Constraint Based Integration of Deep and Shallow Parsing Techniques

Michael Daum and **Kilian A. Foth** and **Wolfgang Menzel**
Natural Language Systems
Department of Computer Science
University of Hamburg
(micha|foth|wolfgang)@nats.informatik.uni-hamburg.de

Abstract

To investigate the contributions of taggers or chunkers to the performance of a deep syntactic parser, Weighted Constraint Dependency Grammars have been extended to also take into consideration information from external sources. Using a weak information fusion scheme based on constraint optimization techniques, a parsing accuracy has been achieved which is comparable to other (stochastic) parsers.

1 Introduction

To investigate the contributions of different shallow processing components like taggers or chunkers to the performance of a deep syntactic parser, Weighted Constraint Dependency Grammars (WCDG) have been extended to also take into consideration information from external sources. Constraints in the WCDG framework are (partially) defeasible conditions on admissible structural configurations in a dependency tree for a given natural language utterance. By defining conditions (1) on word forms and their positional relationships, (2) on links between word forms, and (3) on the labels attached to these links, they license certain dependency relations or combinations thereof and assign a score to them. These scores give an estimate of the appropriateness of a particular partial structure. Constraint optimization techniques are used to obtain the structure with the best overall score, which is chosen as the optimal representation of the utterance. Due to the use of weighted constraints throughout the grammar, WCDG comes with a number of advantages:

- A WCDG grammar is able to accomodate conflicting requirements as they can often be found in natural language, e.g. in case of ordering preferences. WCDG shares this property with Optimality Theory (Prince and Smolensky, forthc).
- Since constraints can be violated, the grammar also accepts unusual or even deviant input (usually with a lower global score) and assigns a structural interpretation to it.
- To a large degree a WCDG parser does not depend crucially on a complete set of constraints. Although performance degrades if some constraints are deactivated, it usually does so in a graceful manner.
- Constraints offer an ideal interface to integrate external processing components as additional sources of evidence into the decision about the optimal structural interpretation.

It is this last advantage which this paper is focussing upon. Shallow processing components like taggers or chunkers can be integrated into the deep analysis simply by providing additional constraints. If a scoring scheme is also available for such an external contribution (and it usually is), it can be combined with the internal scores assigned by the constraints of the grammar. Similar to the arbitration of conflicting requirements within the grammar itself, the general optimization procedure allows the WCDG parser to handle possible conflicts between predictions of different origin. This weak integration of hypotheses not only makes the approach an ideal platform for information fusion. Moreover the individual contributions certain information sources make to the overall performance of the parser can be exactly measured, simply by switching the corresponding constraints on or off.

2 The WCDG language modelling system

In *dependency grammar*, syntactic structure is modelled not by nested constituents but as a set of direct relations between two words. Usually an utterance is represented as a tree with as many nodes as there are words, the finite verb forming the root of the tree. To express different types of dependency relations, the vertices are often enriched with different labels, e.g. to distinguish subjects from objects.

Instead of giving generative rules about how to construct a dependency tree, a *Constraint dependency grammar* declares *constraints*, i.e. properties that a well-formed tree should fulfill. Finding a well-formed syntactic structure takes on the form of a *constraint satisfaction problem* (CSP), which can be solved by a variety of solution methods. This kind of dependency grammar was introduced by Maruyama (Maruyama, 1990).

The constraints about well-formedness are notated as logical formulas. A common rule is to postulate that subjects typically precede their finite verb and objects follow it. This can be expressed by requiring that for all edges with the label 'SUBJ', the governor must occur to the left of the modifier, and the inverse for edges labelled 'OBJA'.

Applying these constraints to an example analysis (Figure 1), we see that two of the dependencies actually violate them. Since this is not incorrect in German but merely marks a topicalization, it is useful to gradate the constraints. This is the defining characteristic of Weighted CDG (Schröder, 2001): The grammar writer can assign different weights to the constraints to indicate their relative strength. The weights of violated constraints are multiplied to obtain the score for an entire dependency tree. Violation of a constraint is tolerated if no other analysis has a higher overall score. This property leads to considerable robustness against ungrammatical or extragrammatical input. Note that a low figure for the weight corresponds to a strong constraint, while a high figure (near 1) has little influence under the multiplicative scheme.

Introducing constraint weights turns the CSP into a constraint optimization problem, which is considerably harder to solve, since one particular solution must be found rather than just one of a number of solutions. Furthermore, weighted constraints do not allow us to prune away an alternative, leading to very large search spaces. In fact, analyzing long sentences with WCDG usually leads to problems that are too large to be solved exactly. However, heuristic methods have been developed that can approximate the optimal solution, where the quality of the solution increases over time as more alternatives are tried (Foth et al., 2000).

3 Experimental setup

For our experiments we analysed 1845 unedited sentences from German online newscasts on buying or selling events, with an average length of 24 word forms (see Table 1 for a distribution of sentence lengths). We employed a handwritten WCDG of German that aims to perform a very thorough analysis; in addition to establishing syntax structure, it assigns each dependency one of 28 labels. It must also determine the exact word class and morphological form of each word. At the same time the lexicon was kept incomplete; of the open word classes of German, only the verbs are modelled while most nouns and adjectives are not[1]. Therefore, about 29% of all tokens are effectively unknown words and receive a totally underspecified representation.

In the experimental setup chosen we run the optimization process on each sentence, and interrupt it if it has not terminated by itself after three minutes. In this case, the best dependency structure established so far is returned as the parsing result. If no additional knowledge sources (like the tagger and the chunker) are used the parser terminates by itself after 134 seconds on average. By the measure of *labelled recall*, which counts how many dependencies are established correctly along with their labels, only 50.7% recall[2] is achieved on the average; although the WCDG usually assigns the desired analysis a near-optimal score, this analysis is often not found simply because the problem

[1] However, nouns and proper names can be recognized automatically by their capital initials.

[2] Since all possible dependency analyses for a sentence have the same number of dependency edges, recall and precision are always equal for a fully disambiguated tree. The term *accuracy* is sometimes used instead.

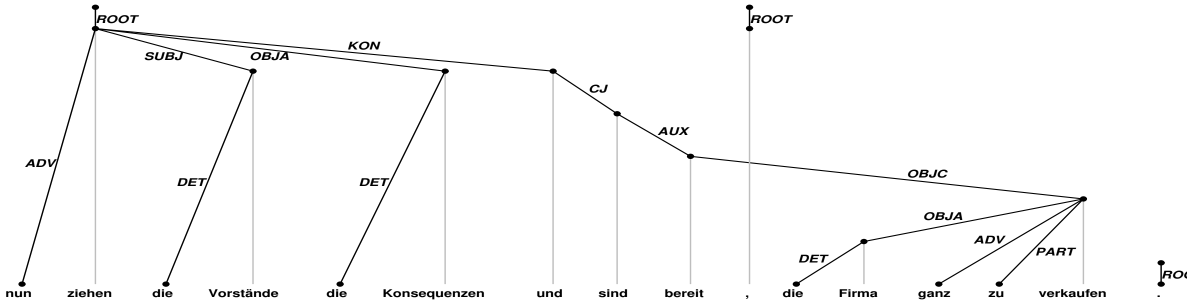

Figure 1: Example of a labelled dependency analysis for a German sentence *(Management now bears the consequences and agrees to sell the company outright.)*

space is too large. To measure the effect of the time limitation on accuracy, we conducted control experiments with the time limited to 6 minutes instead for comparison; the corresponding recall measures are given in Table 2 in brackets. They show that the time limit is not a great source of error.

Obviously the parsing process would benefit from additional information about the correct analysis, even if it is uncertain, as long as it can be produced with little effort. This is the motivation behind the following experiments: to integrate a shallow but efficient information source into a thorough but slow parser to see whether a net gain can be achieved.

length	... 10 ...	20 ...	30 ...	40 ...	50 ...	60 ...
# sentences	160	650	651	274	81	19	10

Table 1: Distribution of sentence length.

4 Integration of a tagger

The most common type of partial language analysis is *part-of-speech tagging*: annotating each word in the input string with its grammatical category. In contrast to *single taggers*, so-called *multi taggers* assign a word different categories with different probabilities. POS tagging is susceptible to statistical analysis, and very high accuracy has been reported on well-known corpora.

Since part of the task of WCDG analysis is to resolve lexical ambiguity, the information compiled by a POS tagger is directly helpful: We can write a constraint that penalizes any mismatch between the lexical category under consideration and the category predicted by the tagger. In the case of multi tagging the penalty is proportional to the score emitted by the tagger. In single tagging this algorithm can only assign a constraint weight of 0 or 1, since our tagger does not report the internal confidence value, so that all but one lexical variants are actually forbidden entirely. To reduce the impact of tagging errors in both tagging modes, the weight of the constraint can be smoothed to ensure that all other categories are merely punished but not eliminated (Foth and Hagenström, 2002).

By adding or omitting a single constraint in the existing dependency grammar, we can accurately measure how *useful* the contribution of the POS tagger is to the parsing process, independently of how *accurate* it is. As a first experiment, we simulated the ideal POS tagger (one which makes no mistakes at all) by giving the tagging constraint access to our manual annotations. Under these idealized conditions, we achieved a labelled recall of 75.8% under the same conditions as before. Also, processing took only a third of the time necessary before. Obviously, the POS information is very useful for solving the optimization problem posed by the WCDG.

Note that the information introduced by the POS tagger is not strictly new: The original grammar already contains rules about what syntactic categories can combine to build larger structures, and thus the parser already solves the tagging problem as part of a much harder problem. The introduction of explicit POS scores simply serves to guide the parser more quickly toward structures that are probably part of the optimal analysis.

Since an ideal POS tagger is unavailable, we then investigated how useful a less accurate but available information source can be. Among the freely available POS taggers for German, the one that performs best on our corpus of online newscasts is the statistical Markov tagger *Trigrams'n'Tags* (TnT) (Brants, 2000). Although trained on a similar corpus of German newspaper text, even this tagger achieves only 92.3% accuracy in multi-tagging mode. One reason for this is that the distribution of word categories differs considerably between the two corpora: Our sentences contain ten times as much foreign-language material (`FM`) as the tagger's training set, and also more names (`NE`), both of which are (in German) very difficult to distinguish from proper nouns. Therefore, in many cases the information provided by the tagger is actually misleading.

However, even though the accuracy of this information source is comparatively low, the benefit is still high. Supplying the tagging constraint with the output of TnT in multi-tagging mode yields a labelled recall of 73.7%. Comparing experiments 3 and 4 in Table 2 shows that almost the full benefit of having access to POS information can be reaped even if the POS tagger is markedly inaccurate. Using TnT in single-tagging mode achieves only 71.1% recall.

A complication arises because the scores introduced from shallow analysis change the original optimization problem. In the extreme case, if the POS tagger makes a serious error and forbids that variant of a word that is part of the desired analysis, a different analysis may become numerically optimal. This means that the optimization procedure would not find the preferred analysis even if no search errors ever occurred. But as noted above, knowledge about syntactic categories is provided redundantly by the constraint grammar as well as by the tagger. Errors of the tagger can thus be partially compensated if the combined evidence of other constraints is sufficient to keep the linguistically preferred structure optimal.

Also, many typical tagger errors do not affect the syntactic structure much, e.g. if the tagger identifies a noun as a proper name. Since both word classes can fill much the same roles, the new optimal analysis will be essentially the same as the old one, except for one lexical selection. Under the measure of labelled recall, this would not be an error. Only if a more disruptive error is made does the structure change drastically. For these reasons, the benefit of the POS tagger outweighs the cost of occasional misclassifications.

5 Integration of a chunker

Another source of shallow syntactic information are the partial parsers known as *chunk parsers*. The object of these is not to build a complete syntactic structure but only to identify coherent word sequences of particular types. The typical application of chunk parsing is the detection of noun phrases, e.g. for information extraction.

Simple chunk parsers only determine the extents of chunks, but neither the interrelationships between them nor their internal structure. However, they sometimes try to assign to each chunk a lexical head which can then represent the entire chunk. The idea is that once a chunk has been detected, only its head need be considered for attachment when building the complete syntax tree.

This notion allows us to write useful disambiguating constraints. Assume that the constraint engine knows, for each word, the extent of the chunk that it belongs to, and the head word of that chunk. A grammar can then constrain all dependency edges to those that either connect two chunk heads or attach a word to the head of its own chunk. Again, this condition can be expressed with a single unary constraint.

The chunk constraint allows the grammar to rule out on structural grounds many subordinations that would otherwise appear perfectly reasonable. Consider the case of determiners. Since German noun phrases can be nested, the distance between a noun and its determiner can become unpredictably large. Where several noun phrases occur in a sentence, this results in a combinatorial number of possible determiner relations, neither of which can be ruled out without considering the whole sentence. If the example sentence from Figure 1 is divided into chunks, each of the definite articles can be unambiguously paired with the following noun:

> Nun/ADV [VC ziehen/VVFIN] [NC die/ART Vorstände/NN] [NC die/ART Konsequenzen/NN] und/KON [VC sind/VAFIN] bereit/ADJD ,/$, [NC die/ART Firma/NN] komplett/ADJD [VC zu/PTKZU verkaufen/VVINF] ./$.

Thus the ambiguity about determiner attachment

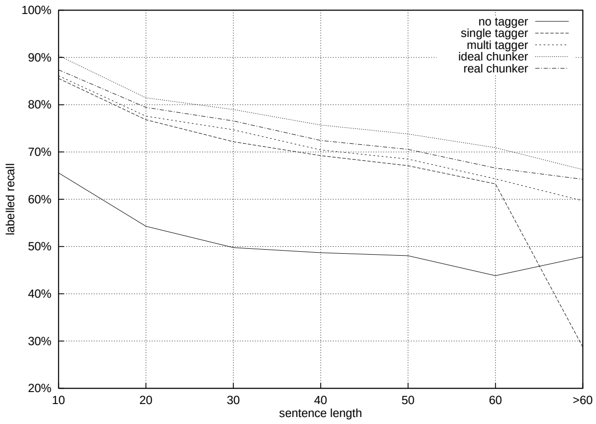

Figure 2: Labelled Recall per problem size.

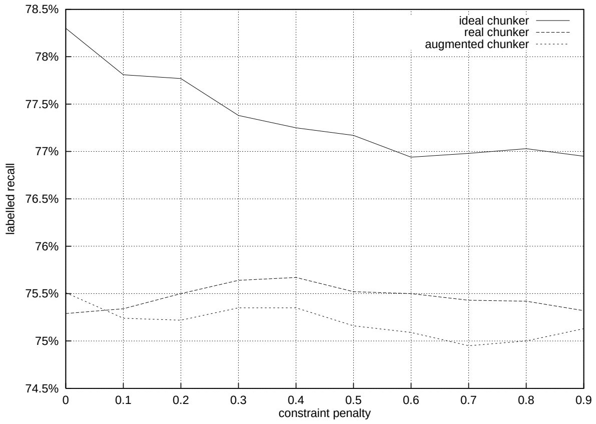

Figure 3: Labelled Recall of different chunkers.

is completely resolved. Many other syntactic relations can profit from similar reasoning.

To find out how much this additional information can contribute to the parsing process, we first simulated the ideal chunk parser by supplying our program with the correct chunk boundaries as present in our gold-standard annotations. As line 7 in Table 2 shows, the syntactic accuracy increases by about 5 percentage points. Obviously, this is the upper limit of utility that chunk information can supply in our case.[3]

To test the performance of an actual chunk parser we used the decision-tree based TreeTagger (Schmid, 1994), a freely available tool that is capable of assigning POS tags and of determining noun, verb and prepositional chunks in German text. Since its tagging accuracy is no better than TnT on our corpus, we only use it to extract chunk information. When run as a chunk parser for German, the TreeTagger correctly recalls 87.4% of the chunks present in our annotations. The chunks that it emits are correct in 82.3% of all cases.

Since the TreeTagger does not compute lexical head words we have to supply simple rules of our own to insert head markers into its output: in noun chunks, the leftmost noun or proper name is the head; in verb chunks, the finite verb is the head; failing that, the rightmost word is the head; in prepositional chunks, the preposition is the head.

When enriching the TreeTagger output with these markers, we have all the necessary information to implement the chunk constraint outlined above. Line 5 in Table 2 shows the results of supplying the grammar with the output of TreeTagger rather than a perfect chunk parser. It can be seen that the inaccuracy of its output has a severe impact on parsing accuracy: the benefit on parsing accuracy of using this chunk parser is only half of what would theoretically be possible.

The raw precision and recall of the TreeTagger may actually be misleading, since a WCDG parser does not explicitly establish any chunks. In fact, many mis-predicted chunk boundaries do not lead to violation of the chunk constraint and consequently cause no harm to the parser, other than not providing a possible benefit. It might be more reasonable to measure how many spurious constraint violations are actually caused by errors of the chunk parser. In the annotations of all test sentences, 702 out of 44099 syntactic dependency edges (1.6%) mistakenly trigger the chunker constraint because the TreeTagger disagrees with our annotations. Measured in this way, even an accuracy of 98.4% already halves the utility of the information.

As distributed, the TreeTagger consistently makes some questionable judgements that disagree with our grammar. For instance, it always groups postpositions with the following rather than the preceding noun phrase. We consider this a modelling error. Since we could not modify the TreeTagger itself, we filtered its output through a finite-state transducer that undoes this and some

[3]The chunk parser actually used in the following experiment requires tagged input to operate so that it is not possible to test its effect in isolation, i.e. without POS tagging. Therefore this idealized experiment also employs the TnT tagger.

	tagger	chunker	weight	rel. time%	unlabelled recall%	labelled recall%
1	none	none	–	100.0	58.2	50.7
2	single	none	–	35.2	75.7	71.1
3	multi	none	–	44.1	78.2	73.7
4	ideal	none	–	31.0	80.2	75.8
5	multi	real	0.4	39.4	80.0 (80.6)	75.7 (76.2)
6	multi	augmented	0.0	36.2	79.9	75.5
7	multi	ideal	0.0	33.4	82.7 (83.0)	78.3 (78.7)
8	ideal	ideal	0.0	22.4	83.9	79.7

Table 2: Influence of partial parsing information on the recall of a deep syntax analysis. Numbers in brackets are for experiments with a relaxed timeout condition.

other incorrect groupings.

There are also some coordinative constructions that the TreeTagger invariably groups incorrectly, such as the following:

> . . . [VC vermietet/VVPP] oder/KON [VC verkauft/VVPP worden/VAPP seien/VAFIN]
> *(. . . had been rented or sold)*

The final verb chunk appears very plausible, but since our grammar actually groups such coordinations as follows:

> [[vermietet/VVPP oder/KON verkauft/VVPP] worden/VAPP seien/VAFIN]

the obvious choice of verb chunks likewise leads to a constraint violation. Because coordinative constructions usually are much more complicated than this case, it would be difficult to handle them all in the output filter; instead we add an explicit exception in the chunker constraint that the complements of conjunctions may cross chunk boundaries to the left, thus anticipating the consistent error of the chunk parser.

With these automatic corrections in place, the number of chunker constraint violations is reduced to 477 out of 44099 (1.1%). As can be seen in line 6 in Table 2, a further benefit of these automatic corrections cannot be shown; in fact, they are often counter-productive, since the modified chunker constraint allows some more correct edges, but also many more incorrect ones.

A different way of guarding against errors in the shallow information source would be to assign a higher score to the constraint that integrates it. If edges that violate chunk boundaries are discouraged rather than forbidden, the other constraints may eventually override the chunk information if the evidence is strong enough, as with the integration of POS information. Of course, this weakens the positive effect of the information source in all other cases. Figure 3 shows that while the overall benefit can be raised for some constraint weights, the improvement is not great.

The question arises why chunk information as an information source suffers more from inaccuracy than POS information, particularly since some chunker errors do not change the optimization problem at all. On the other hand, where an incorrect POS tag affects primarily one lexical selection, an incorrect chunk bracket can penalize several correct syntactic edges at once; it usually forbids the desired structural alternative entirely, forcing the parser to interpret at least part of a sentence differently in a more fundamental way.

6 Related Work

Most current approaches to probabilistic parsing are based on the output of a tagger. Some systems parse only the sequence of single best tags (c.f. (Brants, 1999)). If, however, a lexically sensitive model is desired, the tagger is invoked only as fall back solution for unknown words (c.f. (Collins, 1999)). Both solutions differ from the approach adopted here in that they cannot explicitly measure the contribution of the tagging results to the overall performance of the parser.

(Charniak et al., 1996) performes a comparison of single tagging (where the tagger provides a unique tag assignment) to multi-tagging (which leaves the decision among several possible tags to the parser). He concludes that multi-tagging gives no significant advantage. This result has not been confirmed by our findings, which indeed show a small but reliable advantage of multi tagging over single tagging. The contrast might be explained by considering the different information sources used in both cases. (Charniak et al., 1996) combined two purely stochastic models, which probably capture rather similar kinds of model information from their training data, whereas in our case information of completely different origin has been brought together. While the tagger is geared towards exploiting sequential patterns in a sentence, the parser relies more on possible structural configurations and subcategorization information.

There has also been a number of investigations on how to integrate chunk-boundary information into a parsing procedure. For Italian, (Basili et al., 1998) segment a sentence using a chunker and a clause boundary detection based on verb subcategorization frames. Based on this information inter-chunk dependencies are computed using a special purpose parser which is sensitive to the type of chunks. Recall and precision for inter-chunk dependencies (not full dependency structures) of 75.2% and 72.1% respectively have been achieved on a small test set of 56 sentences (1149 words).

Since chunk-boundary information is ultimatively required by the parsing procedure and considered to be non-defeasible, no figures of merit are available which would allow estimation of the contribution of chunk boundaries to the overall behaviour of the system.

Such figures have been given, however, for an architecture which integrates shallow information (coming from a tagger, a named entity recognizer, an external semantic database and a topological parser) by guiding a deep HPSG-parser towards the most plausible interpretation (Cryssmann et al., 2002). This approach increased the coverage of the overall system from 12.5% to 22.1% on the NEGRA-corpus but failed to reduce the average number of analyses which even grew from 16.19% to 18.53% per sentence. In contrast, due to its inherent robustness properties our parser always returns the single best solution, i.e. it has a coverage of 100% and always achieves full disambiguation.

In this respect our approach is more similar to stochastic parsers. Here, evaluation results for the analysis of dependency relations have been published by (Collins, 1999) among others. In his parser dependency links can be recovered from the head-modifier relationships of a phrase-structure model with 91% accuracy for unlabelled dependencies measured on Wall Street Journal sentences taken from the Penn Treebank.

The parser has also been sucessfully ported to Czech, a highly inflected language with free word-order. Here 80% accuracy for unlabelled dependencies have been achieved (Collins et al., 1999), which is surprisingly close to our results on a German corpus. In the experiment for Czech an indirect approach has been pursued, converting the dependency structures of the Prague Tree Bank into Penn-Treebank-like phrase structure trees. In contrast to this approach, WCDG parses dependency structures directly.

Comparable results are available for a LFG-parser (Riezler et al., 2002) which uses partial parsing and skimming techniques to achieve full coverage, along with a stochastic parse selection model to guarantee full diambiguation. An f-score of 73.0% for the extraction of dependency relations for 700 Wall Street Journal sentences has been reported.

7 Conclusions

By integrating the results from two components for shallow syntactic analysis we were able to advance a dependency parser based on constraint optimization techniques to a new level of robust performance. Thanks to the weak integration of evidence from different sources, a high degree of tolerance against missing lexical information, reduced run time requirements and a considerably improved parsing accuracy have been achieved. Thus, processing of free running text became an option, and results have been obtained which are comparable to other approaches.

Since the system has not been composed as a strict sequence of components, but integrates evidence from different modules in a single decision procedure, the overall performance does not crucially depend on the availability of certain types of information. This would allow us to measure the contribution of individual information sources to the overall performance rather precisely. Three main conclusions can be drawn from the experiments conducted so far:

- Even unreliable information can be useful to contribute to a better performance.
- The system is much more sensitive against chunk errors, than it is against tagging errors.
- Delayed decisions, as in the case of multi tagging, might result in an advantage, as long as the different contributions come from rather complementary sources of evidence.

Several possibilities to further investigate weak information fusion within the hybrid architecture of a WCDG parser can easily be imagined. First of all, other types of chunkers should be considered. LoPar (Schmid and Schulte im Walde, 2002), for instance, which became recently available, can contribute recursive chunks and provides much better chunking results for coordinated structures. Moreover, LoPar does not depend on the availability of external tagging information. Therefore the net contribution of the chunker without an additional tagger could also be determined, a configuration which was not available so far. Furthermore, by providing the necessary constraints, completely different classes of shallow syntactic sources can be included as well. Thus the utility of evidence from a specialized attacher (e.g. for PP-attachment) or a model of lexical associations could be studied in more detail.

Acknowledgements

This research has been partially supported by Deutsche Forschungsgemeinschaft under grant Me 1472/4-1.

References

R. Basili, M. T. Pazienza, and F. M. Zanzotto. 1998. Efficient parsing for information extraction. In *Proc. 13th Europ. Conf. on Artificial Intelligence*, pages 135–139, Brighton, UK.

Thorsten Brants. 1999. Cascaded markov models. In *Proc. 9th Conf. of the Europ. Chapter of the ACL, EACL-99*, pages 118 – 125, Bergen, Norway.

Thorsten Brants. 2000. TnT — a statistical part-of-speech tagger. In *Sixth Applied Natural Language Processing Conf. (ANLP-2000)*, Seattle, WA, USA.

E. Charniak, G. Carroll, J. Adcock, A. Cassandra, Y. Gotoh, J. Katz, M. L. Littman, and J. McCann. 1996. Taggers for parsers. *Artificial Intelligence*, 85:45 – 57.

Michael Collins, Jan Hajič, Lance Ramshaw, and Christoph Tillmann. 1999. A statistical parser for czech. In *Proc. 37th Annual Meeting of the ACL, ACL-99*, pages 505 – 512, College Park, MD.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, Philadephia, PA.

B. Cryssmann, A. Frank, B. Kiefer, S. Müller, G. Neumann, J. Piskorski, U. Schäfer, M. Siegel, H. Uszkoreit, F. Xu, M. Becker, and H.-U. Krieger. 2002. An integrated architecture for shallow and deep processing. In *Proc. 40th Annual Meeting of the ACL*, pages 441 – 448, Philadephia, PA.

Kilian A. Foth and Jochen Hagenström. 2002. Tagging for robust parsers. In *2nd Workshop on Robust Methods in Analysis of Natural Language Data, ROMAND2002*, pages 21 – 32, Frascati, Italy.

Kilian A. Foth, Wolfgang Menzel, and Ingo Schröder. 2000. A Transformation-based Parsing Technique with Anytime Properties. In *4th Int. Workshop on Parsing Technologies*, pages 89 – 100, Trento, Italy.

Hiroshi Maruyama. 1990. Structural disambiguation with constraint propagation. In *28th Annual Meeting of the ACL*, pages 31 – 38, Pittsburgh.

A. Prince and P. Smolensky. forthc. *Optimality Theory: Constraint interaction in generative grammar*. Linguistic Inquiry Monograph Series. MIT Press.

S. Riezler, T. H. King, R. M. Kaplan, R. Crouch, J. T. Maxwell III, and M. Johnson. 2002. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proc. 40th Annual Meeting of the ACL*, pages 271 – 278, Philadephia, PA.

Helmut Schmid and Sabine Schulte im Walde. 2002. Robust german noun chunking with a probabilistic context-free grammar. In *Proc. 18th Int. Conf. on Computational Linguistics, COLING-00*, pages 726 – 732, Saarbrücken, Germany.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Int. Conf. on New Methods in Language Processing*, Manchester, UK.

Ingo Schröder. 2001. *Natural Language Parsing with Graded Constraints*. Phd thesis, Dept. of Computer Science, University of Hamburg, Germany.