Learning Grammar Weights Using Genetic Algorithms

Ingo Schröder and **Horia F. Pop** and **Wolfgang Menzel** and **Kilian A. Foth**

Fachbereich Informatik, Arbeitsbereich NATS
Universität Hamburg, Vogt-Kölln-Straße 30, 22527 Hamburg, Germany
`schroeder|hfpop|menzel|foth@informatik.uni-hamburg.de`

Abstract

Experiments are described in which the parameters of a weighted constraint dependency grammar are learned using genetic algorithms. The results suggest that it is possible to improve (both in terms of quality and parsing time) a manually prepared grammar as well as to derive the weights from a random initial assignment.

1 Introduction

The application of soft computing techniques to natural language processing has a number of advantages, namely the increased robustness against non-standard utterances and the ability to accomodate conflicting information such as preferences or defaults in the model. On the other hand, they introduce the problem of finding appropriate values for the free parameters by means of a suitably designed training procedure. Probabilistic approaches typically compute their parameters by means of well-established estimation procedures independently of the probabilistic processing component itself, often using frequency counts from a set of training data. Other approaches that are based on alternative measures (such as certainty or penalty factors, membership degrees or various kinds of hierarchical preference ordering schemes) require qualitatively different training methods to optimize their behaviour on a given data sample. In particular, since no independent measure of model quality is available, they must allow the inclusion of the processing model into the optimization procedure.

A grammar model of the latter kind is weighted constraint dependency grammar (WCDG) (Heinecke *et al.* 98) for which so far the weights have usually been assigned manually by a grammar writer. Since this is a difficult and error-prone task, an attempt has been made to determine the weights using an evolutionary approach. It tries to optimize the parsing performance on a given training corpus by means of a genetic algorithm.

2 Weighted Constraint Dependency Grammar

Constraint Dependency Grammars (CDG) map natural language utterances to tree-like configurations that describe the structures of a sentence on different levels of analysis, e. g., syntax and/or semantics. CDG have been used by various researchers with different aims during the last decade, namely interactive machine translation (Maruyama 90), speech recognition (Harper *et al.* 00) as well as robust parsing and error diagnosis (Menzel & Schröder 98).

It is characteristic for CDG that the initial space of possible structures is limited only by the set of labelled trees with the words as nodes. Thus, the only prior condition on structures is that words have to be linked to other words (or to a virtual node, the root) and that these linkages have to be labelled. Even if one restricts oneself to connected trees, there are still $(n \cdot l)^{n-1}$ possible structures per level for a sentence with n words and l different labels. From this large set of initial structure candidates, those are selected as the solution that only violate the constraints of the grammar in some minimal way.

In WCDG, constraints can be divided into hard and soft ones. Hard constraints may never be violated by a solution, while soft constraints are used to rank those structures that fulfill all hard requirements.[1] Constraints have weights (from the interval $[0, 1]$) associated with them, and structural configurations are scored using the product of the weights of the violated soft constraints.

WCDG have a number of desirable properties. In particular, they allow without further provision for robust parsing, the inclusion of preferences into the reasoning mechanism and an integrative treatment of different knowledge sources; additionally, they facilitate incremental parsing (Foth *et al.* 00) and have potential for time-adaptive behavior.

[1] The original definition of CDG (Maruyama 90) used only hard constraints.

Because there are usually very many constraints and because the consequences of their interaction are hardly foreseeable, the assignment of appropriate weights to constraints is a labor-intensive and error-prone task. Furthermore, the extension of a given grammar requires revising old constraints and re-adjusting their weights.

3 Learning weights for CDG

3.1 Evolutionary optimization

Genetic algorithms (Michalewicz 96) are optimization techniques inspired by biological systems. They maintain an evolving population of candidates, so-called individuals, throughout an iterative process, i. e., from generation to generation. The initial population either is an educated guess or consists of randomized individuals. Each new generation is created by first adding individuals resulting from recombination and mutation to the population and then selecting the "fittest" ones for the next cycle.

Evolutionary optimization techniques seem appropriate for the problem at hand for two reasons. On one hand, the large size of the search space prohibits the use of traditional (systematic and non-systematic) search methods. Genetic algorithms, in contrast, explore only a small portion of the neighborhood. It is this selective nature that makes it feasible to find an approximate solution in a resonable amount of time. On the other hand, the randomness in genetic algorithms allows jumps in the search space, making it possible to break out of local minima.

3.2 Genetic operators

Each individual consists of the whole set of weights for one particular grammar instance, i. e., a list of real-valued numbers.[2] Two different genetic operators are applied to the population (see Figure 1): crossover and mutation.

$$\begin{array}{c}
.1\ \ .2\ \ .3\ \ .4 \quad \rightarrow \mathrm{M} \rightarrow \quad .1\ \ \mathbf{.25}\ \ .3\ \ \mathbf{.2} \\[1em]
\left.\begin{array}{cccc} .1 & .2 & .3 & .4 \\ .8 & .6 & .3 & .2 \end{array}\right\} \rightarrow \mathrm{AC} \rightarrow \quad \mathbf{.45}\ \ \mathbf{.4}\ \ .3\ \ \mathbf{.3} \\[1em]
\left.\begin{array}{cccc} .1 & .2 & .3 & .4 \\ .8 & .6 & .3 & .2 \end{array}\right\} \rightarrow \mathrm{MC} \rightarrow \left\{\begin{array}{cccc} \mathbf{.1} & \mathbf{.6} & \mathbf{.3} & \mathbf{.4} \\ \mathbf{.8} & \mathbf{.2} & \mathbf{.3} & \mathbf{.2} \end{array}\right.
\end{array}$$

Figure 1: Illustration of different genetic operators

[2]Assuming that the constraints themselves are fixed, an individual completely describes one grammar instance.

Crossover is the analogue of reproduction in nature, and for each pair of individuals two different crossover operators are used (with probabilities p_{mc} and p_{ac} respectively):

Multipoint crossover (MC): The genes of the two involved individuals, i. e., their lists of weights, are "intertwined".

Arithmetic crossover (AC): The descendant is the pairwise weighted arithmetic mean of the two involved individuals.

Mutation (M) is applied to single individuals and can be subdivided into three types (with probability p_r, p_a and p_z respectively):

Relative mutation: A weight is randomly increased or decreased.

Absolute mutation: A weight is assigned a completely random number.

Mutation-to-zero: The weight is set to zero.

During the whole process, the population size is kept constant. Because the individuals produced by the genetic operators are added to the population, a mechanism is needed to select the population members for the new generation from the current population. Selection is based on a fitness measure which is defined as a weighted mean of the f-measure[3] (80%) and a time index[4] (20%): (1) Include the five fittest individuals in the next generation. (2) Discard the five worst individuals. (3) Fill up the new population with randomly selected individuals where the chance of survival of an individual is proportional to its fitness. This procedure guarantees that the best individuals are not lost and the genetic diversity is maintained.

4 Experiments

Two experiments were conducted using a small training set of 220 utterances taken from the Verbmobil domain. In order to estimate what a trivial algorithm can achieve on our data, we counted the number of "trivial" edges (with root attachement and null label) in the annotations. Using a random selection of lexical alternatives a baseline f-measure of 47.9% has been obtained.

[3]The f-measure is an aggregated measure for the quality of a parsing solution: f-measure = 2/(1/accuracy + 1/recall).

[4]The time index is a measure for parsing speed with zero meaning immediate completion and one meaning that the process uses the maximum amount of time allowed by a hard time limit that is necessary for practical reasons.

4.1 Improving a manual grammar

The first experiment tries to answer the question whether it is possible to improve the weights of a carefully crafted manual grammar with 492 constraints (free parameters). In an educated guess the following experimental parameters were selected: $p_{ac} = 0.25$, $p_{mc} = 0.25$, $p_r = 0.02$, $p_a = 0.002$ and $p_z = 0.002$.

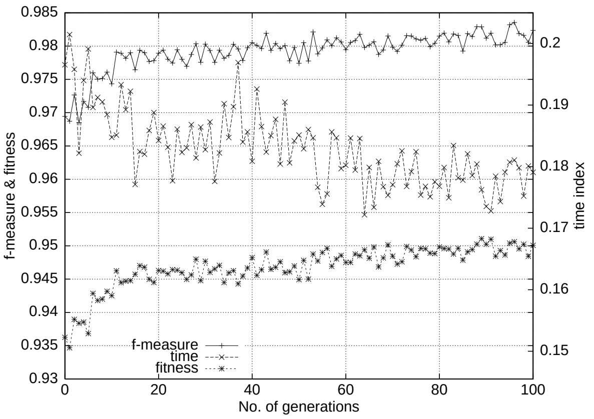

Figure 2: Results for the first experiment

Figure 2 presents the results; the following observations can be made. The f-measure increased from 96.9% to 98.4%. Processing time decreased slightly. However, since the time index was measured over the corpus and may be dominated by some few lengthy parsing runs, no further conclusions can be drawn. The resulting fitness increased from 93.6% to 95.1%. The improvements are small, but it should be noted that the original grammar was already heavily hand-tuned and that the numbers are close to 100%.

4.2 Finding soft weights automatically

The encouraging results from the first experiment suggested a more ambitious goal: Can a similar level of performance be achieved without the hand-crafted initial weights? In the second experiment, soft constraints receive random weights in the initial population; however, we assume here that the grammar designer deliberately distinguished between hard and soft constraints. Therefore, absolute mutation and mutation to zero were de-activated reducing the number of free parameters to 277. The experimental parameters were changed slightly multiple times after the process seemed to enter a saturation phase.

As can be seen from Figure 3, the f-measure[5]

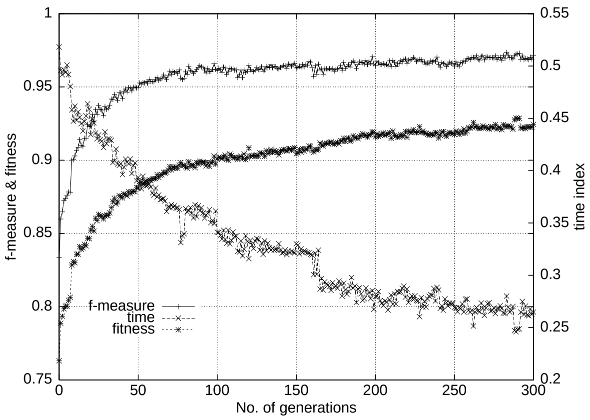

Figure 3: Results for the second experiment

started at 83.3% and finally reached 97.4%; the total processing time decreased by 52.5%. The fitness rose from 76.3% to 92.8%. Two observations can be made. The learning process was able to reach a similar level of performance as the original manual grammar. Even though considerable improvement was achieved and three times as many generations were tested, it was not possible to attain the quality of the optimized grammar that resulted from the first experiment.

4.3 Results on test set

The results reported so far are intra-experimental, i. e., they are tested on the training set. In order to judge the evolved grammars independently, we tested the individuals from the experiments on a held-out test set of 90 more utterances. For this problem set, the manual grammar has a fitness of 91.5%. This value increases for the final individual of the first experiment to 92.3%. The best random assignment of weights that was used to start the second experiment scored a fitness of 77.2% and improved to a fitness of 91.6% (cf. Figure 4). In either case, both a speedup of analysis and an increased f-measure were observed.

In order to confirm the robustness of these results with respect to test set variations, the test set was randomly split into two subsets. Both subsets showed exactly the same tendencies and varied only within a three percent margin. A verification of these results on a larger test set seems desirable.

5 Related work

5.1 Optimality Theory

Optimality Theory (Prince & Smolensky 93) is a linguistic theory that declares exactly those struc-

[5] The number of generations was increased to 300 for the second experiment.

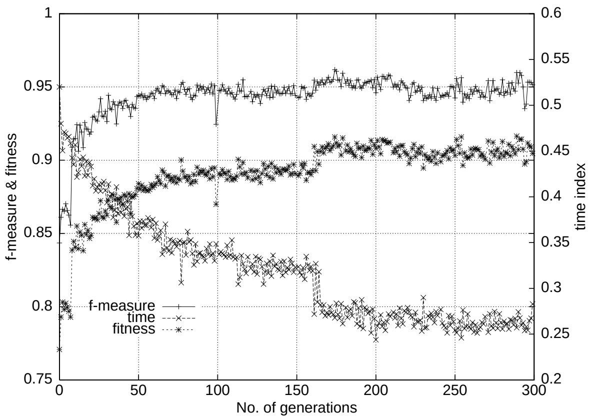

Figure 4: F-measure, time and fitness of the best individuals taken from each generation of the second experiment evaluated on the test set

tures as well-formed that violate only "less important" constraints than any other structure for the utterance. Constraints are strictly ordered and it is this hierarchy that determines their importance. It is assumed that the constraints themselves are universal while the ordering is language-dependent.[6] Although it can be argued that it is a simpler task to decide on a hierarchy of constraints than on individual weights, it nevertheless remains a difficult job for the grammar writer to find the correct constraints and their ordering in OT. So far as learning the hierarchical ordering is concerned, only selected problems within phonology have been studied, mainly from the (cognitively interesting) perspective of unsupervised learning (Tesar & Smolensky 98).

5.2 Statistical NLP

This paper focusses on the task of learning parameters for a given grammar. The more general problem of finding a complete grammar (including the rules themselves) has been tackled by various approaches in the field of statistical NLP. For probabilistic context-free grammars (the traditional representative of this type), the rules are extracted from an annotated corpus and the weights are computed from their relative frequencies in the corpus. If no annotated corpus is available, however, the parameters for the context-free rules can be approximated by an efficient training algorithm (Charniak 93). Lankhorst (93) describes an approach for induction of context-free grammars using genetic algorithms, but deals with small artificial languages only.

[6]A brief comparison between OT and CDG can be found in Schröder *et. al.* (00). .

Collins (97) describes a probabilistic model for dependency parsing based on lexical as well as tag affinities and linear distance. Eisner (96) additionally investigates models based on selectional preferences as well as decisions made during language production, but restricts himself to unlabelled trees. Carroll & Charniak (92) go one step further by trying to learn complete rules for dependency grammars from unannotated corpora. However, they train from artificially created data.

The main difference between statistical methods and the work reported in this paper is that the former calculate the parameters more directly, i. e., in a well-defined deterministic way, and can therefore guarantee a certain rate of convergence. Genetic algorithms, in contrast, rely on the randomized generation of new candidates and a powerful selection strategy. Statistical methods are only applicable if large corpora (most often annotated) are available, since they usually rely on frequency counts in the training set that are used to estimate model parameters. In these models, a separate parameter is needed for each combination of grammatical atoms (e. g., words, tags, grammatical labels etc.) which results in a serious sparse data problem and requires additional back-off or smoothing techniques. In contrast, constraints in our models are (manually derived) abstractions of such combinations and therefore substantially fewer parameters have to be learned. Furthermore, statistical approaches usually restrict themselves to a single level of representation.

5.3 Rule induction for CDG

Constraint induction for CDG has been tried in two related ways so far. Schröder (96) uses an annotated corpus to find a set of constraint instances by means of a small set of constraint templates and smoothed relative frequencies. However, the approach suffers from too small a corpus and limited constraint templates. For hard constraints, White (00) introduces so-called abstract role values which are derived from example utterances. An abstract role value covers a whole set of actual configurations and can therefore be used to generalize the characteristics of the training set.

6 Conclusion

Genetic algorithms based on the principle of "survival of the fittest" have been used to learn the weights of a constraint dependency grammar from a corpus of annotated utterances. The experiments conducted so far have demonstrated a remarkable potential of such evolutionary techniques for improving existing grammars. Improvement has been observed both in terms of parsing quality (as determined by an increase of the f-measure) and as a remarkable speedup in processing time. Most surprisingly, these promising results have been obtained using comparatively little training data. Significant improvements were possible for two quite different initial setups: (a) a manually weighted grammar and (b) a grammar where all soft constraints received an initial assignment of random weights. Although the initial grammar represented a considerable amount of human work, a small but noticable increase of quality and time has been achieved for the first case. In the second case, a level of perfomance was reached close to that of the initial hand-tuned grammar. Both results have been confirmed on an independent test set. Thus, genetic algorithms might well be used as a fully automatic substitute for this difficult and error-prone subtask of grammar development. More experiments, however, will be necessary to challenge the obvious saturation phenomenon observed in both experiments. In particular, a further improvement can be expected if higher mutation rates and/or different recombination schemata are temporarily introduced into the optimization process. This behavior has already been observed after changing the experimental parameters in the second experiment. Ideally, such strategies should make it possible to converge to the same level of quality, irrespective of the initial assignment of constraint weights.

References

(Carroll & Charniak 92) Glenn Carroll and Eugene Charniak. Two experiments on learning probabilistic dependency grammars from corpora. In *Proc. of the Fall Symposium on Probabilistic Approaches to Natural Language, American Association for Artificial Intelligence*, 1992.

(Charniak 93) Eugene Charniak. *Statistical Language Learning*. The MIT Press, Cambridge, MA, 1993.

(Collins 97) Michael John Collins. Three generative, lexicalised models for statistical parsing. In *Proc. of the Second Conference on Empirical Methods in Natural Language Processing*, Brown University, Providence, USA, 1997.

(Eisner 96) Jason M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *Proc. of the 16th International Conference on Computational Linguistics*, pages 340–345, Copenhagen, Denmark, 1996.

(Foth *et al.* 00) Kilian Foth, Wolfgang Menzel, Horia F. Pop, and Ingo Schröder. An experiment on incremental analysis using robust parsing techniques. In *Proc. of the 18th International Conference on Computational Linguistics*, pages 1026–1030, Saarbrücken, Germany, 2000.

(Harper *et al.* 00) Mary P. Harper, C. M. White, Wen Wang, Michael T. Johnson, and Randall A. Helzerman. The effectiveness of corpus-induced dependency grammars for post-processing speech. In *Proc. of the First Conference of the North American Chapter of the Association for Computational Linguistics*, pages 102–109, Seattle, WA, 2000.

(Heinecke *et al.* 98) Johannes Heinecke, Jürgen Kunze, Wolfgang Menzel, and Ingo Schröder. Eliminative parsing with graded constraints. In *Proc. of the Joint Conference COLING/ACL-98*, Montréal, Canada, 1998.

(Lankhorst 93) Marc M. Lankhorst. A genetic algorithm for the induction of context-free grammars. In *Proc. of the fourth CLIN meeting*, Groningen, 1993.

(Maruyama 90) Hiroshi Maruyama. Structural disambiguation with constraint propagation. In *Proc. of the 28th Annual Meeting of the Association of Computational Linguistics*, pages 31–38, Pittsburgh, PA, 1990.

(Menzel & Schröder 98) Wolfgang Menzel and Ingo Schröder. Model-based diagnosis under structural uncertainty. In Henri Prade, editor, *Proc. of the 13th European Conference on Artificial Intelligence*, pages 284–288, Brighton, UK, 1998.

(Michalewicz 96) Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer Verlag, Berlin, 1996. Third, Revised and Extended Version.

(Prince & Smolensky 93) Alan Prince and Paul Smolensky. Optimality theory: Constraint interaction in generative grammar. Technical Report RuCCS #2, Rutgers University, Center for Cognitive Science, Piscataway, NJ, 1993.

(Schröder 96) Ingo Schröder. Integration statistischer Methoden in eliminative Verfahren zur Analyse von natürlicher Sprache. Diplomarbeit, Fachbereich Informatik, Universität Hamburg, 1996.

(Schröder *et al.* 00) Ingo Schröder, Wolfgang Menzel, Kilian Foth, and Michael Schulz. Modeling dependency grammar with restricted constraints. *International Journal Traitement Automatique des Langues: Grammaires de dépendance*, 41(1):97–126, 2000.

(Tesar & Smolensky 98) Bruce Tesar and Paul Smolensky. The learnability of optimality theory. *Linguistic Inquiry*, 29(2):229–268, 1998.

(White 00) Christopher M. White. Rapid grammar development and parsing: Constraint dependency grammar with abstract role values. PhD Thesis, Purdue University, 2000.