YAC – A Recursive Chunker for Unrestricted German Text

Hannah Kermes, Stefan Evert

Institute for Natural Language Processing, University of Stuttgart
Azenbergstr. 12, 70174 Stuttgart, Germany
{kermes,evert}@ims.uni-stuttgart.de

Abstract

YAC is a fully automatic recursive chunker for unrestricted German text. It is especially designed to provide a useful basis for the extraction of linguistic as well as lexicographic information. Consequently, the grammar rules of YAC are implemented such as to make the resulting analysis meet the needs of an ensuing extraction process. The chunks provided by YAC are continuous parts of intra-clausal constituents including recursion but no PP-attachment or sentential elements. The chunks are additionally enriched with information about head lemma, morpho-syntactic features and certain lexical and structural properties.

1. Introduction

YAC is a fully automatic recursive chunker for unrestricted German text. It is especially designed to provide a useful basis for the extraction of linguistic as well as lexicographic information. Consequently, the grammar rules of YAC are implemented such as to make the resulting analysis meet the needs of an ensuing extraction process. The chunks provided by YAC are continuous parts of intra-clausal constituents including recursion in pre-head as well as in post-head position but no PP-attachment or sentential elements. They are additionally enriched with feature attributes including information about head lemma, morpho-syntactic features and certain lexical and structural properties.

YAC is based on a symbolic regular expression grammar written in the CQP query language (CQP is part of the IMS Corpus Workbench (CWB)[1]). The chunker works on a corpus that is tokenised and part-of-speech tagged using the STTS-tagset (Schiller et al., 1999)[2]. The German grammar further needs lemma and agreement information[3]. Our definition of a chunk deviates from the definition given by Abney (1996a), where it is defined as "a non-recursive core of an intra-clausal constituent, extending from the beginning of the constituent to its head". The broader concept of *chunk* is necessary as German commonly includes both pre-head as well as post-head recursion with respect to Nominal Phrases (NP).

1.1. A few words about German

For English the major difficulties encountered are PP-attachment and coordination with ellipsis. They can be treated on the basis of chunks according to Abney's definition. During the extraction process the single elements can be combined relatively easily. In German, such constructions occur less frequently. Instead, we often have pre-head recursion as in (1), which sounds rather awkward in English:

(1) [$_{PP}$ mit [$_{NP}$ [$_{AP}$ kleinen] , [$_{AP}$ [$_{PP}$ über [$_{NP}$ die Köpfe [$_{NP}$ der Apostel]]] gesetzten] Flammen]]
with small, above the heads of the apostles set flames
"with small flames set above the heads of the apostles"

A simple chunk analysis of this example will display phrasal structures more or less like the ones in (2). Depending on which chunker is used, the internal structures of the chunks may or may not be present.

(2) [$_{PC}$ mit [$_{NC}$ kleinen]] , [$_{PC}$ über [$_{NC}$ die Köpfe]] [$_{NC}$ der Apostel] [$_{NC}$ gesetzten Flammen]

Besides, the first NC *kleinen* and consequently the first PC *mit kleinen* will only be analysed as such if the rule for NCs allows headless NCs. Otherwise, the words would not be part of a chunk at all.

In the case of a classic chunk analysis, the extraction process would have to fill the gap between the analyses in (2) and in (1). In the simplest case, an extraction query would have to state that a PP can consist of a sequence of PCs and NCs headed by a PC or preposition, a definition which is neither very reliable nor theoretically motivated. And still, the internal structure would remain obscure. In order to observe the internal structure as well, a far more complex analysis would have to take place. The NC *der Apostel* would have to be identified as a post-head genitive modifier of the preceeding NC *die Köpfe* and embedded together with it in the PC. The last NC *gesetzten Flammen* would have to be split, and *gesetzten* would have to be identified as an adjective that is able to embed a PP into the AP. Then, the complete NP *kleinen, über die Köpfe der Apostel gesetzten Flammen* could be identified. Looking at such complex pre-head embedding, it is obvious that the classical non-recursive chunk analysis is not sufficient for German, and that we need a broader definition.

2. What is annotated

YAC is designed to provide a basis for the extraction of lexicographic as well as linguistic information, but it deliv-

[1] For more information see `http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/index.html`

[2] We used the TreeTagger of Schmid (1994) for tokenisation and part-of-speech tagging

[3] In our case lemma and agreement information was annotated using the IMSLex morphology (Lezius et al., 2000). For more information see `http://www.ims.uni-stuttgart.de/projekte/IMSLex`

ers no full parse. The idea is to build relatively flat annotations of intra-clausal constituents incrementally: adverbial phrases (AdvP), adjectival phrases (AP), noun phrases (NP), prepositional phrases (PP), and verbal complexes (VC). The structures are built as far as this can be done with high reliability and with a limited amount of lexical information. Highly ambiguous attachment decisions, including most PP-attachments, are not made. Coordinations are only taken into account if they are embedded in a larger chunk (e.g. coordination of NPs within a PP). Discontinuous elements of a phrase are left as separate structures. The maximal structures project directly to the sentence without any functional projection in between. Avoiding ambiguous decisions allows us to produce a single parse of a sentence rather than a parse forest. A similar approach was conducted by Steve Abney (Abney, 1991; Abney, 1996b) for English using a cascaded finite-state parser. In contrast to Abney, we enrich the chunks with certain lexical and structural properties as well as agreement information, in addition to the head lemma.

2.1. Recursion

Chunks produced by YAC cover pre-head as well as post-head recursion. Pre-head recursive embedding includes constructions such as *kleine, über die Köpfe der Apostel gesetzte Flammen* (small, above the heads of the apostles set flames), where the NP *die Köpfe der Apostel* is embedded in the larger NP in pre-head position. Post-head recursive embedding includes, e.g., genitive NP modifiers such as the genitive NP *der Apostel* (embedded in post-head position of the NP *die Köpfe der Apostel*), and named entities functioning as post-head modifiers (e.g., *"Endeavour"* in the NP *einer Fahrt des Shuttle "Endeavour"* (a voyage of the shuttle "Endeavour")). While pre-head recursion is performed fully, post-head recursion is restricted to cases where the attachment can be made with high reliability. Therefore, post-head genitive NPs are attached but post-head PPs are not.

2.2. Enriching Chunks

2.2.1. Head Lemma

YAC enriches the chunks with feature attributes relevant for the extraction process. During the extraction the lemma information can easily be accessed and used, e.g., to identify collocations and lexical preferences.

2.2.2. Agreement

The agreement information annotated for chunks consists of all combinations of case, gender, number that are consistent with the morpho-syntactic features of relevant elements. These elements can be terminal nodes (typically the head of the chunk and the determiner in the case of NPs) as well as embedded chunks (e.g., APs). In order for the chunk to be accepted, the single elements have to agree, i.e., there has to be at least one consistent analysis. Thus, even large structures, where the elements may be far from the head, can be assembled with high reliability. The agreement information of chunks is disambiguated as far as can be done without guessing (in contrast to a PCFG parser).

2.3. Lexical and structural properties

In some cases, chunks are enriched with feature attributes specifying the lexical or structural properties of a chunk. In many cases the lexical properties are projected from the head to the chunk. This includes temporal nouns (*Jahr* (year)) and adverbs (*immer* (always), deverbal (*gechunkt* (chunked)) and invariant (*lila* (violet)) adjectives. Some lexical properties are determined by the whole chunk, e.g., dates (*Januar 1995* (January 1995)), addresses (*Musikantengasse 1*), or multiword-cardinals (*264,6 Millionen* (264.6 million)). Some structural properties are also annotated, e.g., whether a chunk is enclosed in quotation marks (*"Wilhelm Meisters Lehrjahre"* (a novel by J.W. von Goethe)). A selection of the lexical and structural feature attributes associated with APs and NPs can be found in Table 1 and Table 2, respectively. These properties are used during the parsing process to restrict the types of chunks allowed in certain positions. Chunks enclosed in quotes, e.g., are taken to be possible post-nominal modifiers. APs whose head is an invariant adjective do not have to agree with the other elements of the NP they are embedded in. During extraction, these properties can be used, e.g., to exclude temporal NPs from subcategorisation frames, or proper nouns from collocations. Besides, deverbal adjectives can be used to collect information about the corresponding verbs (e.g. noun-verb collocations). NPs with the feature *quot* or *brac* can be used to identify named entities (e.g. *"Pizza Hut"*).

3. Technical framework

Our tools are based on the IMS Corpus Workbench[4] (CWB). The CWB is an environment for storage and querying of large corpora with shallow annotations. Currently, the maximum size of a single corpus can be approximately 300 million words, depending on the number and type of annotations. The CWB provides fast access to corpora by constructing a separate lexicon and a full index for each annotation level. The data are stored in a compact proprietary format, and compressed with specialised algorithms (Huffmann coding for the token sequence and a variable-length encoding for the index).

The CWB was initially developed for corpora annotated only at token level (typically with part-of-speech (PoS) and lemma values). Later, support for flat, non-overlapping structural annotation was added (referred to as *structural attributes*). Since this mark-up was mainly intended for the annotation of document structure (e.g., source files, paragraphs, and sentences), the regions of structural attributes are neither combined into hierarchical structures nor do they allow recursion. Besides, compression algorithms are not necessary to store the relatively small number of sentences, paragraphs, etc. in a corpus.

Queries can be specified in terms of regular expressions over tokens and their linguistic annotations, using the Corpus Query Processor (CQP). In contrast to most CFG parsers, the CQP query language allows complex expressions at the basic token level. These include regular ex-

[4] See (Christ, 1994) for an overview. More information on the IMS Corpus Workbench is available from `http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/`

Features associated with adjectival phrases		
feature	description	example
norm	default feature	
attr	attributive adjective	
invar	invariant adjective	*lila, Berliner*
pred	predicative adjective	
vder	deverbal adjective	*geplant, gechunkt*
meas	measure adjective	*Meter hohe, Hektar große*
pp	AP embedding PP	*von seiner Frau geborgten, auf seinen Sohn stolzen*
quot	AP in quotation marks	*“allzu plötzlich”, “abenteurerlich”*

Table 1: Lexical and structural features associated with adjectival phrases

Features associated with noun phrases		
feature	description	example
norm	default feature	
meas	measure noun	*Handvoll, Dollar*
ne	named entities	*Mecklenburg-Vorpommern, Professor Wilhelm Heimeyer*
temp	temporal noun	*Jahr, Wochen später*
date	date	*17. Februar 2002*
street	street address	*Musikantengasse 1*
brac	NP in brackets	*(LKA), (Framingham Heart Study)*
quot	NP in quotes	*“Kommerzielle Koordinierung”*
pron	pronominal NP	*er, sich*

Table 2: Lexical and structural features associated with noun phrases

pression matching of tokens and annotated strings (optionally ignoring case and/or diacritics), tests for membership in user-specified word lists, and arbitrary Boolean expressions over feature-value pairs. Additional, “global” constraints can be used to specify dependencies between arbitrary tokens in a CQP query.

CQP provides a simple macro language, based on string replacement with interpolation of up to 10 arguments. The “body” of a macro (which is substituted for each macro “call” in a query) may contain further (non-recursive) macro invocations. Thus, complex queries can be broken down into small parts, similar to the rules of a context-free grammar. Macro definitions are loaded from text files and can be modified at run-time.

The CWB includes support for feature-set annotations encoded as string values with a special disjunctive notation. This allows, in particular, the treatment of agreement features: all possible combinations of case, gender, and number values a certain word or phrase may have, are stored as a single feature-set annotation. Unification of feature values (which is the basis of most complex grammar formalisms) is equivalent to the (set-theoretic) intersection of the corresponding feature-sets. Special operators implemented as regular expressions are available to test for the presence of features (e.g. an NP which might have genitive case) as well as the uniqueness of feature values (an NP uniquely identified as genitive).

We considered a number of alternative approaches for the parsing stage as well, in particular those for which standard tools are available.

- Complex grammars (e.g., in the LFG or HPSG framework) can model the hierarchical structure of language, and are well-suited for handling attachment ambiguities. Drawbacks include slow parsing speed, lack of robustness, dependence on an extensive lexicon as a prerequisite, and the complex interactions between rules that complicate both grammar development and the adjustment to a particular domain seriously. Furthermore, complex grammars usually return a large number of possible analyses for each sentence, which cannot be stored and queried efficiently for large corpora. Thus, an additional, and probably rather unreliable disambiguation component or labour-intensive manual disambiguation would be necessary.

- Context-free grammars (CFGs) are modular (i.e. there is little interaction between different rules) and allow for fast parsing. In most CFG-based systems, however, modelling agreement and special (lexical) constructions requires large numbers of additional rules, which makes grammars unwieldy and slows down the parsing process. For the automatic analysis of large amounts of text, further “robustness” rules are needed. Partial or full disambiguation of agreement information is difficult to achieve.

- Probabilistic context-free grammars (PCFGs) extend the CFG formalism with a statistical model of lexical information such as subcategorisation frames and collocations. In contrast to complex grammars, probabilistic “lexicon entries” are learned from the input text and training data without human intervention.

PCFG-based parsers are slower than their CFG counterparts and require a considerable amount of working memory for their large parameter sets. A particular problem for PCFGs are marked constructions and other special cases, where the parser almost inevitably prefers a more frequent unmarked alternative. In general, PCFG parsers perform a full disambiguation of agreement features involving guesswork rather than the partial disambiguation that we prefer.

To sum up, the advantages which led us to use the CWB as a framework for our tools are: (i) The possibility to work with large corpora. After compression, the surface forms and lexical annotations (lemma, etc.) require approximately 30 bits/token of disk space, whereas categorical annotations (PoS, agreement features, etc.) require 10 bits/token or less[5]. (ii) CQP efficiently evaluates complex queries on large corpora. Disk files are accessed directly using memory-mapping and do not have to be loaded into memory first. It is this feature which makes a multi-pass algorithm (in which CQP is frequently restarted in order to re-use intermediate results) feasible at all. (iii) The query language is modular and allows easy treatment of special cases (using additional rules, word lists, or structural markup of multiword entities as structural attributes). (iv) The same representation formalism and query language can be used at the parsing stage, for interactive querying of the final results, and for the extraction of lexical information. In order to test a new or changed rule, it is possible to abort the parsing process at any stage. The rule can be applied to the current state of the annotation in an interactive CQP session. The results of the rules can be viewed on the screen. If necessary, the rule can be modified and applied again until the desired results are obtained. The parsing process can then be continued with the new or modified rule.

4. The chunking process

YAC is divided into three levels: (i) a first level, where most of the lexical information is introduced and specific chunks are built, (ii) a second level, where the main parsing is performed, and (iii) a third level, where structures can be checked for correctness and where the hierarchy is built.

4.1. First Level

In the first level most of the lexical information is introduced and added in the form of chunks with special annotations. It is information about certain lexical properties of words that can be relevant and helpful for future extraction processes.

Besides of the lexical information, chunks with a specific internal structure are annotated at the first level. These chunks are "non-recursive" units that do not follow the general patterns but need highly specialised rules. The detection of such chunks is in general triggered by lexical properties. Such properties can sometimes be inferred from the PoS-tags, as is the case for noun chunks representing named entities or proper names (e.g., *General Electric Company; Joint Venture*, and *Johann Sebastian Bach; Goalgetter Rod Poindexter*), where the trigger is the PoS-tag *NE* (proper name). They can also be derived from the lexical information provided in the form of word and phrase lists. In this case, the lexical information is not only introduced to help future extraction but also for the parsing process itself. It is used as a trigger to build up specific structural units, such as: (i) dates, e.g., *Februar 1991* (February 1991); *Anfang April* (begin of April); *19.6.1992*, and (ii) multi-word lexemes, e.g., multi-word cardinals (*264,6 Millionen* (264.6 millions); *fünf Milliarden* (five billions)).

Other chunks with a specific internal structure that are annotated in the first level are: (i) chunks in brackets, e.g. *(Fortune-Liste); (DEC)*, and (ii) chunks in quotation marks, e.g. *"Omaha Beach"; "Wilhelm Meisters Lehrjahre"*. The triggers are textual markers which group together sequences of words and indicate the specific character of the chunk. This information is especially helpful when the PoS-tag (probably a tagging error) does not indicate this property.

The properties of the chunks, whether lexical or structural, are annotated as a set of feature values. The information gathered at the first level, in the form of chunks and feature attributes, can easily be integrated in further parsing steps. It is the basic knowledge source and the foundation for the rules of the main parsing level. The advantages of introducing lexical information and, especially, assembling chunks and phrases with a specific internal structure in a separate level are: (i) the rules can be designed for a specific purpose without interacting with the rules of the main parsing level and the correction level, and (ii) the rules of the main parsing level can be kept relatively simple and general as many special cases have already been dealt with.

4.2. Second Level

4.2.1. Building complex structures

In the second level a set of relatively simple and general rules is applied repeatedly to form larger and larger structures. The rules can be kept relatively simple because they refer to structures built in the first level or in earlier iterations of the second level. The rule for NPs, e.g., simply states that APs can modify a head noun. However, it does not have to state what an AP may consist of as the structure is already annotated in the corpus. The examples in (3) show this.

(3) a. [$_{NP}$ eine [$_{AP}$ verständliche] Sprache]
an understandable language

b. [$_{NP}$ eine [$_{AP}$ für den Anwender verständliche] Sprache]
a for the user understandable language
"a language understandable for the users"

The same rule (NP → Det AP N) can be used for both NPs in (3) The only difference is that the AP that is being embedded in (3-b) is more complex than the AP embedded in (3-a). The size, and thus the complexity of the NP in (3-b) is a result of the complexity of the embedded structure and not of the grammar rule that builds the NP. The same is true

[5]For instance, a 100 million word corpus would require ≈360 MBytes per lexical attribute, and less than 120 MBytes for each categorical annotation

for the PPs in (4-a) and (4-b) below. The complex PP in (4-b) is built by the same relatively simple rule stating that a PP can consist of a preposition and a NP that builds the simple PP in (4-a). Again, the complexity of the PP in (4-b) is a result of embedding a complex phrasal structure (in this case a NP) rather than the result of applying a complex rule.

(4) a. [$_{PP}$ auf [$_{NP}$ dem Giebel]]
on [top of] the gable

b. [$_{PP}$ auf [$_{NP}$ dem westwärts gerichteten Giebel des heute in barockem Gewande erscheinenden Gotteshauses]]
on [top of] the westwards pointed gable of the today in baroque garment appearing Lord's house
"on top of the westwards-pointed gable of the Lord's house that nowadays appears in a baroque garment"

4.2.2. Function of embedded chunks

As in the examples in (3) and (4), chunks can be embedded in another chunk as complements or modifiers. The chunks built in the first level can additionally form the kernel of larger chunks. In the examples in (5), noun chunks are embedded in a larger NP at the position normally occupied by a single head noun. This is exactly what the rule for NPs states: the position of the head can be occupied by a common noun or an NC (noun chunk). In the first case the common noun is taken as head of the NP. In the latter case the head of the noun chunk is projected to the NP.

(5) a. [$_{NP}$ das [$_{NC}$ "Space Hotel"]]
the "Space Hotel"

b. [$_{NP}$ der ehemalige sowjetische [$_{NC}$ Präsident Michail Gorbatschow]]
the former soviet President Michail Gorbachev

4.2.3. Function of feature attributes

Chunks may be embedded independently of the feature attributes associated with them, i.e., they are used according to their phrasal category. Any NC can, e.g., form the kernel of an NP. In (5-a) a NC with the feature attribute *quot* (stating that the NC is enclosed in quotation marks) forms the kernel of the NP, and in (5-b) it is a NC with the feature attribute *ne* (a named entity).

The feature attributes, however, can also be used to specify the characteristics of a chunk that is to be embedded more closely. NPs as post-head modifiers of nouns, e.g., are restricted to genitive NPs, or NPs with the feature *brac*, *quot* or *ne* as the examples in (6) show.

(6) a. [$_{NP}$ dem Slogan [$_{NP}$ "The Windows Solution"]]
the slogan "The Windows Solution"

b. [$_{NP}$ Ulrike Linden [$_{NP}$ (Klavier)]]
Ulrike Linden (piano)

c. [$_{NP}$ die Firma [$_{NP}$ Nordsee]]
the company Nordsee

4.2.4. Considering special cases

In contrast to many CFG-parsers, YAC considers some special cases, including parentheses and quotation marks. This allows us to include PP-attachment in some cases without losing the reliability of the resulting chunks and without introducing ambiguities. If there are surrounding quotation marks, PP-attachment is performed within an NP construction, as the examples in (7) show.

(7) a. [$_{NP}$ "Einladung [$_{PP}$ zur Enthauptung]"]
"invitation to [the] decapitation"

b. [$_{NP}$ die schlagfertige "Frau [$_{PP}$ mit den Hüten]"]
the quick-witted "woman with the hats"

The quotation marks may be in different positions, e.g., surrounding the whole NP as in (7-a) or only part of the NP as in (7-b), but both quotation marks have to be present and the entire PP has to be between them.

4.2.5. Overgeneralisation

The second level allows for overgeneralisation of certain constructions. The rule building APs, e.g., states that every adjective may embed any number of immediately preceding PPs. This leads to a vast overgeneralisation of structures, producing correct APs such as (8-a) as well as incorrect APs (8-b) (the correct structure of (8-b) is shown in (8-c)). These APs are annotated intermediately and marked as hypothetical. The decision whether they are correct or not is postponed to the third level, as it is not possible to make this decision immediately.

(8) a. Lisa hat [$_{NP}$ ihre [$_{AP}$ wegen des Regens nassen] Füße] abgetrocknet.
Lisa has her because of the rain wet feet dried.
"Lisa has dried her feet that where wet because of the rain."

b. Lisa hat [$_{NP}$ [$_{AP}$ wegen des Regens nasse] Füße].
Lisa has because of the rain wet feet.
"Lisa has wet feet because of the rain."

c. Lisa hat [$_{PP}$ wegen des Regens] [$_{NP}$ [$_{AP}$ nasse] Füße].

4.3. Third Level

The third level is responsible for two things: (i) it checks hypothetical structures, and (ii) it builds the hierarchical structure of the chunks.

4.3.1. Checking of hypothetical structures

In order to determine whether to accept or reject a hypothetical structure, its position in the sentence is taken into account. Hypothetical structures are accepted if they are in a "secure context" and rejected otherwise. In the case of the APs in (8) this means that they have to be embedded in a NP with at least one non-hypothetical element preceding the AP, in order to be accepted. The AP in (8-a) is accepted as it is embedded in the NP *ihre wegen des Regens nassen Füße* with the possessive pronoun *ihre* preceding it. In (8-b) the AP is embedded in the NP *wegen des Regens nasse Füße*. However, there is no non-hypothetical element preceding it inside the NP. Thus, the AP is rejected and

built back to the AP-kernel *nasse*. The NP is built back to *nasse Füße*, respectively. The result is the correct structural analysis as in (8-c).

4.3.2. Hierarchy building

At the end of the parsing process, only maximal constituents of each category are annotated. In order to obtain the internal structure of recursive chunks (such as the embedded NPs in (1)), the results of each stage of the incremental parser are collected and combined into a hierarchy. Using the position of the head as a reference, only the largest version of each phrase is included. Hypothetical structures (4.2.5.) may be deleted in the third level.

4.4. Output

The output of YAC is a simple XML format (an example is given in Figure 1), which can be imported into the TIGERSearch program (König and Lezius, 2001a; König and Lezius, 2001b)[6] and displayed with the TIGERSearch graph viewer (Figure 2). Alternatively, the chunks can be automatically "written back" to the original CWB corpus.

The XML format can also be easily converted to HTML for displaying. Data extraction is possible with simple XSLT stylesheets. An example of a XSLT stylesheet extracting adjective-noun collocations is displayed in Figure 3. More complex queries modelling sentence structure (e.g. to identify V+Obj collocations) can be written in the CQP query language.

5. Evaluation

As a gold standard for the evaluation of the NP chunks we used the NEGRA Corpus (Skut et al., 1998)[7] consisting of 355,096 tokens of German newspaper text with manually corrected part-of-speech tagging and parse trees. A reference set of 99,238 NPs was extracted from a version of the NEGRA corpus encoded in the TigerXML format (Mengel and Lezius, 2000) using XSLT stylesheets. Unfortunately, the syntactic annotation scheme of the NEGRA treebank (Skut et al., 1997), which omits all projections that are not strictly necessary to determine the consituent structure of a sentence, is not very well suited for automatic extraction tasks. Thus, it was not sufficient to look for all instances of NPs in the NEGRA corpus that suited our definition of a noun chunk, as not all NPs have an NP node (e.g. single item NPs, NPs embedded in a PP, and cardinal numbers. The latter where removed as it was not possible to filter them out correctly). Besides, some NPs have a different category (e.g. multiword proper nouns (MPN)). Therefore, the various instances ofa NP notation had to be extracted using separate rules.

5.1. Construction of the reference set

The annotation of the NEGRA corpus consists of full parse trees. YAC, however, is a recursive chunker producing only structures according to our chunk definition (repeated here for convenience): continuous parts of intraclausal constituents including recursion in pre-head as well as in post-head position but no PP-attachment or sentential elements. Therefore, the NEGRA NPs have to be reduced so as not to include PP-attachments and clausal attachments, besides discontinous NPs are removed. Because of the characteristics of the NEGRA annotation as described above, this was not a trivial task, requiring complicated and inefficient XSLT stylesheets.

Some categories (e.g. MPN) include material that would usually not be subsumed under the category NP (e.g. all kinds of foreign word material). Because of the rather specific annotation scheme of the NEGRA corpus it was not always possible to filter out all the elements listed above. We plan to refine the stylesheets and check critical cases manually, in order to obtain a true gold standard for future evaluations.

5.2. Evaluation results

For the evaluation we ran YAC on the text of the NEGRA corpus using the manually disambiguated part-of-speech tagging provided. Lemma and agreement information from the IMSLex morphology was added. The 99,238 NPs in the reference set were then compared to the 96,447 NPs identified by YAC. Of those, 81,730 were correct (true positives), corresponding to a precision of 84.74% and a recall of 82.36%.[8] Since the head lemmas of NPs are not explicitly annotated in the NEGRA treebank and would have to be guessed from the part-of-speech tags at token level, we did not evaluate the head lemma annotations provided by YAC.

In real applications manually corrected part-of-speech tagging is not available. For this reason, we also evaluated YAC on a version of the corpus that was automatically part-of-speech tagged with the TreeTagger. Using its standard training corpus and a custom tagger lexicon based on the IMSLex morphology, the TreeTagger achieved a tagging precision of 94.82% (336,692 tokens correct out of 355,096). Most of the tagging errors were proper nouns that where not correctly identified by the morphology.

With this fully automatic process, YAC identified 99,353 NPs, of which 78,391 were correct. This gives a precision of 78.90% and a recall of 78.99%.

5.3. Discussion of the results

YAC was not developed and tested on the NEGRA treebank. Therefore, evaluating YAC on this corpus shows its realistic performance on German newspaper text. The rules of YAC allow it to be easily adapted to specific text types. Thus, it would have been possible to optimise the rules for the specific characteristics of the NEGRA corpus. We deliberately did not do so, as we want to present a tool that is useful for extraction processes on unrestricted text as it performs in a real-life application.

Therefore, we believe the overall system performance (combining TreeTagger, IMSLex, and YAC) with a precision and recall close to 80% to be a good indicator of the usefulness of our tool.

[6] see also `http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERSearch/`

[7] for details, see `http://www.coli.uni-sb.de/sfb378/negra-corpus/negra-corpus.html`

[8] Note that this evaluation strategy is equivalent to computing labelled precision and labelled recall restricted to NP chunks.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<tlipp>
  <!-- Sentence #1 -->
  <s>
    <pp f="|nodet|norm|" h="mit:Flamme">
      <t><a>mit</a></t>
      <np f="|nodet|norm|" h="Flamme">
        <ap f="|attr|norm|" h="klein">
          <ac f="|attr|norm|" h="klein">
            <t><a>kleinen</a></t>
          </ac>
        </ap>
        <t><a>,</a></t>
        <ap f="|attr|hypo|pp|vder|" h="gesetzt">
          <pp f="|norm|" h="über:Kopf">
            <t><a>über</a></t>
            <np f="|norm|" h="Kopf">
               <t><a>die</a></t>
               <t><a>Köpfe</a></t>
               <np f="|norm|" h="Apostel">
                <t><a>der</a></t>
                <t><a>Apostel</a></t>
               </np>
             </np>
           </pp>
           <ac f="|attr|norm|vder|" h="gesetzt">
             <t><a>gesetzten</a></t>
           </ac>
         </ap>
         <t><a>Flammen</a></t>
       </np>
     </pp>
   </s>
</tlipp>
```

Figure 1: XML output format of YAC

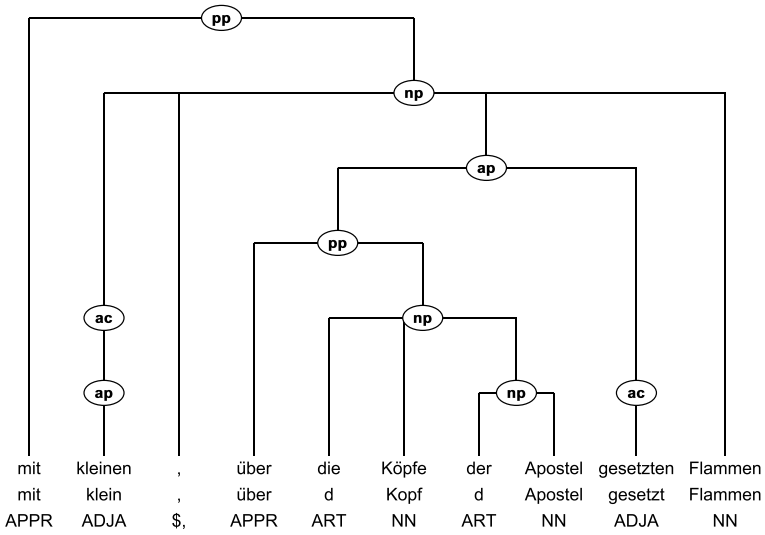

Figure 2: Tree diagram of YAC output

As extracting the reference from the NEGRA treebank proved to be problematic we intend to improve our stylesheets further. Additionally, the reference will have to be manually checked to provide a true gold standard according to our chunk definition. Thus, the evaluation figures presented have to be relativised.

6. References

Steven Abney. 1991. Parsing by chunks. In Robert Berwick, Steven Abney, and Carol Tenny, editors, *Principle-Based Parsing*. Kluwer Academic Publishers.

Steven Abney. 1996a. Chunk stylebook. Working draft.

Steven Abney. 1996b. Partial parsing via finite-state cascades. In *Proceedings of the ESSLLI '96 Robust Parsing Workshop*.

Oliver Christ. 1994. A modular and flexible architecture for an integrated corpus query system. In *Papers in Computational Lexicography COMPLEX '94*, Budapest, Hungary.

Esther König and Wolfgang Lezius. 2001a. The TIGER language - a description language for syntax graphs. Part 1: User's guidelines. Technical report, IMS, University of Stuttgart.

Esther König and Wolfgang Lezius. 2001b. The TIGER language - a description language for syntax graphs. Part 2: Formal definition. Technical report, IMS, University

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<xsl:transform version="1.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" encoding="ISO-8859-1"/>
<xsl:strip-space elements="*"/>

<!-- simply process each <np> element which has <ap> children -->
<xsl:template match="np">
  <xsl:param name="self" select="."/>
  <xsl:for-each select="ap">
    <xsl:value-of select="@h"/>
    <xsl:text>&#32;</xsl:text>
    <xsl:value-of select="$self/@h"/>
    <xsl:text>&#10;</xsl:text>
  </xsl:for-each>
  <xsl:apply-templates/> <!-- process embedded NPs -->
</xsl:template>

<!-- skip the tokens -->
<xsl:template match="t" />

</xsl:transform>
```

Figure 3: XSLT stylesheet extracting adjective-noun collocations

of Stuttgart.

Wolfgang Lezius, Stefanie Dipper, and Arne Fitschen. 2000. IMSLex – representing morphological and syntactical information in a relational database. In Ulrich Heid, Stefan Evert, Egbert Lehmann, and Christian Rohrer, editors, *Proceedings of the 9th EURALEX International Congress, Stuttgart, Germany*, pages 133–139, Stuttgart, Germany.

Andreas Mengel and Wolfgang Lezius. 2000. An XML-based representation format for syntactically annotated corpora. In *Proceedings of the Second International Conference on Language Resources and Engineering (LREC)*, volume 1, pages 121–126, Athens, Greece.

Anne Schiller, Simone Teufel, Christine Stoeckert, and Christine Thielen. 1999. Guidelines fuer das tagging deutscher textcorpora mit stts. Technical report, Universitaet Stuttgart,IMS and Universitaet Tuebingen, November.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK.

Wojciech Skut, Brigitte Krenn, Thorsten Brants, and Hans Uszkoreit. 1997. An annotation scheme for free word order languages. In *Proceedings of the Fifth Conference on Applied Natural Language Processing ANLP-97*, Washington, DC.

Wojciech Skut, Thorsten Brants, Brigitte Krenn, and Hans Uszkoreit. 1998. A linguistically interpreted corpus of german newspaper texts. In *Proceedings of the ESSLLI Workshop on Recent Advances in Corpus Annotation*, Saarbrücken, Germany.