Creating a CCGbank and a wide-coverage CCG lexicon for German

Julia Hockenmaier
Institute for Research in Cognitive Science
University of Pennsylvania
Philadelphia, PA 19104, USA
juliahr@cis.upenn.edu

Abstract

We present an algorithm which creates a German CCGbank by translating the syntax graphs in the German Tiger corpus into CCG derivation trees. The resulting corpus contains 46,628 derivations, covering 95% of all complete sentences in Tiger. Lexicons extracted from this corpus contain correct lexical entries for 94% of all known tokens in unseen text.

1 Introduction

A number of wide-coverage TAG, CCG, LFG and HPSG grammars (Xia, 1999; Chen et al., 2005; Hockenmaier and Steedman, 2002a; O'Donovan et al., 2005; Miyao et al., 2004) have been extracted from the Penn Treebank (Marcus et al., 1993), and have enabled the creation of wide-coverage parsers for English which recover local and non-local dependencies that approximate the underlying predicate-argument structure (Hockenmaier and Steedman, 2002b; Clark and Curran, 2004; Miyao and Tsujii, 2005; Shen and Joshi, 2005). However, many corpora (Böhomvá et al., 2003; Skut et al., 1997; Brants et al., 2002) use dependency graphs or other representations, and the extraction algorithms that have been developed for Penn Treebank style corpora may not be immediately applicable to this representation. As a consequence, research on statistical parsing with "deep" grammars has largely been confined to English. Free-word order languages typically pose greater challenges for syntactic theories (Rambow, 1994), and the richer inflectional morphology of these languages creates additional problems both for the coverage of lexicalized formalisms such as CCG or TAG, and for the usefulness of dependency counts extracted from the training data. On the other hand, formalisms such as CCG and TAG are particularly suited to capture the crossing dependencies that arise in languages such as Dutch or German, and by choosing an appropriate linguistic representation, some of these problems may be mitigated.

Here, we present an algorithm which translates the German Tiger corpus (Brants et al., 2002) into CCG derivations. Similar algorithms have been developed by Hockenmaier and Steedman (2002a) to create CCGbank, a corpus of CCG derivations (Hockenmaier and Steedman, 2005) from the Penn Treebank, by Çakıcı (2005) to extract a CCG lexicon from a Turkish dependency corpus, and by Moortgat and Moot (2002) to induce a type-logical grammar for Dutch.

The annotation scheme used in Tiger is an extension of that used in the earlier, and smaller, German Negra corpus (Skut et al., 1997). Tiger is better suited for the extraction of subcategorization information (and thus the translation into "deep" grammars of any kind), since it distinguishes between PP complements and modifiers, and includes "secondary" edges to indicate shared arguments in coordinate constructions. Tiger also includes morphology and lemma information.

Negra is also provided with a "Penn Treebank"-style representation, which uses flat phrase structure trees instead of the crossing dependency structures in the original corpus. This version has been used by Cahill et al. (2005) to extract a German LFG. However, Dubey and Keller (2003) have demonstrated that lexicalization does not help a Collins-style parser that is trained on this corpus, and Levy and Manning (2004) have shown that its context-free representation is a poor approximation to the underlying dependency structure. The resource presented here will enable future research to address the question whether "deep" grammars such as CCG, which capture the underlying dependencies directly, are better suited to parsing German than linguistically inadequate context-free approximations.

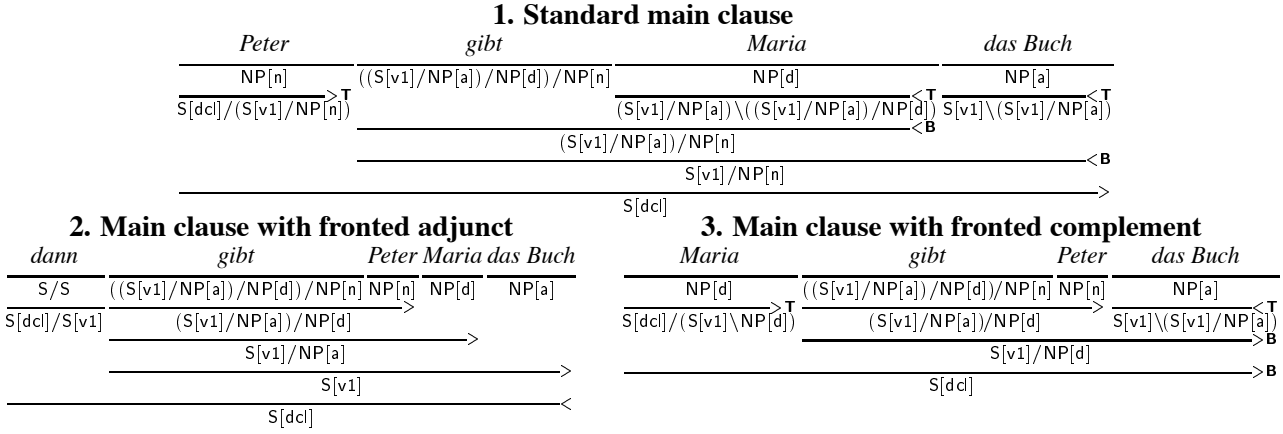


Figure 1: CCG uses topicalization (1.), a type-changing rule (2.), and type-raising (3.) to capture the different variants of German main clause order with the same lexical category for the verb.

2 German syntax and morphology

Morphology German verbs are inflected for person, number, tense and mood. German nouns and adjectives are inflected for number, case and gender, and noun compounding is very productive.

Word order German has three different word orders that depend on the clause type. Main clauses (1) are verb-second. Imperatives and questions are verb-initial (2). If a modifier or one of the objects is moved to the front, the word order becomes verb-initial (2). Subordinate and relative clauses are verb-final (3):

(1) a. *Peter gibt Maria das Buch.*
Peter gives Mary the book.
b. *ein Buch gibt Peter Maria.*
c. *dann gibt Peter Maria das Buch.*

(2) a. *Gibt Peter Maria das Buch?*
b. *Gib Maria das Buch!*

(3) a. *dass Peter Maria das Buch gibt.*
b. *das Buch, das Peter Maria gibt.*

Local Scrambling In the so-called "Mittelfeld" all orders of arguments and adjuncts are potentially possible. In the following example, all 5! permutations are grammatical (Rambow, 1994):

(4) *dass [eine Firma] [meinem Onkel] [die Möbel] [vor drei Tagen] [ohne Voranmeldung] zugestellt hat.*
that [a company] [to my uncle] [the furniture] [three days ago] [without notice] delivered has.

Long-distance scrambling Objects of embedded verbs can also be extraposed unboundedly within the same sentence (Rambow, 1994):

(5) *dass [den Schrank] [niemand] [zu reparieren] versprochen hat.*
that [the wardrobe] [nobody] [to repair] promised has.

3 A CCG for German

3.1 Combinatory Categorial Grammar

CCG (Steedman (1996; 2000)) is a lexicalized grammar formalism with a completely transparent syntax-semantics interface. Since CCG is mildly context-sensitive, it can capture the crossing dependencies that arise in Dutch or German, yet is efficiently parseable.

In categorial grammar, words are associated with syntactic categories, such as $S\backslash NP$ or $(S\backslash NP)/NP$ for English intransitive and transitive verbs. Categories of the form X/Y or $X\backslash Y$ are functors, which take an argument Y to their left or right (depending on the the direction of the slash) and yield a result X. Every syntactic category is paired with a semantic interpretation (usually a λ-term).

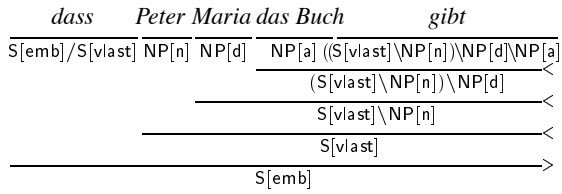
Like all variants of categorial grammar, CCG uses function application to combine constituents, but it also uses a set of combinatory rules such as composition (**B**) and type-raising (**T**). Non-order-preserving type-raising is used for topicalization:

Application:	X/Y	Y	$\Rightarrow$	X
	Y	$X\backslash Y$	$\Rightarrow$	X
Composition:	X/Y	Y/Z	$\Rightarrow_{\mathbf{B}}$	X/Z
	X/Y	$Y\backslash Z$	$\Rightarrow_{\mathbf{B}}$	$X\backslash Z$
	$Y\backslash Z$	$X\backslash Y$	$\Rightarrow_{\mathbf{B}}$	$X\backslash Z$
	Y/Z	$X\backslash Y$	$\Rightarrow_{\mathbf{B}}$	X/Z
Type-raising:	X		$\Rightarrow_{\mathbf{T}}$	$T\backslash(T/X)$
Topicalization:	X		$\Rightarrow_{\mathbf{T}}$	$T/(T\backslash X)$

Hockenmaier and Steedman (2005) advocate the use of additional "type-changing" rules to deal with complex adjunct categories (e.g. $(NP\backslash NP) \Rightarrow S[ng]\backslash NP$ for *ing*-VPs that act as noun phrase modifiers). Here, we also use a small number of such rules to deal with similar adjunct cases.

3.2 Capturing German word order

We follow Steedman (2000) in assuming that the underlying word order in main clauses is always verb-initial, and that the sententce-initial subject is in fact topicalized. This enables us to capture different word orders with the same lexical category (Figure 1). We use the features S[v1] and S[vlast] to distinguish verbs in main and subordinate clauses. Main clauses have the feature S[dcl], requiring either a sentential modifier with category S[dcl]/S[v1], a topicalized subject (S[dcl]/(S[v1]/NP[nom])), or a type-raised argument (S[dcl]/(S[v1]\X)), where X can be any argument category, such as a noun phrase, prepositional phrase, or a non-finite VP. Here is the CCG derivation for the subordinate clause (S[emb]) example:



For simplicity's sake our extraction algorithm ignores the issues that arise through local scrambling, and assumes that there are different lexical category for each permutation.[1]

Type-raising and composition are also used to deal with wh-extraction and with long-distance scrambling (Figure 2).

4 Translating Tiger graphs into CCG

4.1 The Tiger corpus

The Tiger corpus (Brants et al., 2002) is a publicly available[2] corpus of ca. 50,000 sentences (almost 900,000 tokens) taken from the *Frankfurter Rundschau* newspaper. The annotation is based on a hybrid framework which contains features of phrase-structure and dependency grammar. Each sentence is represented as a graph whose nodes are labeled with syntactic categories (NP, VP, S, PP, etc.) and POS tags. Edges are directed and labeled with syntactic functions (e.g. head, subject, accusative object, conjunct, appositive). The edge labels are similar to the Penn Treebank function tags, but provide richer and more explicit information. Only 72.5% of the graphs have no crossing edges; the remaining 27.5% are marked as discontinuous. 7.3% of the sentences have one or more "secondary" edges, which are used to indicate double dependencies that arise in coordinated structures which are difficult to bracket, such as right node raising, argument cluster coordination or gapping. There are no traces or null elements to indicate non-local dependencies or wh-movement.

Figure 2 shows the Tiger graph for a PP whose NP argument is modified by a relative clause. There is no NP level inside PPs (and no noun level inside NPs). Punctuation marks are often attached at the so-called "virtual" root (VROOT) of the entire graph. The relative pronoun is a dative object (edge label DA) of the embedded infinitive, and is therefore attached at the VP level. The relative clause itself has the category S; the incoming edge is labeled RC (relative clause).

4.2 The translation algorithm

Our translation algorithm has the following steps:

```
translate(TigerGraph g):
    TigerTree t = createTree(g);
    preprocess(t);
    if (t ≠ null)
        CCGderiv d = translateToCCG(t);
        if (d ≠ null);
            if (isCCGderivation(d))
                return d;
            else fail;
        else fail;
    else fail;
```

1. Creating a planar tree: After an initial preprocessing step which inserts punctuation that is attached to the "virtual" root (VROOT) of the graph in the appropriate locations, discontinuous graphs are transformed into planar trees. Starting at the lowest nonterminal nodes, this step turns the Tiger graph into a planar tree without crossing edges, where every node spans a contiguous substring. This is required as input to the actual translation step, since CCG derivations are planar binary trees. If the first to the ith child of a node X span a contiguous substring that ends in the jth word, and the $(i+1)$th child spans a substring starting at $k > j+1$, we attempt to move the first i children of X to its parent P (if the head position of P is greater than i). Punctuation marks and adjuncts are simply moved up the tree and treated as if they were originally attached to P. This changes the syntactic scope of adjuncts, but typically only VP modifiers are affected which could also be attached at a higher VP or S node without a change in meaning. The main exception

[1] Variants of CCG, such as Set-CCG (Hoffman, 1995) and Multimodal-CCG (Baldridge, 2002), allow a more compact lexicon for free word order languages.

[2] http://www.ims.uni-stuttgart.de/projekte/TIGER

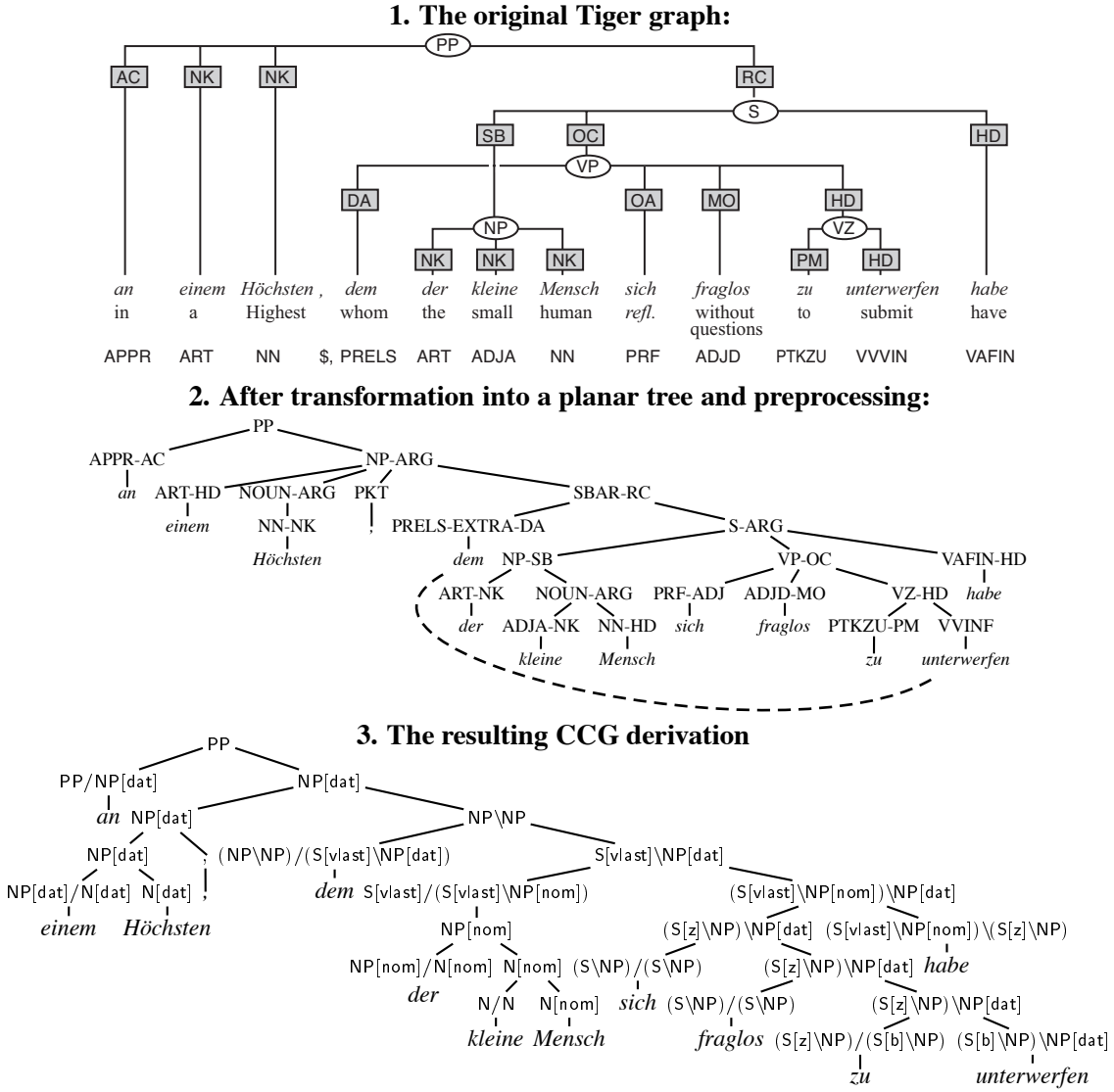


Figure 2: From Tiger graphs to CCG derivations

are extraposed relative clauses, which CCG treats as sentential modifiers with an anaphoric dependency. Arguments that are moved up are marked as extracted, and an additional "extraction" edge (explained below) from the original head is introduced to capture the correct dependencies in the CCG derivation. Discontinuous dependencies between resumptive pronouns ("place holders", PH) and their antecedents ("repeated elements", RE) are also dissolved.

2. Additional preprocessing: In order to obtain the desired CCG analysis, a certain amount of preprocessing is required. We insert NPs into PPs, nouns into NPs[3], and change sentences whose first element is a complementizer (*dass*, *ob*, etc.) into an SBAR (a category which does not exist in the original Tiger annotation) with S argument. This is necessary to obtain the desired CCG derivations where complementizers and prepositions take a sentential or nominal argument to their right, whereas they appear at the same level as their arguments in the Tiger corpus. Further preprocessing is required to create the required structures for wh-extraction and certain coordination phenomena (see below).

In figure 2, preprocessing of the original Tiger graph (top) yields the tree shown in the middle (edge labels are shown as Penn Treebank-style function tags).[4]

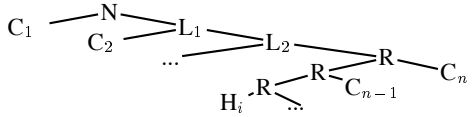
We will first present the basic translation algorithm before we explain how we obtain a derivation which captures the dependency between the relative pronoun and the embedded verb.

[3]The span of nouns is given by the NK edge label.

[4]We treat reflexive pronouns as modifiers.

3. The basic translation step Our basic translation algorithm is very similar to Hockenmaier and Steedman (2005). It requires a planar tree without crossing edges, where each node is marked as head, complement or adjunct. The latter information is represented in the Tiger edge labels, and only a small number of additional head rules is required. Each individual translation step operates on local trees, which are typically flat.

Assuming the CCG category of N is X, and its head position is i, the algorithm traverses first the left nodes $C_1...C_{i-1}$ from left to right to create a right-branching derivation tree, and then the right nodes $(C_n...C_{i+1})$ from right to left to create a left-branching tree. The algorithm starts at the root category and recursively traverses the tree.

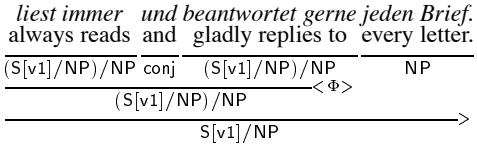


The CCG category of complements and of the root of the graph is determined from their Tiger label. VPs are S[.]\NP, where the feature [.] distinguishes bare infinitives, *zu*-infinitives, passives, and (active) past participles. With the exception of passives, these features can be determined from the POS tags alone.[5] Embedded sentences (under an SBAR-node) are always S[vlast]. NPs and nouns (NP and N) have a case feature, e.g. [nom].[6] Like the English CCGbank, our grammar ignores number and person agreement.

Special cases: Wh-extraction and extraposition In Tiger, wh-extraction is not explicitly marked. Relative clauses, wh-questions and free relatives are all annotated as S-nodes,and the wh-word is a normal argument of the verb. After turning the graph into a planar tree, we can identify these constructions by searching for a relative pronoun in the leftmost child of an S node (which may be marked as extraposed in the case of extraction from an embedded verb). As shown in figure 2, we turn this S into an SBAR (a category which does not exist in Tiger) with the first edge as complementizer and move the remaining children under a new S node which becomes the second daughter of the SBAR. The relative pronoun is the head of this SBAR and takes the S-node as argument. Its category is S[vlast], since all clauses with a complementizer are verb-final. In order to capture the long-range dependency, a "trace" is introduced, and percolated down the tree, much like in the algorithm of Hockenmaier and Steedman (2005), and similar to GPSG's slash-passing (Gazdar et al., 1985). These trace categories are appended to the category of the head node (and other arguments are type-raised as necessary). In our case, the trace is also associated with the verb whose argument it is. If the span of this verb is within the span of a complement, the trace is percolated down this complement. When the VP that is headed by this verb is reached, we assume a canonical order of arguments in order to "discharge" the trace.

If a complement node is marked as extraposed, it is also percolated down the head tree until the constituent whose argument it is is found. When another complement is found whose span includes the span of the constituent whose argument the extraposed edge is, the extraposed category is percolated down this tree (we assume extraction out of adjuncts is impossible).[7] In order to capture the topicalization analysis, main clause subjects also introduce a trace. Fronted complements or subjects, and the first adjunct in main clauses are analyzed as described in figure 1.

Special case: coordination – secondary edges Tiger uses "secondary edges" to represent the dependencies that arise in coordinate constructions such as gapping, argument cluster coordination and right (or left) node raising (Figure 3). In right (left) node raising, the shared elements are arguments or adjuncts that appear on the right periphery of the last, (or left periphery of the first) conjunct. CCG uses type-raising and composition to combine the incomplete conjuncts into one constituent which combines with the shared element:



[5] Eventive (*"werden"*) passive is easily identified by context; however, we found that not all stative (*"sein"*) passives seem to be annotated as such.

[6] In some contexts, measure nouns (e.g. *Mark*, *Kilometer*) lack case annotation.

[7] In our current implementation, each node cannot have more than one forward and one backward extraposed element and one forward and one backward trace. It may be preferable to use list structures instead, especially for extraposition.

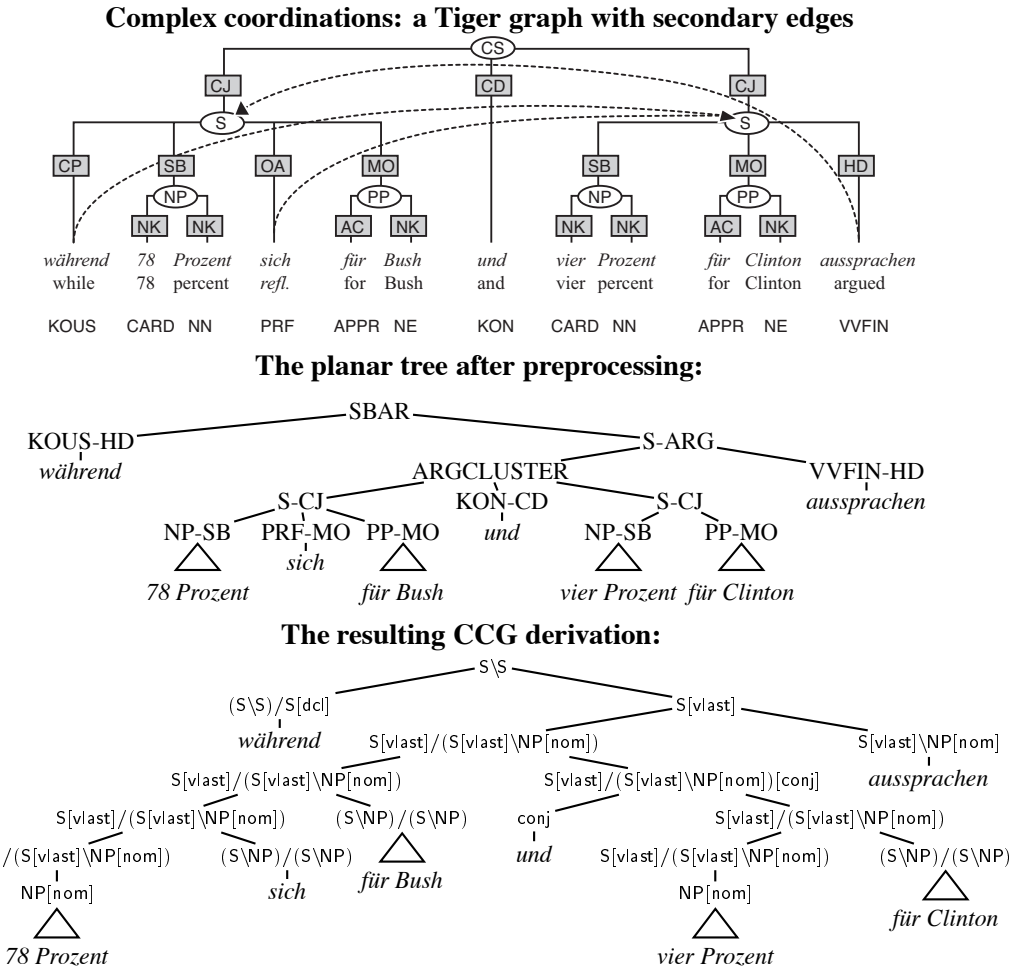


Figure 3: Processing secondary edges in Tiger

In order to obtain this analysis, we lift such shared peripheral constituents inside the conjuncts of conjoined sentences CS (or verb phrases, CVP) to new S (VP) level that we insert in between the CS and its parent.

In argument cluster coordination (Figure 3), the shared peripheral element (*aussprachen*) is the head.[8] In CCG, the remaining arguments and adjuncts combine via composition and typeraising into a functor category which takes the category of the head as argument (e.g. a ditransitive verb), and returns the same category that would result from a non-coordinated structure (e.g. a VP). The result category of the furthest element in each conjunct is equal to the category of the entire VP (or sentence), and all other elements are type-raised and composed with this to yield a category which takes as argument a verb with the required subcat frame and returns a verb phrase (sentence). Tiger assumes instead that there are two conjuncts (one of which is headless), and uses secondary edges to indicate the dependencies between the head and the elements in the distant conjunct. Coordinated sentences and VPs (CS and CVP) that have this annotation are rebracketed to obtain the CCG constituent structure, and the conjuncts are marked as argument clusters. Since the edges in the argument cluster are labeled with their correct syntactic functions, we are able to mimic the derivation during category assignment.

In sentential gapping, the main verb is shared and appears in the middle of the first conjunct:

(6) *Er trinkt Bier und sie Wein.*
He drinks beer and she wine.

As in the English CCGbank, we ignore this construction, which requires a non-combinatory "decomposition" rule (Steedman, 1990).

5 Evaluation

Translation coverage The algorithm can fail at several stages. If the graph cannot be turned into a tree, it cannot be translated. This happens in 1.3% (647) of all sentences. In many cases, this is due

[8] *Während* has scope over the entire coordinated structure.

to coordinated NPs or PPs where one or more conjuncts are extraposed. We believe that these are anaphoric, and further preprocessing could take care of this. In other cases, this is due to verb topicalization (*gegeben hat Peter Maria das Buch*), which our algorithm cannot currently deal with.[9] For 1.9% of the sentences, the algorithm cannot obtain a correct CCG derivation. Mostly this is the case because some traces and extraposed elements cannot be discharged properly. Typically this happens either in local scrambling, where an object of the main verb appears between the auxiliary and the subject (*hat das Buch Peter...*)[10], or when an argument of a noun that appears in a relative clause is extraposed to the right. There are also a small number of constituents whose head is not annotated. We ignore any gapping construction or argument cluster coordination that we cannot get into the right shape (1.5%), 732 sentences).

There are also a number of other constructions that we do not currently deal with. We do not process sentences if the root of the graph is a "virtual root" that does not expand into a sentence (1.7%, 869). This is mostly the case for strings such as *Frankfurt (Reuters)*), or if we cannot identify a head child of the root node (1.3%, 648; mostly fragments or elliptical constructions).

Overall, we obtain CCG derivations for 92.4% (46,628) of all 54,0474 sentences, including 88.4% (12,122) of those whose Tiger graphs are marked as discontinuous (13,717), and 95.2% of all 48,957 full sentences (excluding headless roots, and fragments, but counting coordinate structures such as gapping).

Lexicon size There are 2,506 lexical category types, but 1,018 of these appear only once. 933 category types appear more than 5 times.

Lexical coverage In order to evaluate coverage of the extracted lexicon on unseen data, we split the corpus into segments of 5,000 sentences (ignoring the last 474), and perform 10-fold cross-validation, using 9 segments to extract a lexicon and the 10th to test its coverage. Average coverage is 86.7% (by token) of all lexical categories. Coverage varies between 84.4% and 87.6%. On average, 92% (90.3%-92.6%) of the lexical tokens that appear in the held-out data also appear in the training data. On these seen tokens, coverage is 94.2% (93.5%-92.6%). More than half of all missing lexical entries are nouns.

In the English CCGbank, a lexicon extracted from section 02-21 (930,000 tokens) has 94% coverage on all tokens in section 00, and 97.7% coverage on all seen tokens (Hockenmaier and Steedman, 2005). In the English data set, the proportion of seen tokens (96.2%) is much higher, most likely because of the relative lack of derivational and inflectional morphology. The better lexical coverage on seen tokens is also to be expected, given that the flexible word order of German requires case markings on all nouns as well as at least two different categories for each tensed verb, and more in order to account for local scrambling.

6 Conclusion and future work

We have presented an algorithm which converts the syntax graphs in the German Tiger corpus (Brants et al., 2002) into Combinatory Categorial Grammar derivation trees. This algorithm is currently able to translate 92.4% of all graphs in Tiger, or 95.2% of all full sentences. Lexicons extracted from this corpus contain the correct entries for 86.7% of all and 94.2% of all seen tokens. Good lexical coverage is essential for the performance of statistical CCG parsers (Hockenmaier and Steedman, 2002a). Since the Tiger corpus contains complete morphological and lemma information for all words, future work will address the question of how to identify and apply a set of (non-recursive) lexical rules (Carpenter, 1992) to the extracted CCG lexicon to create a much larger lexicon. The number of lexical category types is almost twice as large as that of the English CCGbank. This is to be expected, since our grammar includes case features, and German verbs require different categories for main and subordinate clauses. We currently perform only the most essential preprocessing steps, although there are a number of constructions that might benefit from additional changes (e.g. comparatives, parentheticals, or fragments), both to increase coverage and accuracy of the extracted grammar.

Since Tiger corpus is of comparable size to the Penn Treebank, we hope that the work presented here will stimulate research into statistical wide-coverage parsing of free word order languages such as German with deep grammars like CCG.

[9] The corresponding CCG derivation combines the remnant complements as in argument cluster coordination.

[10] This problem arises because Tiger annotates subjects as arguments of the auxiliary. We believe this problem could be avoided if they were instead arguments of the non-finite verb.

Acknowledgments

I would like to thank Mark Steedman and Aravind Joshi for many helpful discussions. This research is supported by NSF ITR grant 0205456.

References

Jason Baldridge. 2002. *Lexically Specified Derivational Control in Combinatory Categorial Grammar*. Ph.D. thesis, School of Informatics, University of Edinburgh.

Alena Böhomvá, Jan Hajič, Eva Hajičová, and Barbora Hladká. 2003. The Prague Dependency Treebank: Three-level annotation scenario. In Anne Abeillé, editor, *Treebanks: Building and Using Syntactially Annotated Corpora*. Kluwer.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lexius, and George Smith. 2002. The TIGER treebank. In *Workshop on Treebanks and Linguistic Theories*, Sozpol.

Aoife Cahill, Martin Forst, Mairead McCarthy, Ruth O'Donovan, Christian Rohrer, Josef van Genabith, and Andy Way. 2005. Treebank-based acquisition of multilingual unification-grammar resources. *Journal of Research on Language and Computation*.

Ruken Çakıcı. 2005. Automatic induction of a CCG grammar for Turkish. In *ACL Student Research Workshop*, pages 73–78, Ann Arbor, MI, June.

Bob Carpenter. 1992. Categorial grammars, lexical rules, and the English predicative. In Robert Levine, editor, *Formal Grammar: Theory and Implementation*, chapter 3. Oxford University Press.

John Chen, Srinivas Bangalore, and K. Vijay-Shanker. 2005. Automated extraction of Tree-Adjoining Grammars from treebanks. *Natural Language Engineering*.

Stephen Clark and James R. Curran. 2004. Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain.

Amit Dubey and Frank Keller. 2003. Probabilistic parsing for German using Sister-Head dependencies. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 96–103, Sapporo, Japan.

Gerald Gazdar, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag. 1985. *Generalised Phrase Structure Grammar*. Blackwell, Oxford.

Julia Hockenmaier and Mark Steedman. 2002a. Acquiring compact lexicalized grammars from a cleaner Treebank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)*, pages 1974–1981, Las Palmas, Spain, May.

Julia Hockenmaier and Mark Steedman. 2002b. Generative models for statistical parsing with Combinatory Categorial Grammar. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 335–342, Philadelphia, PA.

Julia Hockenmaier and Mark Steedman. 2005. CCGbank: Users' manual. Technical Report MS-CIS-05-09, Computer and Information Science, University of Pennsylvania.

Beryl Hoffman. 1995. *Computational Analysis of the Syntax and Interpretation of 'Free' Word-order in Turkish*. Ph.D. thesis, University of Pennsylvania. IRCS Report 95-17.

Roger Levy and Christopher Manning. 2004. Deep dependencies from context-free statistical parsers: correcting the surface dependency approximation. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330.

Yusuke Miyao and Jun'ichi Tsujii. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 83–90, Ann Arbor, MI.

Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. 2004. Corpus-oriented grammar development for acquiring a Head-driven Phrase Structure Grammar from the Penn Treebank. In *Proceedings of the First International Joint Conference on Natural Language Processing (IJCNLP-04)*.

Michael Moortgat and Richard Moot. 2002. Using the Spoken Dutch Corpus for type-logical grammar induction. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC)*.

Ruth O'Donovan, Michael Burke, Aoife Cahill, Josef van Genabith, and Andy Way. 2005. Large-scale induction and evaluation of lexical resources from the Penn-II and Penn-III Treebanks. *Computational Linguistics*, 31(3):329 – 365, September.

Owen Rambow. 1994. *Formal and Computational Aspects of Natural Language Syntax*. Ph.D. thesis, University of Pennsylvania, Philadelphia PA.

Libin Shen and Aravind K. Joshi. 2005. Incremental LTAG parsing. In *Proceedings of the Human Language Technology Conference / Conference of Empirical Methods in Natural Language Processing (HLT/EMNLP)*.

Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. 1997. An annotation scheme for free word order languages. In *Fifth Conference on Applied Natural Language Processing*.

Mark Steedman. 1990. Gapping as constituent coordination. *Linguistics and Philosophy*, 13:207–263.

Mark Steedman. 1996. *Surface Structure and Interpretation*. MIT Press, Cambridge, MA. Linguistic Inquiry Monograph, 30.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, MA.

Fei Xia. 1999. Extracting Tree Adjoining Grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium (NLPRS-99)*.