found to be inadequate because, for example, it is often unclear which parameters are incorrect. Recent models typically undertake extensive trial and error, selecting a grammar and then testing to see whether it will parse *i* (e.g., Gibson & Wexler, 1994; Clark, 1992; Yang, 2002). The models that we have developed use the parsing routines instead to identify needed changes to the current grammar (Sakas & Fodor, 2001). However, this technique has its limits. It can reliably identify *one* grammar that generates *i*, but not more than one without exceeding standardly accepted limits on the capacity of the human parsing mechanism.

Task B (identifying *all* grammars compatible with *i*) is a challenge of a higher order. The natural language domain is highly ambiguous, with most sentence types compatible with multiple grammars (Clark, 1989). It is also a very large search space, possibly on the order of billions of grammars (2^n for *n* independent binary parameters), so the workload would be prohibitive if indeed every grammar must be checked whenever LM is considering adopting a new one. It is clearly beyond the bounds of psychological plausibility to suppose that a child runs a billion parse tests, each with a different grammar, on a single input sentence to see which grammars succeed. To solve this problem, a completely different approach to SP is required which does *not* require exhaustive knowledge of all grammars that license *i*, as we discuss below.

Task C (discovering subset relations between grammars that license *i*) might be achieved by comparing languages (sets of sentences) on-line, but this too would exceed plausible computational resources. An alternative approach would be to assume that LM is equipped with prior information as to which languages are subsets of which others. Ideally, these subset relations between languages would be transparently reflected in formal relations between their grammars, so that LM could simply inspect two grammars to find out whether one generates a subset of the other.

This was proposed by Manzini & Wexler (1987), who suggested that each parameter has a default value and a marked value (notated 0 and 1 respectively) and that subset relations between grammars are due exclusively to these values: for any pair of grammars differing with respect to the value of a parameter P, the language with value 0 for P is a proper subset of the language with value 1 for P; and no other subset-superset relations hold between any grammars in the domain. We have called this the *Simple Defaults Model* (Fodor & Sakas, 2005). If it were true of natural languages, it would strongly limit the number of subset relations in the domain, thus reducing the scale of Task C. And it would provide LM with a trivially easy way to identify all the subsets of a language L(G): they would be all and only those languages whose grammars differ from G by having value 0 for one or more parameters for which G has value 1.

Unfortunately, it seems that this optimal situation does not obtain in the case of natural languages. For our parameter-setting simulation experiments we have created a domain of 3,072 artificial languages, defined by 13 syntactic parameters and designed to be as much like real natural languages as possible despite necessary simplifications. In this domain the Simple Defaults Model fails. A high proportion (over 42%) of the subset relations that hold between grammars are *not* predictable from the subset values of individual parameters; they are due instead to interactions, often quite unruly, among two or more parameters. Therefore, any SP-implementation based on the Simple Defaults Model would under-report the subsets a language has, and would fail to protect LM against overgeneration errors. Simulation data confirm this expectation; we observe 64% failures for a model that performs without error when supplied with full information about subset relations.

Perhaps other linguistic theories might offer better ways of predicting subset relations between languages based on their grammars, but none is known at present and in fact there are good reasons to suspect that the relationship between grammars and the languages they generate is bound to be disorderly: a small change in a grammar can completely change the set of sentences (word strings) it generates, and word strings generated by quite different grammars may happen to coincide. It therefore becomes important to consider what theoretical options there are, if it does turn out that subset relations cannot be projected on-line by LM. It seems unavoidable to suppose, in that case, that LM has access to an innate database of some kind which provides subset-superset information. The biological origin of such a knowledge structure may be a mystery in the present state of understanding, and remains to be explored, but a first step is to find out whether, if it *did* exist, it would permit Task C to be achieved without incurring an unreasonable computational workload.

From Enumeration to Lattice

Formal learnability studies in the tradition of Gold (1967) assumed that the learning algorithm was provided with subset-superset information in the form of an *enumeration* of grammars: a total ordering of all the possible grammars, in which any subset grammar precedes all of its superset grammars. (Note that for convenience from now on we refer to subset relations between *grammars*, as a shorthand for subset relations between the languages that the grammars generate.) Because of its foundational status in formal learning theory, it is worthwhile to see whether an enumeration can be adapted for psychological purposes.

The enumeration could serve as the innate database about subset-superset relations that LM would consult for Task C. It could also provide dynamic guidance for LM in its on-line process of grammar hypothesization. If LM hypothesizes grammars strictly in accord with the enumeration ordering, moving on to the next one only when the previous one has proven incompatible with the input, it will have obeyed SP without explicitly applying it. In particular, an enumeration-based LM obeys SP without exhaustively identifying and comparing all candidate grammars; thus, the enumeration does away with Task B. It does so by rendering illicit grammars (i.e., superset grammars) *inaccessible* to LM; LM has access to a grammar only after all its subsets, prior to it in the ordering, have been disconfirmed. Also inaccessible are all previously disconfirmed grammars, since they are necessarily prior in the enumeration to LM's current grammar; so those hypotheses are not revisited and convergence is thereby speeded.

Thus a classic Gold-type enumeration makes short work of Tasks B and C. It falters, however, on Task A: selecting a new grammar compatible with the current input sentence. The enumeration gives LM no choice with respect to its next grammar hypothesis: when its current grammar fails to license input *i,* LM must try out the immediately next grammar in the enumeration. This has the obvious disadvantage that a grammar late in the ordering can be attained only after eliminating all billion-or-so grammars prior to it in the enumeration. As described so far, the model has no way to use the properties of the input sentence to move *directly* to an appropriate grammar, skipping over irrelevant ones in between. More importantly, we cannot *introduce* any devices that would do this, because once intervening grammars are allowed to be passed over, the role of the enumeration as the enforcer of SP is lost. The danger of LM passing over an intervening *subset* grammar would obviate the whole purpose of the grammar ordering. However, without the ability to move faster through the sequence by skipping grammars along the way, enumeration-based learning is generally regarded as being unredeemably slow and has not been embraced by psychological models of language acquisition (Pinker, 1979).

The excessive rigidity of the classic enumeration can be remediated, however, by shifting to a *partial* ordering of grammars, which places all subset grammars prior to their superset grammars but does not impose a fixed order otherwise. The partial ordering is sufficient to ensure compliance with SP, but in other respects it leaves LM free to move around the grammar search space, from less profitable to more profitable regions, using whatever skills it may possess for identifying a likely grammar to license *i.* On this proposal the database of grammars takes the form of a *lattice* (or strictly, a *poset*), as illustrated in Figure 1.

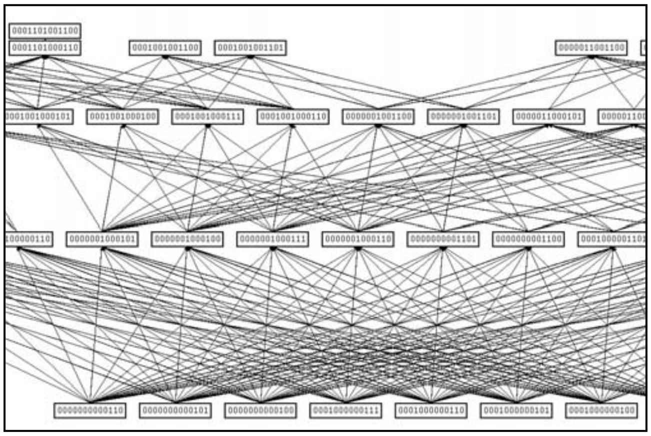

Figure 1: A small fragment (less than 1%) of a lattice representation of the domain of 3,072 parameterized languages used in the simulation experiments described below. Supersets are above subsets.

Observe that the classic one-dimensional enumeration has been reshaped here. The smallest subset grammars in the domain are presented at the lower edge of the lattice, with their supersets above them. The lowest grammars are all and only those that constitute legitimate hypotheses for LM at the initial stage of learning prior to any input. A grammar that is higher in the lattice may be adopted only after all the grammars it dominates have been tested and disconfirmed by the input. This means that higher grammars will be attained more slowly on average than lower grammars, but the disparity is far less than between the earliest and latest grammars in a classic enumeration: the maximum depth of the lattice for our natural-language-like domain of grammars is 7; the mean depth is 3.4. It can be supposed that as lower grammars are disconfirmed they are deleted from the lattice, so that the set of grammars accessible to LM, at the lower edge of the lattice, gradually changes over time. A grammar that has many subsets will start out high in the lattice but will work its way down if and when the subset grammars beneath it are erased.

As far as SP is concerned, LM may choose freely from among the accessible grammars at the bottom of the lattice. It might do so by random trial and error if no better mechanism is available. But the lattice has a considerable advantage over a total enumeration in that it leaves LM some elbowroom to apply useful grammar-guessing strategies. Any linguistic knowledge that LM may have can be put to work to extract relevant properties of input sentences to guide its grammar choices. The family of learning models that we have proposed, known as *Structural Triggers Learners* (*STL*s; Fodor & Sakas, 2004), can do this. STLs use the technique noted above, of employing the parsing routines for on-line detection of how the current grammar can be supplemented to accommodate input *i.* It works as follows. The parser applies LM's current grammar, $G_{current}$, to the input sentence. If the parse succeeds, LM retains $G_{current}$. If the parse fails, then at the specific locus of that failure in the word string, the parser is permitted to draw on other linguistic resources (specifically, previously unused parameter values) as necessary in order to complete the parse. LM then adopts whichever new parameter values contributed to rescuing the parse.

We call this *decoding* the input sentence. The parser does not merely register *whether* a given grammar licenses *i* or not, but actively *finds* a grammar that licenses *i.* As noted above, it cannot realistically be assumed that for an ambiguous input the human parser computes *every* grammar that could license it. Moreover, the one grammar it does find may not be the correct grammar for *i* in the target language, but it is at least a genuine candidate hypothesis, one that might be correct or could lead LM in the direction of one that is. The consequence of combining this decoding strategy with the lattice representation of possible grammars is that LM does not waste effort checking grammars that have no relation to the current input. Instead, the work of testing and discarding grammars in the lattice is highly focused on grammars that do license sentences in the learner's input sample. In many regions of the lattice there may be no activity at all, because the grammars there are unable to parse target language sentences (e.g., they generate head-final constructions while the target language is head-initial). The lattice representation combined with input decoding thus may be a step towards an optimal grammar search strategy.

To summarize: Like an enumeration, the partial ordering of grammars in a lattice encodes essential information about the subset-superset relations in the domain. Also like an enumeration, it blocks LM's access to unsafe (superset) grammars, so that LM can avoid them without engaging in resource-heavy comparisons between grammars. Unlike an enumeration, it does not insist on a single fixed sequence of grammar hypotheses. Subset-superset grammars must be ordered because of SP, but other grammars are freely accessible. This decreases the learning time discrepancy between the least and most accessible grammars in the domain, and also permits LM to take advantage of linguistic information (cues, triggers) in the input to *guide* its search through the lattice for the target grammar. The simulation data we present below show that while there are better and worse ways for a learner to make use of a lattice, a lattice-based model can indeed reliably prevent overgeneration without exceeding reasonable computation loads.

The Undergeneralization Problem

The erasure of disconfirmed grammars from the lattice offers a straightforward solution to the problem of undergeneralization that can afflict incremental learners. Incremental learning is widely favored over batch learning from a psychological point of view, because it presupposes neither memory for the entire input sample, nor methods for fitting a grammar to a large corpus. However, there is a fundamental incompatibility between incremental learning and the conservative learning that is needed for avoiding overgeneration. SP is often cast informally as the requirement that LM should select the least inclusive grammar compatible with the input. But if the only input accessible to LM is the current sentence, the least inclusive grammar compatible with it will generate a very small language indeed; it is likely to lack many language phenomena that were acquired from previous inputs no longer in memory. For instance, all long-distance movement would be lost if the current sentence has none. The fact that the previously acquired phenomena are generated by LM's current grammar does not protect them from loss. Conservative learning requires that all contents of the current grammar be given up when a new grammar is about to be adopted, *except those that are known to be correct.* Otherwise LM's grammar would just keep growing as the sum of all its previous false hypotheses, and overgeneration would be rife. However, since most learning models hypothesize grammars on the basis of ambiguous input (and most cannot even tell which inputs are ambiguous and which are not), LM can rarely be certain that some phenomenon it previously 'acquired' is veridical. Hence SP (or comparable conservative learning principles) would repeatedly force the learner to regress to very limited languages compatible with just the current input. (See Fodor & Sakas, 2005, for additional discussion of this problem of *excessive retrenchment.*)

Since there is no evidence that child learners are afflicted with this problem, it should not occur in our learning models either. A simple solution would be to abandon incremental learning entirely. If it were assumed instead that LM holds in memory all or many of its prior input sentences, it could not be forced by SP to adopt a language smaller than the minimal one that contains all of those sentences. Psychological models of parameter setting that base each grammar hypothesis on a collection of many sentences (unlike 'triggering') may well be of interest but no standard implementation currently exists (though see Kapur, 1994). An alternative approach, which avoids giving up the psychologically desirable aspects of incremental learning, is to eliminate languages from the hypothesis space as and when they are found to be too limited to include the input. As learning proceeds, languages that are excessively small will be ruled out; the smallest languages in the pool will be larger and larger, and LM can now adopt them even in response to a single input sentence. E.g., once languages without long-distance movement have been eliminated by previous input exhibiting long-distance movement, LM will necessarily adopt a grammar that licenses long-distance movement even if the current sentence exhibits no movement at all. Elimination of disconfirmed grammars is very natural in a lattice-based model, as sketched above. Note that this antidote to excessive retrenchment adds memory to the incremental model in order to solve the undergeneration problem, but it does so in an economical way. A lattice model with erasure retains the fruits of past learning not by accumulating memory traces of prior events, but by unburdening long-term memory as the innate lattice representation is progressively simplified.

Computational Evaluation of Lattice Models

Our simulation studies are conducted on the domain of constructed languages described above, defined by familiar syntactic parameters that govern word order, null subjects, wh-movement and so forth. To isolate syntactic parameter setting from the acquisition of lexical items, the sentences are pre-coded as strings of part-of-speech labels (cf. Gibson & Wexler, 1994). A detailed description and examples of the languages can be found in Sakas (2003). The sentences of a target language are fed to a learning model which guesses a grammar after each one. In the simulations reported below, each learning model was run 100 times on each language in the domain, with a ceiling of 10,000 input sentences on any trial. We record the percent of successful convergence on the target grammar, and the average number of input sentences consumed before convergence. These measures allow us to quantify the reliability and efficiency of a wide variety of alternative models.

Six variants of the lattice-based model outlined above have been tested in this environment. They differ from each other as indicated below. Some make use of the lattice but do not decode the input; some do decoding but do not make full use of the lattice. Our purpose in comparing this range of models was to assess the relative contributions of these two components, and to identify limits on the usefulness of the lattice concept. The results are shown in Table 1. Note that *SL* denotes the set of 'smallest languages' at the lower edge of the lattice. The descriptions indicate what the learning model does on receiving a novel input *i* which Gcurrent does not license; its task is to find and adopt an SP-compatible grammar that parses *i.* Unless otherwise specified, a grammar that has failed on *i* is erased from the lattice before the next input is processed.

M1: No Decoding, SL: If Gcurrent fails, select any grammar G in SL; run parse-test; if G fails erase it from the lattice and retain Gcurrent; if G succeeds, adopt it as Gcurrent.

M2: No Decoding, SL, Activation: Like M1 except that every grammar has an activation score. If Gcurrent fails, select the grammar G in SL with the highest activation; run parse-test; if G fails, erase it from the lattice and select the grammar with the next highest activation as the new Gcurrent; if G succeeds, adopt it as Gcurrent and add one activation unit to all grammars that dominate it in the lattice (since these all also license *i*).

M3: Decoding and SL: Decode *i* (i.e., use Gcurrent to initiate a parse of *i*; if it succeeds, retain Gcurrent; if it fails, patch the parse tree with new parameter values as necessary and adopt them into Gcurrent), but subject to the condition that only values in the grammars in SL are available for adoption. If decoding fails, as it may due to this restriction, select a grammar at random from SL to be the next Gcurrent.

M4: Decoding (Defaults), SL as Filter: Decode *i* (see above), favoring subset (i.e., default) values of parameters if there are alternative parses of *i;* if the decoded grammar is in SL, adopt it; else retain Gcurrent.

M5: Decoding (Random), SL as Filter: Decode *i* (see above), making a random choice if there are alternative parses of *i;* if the decoded grammar is in SL, adopt it; else retain Gcurrent.

M6: Decoding (Random), Track Downward: Like M5, but if the decoded grammar G′ is not in SL, run parse tests on daughters of G′ until a grammar is found that parses *i;* repeat recursively on its daughters until a grammar is found with no daughters that parse *i;* adopt that grammar. (See discussion of this strategy below.)

Table 1: Measures of reliability and efficiency for some lattice-based learning models

Model	% success	Average sentences	Average for 99%	# parses per sentence
M1	100	858	1,454	1
M2	100	900	968	1
M3	100	140	286	1
M4	12	23	71	1
M5	31	2,631	6,032	1
M6	96	190	694	mean 4.8

Note that the fourth column of Table 1 shows how many input sentences were required, averaged across languages, for 99 of the 100 trials of the learning model on a given language to attain the target grammar. Since the vast majority of children do acquire their target language, this is an appropriate and rigorous estimate of a model's performance.

Discussion of Results

The data make it evident that not every way of incorporating a lattice representation into a learning model is helpful, but at least one of the designs we tested is both reliable and speedy. Not unexpectedly this is version M3, which is the only one that fully integrates partial decoding and the lattice representation. It required fewer than 300 input sentences for 99% convergence on grammars in this domain. This compares favorably with the performance of decoding learning models that we have tested in the past which lacked any machinery for applying SP (so that it had to be externally imposed by an oracle that blocked adoption of overgenerating grammars).

Other noteworthy outcomes include the fact that model M1, which employs the lattice without taking advantage of the opportunity to do decoding, is very slow, as is characteristic of models that rely on trial and error in selecting which grammars to test. Models M4 and M5 use the lattice not to help *select* their hypotheses but only to *filter* them after selection, and they are both extremely slow, with many 'time-out' failures (88% and 69% respectively). M4 is speedy only for a handful of target languages near the bottom of the lattice, for which it does succeed; M5 is a generally slow trial-and-error system.

Despite a few time-outs, model M6 mostly works fast in terms of number of input sentences, but it does extra work in processing each one, to make up for the fact that it does not restrict its hypotheses to the 'safe' grammars at the bottom of the lattice. This gives LM the freedom to focus on a preferred grammar, but the cost is that multiple parse tests are then needed to identify any SP-compatible grammars beneath it in the lattice, since they must take precedence. The average of approximately 5 parse-tests per sentence is less than we had anticipated but is still an implausible amount of computation to be performed every time a child hears a sentence.

A promising finding for future work is that M2's activation levels for grammars confers some advantage compared with M1, at least in the 99% convergence score. This activation strengthens grammars in proportion to how many target language sentences they have been observed to license, and thus helps to attract the learner toward profitable areas of the lattice. It may be that incorporating activation into model M3, which already successfully combines SL and decoding, will yield the best performance of all but we have not yet investigated this.

The Source of the Lattice

What may be hard to swallow about the subset-superset lattice is why it should exist at all in the minds of language learners. Perhaps the best that can be said in its defense is that it is not an impossibility if the domain of grammars is finite. But it is likely to be very large, and it serves no other apparent purpose than SP. The principles-and-parameters theory assumes that the set of all possible grammars is determined by the innately-given parameters and their values. So why should each grammar also be individually specified in a lattice? Clearly, the lattice-based model for SP-implementation would be more palatable if, as we considered above, there were general principles for projecting not just the set of grammars but all the subset relations between them. However, the facts of our constructed language domain do not encourage confidence in this. In Figure 1, which is quite typical of the domain as a whole, it can be seen that there are pairs of grammars for which one of the subset-superset parameters (P4-P7, P10-P13) has value 0 in the superset grammar and value 1 in the subset grammar, contrary to the Simple Defaults Model. For instance, there are over 50 such reversals just for parameter P13 (Question Inversion). We are currently exploring richer predictive schemas for identifying subset relations between languages by inspection of their parameter values, in hope that the subset relations in the lattice will ultimately prove to be fully projectible on-line rather than needing to be hardwired. But in the meantime we hope to have shown here that a lattice representation of subset relations – or some projectible version of it – is worthy of study since it offers solutions to two thorny problems in modeling conservative learning from text.

Acknowledgements

This research was supported in part by grants 67827-00 36 and 67560-00 36 to the first two authors, from the Professional Staff Congress of the City University of New York.

References

Berwick, R. C. (1985). *The acquisition of syntactic knowledge.* Cambridge, MA: MIT Press.

Chomsky, N. (1981). *Lectures on government and binding*. Dordrecht: Foris Publications.

Clark, R. (1989). On the relationship between the input data and parameter setting. *Proceedings of the 19th Annual Meeting of the North East Linguistic Society* (pp. 48–62).

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

Fodor, J. D. & Sakas, W. G. (2004). Evaluating models of parameter setting. *Proceedings of the 28th Annual Boston University Conference on Language Development* (pp. 1-27). Boston: Cascadilla Press.

Fodor, J. D. & Sakas, W. G. (2005). The Subset Principle in syntax: Costs of compliance. *Journal of Linguistics*, *41*, 513-569.

Gibson, E. A .F. & Wexler, K. (1994). Triggers. *Linguistic Inquiry*, *25*, 407-454.

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

Kapur, S. (1994). Some applications of formal learning theory results to natural language acquisition. In B. Lust, G. Hermon & J. Kornfilt (eds.) *Binding, dependencies, and learnability*, Vol. 2 of *Syntactic theory and first language acquisition: Cross-linguistic perspectives*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Manzini, R. & Wexler, K. (1987). Parameters, binding theory, and learnability. *Linguistic Inquiry*, *18*, 413-444.

Marcus, G. F. (1993). Negative evidence in language acquisition. *Cognition*, *46*, 53-85.

Pinker, S. (1979). Formal models of language learning. *Cognition*, *7*, 217-283.

Sakas, W. G. (2003). A word-order database for testing computational models of language acquisition. *Proceedings of the 41st Annual Meeting of The Association for Computational Linguistics* (pp. 415-422). East Strasburg, PA: Association of Computational Linguistics.

Sakas, W. G. & Fodor, J. D. (2001). The Structural Triggers Learner. In S. Bertolo (ed.), *Language acquisition and learnability*. Cambridge: Cambridge University Press.

Tenenbaum, J. B. & Griffiths, T. L. (2001). Generalization, similarity, and Bayesian inference. *Behavioral and Brain Sciences*, *24*, 629-641.

Yang, C. D. (2002). *Knowledge and learning in natural language*. New York: Oxford University Press.