Evolutionary Parsing for a Probabilistic Context Free Grammar

L. Araujo

Dpto. Sistemas Informáticos y Programación. Universidad Complutense de Madrid. Spain. lurdes@sip.ucm.es

Abstract. Classic parsing methods are based on complete search techniques to find the different interpretations of a sentence. However, the size of the search space increases exponentially with the length of the sentence or text to be parsed, so that exhaustive search methods can fail to reach a solution in a reasonable time. Nevertheless, large problems can be solved approximately by some kind of stochastic techniques, which do not guarantee the optimum value, but allow adjusting the probability of error by increasing the number of points explored. Genetic Algorithms are among such techniques. This paper describes a probabilistic natural language parser based on a genetic algorithm. The algorithm works with a population of possible parsings for a given sentence and grammar, which represent the chromosomes. The algorithm produces successive generations of individuals, computing their "fitness" at each step and selecting the best of them when the termination condition is reached. The paper deals with the main issues arising in the algorithm: chromosome representation and evaluation, selection and replacement strategies, and design of genetic operators for crossover and mutation. The model has been implemented, and the results obtained for a number of sentences are presented.

keywords: Evolutionary programming, Parsing, Probabilistic Grammar

1 Introduction

Classic parsing methods are based on complete search techniques to find the different interpretations of a sentence. However, experiments on human parsing suggest that people do not perform a complete search of the grammar while parsing. On the contrary, human parsing seems to be closer to a heuristic process with some random component. This suggest exploring alternative search methods in order to improve the efficiency. Another central point when parsing is the need of selecting the "most" correct parsing from the multitude of possible parsings consistent with the grammar. In such a situation, some kind of disambiguation is required. Statistical parsing helps to tackle the previous questions, that is, avoids an exhaustive search and provides a way of dealing with disambiguation.

Stochastic grammars [1], obtained by supplementing the elements of algebraic grammars with probabilities, represent an important part of the statistical methods in computational linguistics. They have allowed important advances in areas such as disambiguation and error correction. Another stochastic methods are genetic algorithms (GAs). They have been already applied to different issues of natural language processing. Davis and Dunning [3] use them for query translation in a multi-lingual information retrieval system. GAs have also been applied to the inference of context-free grammars [2]. Wyard [6] devised a genetic algorithm for the language of correctly balanced nested parentheses, while Smith and Witten [5] proposed a genetic algorithm for the induction of non recursive s-expressions.

This paper presents a stochastic parser based on a genetic algorithm which works with a population of possible parsings. The algorithm produces successive generations of individuals, computing their "fitness" at each step and selecting the best of them when the termination condition arises. Apart from the characteristic efficiency of these stochastic methods, the nature of the generation of solutions in a genetic algorithm brings the advantages of statistical approaches.

The rest of the paper proceeds as follows: Section 2 describes the evolutionary parser, presenting the main elements of the genetic algorithm; section 3 presents and discusses the experimental results, and section 4 draws the main conclusions of this work.

2 Evolutionary Parsing

The syntactic structure of a sentence is a necessary previous step to determine its meaning. Such structures assign a syntactic category (verb, noun, etc) to each word in the sentence and specify how these categories are clustered to form higher level categories (np, vp, etc) until building the whole sentence. The grammar specifies the permitted structures in a language. Context free grammars (GFGs), whose rules present a single symbol on the left-hand-side, are a sufficiently powerful formalism to describe most of the structure in natural language, while at the same time is sufficiently restricted as to allow efficient parsing.

Parsing according to a grammar amounts to assigning one or more structures to a given sentence of the language the grammar defines. If there are sentences with more than one structure, as in natural language, the grammar is ambiguous. Parsing can be sought as a search process that looks for correct structures for the input sentence. Besides, if we can establish some kind of preference between the set of correct structures, the process can be regarded as an optimization one. This suggests considering evolutionary programming techniques, which are acknowledged to be practical search and optimization methods [4].

Probabilistic grammars [1] offer a way to establish preferences between parsings. In a probabilistic CFG a weight is assigned to each rule in the grammar. The probability of each parsing is the product of the probabilities of all the rules used in the parsing. Probabilistic grammars not only offer a way to deal with issues such as ambiguity or ungrammaticality [1], but can also lead to an improvement in performance. Genetic algorithms and probabilistic grammars complement each other, for at least two reasons:

a) Large populations in a GA lead to a higher diversity at the expense of slowing down the convergence process, while higher percentages in the applications of genetic operators hasten the process but increase the selective pressure. The use of probabilistic grammars help to accelerate the convergence process. Although the selective pressure is increased for individuals composed of grammar rules of high probability, this will lead to better individuals for most sentences (since they will correspond to the most probable rules). Thus, in general there will not be a premature convergence to a wrong individual.
b) The nature of the GAs, which favours the exploration of new areas of the search space, helps to reach a correct result, even if the sentence to parse requires applying rules of low probability.

According to the previous considerations, a probabilistic GA has been designed, in which the parsings that compose the population correspond to a probabilistic CFG. When the algorithm finishes with correct parsings, the one for which the product of the probabilities of its genes is the largest is chosen. This is the answer of the algorithm for the most probable parsing of the sentence.

2.1 Chromosome Representation

Our system chromosomes represent parsings for the input sentence, corresponding to a fixed context-free grammar. The input sentence is given as a sequence of words with their set of categories attached to them (if they belong to several categories every of them is added). Nevertheless, this information could be easily obtained from a lexicon in a preprocessing step. Let us consider a simple example. The sentence “the man sings a song” will be given as *the(Det) man(Noun) sings(Verb) a(Det) song(Noun)*.

A chromosome is represented as a data structure containing the following information:

- Fitness of the chromosome.
- A list of *genes*, which represents the parsing of different sets of words in the sentence.
- The number of genes in the chromosome.
- The depth of the parsing tree.

Each gene represents the parsing of a consecutive set of words in the sentence. If this parsing involves no terminal symbols, the parsing of the subsequent partitions of the set of words is given in later genes. Accordingly, the information contained in a gene is the following:

- The sequence of words in the sentence to be analyzed by the gene. It is represented by two data: the position in the sentence of the first word in the sequence, and the number of words of the sequence.
- The rule of the grammar used to parse the words in the gene.
- If the right hand side of the rule contains no terminal symbols, the gene also stores the list of references to the genes corresponding to the parsing of these symbols.

```
NP: The man             NP: The man
VP: sings a song        VP: sings a song

NP -> Det,NP:           NP -> Adj,NP:
   Det: The                Adj: The
   NP: man                 NP: man

   NP -> Noun              NP -> Noun
      Noun: man               Noun: man

VP -> Verb, NP:         VP -> Verb, PP:
   Verb: sings             Verb: sings
   NP: a song              PP: a song

   NP -> NP, AP            PP -> Prep, NP
     NP -> Noun              Prep: a
        Noun: a              NP -> Noun
     AP -> Adj                  Noun: song
        Adj: song
Chromosome 1            Chromosome 2
```

Fig. 1. Possible chromosomes for the sentence *The man sings a song.* NP stands for nominal phrase, VP for verb phrase, *Det* for determiner, *Adj* for adjective, PP for prepositional phrase and AP for adjective phrase.

– The depth of the node corresponding to the gene in the parsing tree. It will be used in the evaluation function.

Figure 1 presents some possible chromosomes for the sentence of the example.

Initial Population The initial population consists of PS randomly generated (according to the probabilities of the different rules) individuals. The steps for the creation of chromosomes in the initial population are the following:

– The set of words in the sentence is randomly partitioned, making sure that there is at least one verb in the second part, which corresponds to the main VP.
– The set of words corresponding to the NP is parsed by randomly generating (consistently with the assigned probabilities) any of the possible NP rules. The same is done for generating the parsing of the VP with the VP rules. The process is improved by enforcing the application of those rules able to parse the right number of words of the gene.
– If the rules applied contain some non terminal symbol in its right hand side, the parsing process is applied to the set of words which are not yet assigned a category.
– The process continues until there are no terminal symbols left pending to be parsed.

2.2 Fitness: Chromosome Evaluation

Adaptation of individuals is revised after each new generation, testing the ability of every chromosome to parse the objective sentence. The evaluation of individuals is a crucial point in the evolutionary algorithms since the opportunities of an individual for survival depends on its fitness.

Fitness is computed as

$$\text{fitness} = \frac{\text{Number of coherent genes} - \sum_{i \in \text{incoherent genes}} \frac{\text{penalization}}{\text{depth}(i)}}{\text{Total number of genes}}$$

This formula is based on the relative number of *coherent* genes. A gene will be considered *coherent* if

a) it corresponds to a rule whose right hand side is only composed by terminal symbols, and they correspond to the categories of the words to be parsed by the rule.
b) it corresponds to a rule with non-terminal symbols in its right hand side and each of them is parsed by a coherent gene.

The formula takes into account the relative relevance of the genes: the higher in the parsing tree is the node corresponding to an incoherent gene, the worse is the parsing. Thus the fitness formula presents a penalization factor which decreases with the depth of the gene.

2.3 Genetic Operators

Chromosomes in the population of subsequent generations which did not appear in the previous one are created by means of two genetic operators: *crossover* and *mutation*. The crossover operator combines two parsings to generate a new one; mutation creates a new parsing by replacing a randomly selected gene in a previous chromosome. The rates of crossovers and mutations performed at each step are input parameters. The efficiency of parsing is very sensitive to them. At each generation a number of chromosomes equal to the number of offsprings is selected to be replaced. The selection is performed with respect to the relative fitness of the individuals: a chromosome with a worse than average fitness has higher chances to be selected for replacement. On the contrary, chromosomes adapted over the average have higher probability to be selected for reproduction.

Reproduction Crossover operator generates a new chromosome that is added to the population in the new generation. The part of one parent after a point randomly selected is exchanged with the corresponding part of the other parent to produce two offsprings, under the constraint that the genes exchanged correspond to the same type of parsing symbol (NP, VP, etc) in order to avoid wrong references of previous genes in the chromosome. Of course those exchanged which produce parsings inconsistent with the number of words in the sentence must be avoided. Therefore, the crossover operation performs the following steps:

- Select two parent chromosomes, C_1 and C_2.
- Randomly select a word from the input sentence.
- Identify the inner most gene to which the selected word corresponds in each parent chromosome.
- If the genes correspond to different sets of words, the next gene in the inner most order is selected. This process continues until the sequences of words whose parsings are to be exchanged are the same, or until the main NP or VP are reached.
- If the two selected genes parse the same sequence of words the exchange is performed.
- If the process to select genes lead to the main NP or VP, and the sequence of words do not match yet, the exchange can not be performed. In this case a new procedure is followed: in each parent one of the two halves is maintained while the other one is randomly generated to produce a parsing consistent with the number of words of the sentence. This produces four offsprings, out of which the best is selected.
- Finally, the offspring chromosome is added to the population.

Mutation Selection for mutation is done in inverse proportion to the fitness of a chromosome. Mutation operation changes the parsing of some randomly chosen sequence of words. The mutation operation performs the following steps:

- A gene is randomly chosen from the chromosome.
- The parsing of the selected gene, as well as every gene corresponding to its decomposition, are erased.
- A new parsing is generated for the selected gene.

3 Experimental Results

The algorithm has been implemented using C language and run on a Pentium II processor. In order to evaluate its performance we have considered the parsing of the sentences appearing in Table 1. The average length of the sentences is around 10 words. However, they present different complexities for the parsing, mainly the length and the number of subordinate phrases.

1	Jack(noun) regretted(verb) that(wh) he(pro) ate(verb) the(det) whole(adj) thing(noun)
2	The(det) man(noun) who(wh) gave(verb) Bill(noun) the(det) money(noun) drives(verb) a(det) big(adj) car(noun)
3	The(det) man(noun) who(wh) lives(verb) in(prep) the(det) red(adj) house(noun) saw(verb) the(det) thieves(noun) in(prep) the(det) bank(noun)

Table 1. Sentences used in the parsing experiments.

The results reported in this section have been obtained as the average of five runnings with different seeds. Results show that in most cases the correct parsing is reached in a small number of steps, less than 10 for populations of above 300 individuals.

Results obtained with a deterministic context free grammar are compared to the ones obtained using a probabilistic grammar. Figure 2 shows the results obtained for the sentences when using rates of crossover of 50% and one of mutation of 20%. Results clearly improve in all the cases by using the probabilistic grammar. The first observation is that while the deterministic CFG produces irregular convergence processes, the probabilistic one leads to highly regular processes as the population size grows. This indicates a higher robustness of the genetic algorithm. The difference between the results from the two kinds of grammar increases with the complexity of the sentence. Thus while the deterministic CFG leads to quick convergence for the sentence 1, the process is quite irregular for sentences 2 and 3. Another observation is that a threshold population size is required to achieve convergence.

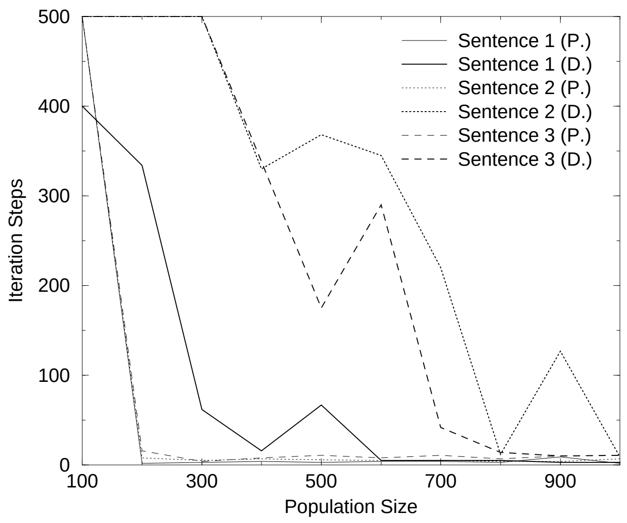

Fig. 2. Number of iteration required to reach the correct parsing with a probabilistic grammar (P) and a deterministic grammar (D).

The most relevant GA parameters have been studied (data not shown). It is clear that the population diversity and the selection pressure are related to the *population size*. If the population size is too small the genetic algorithm will converge too quickly to a bad result (all individuals correspond to similar incorrect parsings), but if it is too large the GA will take too long to converge. Results show that the behavior is quite different for each sentence: the higher the "sentence complexity", the larger the population size required to reach the correct parsing in a reasonable number of steps. The sentence complexity depends on its length and on the number of subordinate phrases it contains. Besides, as the population size increases, higher rates of crossover and mutation are required to increase the efficiency of the algorithm.

4 Conclusions

This paper presents a genetic algorithm that adapts a population of possible parsings for a given input sentence and a given grammar. Genetic algorithms allow a statistical treatment of the parsing process, providing at the same time the typical efficiency of stochastic methods.

Results from a number of tests indicate that the GA is a robust approach for parsing positive examples of natural language. A number of issues of the GA have been tackled, such as the design of the genetic operators and a study of the GA parameters. The tests indicate that the GA parameters need to be suitable for the input sentence complexity. The more complex the sentence (length and number of subordinate phrases), the larger the population size required to quickly reach a correct parsing.

Probabilistic grammars and genetic algorithms have been shown to complement each other. The use of a probabilist context free grammar instead of a deterministic one for the generation of the population of parsings in the algorithm has been investigated. Results obtained for these experiments show a clear improvement in the performance. For short sentences, though, greedy parser algorithm can be at least as fast. Nevertheless, the method proposed herein also allows dealing with problems such as ambiguity or ungrammaticality, and are expected to be advantageous for parsing long texts. Work along this line is currently in process.

References

1. E. Charniak. *Statistical Language Learning.* MIT press, 1993.
2. Paul Cohen and Ed Feigenbaum. Grammatical inference. In *HandBook of Artificial Intelligence*, volume 3, pages 494–511. Pitman Books Limited, 1984.
3. T. Dunning M. Davis. Query translation using evolutionary programming for multilingual information retrieval II. In *Proc. of the Fifth Annual Conf. on Evolutionary Programming.* Evolutionary Programming Society, 1996.
4. Z. Michalewicz. *Genetic algorithms + Data Structures = Evolution Programs.* Springer-Verlag, 2nd edition, 1994.
5. I.H. Witten T.C. Smith. A genetic algorithm for the induction of natural language grammars. In *Proc. IJCAI-95 Workshop on New Approaches to Learning Natural Language*, pages 17–24, Montreal, Canada, 1995.
6. P. Wyard. Context free grammar induction using genetic algorithms. In *Proc. of the 4th Int. Conf. on Genetic Algorithms*, pages 514–518, 1991.