Exploring HPSG-based Treebanks for Probabilistic Parsing

Günter Neumann and Berthold Crysmann

DFKI GmbH & Saarland University
Stuhlsatzenhausweg 3
D–66123 Saarbrücken
{neumann, crysmann}@dfki.de

Abstract

We describe a method for the automatic extraction of a Stochastic Lexicalized Tree Insertion Grammar from a linguistically rich HPSG Treebank. The extraction method is strongly guided by HPSG–based head and argument decomposition rules. The tree anchors correspond to lexical labels encoding fine–grained information. The approach has been tested with a German corpus achieving a labeled recall of 77.33% and labeled precision of 78.27%, which is competitive to recent results reported for German parsing using the Negra Treebank.

1. Introduction

In (Neumann, 2003) we applied the idea of data–oriented parsing (DOP) for achieving domain-adaptation to HPSG. The basic idea of HPSG–DOP is to parse all sentences of a representative training corpus using an HPSG grammar and parser in order to automatically acquire from the parsing results a *stochastic lexicalized tree grammar*. The decomposition operation is guided by the *head feature principle* of HPSG. A major drawback of this approach was that non–headed constructions were not factored out consequently due to the lack of structural refinements.

However in (Chiang, 2000) (and others) a number of approaches for the automatic extraction of Tree Adjoining Grammars (TAGs) from treebanks are presented, which treat the factorization of modifier constructions more systematically. In this paper, we extend HPSG–DOP by combining it with Chiang's method and apply it on a linguistically rich HPSG treebank for German which is based on the recently developed Redwoods Treebank (cf. (Oepen et al., 2002) and sec. 3.). To our knowledge, our approach is the first time that a rich linguistic theory together with a stochastic TAG is applied to the German language. This is not a trivial task, as recently (Dubey and Keller, 2003) and (Levy and Manning, 2004) have shown that treebank parsing for German yields substantial lower performance compared to English Penn treebank parsing, probably due to the fact that differences in both languages and treebank annotation may be involved.

2. Stochastic Lexicalized Tree Grammars

The set of lexically anchored trees extracted via the original HPSG–DOP method already characterizes a lexical tree–substitution grammar, i.e., a tree–adjoining grammar with no auxiliary trees, cf. (Schabes, 1990). In (Neumann, 1998), and subsequently in (Xia, 1999), (Chen and Vijay-Shanker, 2000), and (Chiang, 2000) it is shown how tree adjoining grammars can be extracted from the Penn Treebank by performing a re–construction of the derivations using head–percolation rules. Here, we follow the approach developed in (Chiang, 2000), because his approach only requires a minimal amount of treebank preprocessing, which makes it easier to adapt it to other kind of treebanks.[1]

For efficiency reasons, a restricted form of lexicalized tree adjoining grammars is considered viz. lexicalized tree insertion grammars (LTIGs). LTIG has been introduced in (Schabes and Waters, 1995) as a TAG–formalism in which all auxiliary trees are either left or right auxiliary trees. No elementary wrapping auxiliary trees or elementary empty auxiliary trees are allowed. Furthermore, left (right) auxiliary trees cannot be adjoined to a node that is on the spine of an elementary right (left) auxiliary tree; and there is no adjunction allowed to the right (left) of the spine of an elementary left (right) auxiliary tree (cf. figure 1).

The parameters of a probabilistic TAG which control the combination of trees by the substitution and adjunction are:

$$\sum_{\alpha} P_i(\alpha) = 1$$

$$\sum_{\alpha} P_s(\alpha \mid \eta) = 1$$

$$\sum_{\beta} P_a(\beta \mid \eta) + P_a(\text{NONE} \mid \eta) = 1$$

$$\sum_{\alpha} P_{sa}(\beta \mid \eta, i, X) + P_{sa}(\text{STOP} \mid \eta, i, X) = 1$$

where α ranges over initial trees, and β over auxiliary trees, and η over nodes. $P_i(\alpha)$ is the probability of beginning a derivation with α; $P_s(\alpha \mid \eta)$ is the probability of substituting α at η; $P_a(\beta \mid \eta)$ is the probability of adjoining β at η; $P_a(\text{NONE} \mid \eta)$ is the probability of nothing adjoining at η; $P_{sa}(\beta \mid \eta, i, X)$ is the probability of sister–adjoining, and $P_{sa}(\text{STOP} \mid \eta, i, X)$ is the probability of no further sister–adjunction. X is the root label of the previous tree to sister–adjoin at the site (η, i), or START if none. The probability of a derivation can then be expressed as the product of the probabilities of the individual operations of the derivation, cf. (Chiang, 2004) for more details.

LTIGs have context–free power and can be parsed in O(n^3). Two parseres are available to us: a two–phase Early–style

[1] And because his approach can be seen as a substantial improvement of the initial work we have layed out and described in (Neumann, 1998).

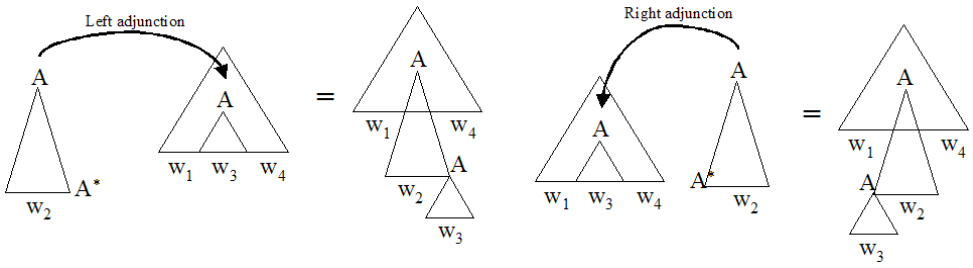

Figure 1: Left and right adjunction.

LTIG parser based on (Schabes and Waters, 1995) written in Lisp at our Lab, and a CKY–style bottom–up parser based on (Schabes and Waters, 1993) written in C by David Chiang. For the experiments reported in this paper in sec. 5., we are using David's parser, because currently, it is much faster than the Early–based Lisp parser, and can be handled much more flexible. The CKY–parser implements sister–adjunction, and uses a beam search, computing the score of an item $[\eta, i, j]$ by multiplying it by the prior probability $P(\eta)$. All items with score less than a given threshold compared to the best item in a cell are pruned.

3. HPSG TreeBank

The HPSG treebank (codename *Eiche*) we use in our study is based on a subset of the Verbmobil corpus which has been automatically annotated with a German HPSG grammar. The analyses provided by the grammar have then been manually disambiguated using the Redwoods treebanking technology, cf. (Oepen et al., 2002).

The underlying HPSG grammar itself has originally been developed as a large-scale competence grammar of German by Stefan Müller and Walter Kasper in the context of the Speech-to-Speech machine translation project Verbmobil (see (Müller and Kasper, 2000)), and has subsequently been ported to the LKB (Copestake, 2001) and PET (Callmeier, 2000) processing platforms. In 2002, grammar development has been taken over by Berthold Crysmann. Since then, the grammar has undergone several major changes, most importantly the treatment of verb placement in clausal syntax (Crysmann, 2003).

3.1. Some basic properties of German syntax

The syntax of German features a variety of phenomena that makes syntactic analysis much harder than that of more configurational languages. Chief among these is the relative free word order in which syntactic arguments of a verb can appear within the clausal domain.

(1) a. weil der Lehrer dem Schüler das Buch schenkte
because the teacher.NOM the pupil.DAT the book.ACC donated
'because the teacher gave the book to the pupil as a present'

b. weil der Lehrer das Buch dem Schüler schenkte

c. weil dem Schüler der Lehrer das Buch schenkte

d. weil dem Schüler das Buch der Lehrer schenkte

e. weil das Buch der Lehrer dem Schüler schenkte

f. weil das Buch dem Schüler der Lehrer schenkte

Almost anywhere between the arguments modifiers can be interspersed quite freely.

(2) weil (gestern) der Lehrer (gestern) dem Schüler (gestern) das Buch (gestern) schenkte
because (yesterday) the teacher.NOM (yesterday) the pupil.DAT (yesterday) the book.ACC (yesterday) donated
'because yesterday the teacher gave the book to the pupil as a present'

This situation is further complicated by the combined effects of verb cluster formation and argument composition, which permit permutation even amongst the arguments of different verbs within the cluster.

(3) a. weil der Lehrer das Buch zu kaufen versprach
because the teacher.NOM the book.ACC to buy promised
'because the teacher promised him to buy the book.'

b. weil das Buch der Lehrer zu kaufen versprach
because the book.ACC the teacher.NOM to buy promised
'because the teacher promised him to buy the book.'

Furthermore, realisation of the verb cluster is often discontinuous, typically in matrix clauses.

(4) a. da versprach der Lehrer.NOM das Buch zu kaufen
there promised the teacher.NOM the book.ACC to buy
'There, the teacher promised him to buy the book.'

b. da versprach das Buch der Lehrer zu kaufen
there promised the book.ACC the teacher.NOM to buy
'There, the teacher promised him to buy the book.'

Assuming continuous constituents only, the argument structure is therefore only partially known in bottom-up parsing, until the other member of the discontinuous verb cluster is found.

In German matrix clauses, the finite verb typically surfaces in second position, the first position being occupied by some fronted, i.e. extracted, constituent. Thus, in contrast to English, presence of non-local dependencies is the norm, rather than the exception.

Taken together, permutation of arguments, modifier interspersal, discontinuous complex predicates and the almost categorial presence of non-local dependencies give rise to a considerable degree of variation in tree structure. As a consequence, we expect data-driven approaches to parsing to be more prone to the problem of data-sparseness. In the context of grammar induction from treebanks, it has already been observed, e.g., by (Dubey and Keller, 2003) that methods which are highly successful in a more configurational language, such as Collins PCFG parser for English, cf. (Collins, 1997), give less optimal results when applied to German.

This problem is further enhanced by the fact that German is a highly inflectional language, with 4 distinct cases, 3 gender and 2 number distinctions, all of which enter into agreement relations. The same holds for the verbal domain, where up to 5 person/number combinations are clearly distinguished.

3.2. The grammar

In the spirit of HPSG as a highly lexicalised grammatical theory, most of the information about an items combinatorial potential is encoded in the lexical entries itself, in terms of typed feature structures. Syntactic composition is then performed by means of highly general rule schemata, again, implemented as typed feature structures, which specify the flow of information within syntactic structure. As a result, the DFKI German HPSG specifies only 87 phrase structure schemata, as compared to some 280+ leaf types for the definition of parameterised[2] lexical entries, augmented by 56 lexical rules and 286 inflectional rules.

The rule schemata, which make up the phrase structure backbone of the HPSG grammar, correspond quite closely to principles of syntactic composition: by themselves they encode basic functional relations between daughter constituents, such as head-subject, head-complement, or head-adjunct, rather than intrinsic properties of the node itself. Thus, a rule like `h-comp` can be used to saturate a subcategorised complement of a preposition, a verb, or, a noun. Similarly, which constituents can function as the complement daughter of the `h-comp` rule is mainly determined by the information represented on the SUBCAT list of the lexical head. The rule schemata merely ensure that the subcategorisation constraints formulated by the head will actually be imposed on the complement daughter, and that the saturated valence requirement will be canceled off.

[2]Lexical entries may get further specialised beyond the information encoded in the lexical leaf type: typically, this includes subcategorisation for lexical case, selection of prepositional complements and verb particles, specification of auxiliary type (*have* vs. *be*), as well as sortal restrictions on complements.

Since the underlying processing platforms (LKB/PET) do not currently support the segregation of immediate dominance and linear precedence, some rule schemata are further specialised according to the position of the head: alongside `h-adjunct`, `h-subj` and `h-comp` rules for verb-initial clauses and *pre*positional phrases, the grammar also defines their head-final counterparts (`adjunct-h`, `subj-h`, `comp-h`), required for verb-final clauses, adjectival phrases and *post*positional phrases. Within NPs some modifiers, e.g. adjectives are licensed by adjunct-h structures, whereas PPs are licensed in post-head position only. To summarise, the rules of the CF backbone provide crucial information about the position of the syntactic head, as well as the functional status of the non-head daughter.

Scrambling of complements is licensed in the German grammar by special lexical rules that permute the elements on a head's SUBCAT list. Modifier interspersal and scrambling across the subject are accounted for by permitting the application of h-subj, h-comp, and h-adjunct rules in any order.

Argument composition and scrambling of arguments from different verbs is captured by shuffling the SUBCAT lists of the upstairs and downstairs verb (e.g., `vcomp-h-0` ... `vcomp-h-4`). Discontinuous verb clusters are modelled by means of simulated verb movement ((Müller and Kasper, 2000) expanding an earlier idea proposed by (Kiss and Wesche, 1991)). Essentially, the subcategorisation requirements of the initial verb are percolated down the tree to be shuffled with those of the final verb.

Finally, extraction is implemented in a fairly standard way using slash feature percolation. Slash introduction is performed, at the gap site, by a unary rule. For subjects and complements, slash introduction saturates an argument requirement of the head by inserting its LOCAL value into the SLASH list. For adjuncts, the slash introduction also inserts a *local* object into SLASH, but since there is no valency to be saturated, it only semantically attaches the extracted modifier to the head. At the filler-site, SLASH specifications are retrieved, under unification: for semantic reasons, the grammar crucially distinguishes here between wh-fillers (`wh-h` rule) and non-wh-fillers (`filler-h` rule).

Besides these more basic constructions, the grammar also provides rule schemata for different types of coordinate structures, extraposition phenomena (Crysmann, in press), dislocation, as well as some constructions more specific to German, such as auxiliary flip and partial VP fronting.

3.3. The treebank

The version of the HPSG formalism underlying the LKB and PET processing systems assumes continuous constituents only. Thus, the derivation tree of a sentence analysed by the grammar corresponds to a context free phrase structure tree. Given a grammar, the full HPSG analysis of a sentence can therefore always be reconstructed deterministically, once the derivation tree is stored together with the unique identifiers of the lexical entries on the terminal nodes. This fact is actually exploited by the Redwoods treebanking infrastructure to provide a compact representation format. From the fully reconstructed feature structure representation of a parse, it is possible to extract

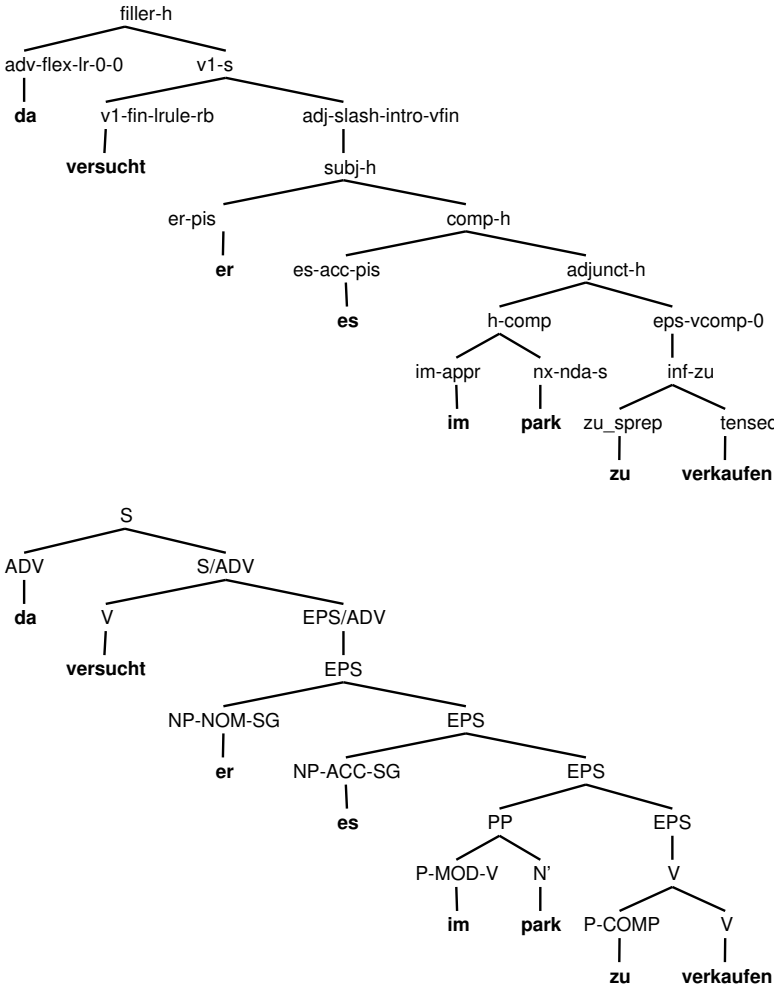

Figure 2: Examples of a derivation tree and its corresponding phrase tree representation. See text below for an explanation of the different symbols.

additional derived structures: one such auxiliary structure that deserves particular mentioning is an isomorphic constituent tree decorated with more conventional node labels, such as S, NP, VP, PP, etc. These labels are obtained by testing the unifiability of a feature structure description against the AVM associated with the node, and assigning the label of the first matching description. Since these derived trees are isomorphic to the derivation history, the "functional" decorations provided by the rule backbone can be enriched straightforwardly with "categorial" information, providing for a very rich annotation.

As already mentioned before, the primary data used for the construction of the *Eiche* treebank are taken from the Verbmobil test corpora. To give the reader an idea about the complexity of the disambiguation task, the grammar assigns on average around 16 distinct analyses to each sentence. In order to minimise duplication of annotation effort, only unique sentence strings have been incorporated into the treebank. Thus, redundancy in the data is limited to partial structures.

4. HPSG–Supertag Extraction

The main purpose of the grammar extraction process is twofold: 1) extract automatically all possible supertags, i.e., an LTIG, and 2) to obtain a maximum–likelihood estimation of the parameters of the extracted LTIG. The grammar extraction process actually re–constructs TAG derivations underlying the parse trees and is quite similar to the head–driven decomposition operation used in HPSG–DOP, but now adapted for the case of LTIG extraction.

4.1. The extraction method

Similar to (Magerman, 1995) and (Chiang, 2000), we use head–percolation and argument rules that classify for each node η exactly one child of η as the head and the others as either argument or modifier. However, as we will discuss below, our rules are based on HPSG and as such, are much more smaller in number and less heuristic in nature as those defined in (Chiang, 2000). Using these rules, the derivations are re–constructed using the method described in (Chiang, 2000), and summarized here for your convenience:

- If η is an adjunct, excise the subtree rooted at η to form a modifier tree.
- If η is an argument, excise the subtree rooted at η to form an initial tree, leaving behind a substitution node.

- If η has a right corner θ which is an argument with the same label as η (and all intervening nodes are heads), excise the segment from η down to θ to form an auxiliary tree.

From the determined structures, supertags are generated in two steps: first the tree template (i.e., the elementary tree minus its anchor), then the anchor. From there, the probabilities are decomposed accordingly and three back-off levels are computed, as described in (Chiang, 2000). Furthermore, all words seen n or fewer times in training are treated as a single symbol UNKNOWN, in order to handle unknown words.

4.2. The rule definition

The following two tables contain the HPSG–based head and argument rules currently in use:

Parent:	Child:
SUBJ-H	last *
ADJUNCT-H	last *
COMP-H	last *
FILLER-H	last *
WH-H	last *
POS-ES	last *
DET-NBAR	last *
NP-NBAR	last *
VCOMP-H-0	last *
VCOMP-H-1	last *
VCOMP-H-2	last *
VCOMP-H-3	last *
VCOMP-H-4	last *
BINARY-COORD	last *
RECURSIVE-EV-COORD	last *
RECURSIVE-NOM-COORD	last *
*	first *

Table 1: Head rules for the HPSG Treebank. The symbol * stands for any label.

Parent:	Child:
SUBJ-H	first *
H-SUBJ	last *
COMP-H	first *
H-COMP	last *
H-COMP-EXTRAPOSED	last *
H-SUBJ-EXTRAPOSED	last *

Table 2: Arg rules for the HPSG Treebank. The symbol * stands for any label.

The list of rules is processed in the order specified and the first rule that fires is applied. A rule fires if the label of the current node matches with one of the parent node labels specified in the rule list. A head rule like "SUBJ-H last *" determines that the last child of a parent node with label SUBJ−H is the head, regardless of the child's label. The head rule "* first *" means that for a parent with an arbitrary node label its leftmost child is chosen as the head daughter. This rule plays the role of a default head rule. The argument rules work in the same way. For an explanation of the linguistic content of these rules, cf. sec. 3..

5. Experiments

We performed a ten–fold cross–validation over a corpus of 3528 sentences from the Verbmobil domain with an average sentence length of 7.2 words. The anchors of the extracted supertags consist of the preterminals of the derivation trees and are lexical labels (LEX). These are much more fine–grained than Penn Treebank preterminal tags, covering information about POS, morpho-syntactic, valence and other information. The UNKNOWN symbol relates to corresponding words in the training set (it maps words seen fewer than N times to this symobl), i.e., stems that only occur in the test set, but not in the training set, are not covered by the grammar. Hence, the parser will deliver no result for sentences which contain "out–of–vocabulary" stems.
We trained and tested our method on the full encoding of the symbols, which among others encode values for gender, number, person, case, tense and mood. Furthermore, the symbols also encode the valency of verbs.
It seems clear that using lexical labels as anchors will effect at least the coverage and recall. In order to test this, we also run an experiment, where we used only the Part–of–Speech (POS) of the lexical labels, which are retrieved from the yield of the corresponding phrase tree. This will lead to a much more coarse–grained classification of word forms, but probably also to a less restrictive tree selection. The table below presents our current results:

Anchor	Cov.	LR(t.)	LP(t.)	LR(c.)	LP(c.)
LEX	77.47	57.68	77.07	77.33	78.27
POS	98.12	76.42	78.36	77.92	78.44

where LR(t.)/LP(t.) – t. stands for total – is measured over all sentences, and LR(c.)/LP(.) – c. stands for coverage – over the parsed sentences, i.e., for sentences without out–of–vocabulary stems.

6. Discussion

To date, there is only little work on full probabilistic parsing of German from treebanks. The first probabilistic treebank parser for German (using the Negra Treebank) is presented in (Dubey and Keller, 2003). They obtain (for sentence length of ≤ 40): LR=71.32% and LP=70.93% (coverage = 95.9%). (Müller et al., 2003) also present a probabilistic parser for Negra. They study the consequences that the Negra implies for probabilistic parsing, and concentrate on the role of two factors (1) lexicalization and (2) grammatical functions. The results they report: LR=71.00% and LP=72.85% (coverage = 100%). Furthermore, (Levy and Manning, 2004) present experiments on probabilistic parsing using Negra concentrating on non–local dependency reconstruction. Their results also suggest that current state–of–art statistical parsing is far better on Penn Treebank than on the Negra Treebank.

7. Related Work

Current stochastic approaches for HPSG basically focus on parse tree disambiguation using the English Redwoods Treebank, cf. (Oepen et al., 2002). For example, (Toutanova et al., 2002), present a parse selection method

using conditional log-linear models built over the levels of derivation tree, phrase structure tree, and semantic dependency graph in order to analyse the effect of different information levels represented in the Redwoods Treebank. The best reported result (in terms of accuracy) is obtained for the derivation tree representation and by implementing an extended PCFG that conditions each node's expansion on several of its ancestors in the derivation tree (with a manually specified upper bound of 4 ancestors). They report an exact parse accuracy of 81.80% for such an extended PCFG, which was only slightly improved when combining it with a PCFG based on the semantic dependency graph representation (82.65%). In (Toutanova and Manning, 2002) this work is extended by the integration of automatic feature selection methods based on decision trees and ensembles of decision trees. Using this mechanism, they are able to improve the parse selection accuracy for the derivation tree based PCFG from 81.82% to 82.24%.

8. Conclusion and Future Work

We have presented an approach of extracting supertags from a HPSG–based treebank, and have evaluated the performance of the grammar using a stochastic LTIG parser. In future work, we will consider the following aspects. First, we will explore how the current results can be improved by either adding more information to the tree labels or by generalizing those tree labels which are currently too specific. Second, we will investigate how this technology can be used to provide the N–best derivation trees and to use them as input for the deterministic feature structure expansion step using the HPSG–source grammar. In this way, a preference–based parsing schema for HPSG using a treebank model will function as a filter.

Acknowledgements

The work presented here was partially supported by a research grant from the German Federal Ministry of Education, Science, Research and Technology (BMBF) to the DFKI project `Quetal` (FKZ: 01 IW C02) and the EC–funded project DeepThought. We thank very much David Chiang for making available to us his TIG training and parsing system. We also thank the Redwoods Treebank team for making their tools open–source, and especially Stephan Oepen for his kind support.

9. References

U. Callmeier. 2000. PET — a platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering*, 6(1):99–108.

J. Chen and K. Vijay-Shanker. 2000. Automated extraction of tags from the penn treebank. In *IWPT'2000*, Italy.

D. Chiang. 2000. Statistical parsing with an automatically–extracted tree adjoining grammar. In *ACL*.

D. Chiang. 2004. *Evaluating Grammar Formalisms for Applications to Natural Language Processing and Biological Sequence Analysis*. Ph.D. thesis, University of Pennsylvania.

M. Collins. 1997. Three generative, lexicalised models for statistical parsing. In *ACL*.

A. Copestake. 2001. *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford.

B. Crysmann. 2003. On the efficient implementation of German verb placement in HPSG. In *RANLP 2003*.

B. Crysmann. in press. Relative clause extraposition in German: An efficient and portable implementation. *Research on Language and Computation*.

A. Dubey and F. Keller. 2003. Probabilistic parsing for german using sister-head dependencies. In *ACL*.

T. Kiss and B. Wesche. 1991. Verb order and head movement. In O. Herzog and C. Rollinger, editors, *Text Understanding in LILOG*, number 546 in LNAI. Springer, Berlin.

R. Levy and C. D. Manning. 2004. Deep dependencies from context-free statistical parsers: Correcting the surface dependency approximation. In *ACL*.

D. Magerman. 1995. Statistical decisiontree models for parsing. In *ACL*, pages 276–283.

S. Müller and W. Kasper. 2000. Hpsg analysis of German. In W. Wahlster, editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, Artificial Intelligence, pages 238–253. Springer-Verlag, Berlin Heidelberg New York.

K. Müller, D. Prescher, and K. Sima'an. 2003. Grammatical functions and parsing the german negra treebank. In *Slides from CLIN 2003 presentation, http://staff.science.uva.nl/ kmueller/Onlinepapers/ CLIN03_slides.pdf*.

G. Neumann. 1998. Automatic extraction of stochastic lexicalized tree grammars from treebanks. In *TAG+ workshop*, Philadelphia, PA, USA, August.

G. Neumann. 2003. Data-driven approaches to head-driven phrase structure grammar. In Rens Bod, Remko Scha, and Khalil Sima'an, editors, *Data Oriented Parsing*. CSLI Publications, University of Chicago Press, Stanford:CA, USA.

S. Oepen, K. Toutanova, S. Shieber, C. Manning, D. Flickinger, and T. Brants. 2002. The lingo redwoods treebank: Motivation and preliminary applications. In *COLING*.

Y. Schabes and R. Waters. 1993. Stochastic lexicalized context-free grammar. In *IWPT'93*.

Y. Schabes and R. Waters. 1995. Tree insertion grammar: A cubic-time parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics*, 21:479–513.

Y. Schabes. 1990. *Mathematical and Computational Aspects of Lexicalized Grammars*. Ph.D. thesis, University of Pennsylvania, Philadelphia, USA.

K. Toutanova and C. Manning. 2002. Feature selection for a rich hpsg grammar using decision trees. In *Proceedings of the Sixth Conference on Natural Language Learning (CoNLL)*.

K. Toutanova, C. Manning, S. Shieber, D. Flickinger, and S. Oepen. 2002. Parse disambiguation for a rich hpsg grammar. In *In First Workshop on Treebanks and Linguistic Theories (TLT2002)*, pages 253–263.

F. Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *NLPRS-99*, Beijing, China.