Topological Field Chunking for German

Jorn Veenstra and **Frank Henrik Müller** and **Tylman Ule**
Seminar für Sprachwissenschaft, Universität Tübingen
{veenstra,fhm,ule}@sfs.uni-tuebingen.de

Abstract

In this paper[1] we compare three different approaches to the analysis of the basic structure in German sentences: the sentence brackets in the *topological field* framework in German (Höhle, 1986). The first approach is based on hand-written Finite-State Automata (FSA); the other two are trained on corpus data. One is a Probabilistic Context-Free Grammar (PCFG) approach, the other is a classification-based Memory-Based Learning (MBL) approach. The three approaches are evaluated on a manually annotated corpus. We will show that the $F_{\beta=1}$ value for this task is around 94% for all three approaches, which suggests that this is a fruitful first step for parsing and analysing German text.

1 Introduction

Sentence structure in German is known for its complexity. Hence, for parsing German it is useful to first assign a basic structure to the sentence and then base subsequent processing on this basic structure, following the divide and conquer strategy.

A well defined basic structure for German sentences is the topological field structure. Topological fields describe sections in the German sentence with regard to the distributional properties of the verb. Topological fields give a handle to break up sentences into smaller, easier to process parts. In this paper we describe three different approaches to the detection of the verbal parts of the topological fields (TopFChunking): i) hand-written rules implemented in Finite State Automata (FSA); ii) Probabilistic Context-Free Grammar rules derived from a corpus (PCFG); and iii) a machine learning approach, Memory-Based Learning (MBL), in which TopFChunking is approached as a tagging task.

This paper is structured as follows. In Section 2 we introduce the concept of Topological Fields. In Section 3 we discuss the corpus. In Section 4 we give a short explanation of the three methods compared in this paper. In Section 5 we see the results. Finally, in Section 6 we draw our conclusions and propose some further research.

2 The topological field model

2.1 An outline of the model

The topological field model (cf. Höhle (1986)) is a well-established descriptive model of the constituent order in German and several other Germanic languages. Topological fields describe sections in the German sentence with regard to the distributional properties of the verb (and the subordinator in subclauses). There are three different types of clauses (see also Table 1):[2] verb-last clauses (VL), verb-first clauses (V1) and verb-second clauses (V2). VL clauses comprise all introduced subclauses, V1 clauses mainly comprise imperatives and yes/no questions and V2 clauses mostly comprise affirmative clauses. The topological fields C/LK and VC constitute the sentence bracket, relative to which the other fields can be described[3]. The section preceding the left part of the sentence

[1]The research reported here was supported by the German Research Council (DFG) as part of the Sonderforschungsbereich 441 "Linguistische Datenstrukturen" (Linguistic Data Structures). The second author was also supported by a grant of the graduate school "Integriertes Linguistikstudium" funded by the DFG.

[2]Cf. Figure 1 for an illustrative example.

[3]Obligatory fields are in bold type. The fields KOORD (coordination field) and LV (Linkversetzung, topicalization) will not be discussed in this paper. The representation in Table 1 is slighty simplified.

Table 1: The topological field model

clause type	topological fields						
VL:	KOORD	LV		**C**	MF	**VC**	NF
V1:	KOORD	LV		**LK**	MF	VC	NF
V2:	KOORD	LV	**VF**	**LK**	MF	VC	NF

bracket is called the *Vorfeld* (VF; initial field; only in V2 clauses), the section included in the sentence bracket is called the *Mittelfeld* (MF; middle field) and the section following the right part of the sentence bracket is called the *Nachfeld* (NF; final field). While the ordering of other constituents is relatively free in German, the ordering of topological fields is subject to syntactic restrictions which adhere to the unvarying pattern outlined in Table 1. Figure 1 shows that the topological field model is also capable of accounting for recursive structures. Subordinate clauses may be embedded in topological fields and, with this model, it is possible to leave their attachment open until disambiguation in further annotation steps.

2.2 The linguistic perspective

The topological field model is primarily a distributional model. It does not give any account of the verb-complement structure and it does not reveal the relation of the constituents within the topological fields, either. However, as argued by Meurers (2002), topological field annotation can "significantly help in using corpora from the perspective of theoretical linguistics". This becomes clear when looking at Figure 1: The structure of topological fields along with POS tags marks the borders and the type of subclauses in a sentence. This shallow annotation thus provides the user with what one can call the skeleton of the sentence. Without the annotation of topological fields, it is by no means clear where the borders of the subclauses are and it is not clear where the potential complements of the respective verbs are. A query for a linguistic structure would therefore highly overgenerate, thus producing a high amount of false matches.

The shallow annotation as shown in Figure 1 gives the user of an annotated corpus the possibility to investigate various distributional phenomena which have to be described relative to the topological fields. The location of the embedding of subclauses is one of them, which has to be discussed in every linguistic theory and which is by no means agreed upon by linguists. Eisenberg (1999) is an adequate example: "Der Besetzung des Nachfeldes wird häufig eine kommunikativ-pragmatische Funktion zugeschrieben, [...]"(p. 391)[4]. If linguists want to examine such a hypothesis, they are in need of an annotation which provides them with topological fields. Otherwise queries to corpora yield too many false matches.

2.3 The computational perspective

Concerning automatic annotation, one of the main advantages which can be acquired through the annotation of topological fields is that they are the skeleton of the sentence and thus considerably reduce the scope of ambiguity. Without the annotation of topological fields the scope of complements of the verb is much wider, especially in complex sentences, in which, additionally, it is not clear which potential complements belong to which verb. Figure 1 shows how the annotation of topological fields reduces the scope of possible complements for the verbs by dividing the sentence into fields and subclauses. The subclauses can now even be dealt with as single units, thus using a divide and conquer strategy similar to the one outlined in Peh and Ting (1996).

By first annotating topological fields, one can use a strategy termed *containment of ambiguity* by Abney (1996). This strategy proposes to annotate higher levels first if "reliable markers" are present because this considerably limits the number of possible attachment sites. Annotating topological fields first and verb-complement structure later transfers this strategy, which was mainly used for chunking (i.e. basic phrase recognition), to the topological fields and clause level.[5] That way, it is also possible to construct a hybrid annotation system, in which every linguistic phenomenon is tackled with methods especially suited for it (cf. Hinrichs et al. (2002)).

[4] "A communicative-pragmatic function is very often attributed to the occupation of the Nachfeld, [...]" (our translation).

[5] A similar strategy has already been used to prestructure sentences for an information retrieval system (cf. Braun (1999) and Neumann et al. (2000)).

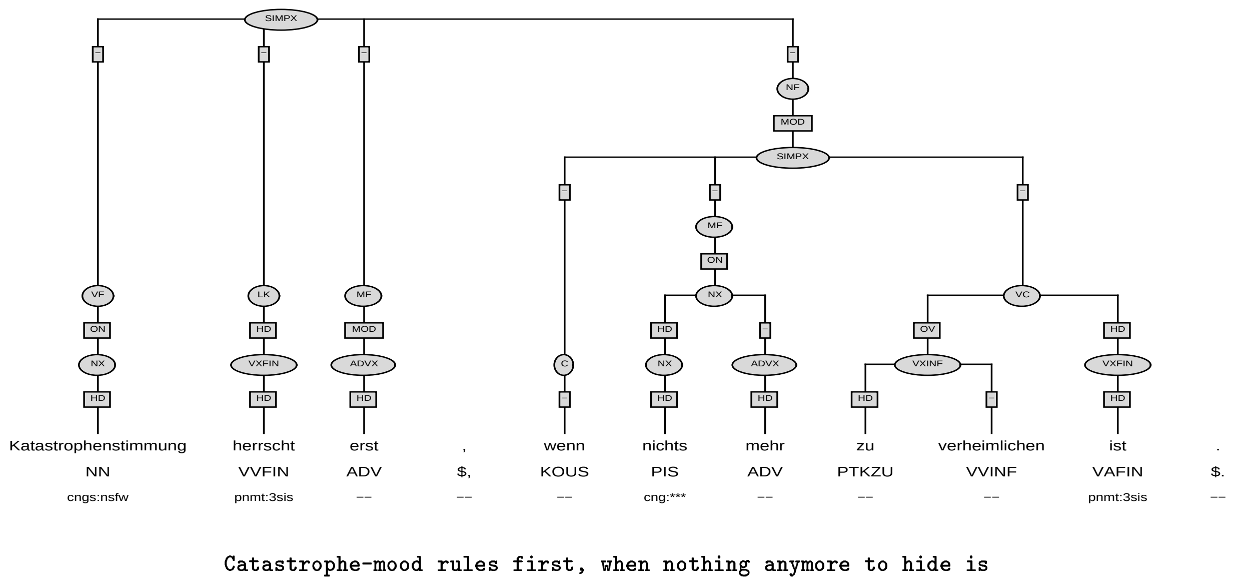


Figure 1: 'You are in a catastrophe mood only when there is nothing more to hide.'

3 Data: the TüBaD/Z Baumbank

The Tübinger Baumbank Deutsch / Zeitung (TüBaD/Z) is used as a "gold standard" for training and testing the parsers. TüBaD/Z currently consists of the last 7300 sentences from the taz newspaper corpus (articles from May 6 and 7, 1999). The sentences are annotated manually according to the annotation scheme developed for the Verbmobil spoken dialogue corpus (Stegmann et al., 2000). The scheme is slightly extended when applied to written texts. The training data is manually annotated with POS, morphology, chunks and phrases (including head information and grammatical functions), topological fields, and clause structure. TüBa-D/Z is not yet finished, so that minor additions, changes or corrections are still to be expected.

Of this complete annotation we have used the VC, C and LK chunks for our experimements. When these three fields are located, the VF, MF and NF can be easily found: the VF is the part of the clause preceding the LK; the MF is between the C or LK and the VC, the NF is following the VC. The three chunkers are trained on (PCFG and MBL) or inspired by (FSA) 4523 manually annotated sentences of the TüBaD/Z corpus. We have kept 1613 sentences apart for testing[6]. These sentences have not been seen during the development of the chunkers.

The sentences are POS tagged with the TnT tagger (Brants, 2000). This tagger has an overall accuracy of 94.7%[7] on our data. For the relevant (verb) tags, however, the accuracy is much lower, around 90.6%. The distinction between finite and infinitive verbs, relevant for TopFChunking, is difficult to learn for a POS tagger that works with local context only, since this distinction often depends on long distance dependencies. For the infinitive verb POS tag (VVINF) the accuracy of the TnT tagger is 82.5%.

For all three methods (and for the baseline) we compare the TopFChunkers trained and tested on the TnT tagged material with chunkers trained and tested on the "gold standard" material as avaible in the corpus, see Table 2.

4 Methods

4.1 The FSA chunker

An FSA receives a sequence of input symbols and accepts them according to a transition function. For the FSA chunker at hand, the input symbols are the POS tags of the input text, and the transition function is defined by a set

[6]Some sentences do not contain relevant TopFs; these sentences are ignored for training and testing.

[7]This relatively poor performance can be explained by the fact that the tagger is trained on Negra material.

of manually devised rules. It acts as a transducer, providing additional structure indicating the topological fields.

It is easy to devise an FSA for the recognition of topological fields because one of the main features of topological fields is that they are an unvarying pattern in the German sentence, i.e. they obey to absolute syntactic restrictions. Because the structure of topological fields is both well-known and simple, the rules for the FSAs can be manually constructed using linguistic knowledge extracted from grammars and deduced from automatically annotating unannotated corpora and manually comparing the results. The rules were improved after evaluation of the parser against the train subset of the corpus.

The construction of the parser is such that it is a pipe of FSA transducers in which the output of one level is the input to the next level. That way, recursive structures, which do occur in topological fields, can be covered. Ambiguities are resolved by a longest match strategy. We use the TTT suite of tools (Grover et al., 2000) available from the LTG Edinburgh (`http://www.ltg.ed.ac.uk/`) for our purposes. The annotation of topological fields is in fact the first step in annotation. One reason for this is that topological fields show the outline of the sentence. The sentence bracket annotation evaluated here is only one part of the linguistic information annotated by the parser.

4.2 The MBL chunker

Memory-Based Learning (MBL) is a machine learning algorithm in which the learning stage is memory-based: train examples are stored in a memory base without abstraction. The classification stage is similarity-based: classification is based on the most similar instances in the memory base. We have used the MBL implementation Timbl (Daelemans et al., 2001), which is available from `http://ilk.kub.nl`.

For TopFChunking we window over the text. To each word we assign one of three tags to determine whether this word is inside a chunk (I), outside a chunk (O), or between two chunks of the same type (B), yielding a total of 9 IOB tags, 3 for each chunk type (LK; VC; C), cf. Ramshaw and Marcus (1995).

To find the optimal parameter setting and feature selection for this TopFChunking task we took 10% of the train set apart and used this as a validation set. The other 90% of the train set was used as a validation train set. It turned out that working only with the POS tag features gives the best results. For an optimal learning algorithm we found IGTree (a decision tree variant of MBL) with IG as feature weighting method and the number of nearest neighbours equal to 1 ($k = 1$).

We have used Information Gain (IG) (Daelemans et al., 2001) to weight the feature relevance. What we can see is that the word form features have a low IG value, this shows that we suffer from sparse data for features with too many values (for word forms we have almost 20,000 different values, for the POS features just 55). This sparse data problem is due to the relatively small size of the TübaD/Z corpus: the train set contains no more than 100,000 words. We can expect that with a larger corpus the word form features will gain relevance, and with this we can expect better results for the TopFChunking task.

4.3 The PCFG chunker

A probabilistic context-free grammar (PCFG) model learns context-free rules from a training corpus, and also the probability for each non-terminal symbol to expand into a sequence of terminal and non-terminal symbols. The terminal symbols, in our case, are POS tags. Conditioning is performed only on the rules' left hand side, i.e. the PCFG is also probabilistically context-free. It does not take into account any lexical information, neither for terminals nor for conditioning.

Some sub-classes of phrases receive the same label in the TüBaD/Z annotation scheme but show complementary distribution in different fields, which is impossible to learn for a PCFG. Most prominently, the C field in VL-clauses has to contain a subordinator. While subordinating conjunctions are usually a direct child to the C field and therefore captured by CFG rules, relative and interrogative pronouns are contained in a separate noun chunk and are therefore indistinguishable from other noun chunks. As a consequence, a CFG not distinguishing these subtypes of NCs is not able to suppress a standard $[_{NX}$ PPOSAT NN $]$ noun chunk in C fields, and only to allow $[_{NX-C}$ PWS $]$, which contains an interrogative pronoun. The treebank was therefore changed to assign unique node names to all non-terminal child nodes of C fields (see Figure 2). Another peculiarity of VL-clauses is that their VC field contains a finite verb form, as opposed to the VC field for V1- and V2-clauses. Again, all VC child nodes and also the VC nodes themselves are systematically renamed as shown in Figure 2 for the VC of the sentence in Figure 3. The impact on performance is quite dramatic for the changes in the C field, improving from 82% to 92%. The improvement proved to be negligable for the VC fields, however.

```
SIMPX  →  C MF VC-F
C      →  NX-C
NX-C   →  PWS
MF     →  NX NX
NX     →  PPOSAT NN
NX     →  ART PIDAT NN
VC-F   →  VXFIN
VXFIN  →  VVFIN
```

Figure 2: Modified CFG rules for the subclause in Figure 3

The lopar (Schmid, 2000) parser (downloadable from: http//www.ims.uni-stuttgart.de/) is used to train the PCFG on the modified treebank and to parse the test set with the resulting grammar model. The task of assigning the verbal frame is performed as a subtask of parsing the whole sentence, i.e. the best parse of the PCFG is determined for a sentence, but only the annotation of the verbal frame is evaluated.

Another modification is applied to the treebank before extracting the rules, because lopar does not allow cyclic grammars, which may well occur in topological field analyses. All nodes that dominate (directly or indirectly) a node of the same category are renamed by adding the number of parent nodes of the same type to their category name, removing cycles in the resulting grammar. The nodes that do not contain any nodes of the same category are not changed, so that the unchanged node categories refer to chunk nodes. In order to focus on the TopFChunking task, all non-chunk nodes are removed whenever they do not dominate a field node, simplifying the structure and reducing the number of CFG rules.

Supervised training on the treebank data yields the results given in Table 2. Due to limited memory, lopar did not parse two sentences, and these two sentences containing 20 TopFs were removed from the test set for the PCFG chunker. Lopar also supports an unsupervised training mode. However, unsupervised training did not increase the performance of the PCFG model. $F_{\beta=1}$ dropped to 94.0% when training on 5000 additional POS-annotated sentences, and to 93.5% and 88.9% when training on 34269 and 90000 sentences, respectively[8].

4.4 Baseline

To give an idea of the complexity of the task we have computed a baseline for both the gold standard and the TnT POS tagged data. Per POS tag we have taken the most frequent TopF class, based on the frequency in the train set. In the test set we have assigned these most frequent TopF classes to each word. The baseline is computed over this assignment.

5 Results

In Table 2 we give an overview of the results of the three approaches and the baseline results. The "ALL" column gives a weighted average between precision and recall for all three tasks ($F_{\beta=1}$). The other columns give the $F_{\beta=1}$ result for each task separately. The *gold* rows give the results with the gold standard POSs from the corpus. This gives an idea of the influence of POS errors. The gold POS results are trained on gold POS data, the TnT results on TnT tagged data.

6 Conclusion and Future Research

6.1 Conclusion

In this paper we have seen that all three approaches show a $F_{\beta=1}$ score on the TopFChunking task around 94%. Such a high performance gives a solid basis for parsing and other text analysis tasks, such as information extraction and grammatical function assignment.

We have compared an FSA chunker, an MBL chunker and a PCFG chunker, which all have well-known advantages. An FSA chunker is easy to implement and is very efficient in terms

[8]The first 34269 sentences of the additional training set have 15 to 25 words, and the first 5000 sentences have 24 to 25 words. The 90000 sentence set contains sentences from 1 to 50 words.

Table 2: Results ($F_{\beta=1}$ in %) on the Topological Field Chunking task

	ALL	LK	VC	C
FSA TnT	94.1	96.2	92.0	93.8
FSA gold	98.4	98.8	98.3	97.5
PCFG TnT	94.4	97.0	92.2	92.3
PCFG gold	98.1	98.9	98.1	96.1
MBL TnT	93.3	96.0	90.0	91.6
MBL gold	97.2	98.0	96.7	96.6
baseline TnT	75.5	75.2	72.3	83.2
baseline gold	77.9	75.0	79.1	86.2

of time and space. On the other hand, rules derived from annotated examples are sometimes preferred to manually edited rules. The MBL chunker is also efficient in terms of speed, and it is a learning method that does not stipulate a certain kind of grammatical structure for the predicted events. However, its results are lowest, while still very close to the others. The PCFG chunker is most costly in terms of space and execution time, but the grammar model seems to be well suited for the task at hand.

We can see that the PCFG and the FSA chunker show a better performance than the MBL chunker. This is due to the relatively small size of the corpus for which MBL is more sensitive, and to the fact that in its present form the MBL chunker uses a windowing approach with a relatively small window, whereas the other two chunkers can take long distance depencies into account.

Now that we have found the LK, C, and VC in the sentence it is straightforward to detect the basic structure of the sentence. In Figure 3 we give another example of a complex sentence with a shallow annotation. We can see that this sentence is rather difficult to analyse without knowledge of the topological fields. Once we have chunked the LK, C and VC, the sentence is much easier to parse: Since we know that "*macht*" has to be the LK of the second clause (because it is the last LK in the sentence) we can assign MF to "*sich entweder lächerlich oder legendär*" and we know where the first clause ends. In the first clause, we know that the MF is to be found between the C field "*Wer*" and the VC field "*prophezeit*". What is more, once we know what sentence type we are dealing with (V1, V2, or VL), and once we have information about morphology and the valency of the verb, we can expect certain constituents in the MF, and their grammatical function. In our example, the MF contains the Direct Object for "*seinem Publikum*" and the Accusative Object for "*eine solche Reaktion*", and the C field "*Wer*" contains the Nominative Object.

6.2 Future Research

We see two major areas for future research: a combined classifier and a qualitative error analysis. A combination of the different TopFChunkers seems promising, because they seem to have different strengths judged by the given results that differ for each field type. The three chunkers may provide the input for a stacked classifier which may also take other features into account.

A combined classifier will work best if the three chunkers make different kinds of errors. Therefore, a qualitative analysis of the chunking errors will be useful, which will also help to improve the individual chunkers.

Other areas of future research include further analysis along the lines of TopFs, i.e. assigning recursive sentence structure based on the C/LK/VC fields; classification of clauses (V1, V2 or VL) and further syntactic annotation within fields.

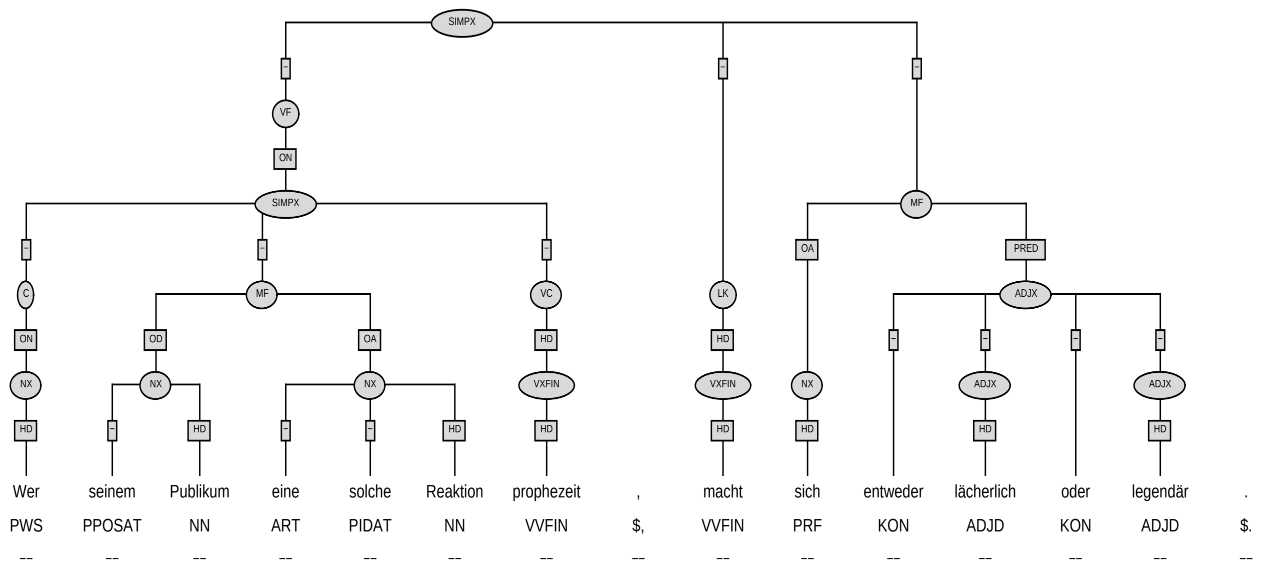
Subcategorisation information is readily available from the topological field structure and the constituents in them. This is an important first step to finding linguistically interesting phenomona; and also to information processing, such as question-answering systems.

References

Steven Abney. 1996. Partial Parsing via Finite-State Cascades. In *Proceedings of the ESSLLI-96 Workshop on "Robust Parsing"*, Prague.

Thorsten Brants. 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, Seattle, April 29 – May 3.

Christian Braun. 1999. Flaches und robustes Parsen deutscher Satzgefüge. Diplomarbeit, Universität des Saarlandes, Saarbrücken.



```
Who his audience a such reaction foretells, makes himself either ridiculous or legendary.
```

Figure 3: 'He who foretells his audience such a reaction, makes himself either ridiculous or legendary'

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2001. Tilburg Memory Based Learner, version 4.0, Reference Guide. Technical Report 01-04, ILK, Tilburg.

Peter Eisenberg. 1999. *Grundriß der deutschen Grammatik*, volume 2: Der Satz. Metzler, Stuttgart.

C. Grover, C. Matheson, A. Mikheev, and M. Moens. 2000. LT TTT - a Flexible Tokenisation Tool. In *Proceedings of the Second Language Resources and Evaluation Conference (LREC2000)*, 31 May–2 June.

Erhard W. Hinrichs, Sandra Kübler, Frank H. Müller, and Tylman Ule. 2002. A Hybrid Archictecture for Robust Parsing of German. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, Las Palmas, Gran Canaria.

Tilman Höhle. 1986. Der Begriff 'Mittelfeld', Anmerkungen über die Theorie der topologischen Felder. In *Akten des Siebten Internationalen Germanistenkongresses*, pages 329–340, Göttingen.

Walt Detmar Meurers. 2002. On the use of electronic corpora for theoretical linguistics. Case studies from the syntax of German. *Lingua*. Forthcoming.

Günter Neumann, Christian Braun, and Jakub Piskorski. 2000. A Divide-and-Conquer Strategy for Shallow Parsing of German Free Texts. In *Proceedings of the 6th Conference on Applied Natural Language Processing (ANLP 2000)*, pages 239–246, Seattle, Washington.

Li-Shiuan Peh and Christopher H. Ting. 1996. A Divide-and-Conquer Strategy for Parsing. In *Proceedings of the ACL/SIGPARSE Fifth International Workshop on Parsing Technologies*, pages 57–66.

L.A. Ramshaw and M.P. Marcus. 1995. Text Chunking using Transformation-Based Learning. In *Proc. of third workshop on very large corpora*, pages 82–94, June.

Helmut Schmid. 2000. Lopar: Design and Implementation. Arbeitspapiere des Sonderforschungsbereichs 340, Linguistic Theory and the Foundations of Computational Linguistics, 149, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.

Rosmary Stegmann, Heike Telljohann, and Erhard W. Hinrichs. 2000. Stylebook for the German Treebank in VERBMOBIL. Technical Report Verbmobil-Report 239, Universität Tübingen.