Chapter 7

Conclusion

In the preceding chapters, experiments have been conducted to answer the question posed in the introduction chapter on how a finite-state approximation of a wide-coverage natural language parser can be constructed and used in order to improve parsing performance. In chapters 2 and 3, the principles of finite-state models and inference, in particular the inference of a HMM, were explained. In the rest of the work, the use of inferred finite-state models to increase parsing performance was discussed. Now the results attained in these chapters will be presented as an answer to the central research question.

7.1 Approximation through inference

The method of inference from parser-generated data was shown to be an appropriate means of arriving at a finite-state approximation of a wide-coverage parser for the following reasons.

First, a wide-coverage parsing system may contain separate modules to deal with various aspects of parsing. As the process of approximation through inference is an indirect one in which a model is derived from data created by some system, inference will capture the behavior of the system as a whole without having to deal with the separate modules. (An example of such a module is Alpino’s stochastic disambiguation model.) This also implies that the technique can be used in quite different parsing systems; their internal workings and type of grammar used are not an issue, as only the annotated output is relevant to the process of inference.

Second, inference is an automatic method (as opposed to constructing a finite-state representation of grammar rules by hand) and as such it allows for easy construction of an approximation even if the original system is continuously undergoing modifications.
Third, stochastic models such as the HMM inferred from annotated data will become more accurate as more data is available, and as the process is automatic, large amounts of data can relatively easily be generated using the parser. Such amounts of data would not be as easily obtainable if these had to be annotated by hand.

7.2 Reduction of lexical ambiguity

The above three conclusions regarding finite-state approximation through inference where drawn out of the positive results of experiments on lexical ambiguity reduction through POS tag filtering. The Alpino wide-coverage parser was approximated in a hidden Markov model by means of inference. The POS tagging HMM was then used in a POS tagger to reduce the number of tags assigned by the Alpino parser during lexical analysis. In this way, the POS tagging behavior of the parser as reflected in the training data was applied before full parsing took place. A large increase in parsing efficiency was observed when using the POS tag filter. At the same time, parsing accuracy increased slightly. These results are shown in table 7.1.

<table>
<thead>
<tr>
<th></th>
<th>parser accuracy</th>
<th>mean CPU time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>using filter</td>
<td>85.0%</td>
<td>14.1</td>
</tr>
<tr>
<td>not using filter</td>
<td>82.8%</td>
<td>53.5</td>
</tr>
</tbody>
</table>

Table 7.1: Parsing results in terms of parser accuracy and mean CPU time per sentence, with and without the POS tag filter.

In using the HMM POS tagging model to do POS tagging for Dutch, it was observed that the most frequent tagging error involved a mix-up of infinitive and finite plural verb forms, which in Dutch look the same. In order to improve the model, in chapter 5 it was extended with contextual information aimed at avoiding this most frequent error. The necessary information, consisting of a flag for every position in the sentence indicating whether or not the finite verb for the current clause has already been encountered, was added by the parser to the training data from which the model was inferred. The extra information was incorporated in the model by adding it as a separate feature to the model, instead of the less attractive alternative of extending the tagset. In experiments, the error rate for this most frequent type of mistake, as well as the overall error rate, was reduced when using the extended model. Using the extended model in the POS tag filter, a small increase in parsing efficiency was observed.
7.3 Reduction of structural ambiguity

In chapter 6, the problem of structural ambiguity was addressed. A model was inferred from data annotated with NP chunk tags by the parser following the IOB2 chunk tagging scheme. This model was used to recognize chunk brackets in new sentences prior to parsing, and the parser was forced to use the brackets, reducing the number of possible parses for that sentence.

Using an approach in which chunk tags were assigned in two steps, first assigning POS tags to words and then assigning concatenations of POS tags and chunk tags to those POS tags, results were attained in stand-alone chunking on a standard WSJ data set that are comparable to other published results. However, the use of the chunker as a way of reducing ambiguity in the Alpino parser, as described above and using the current implementation of the idea, was not successful. Future work in this direction could still lead to improvements in this respect. Suggestions for further research are the use of a different type of chunk (as the one currently being used is relatively hard to recognize), use of an approach to assigning brackets based on recognized chunk tags that is better suited to the chunking techniques that currently achieve worse results than expected, telling the parser where a bracket is not allowed instead of where one should be, and experimenting with different models in addition to the current model that was also used in chapter 5.