A Genetic Algorithm for the Induction of Natural Language Grammars

Tony C. Smith and **Ian H. Witten**
Department of Computer Science, University of Waikato, Hamilton, New Zealand
Email tcs@waikato.ac.NZ; phone: +64 (7) 838–4453; fax: +64 (7) 838–4155

Abstract

Strict pattern-based methods of grammar induction are often frustrated by the apparently inexhaustible variety of novel word combinations in large corpora. Statistical methods offer a possible solution by allowing frequent well-formed expressions to overwhelm the infrequent ungrammatical ones. They also have the desirable property of being able to construct robust grammars from positive instances alone. Unfortunately, the "zero-frequency" problem entails assigning a small probability to all possible word patterns, thus ungrammatical n-grams become as probable as unseen grammatical ones. Further, such grammars are unable to take advantage of inherent lexical properties that should allow infrequent words to inherit the syntactic properties of the class to which they belong.

This paper describes a genetic algorithm (GA) that adapts a population of hypothesis grammars towards a more effective model of language structure. The GA is statistically sensitive in that the utility of frequent patterns is reinforced by the persistence of efficient substructures. It also supports the view of language learning as a "bootstrapping problem," a learning domain where it appears necessary to simultaneously discover a set of categories and a set of rules defined over them. Results from a number of tests indicate that the GA is a robust, fault-tolerant method for inferring grammars from positive examples of natural language.

keywords: grammar induction, genetic algorithms, statistical methods, context-free grammars.

Introduction

Grammatical inference is the gradual construction of a correct grammar based on a finite set of sample expressions. In general, the training set may contain both positive and negative examples from the language under study. However, in the case of natural languages, it is widely held that children learn a grammar from positive instances alone, and that parents tend not to offer overt correction during the learning process. This proves problematic as it is known to be impossible to infer a correct grammar from positive instances only (Gold 1967).

Probabilistic approaches offer a solution to this problem by allowing frequent, well-formed expressions to statistically overwhelm the infrequent ungrammatical ones. More precisely, probabilities for word combinations (n-grams) are derived in accordance with their observed frequencies. Parsing from models constructed in this way can thus be made more efficient than standard Phrase Structure Grammars (PSGs), which must treat all possible strings as equiprobable. Examples of frequency-based grammars include Probabilistic Context-Free Grammars (PCFGs) (Charniak 1993), which assign a probability to each string such that the probabilities of all strings sum to one, and Hidden Markov Models (HMMs) (Rabiner & Juang 1993; Kupiec 1989), which assign a probability to each string such that the probabilities for all strings of a given length sum to one.

Inductive methods for constructing such grammars derive probabilities for each of their transitions by recording frequencies for n-grams garnered from the training set. As the training set will not contain all possible n-grams of the language, some means for assigning probabilities to unobserved transitions must be incorporated. Solutions to this so-called "zero-frequency problem" usually entail assigning a small but equal probability to all unseen patterns. The result, however, is that ungrammatical n-grams become as probable as unseen grammatical ones.

A further difficulty with such models is that they will assign lower probabilities to transitions involving infrequent words than to those involving more common words of the same syntactic category. For this reason they are unable to take advantage of any lexical properties known to be characteristic of the language.

The extent of these limitations can be reduced by applying frequency analysis to the lexical categories of a tagged-text. However, recent work

indicates that grammar induction is an instance of a "bootstrapping problem" (Finch & Chater 1992; Smith & Witten 1995), a learning domain where it appears necessary to simultaneously discover a set of categories and a set of rules defined over them. That is, lexical tagging for the purpose of inferring syntactic structure should not be constrained by existing notions of syntactic categories—the rules and categories must be derived together.

This paper outlines a grammar induction method that incorporates the frequency-based conditioning aspects of probabilistic techniques while supporting the notion of language learning as a bootstrapping problem. The approach uses a genetic algorithm (GA) to adapt hypothesis grammars toward more effective models of the language under study—in this case, English.

Following the approach of evolutionary programming (Wijkman 1994), a population of grammars is allowed to evolve for a fixed number of generations before any culling decisions are made. In this way, apparent shortcomings that arise from a single adaptation are given a chance to reveal possible longer term benefits. Grammars are represented as LISP AND-OR s-expressions, and evolution is achieved using standard GA techniques for tree transformations (Bickel & Bickel 1987). Individual grammars are selected for reproduction and mutation in proportion to their size. The strength of the population as a whole is measured by its ability to parse the training set.

The GA is statistically sensitive in that the utility of frequent patterns is reinforced during the random selection of tree nodes. The more often a given node can account for substrings within training expressions, the more likely it will persist as a recurring feature in subsequent generations of grammars. Its presence in a large number of candidate grammars thereafter gives it a higher probability of continued selection.

The view that language learning is a bootstrapping process is supported because lexical categories emerge as nodes most efficient at capturing regularities in the training set. For example, an inferred structure in which an OR-node of nouns is followed by an OR-node of verbs is a more efficient generalisation of the combinatoric capacity of these words than is an OR-node whose arguments are all possible sequences combining one noun and one verb. By using grammar size as a measure of fitness, the most simple representation has the best chance of being carried forward into the next generation.

Grammar induction using genetic algorithms

Genetic algorithms are an analogue of natural selection. They adapt principles of reproduction, mutation and “survival of the fittest” to evolve successively better generations of computer programs for a particular task. For grammar induction, the “chromosome” is usually a context-free grammar whose fitness is measured by its ability to cover (i.e. parse, accept, generate, etc.) a training set of sample strings. Reproduction is simulated by dissecting two disparate grammars at suitable, randomly chosen points and joining the first half of one grammar with the second half of the other, and similarly joining the remaining halves. Mutation occurs by changing any single randomly chosen symbol of a grammar with another symbol taken randomly from the set of symbols for the language.

Some effort has been directed towards applying GAs to the inference of context-free grammars. Koza (Koza 1992) outlined a GA for detecting exons of length 5 within DNA. After 35 generations, the resulting 61-node tree was 100% successful at identifying all exons in a segment of 1000 neucleoitide bases. Wyard (Wyard 1991) devised a genetic algorithm for the language of correctly balanced nested parentheses which successfully inferred a concise correct grammar from 20 test strings in just 3 generations. Wyard also constructed a GA for inferring the more complex language of all strings containing equal numbers of a’s and b’s, but was unable to produce a successful grammar due to the model becoming stuck in local maximum.

In developing a genetic algorithm for inferring a context-free grammar description for a more complex language such as English, aspects of prior successful GA models were retained while those of failed attempts were avoided. Koza’s successful exon identifier represented a grammar as a logical s-expression using AND, OR and NOT, while Wyard’s failed “a-b language” recogniser used the more traditional Chomsky Normal Form (CNF). Many CNF grammars trivially translate into logical s-expressions. For example, the CNF grammar

S	⇒	NP	VP
NP	⇒	Det	N
VP	⇒	V	NP
Det	⇒	a	
Det	⇒	the	
N	⇒	dog	
N	⇒	cat	
V	⇒	saw	
V	⇒	bit	

expands to

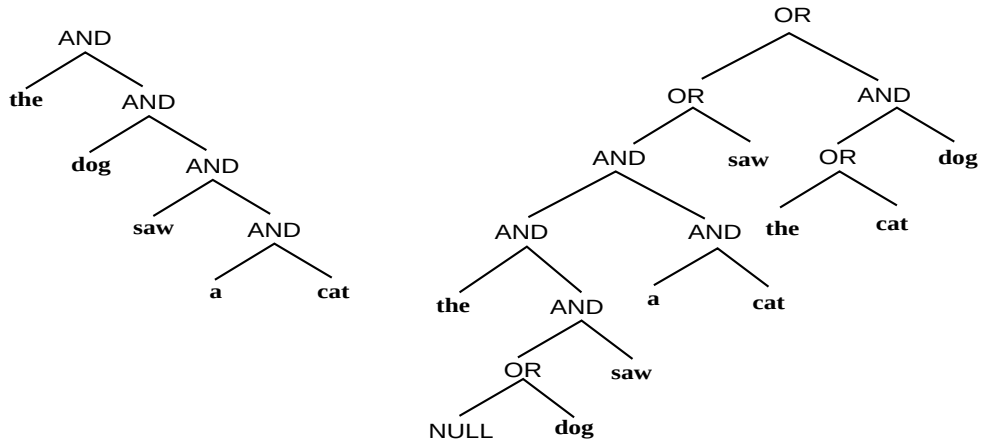

Figure 1: Two sample grammars for "the dog saw a cat".

S	⇒	Det	N	V	Det	N
Det	⇒	a				
Det	⇒	the				
N	⇒	dog				
N	⇒	cat				
V	⇒	saw				
V	⇒	bit				

which has the following equivalent s-expression

(AND (OR **a the**)(OR **dog cat**)(OR **saw bit**) (OR **a the**)(OR **dog cat**))

This representation is not a true s-expression in that the AND symbol is used to indicate catenation, and thus imposes a strict ordering on its arguments. The OR symbol indicates that any one of its operands may substitute for the complete node, and thus imposes no ordering. This representation is used throughout the rest of the paper.

Recursive CFGs cannot be translated to s-expressions in this way, thus our decision to use s-expressions as the representation has certain limitations. However, it is quite difficult to develop an efficient evaluation function for recursive grammars. First, a breadth-first parser would be needed for testing grammars, adding a great deal of additional computation to the evaluation process. S-expressions are easily tested bottom-up, allowing more time for the evolutionary search. Second, simplistic evolution on recursive CFGs permits the constant threat of infinite looping in the form of, say, $S \Rightarrow S$, and thus requires additional constraints in testing for fitness. Because the training set consists of finite length strings, the resulting s-expressions are a clean, simple intermediate representation from which questions about recursive substructure can be addressed at a later time. Furthermore, questions such as whether English allows infinite center embedding (e.g. "the cheese that the mouse that the cat ... chased ate smelled.") or merely embedding to a fixed depth indicate that the issue of a recursive natural language grammar has not yet been fully resolved.

In a further step towards simplification, we limit the set of operators to just AND and OR. It is worth noting that NOT is seldom included in CFG formulations anyway. In addition, n-ary AND and OR expressions are rendered in their binary equivalents to simplify many of the GA functions.

The evolutionary process

The GA proceeds from the creation of a random population of diverse grammars based on the first sample string. The vocabulary of the expression is added to an initially empty lexicon of terminal symbols, and these are combined with randomly chosen operators in the construction of a candidate grammar. A NULL symbol is also included in the lexicon to support the possibility of optionally omitted constructs within a grammar. If the candidate grammar can parse the first string, it is admitted into the initial population. Figure 1 shows two possible grammars for "the dog saw a cat".

Early experiments in which initial grammars were constructed in a completely random manner often took a considerable amount of time to produce even a single grammar capable of parsing the first expression. Constraints, such as a maximum depth for nesting of randomly constructed nodes, were subsequently placed on the grammar-generator to guide it towards suitable grammars more quickly. Additional simplification steps include the substitution of (OR **x x**) and (AND **x** NULL) with just **x** to preempt unnecessary steps during parsing. Also, in

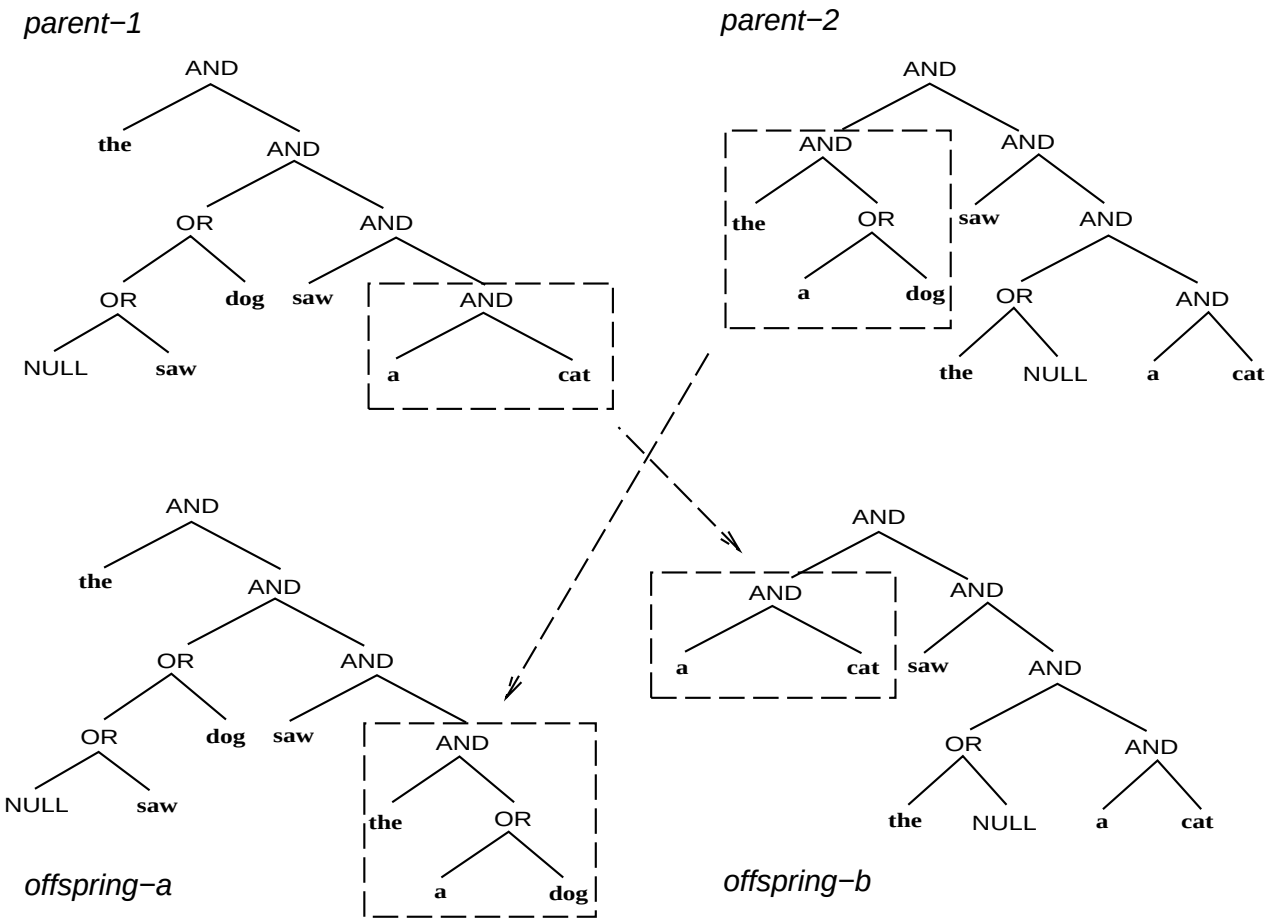

Figure 2: Reproduction via node exchange.

keeping with Mauldin's (Mauldin 1984) suggestion for maintaining diversity, duplicate grammars are always removed from the population.

Reproduction

At each presentation of a new sample string, the GA adds any new vocabulary items to the set of terminal symbols. Two grammars from the existing population are randomly chosen for "sexual" reproduction. Selection is carried out in inverse proportion to a grammar's size. That is, grammars with fewer nodes are given a higher probability than those with a larger number of nodes under the assumption that smaller grammars are more effective.

A single, randomly selected node from one parent is exchanged with one from the other parent to produce two offspring. If an offspring is able to parse at least one string from the training set then it is added to the existing population, otherwise it is discarded. The process is non-destructive so the parents also persist. Figure 2 shows parents and offspring from one reproductive event.

Mutation

Whenever two grammars are chosen for reproduction, another is also randomly selected for mutation. Selection here is done in direct proportion to a grammar's size (i.e. a larger grammar is more likely to be selected than a smaller one) under the assumption that a larger grammar is not adding to the fitness of the population as a whole. Moreover, a larger size indicates the presence of extraneous nodes which further implies a greater capacity to benefit from mutation. A single node is chosen randomly from the grammar. If it is a leaf node, it is replaced with a randomly selected terminal symbol (possibly the NULL terminal) from the lexicon. If it is an internal node, it is replaced with the complement operator. The mutated grammar is added to the existing population if it can parse at least one

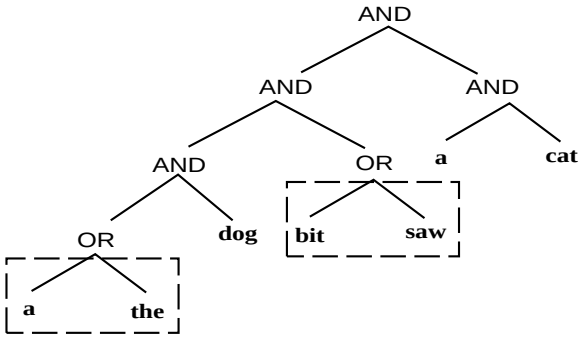

Figure 3: Lexical categories emerge in OR nodes.

of the test strings. Its original form is also allowed to persist.

GAs traditionally assign probabilities to both reproduction and mutation. Mutation is given a low probability (presumably) to allow the adaptation procedures to play the greater part in dictating the direction of the evolutionary process. The primary function of mutation is to reduce the possibility of becoming stuck in local maximum, though this is not always successful if culling decisions are made too early. We needed a method of adding newly acquired lexical items into the population of grammars. One option was to simply add a conjunctive representation of the new sentence as a disjunctive option at the root of one or all grammars. However, this was seen as too strong an intervention in the adaptative process. We decided on the more passive approach of allowing new words to be swapped in during a mutation. Because new words would be given the same chance of selection as all other vocabulary items, mutation was allowed to occur frequently so that the population did not flounder too long in its search for a correct grammar.

Fitness

Adaptation occurs in cycles of a fixed number of reproductions and mutations. After each cycle, all grammars are tested for their ability to parse the existing test set. Parsing is carried out bottom-up, and grammars able to parse the greatest number of expressions in the test set are allowed to persist into the next cycle. Thus if no grammars have evolved to parse the complete test set, the grammars that could parse all but the latest addition will still be present in the population. Any new grammars that parse an equal number of expressions are also allowed to persist, and the population is subjected to another cycle of adaptation. Thus the population could conceivably continue to grow indefinitely. If, however, grammars have evolved to cover the entire test set, then they alone survive into the next generation. At this point a new string is added to the test set and a new evolutionary cycle is initiated. The shortest grammar in the population is output as the "best-of-generation" after each cycle.

This cyclic approach to population adaptation is in accordance with principles of evolutionary programming. It has the benefit of allowing seemingly poor changes that occur through reproduction or mutation to reveal their possible advantages at a later stage of development. For example, the offspring of a reproductive event might not be able to parse as well as its parents. It may be extraordinarily large and complicated. However, it could also be that a simple, random change in one of its operators (brought about through mutation) renders it a powerful grammar able to parse many more well-formed expressions than any of its ancestors.

Results

A series of experiments were devised to examine the behaviour of the GA. The first test sought to compare the GA's response to a small subset of English. A population of 10 randomly constructed grammars was generated for the sentence "the dog saw a cat", the best (i.e. smallest) of which was the following 13 node s-expression.

(AND (AND (AND **the dog**)(AND **saw** (OR (OR **cat** NULL) **a**))) **cat**)

The decision to start with 10 grammars was arbitrary, intended merely to avail the algorithm of some initial diversity.

The sentence "a dog saw a cat" was added to the training set and the initial population was allowed to evolve for one cycle of 25 generations. The cycle length was arrived at through trial and error. Successful grammars were often found in one or two generations, but a larger period allowed a wider

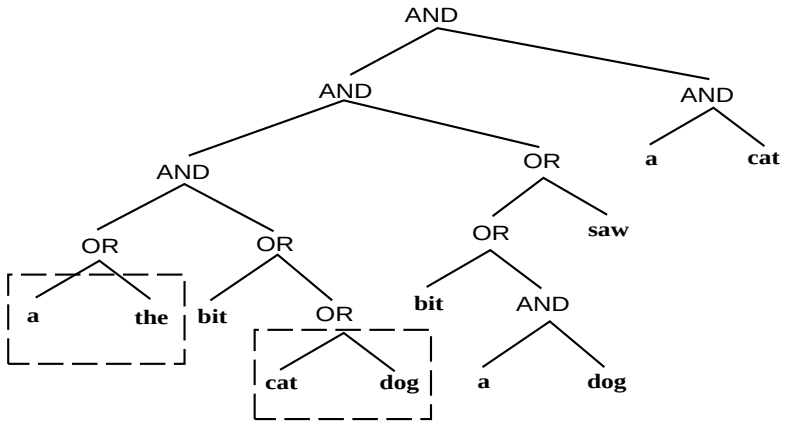

Figure 4: The grammar after four sample strings.

exploration of the search space and a better chance for the emergence of more efficient grammars.

Five correct grammars resulted from the first cycle, the best of which was

(AND (AND (AND (OR **a the**) **dog**) (AND **dog saw**)) (AND (OR **a the**)**a cat**))

This grammar has grouped the determiners into a single disjunctive node—a desirable result from a grammar induction system. It has also fortuitously copied the determiner node into the direct object nounphrase, and thus has inferred a new string.

The sentence "the dog bit a cat" was added to the training set to see if a similar grouping effect could be achieved for regular verbs. The population obtained from the previous cycle was evolved for another 2 cycles (50 generations) before it developed any grammars that could parse all three test strings. The best of the two successful grammars was this 13-node grammar

(AND (AND (AND (OR **a the**) **dog**)(OR **bit saw**))(AND **a cat**))

and its corresponding tree is shown in Figure 3. As was hoped, the verbs have emerged within a single node, though the serendipitous determiner node in the direct object nounphrase has been lost in the process.

Finally, the sentence "the cat saw a cat" was added to the training set to see if nouns will also group together. After 3 cycles (75 generations), 22 grammars evolved that could parse the complete test set. The shortest of these,

(AND (AND (AND (OR **a the**) (OR **bit** (OR **cat dog**)))(OR (OR **bit** (AND **a dog**)) **saw**)) (AND **a cat**))

has successfully collapsed the nouns of the subject nounphrase under a single node. The corresponding tree for this grammar is shown in Figure 4. From the four training expressions, four additional well-formed sentences have been inferred. Even so, two aspects have emerged that weaken the grammar's overall appeal. First, the word "bit" has been erroneously incorporated into the set of subject nouns (the fact that consequent sentences might be argued as grammatical is simply an accident of the vocabulary used). This extraneous node adds nothing to the grammar's ability to cover the training set, and would be removed if enough time were allowed because the result would be a more concise grammar and therefore more likely to persist into future generations.

Second, the direct object nounphrases "a dog" and "a cat" have been distributed between two dissociated nodes. This also is the result of an inadequate amount of evolutionary time, as the seemingly obvious advantage of

(AND **a** (OR **dog cat**))

over

(OR (AND **a dog**)(AND **a cat**))

has not yet emerged. To increase the combinatoric cost of failing to generalise the object nouns, we added the strings "the dog saw a mouse" and "a cat chased the mouse" to the training set in two successive cycles. The best grammar obtained from the additional 100 generations is shown in Figure 5 and, as expected, the additional complexity of the training set has led the GA toward a more effective generalisation of syntactic components—including the emergence of the $S \Rightarrow NP\ VP$ general form often used in stochastic context-free grammars for English.

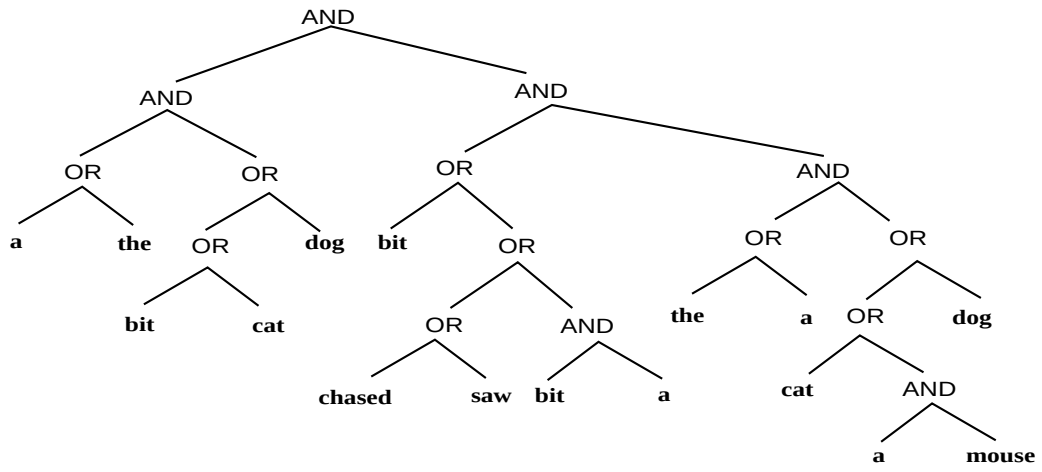

Figure 5: A $S \Rightarrow NP\ VP$ structure emerges after 6 sample strings.

Earlier, we mentioned that many CNF grammars translate into s-expressions. The reverse is also possible. If we perform this transformation on the grammar shown in Figure 5, using nonterminal symbols commonly found in natural language PSRs to stand in for some of the expanded nodes, the following rules are obtained.

S	⇒	NP1	VP
NP1	⇒	Det	N1
VP	⇒	V	NP2
NP2	⇒	Det	N2
Det	⇒	a	
Det	⇒	the	
N	⇒	cat	
N	⇒	dog	
N1	⇒	N	
N1	⇒	bit	
N2	⇒	N	
N2	⇒	a	mouse
V	⇒	bit	
V	⇒	chased	
V	⇒	saw	
V	⇒	bit	a

This description differs in detail only slightly from the grammar presented as an example of CNF earlier in the paper. The differences in its two noun-phrase rules would probably be eliminated if sufficient time was given to discover the weaknesses of their errant nodes. Furthermore, the verbphrase-like structure would very likely adapt to one more closely resembling those found in standard CFGs if enough time were allotted to eradicate its "bit a" option.

The seemingly persistent occurrence of extraneous nodes may be an inherent penalty paid for the over-use of mutation. However, minor discrepancies such as these do not necessarily invalidate the grammar, as

> ... the very elegance and simplicity of [an inferred grammar] is rather more evidence *against*, than evidence for, it being the grammar our brain is built to use ... [since] adaptations are typically *not* maximally efficient engineering solutions to the problems they solve. (Devitt & Sterelny 1987, 145–146)

That slight perturbations are so often present in inferred grammars, irrespective of the approach adopted for their construction, is the consequence of building a description based on facts rather than speculation or intuition (Stich 1980). Grammar induction as a whole seems to be bringing an end to the view that a good grammar will generate all and only the grammatical strings of a language (M. & K. 1987).

Discussion

Recent years have seen a great deal of interest and activity in developing new approaches to learning for natural language processing. A variety of connectionist models have been devised that are apparently able to learn the underlying structure of a subset of English expressions (Rager & Berg 1990). While it is often difficult to assess the quality of a model within the resulting multi-dimensional maps, progress towards representation in classical terms is being made (Moisl 1992), and the full benefit of connectionist techniques may gradually become more apparent.

Symbolic approaches have long been the principal methodology for grammar induction. In general, they seek to characterize language structure through basic pattern recognition techniques, and

have proven quite useful when applied to artificial languages, such as the L-strings of graphics programming (Nevill-Manning, Witten, & Maulsby 1994). However, repeating patterns of more than four or five words are uncommon in samples of natural language, thus models produced in this way tend to be insufficient for processing novel sentences even when very large corpora are used for training. In addition, unlike connectionist models, frequent patterns do not reinforce the model any more than infrequent ones. Once a concept is learned, it tends to persist unchanged throughout the training period.

Statistical models attempt to circumvent problems arising from novel word patterns by assigning small probabilities to all unseen n-grams. But ungrammatical patterns are consequently viewed as being just as likely as unseen grammatical ones and, because any possible word combination must be assigned some probability, the model eventually degenerates to a complete graph. Even so, this comprehensive nature of statistical models makes them extremely robust when processing novel expressions. In addition, frequently observed n-grams reinforce learned concepts, and the resulting model can greatly assist processing tasks, such as parsing, by directing searches to more likely solutions first.

The GA described in this paper is a compromise between these three approaches. Like connectionism, it is an adaptive process whereby the model is gradually conditioned by the training set. Recurring patterns help to reinforce partial inferences, but intermediate states of the model may include incorrect generalisations that can only be eradicated by continued evolution. This is not unlike the developing grammar of a child which includes mistakes and overgeneralisations that are slowly eliminated as their weaknesses are made apparent by increasing positive evidence.

Like statistical methods, a GA is sensitive to pattern frequency. The more often a learned concept can account for substrings of the sample expressions, the more likely it will persist as a recurring feature in subsequent generations of the model. From an evolutionary programming perspective, the presence of a concept in a proportionately large number of models in the population also gives it a higher probability of continued survival.

Processors using the grammar produced by this GA do not have explicit access to the frequencies used in its construction and thus are not able to take advantage of this information for processing tasks. The output of the GA is instead a grammar that, as has been shown, is easily translated into the CNF representation typical of symbolic learners. Such grammars can in principle be incorporated into any existing processor that uses CNF grammars. The GA we have described is an effective, fault-tolerant method for inferring practical natural language grammars.

References

Bickel, A. S., and Bickel, R. W. 1987. Tree structured rules in genetic algorithms. In Davis, L., ed., *Genetic Algorithms and Simulated Annealing*. Pittman.

Charniak, E. 1993. *Statistical language learning*. Massachusetts: MIT Press.

Finch, S., and Chater, N. 1992. Bootstrapping syntactic categories using statistical methods. In Daelemans, W. & Powers, D., ed., *Background and Experiments in Machine Learning of Natural Language*, 229–236. Tilburg, NL.: ITK.

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

Koza, J. R. 1992. *Genetic Programming*. MIT Press.

Kupiec, J. M. 1989. Augmenting a hidden markov model for phrase-dependent word tagging. In *Proceedings of the 1989 DARPA Speech and Natural Language Workshop*, 92–98. Philadelphia: Morgan Kaufmann.

M., D., and K., S. 1987. *Language and Reality: An Introduction to the Philosphy of Language*. Oxford: Blackwell.

Mauldin, M. L. 1984. Maintaining diversity in genetic search. In *Proceedings of the National Conference on AI*, 247–250. AAAI.

Moisl, H. 1992. Connectionist finite state natural langugae processing. *Connection Science* 4(2):67–91.

Nevill-Manning, C. G.; Witten, I. H.; and Maulsby, D. L. 1994. Compression by induction of hierarchical grammars. In Storer, J. A., and Cohen, M., eds., *Proceedings of the Data Compression Conference*, 244–253. Los Alamitos, California: IEEE Press.

Rabiner, L., and Juang, B. H. 1993. *Fundamentals of speech recognition*. Prentice Hall.

Rager, J., and Berg, G. 1990. A connectionist model of motion and government in Chomsky's government-binding theory. *Connection Science* 2(1 & 2):35–52.

Smith, T. C., and Witten, I. H. 1995. Probability-driven lexical classification: A corpus-based approach. In *Proceedings of PACLING-95*. accepted.

Stich, S. 1980. Grammar, psychology, and indeterminacy. In Block, N., ed., *Readings in Philosophy of Psychology, Volume 2*. London: Methuen and Co. 208–222. reprint.

Wijkman, P. A. I. 1994. A framework for evolutionary computation. In *The Third Turkish Symposium on Artificial Intelligence and Neural Networks*.

Wyard, P. 1991. Context free grammar induction using genetic algorithms. In *Proceedings of the 4th International Conference on Genetic Algorithms*, 514–518.