

Interactive Corpus Annotation

Thorsten Brants, Oliver Plaehn

Computational Linguistics

Saarland University

66041 Saarbrücken, Germany

{brants,plaehn}@coli.uni-sb.de

In Second International Conference on Language Resources and Evaluation LREC-2000

May 31 – June 2, 2000, Athens, Greece

Abstract

We present an easy-to-use graphical tool for syntactic corpus annotation. This tool, Annotate, interacts with a part-of-speech tagger and a parser running in the background. The parser incrementally suggests single phrases bottom-up based on cascaded Markov models. A human annotator confirms or rejects the parser's suggestions. This semi-automatic process facilitates a very rapid and efficient annotation.

1. Introduction

During the creation of the NEGRA corpus¹ (Skut et al., 1997), we developed very efficient interactive annotation tools. An easy-to-use graphical tool, Annotate, is used to manipulate syntactic structures. Annotate interacts with a part-of-speech tagger and a parser running in the background (Brants, 1999; Brants, 2000), thus facilitating rapid semi-automatic corpus annotation.

Section 2 describes the graphical annotation tool and the annotation process. In section 3, we present a typical example of the interactive semi-automatic annotation process supported by our annotation tool and the tagger/parser running in the background. The parser is described in section 4. The internal processes employed in the parser during the annotation of the sample sentence from section 3 are exemplified in section 5. Section 6, finally, gives some conclusions.

2. The Annotation Tool

Annotate provides a comprehensive set of commands for efficient creation and manipulation of syntactic structures. The structures consist of trees with possibly crossing branches, labeled terminal nodes (part-of-speech tags and morphological information), labeled edges (grammatical functions), and labeled non-terminal nodes (phrase categories). In addition, so-called secondary links can be drawn between arbitrary nodes, forming directed graphs. These can be used for incorporating structure sharing information, e.g. for multiple PP attachment, to represent ellipses, or to resolve anaphors. Annotate provides immediate graphical feedback on all changes applied to the syntactic structure. Other features of Annotate include a search function, basic tokenization commands (for splitting or merging words, moving sentence boundaries, etc.), postscript output, an undo function, and online help for the different label sets.

Annotate's most notable feature is its interface to an external parser>tagger, which allows for a semi-automatic annotation procedure. First, the tagger determines a part-of-speech tag for each word of the current sentence. Based on

the tags' probabilities, the tagger distinguishes reliable and unreliable assignments. Unreliable assignments are highlighted and the annotator is prompted for confirmation or correction. Second, the syntactic structure is built bottom-up. The parser incrementally suggests new phrases, based on the already constructed partial syntactic structure and the part-of-speech tags. If the suggested phrase is correct, the annotator accepts it. Otherwise, she rejects it and may call the parser for a new suggestion. Alternatively, she may decide to insert a new phrase manually. The internal structures used by the parser are updated, and it again suggests a new phrase based on the existing partial structure. In addition to phrase hypotheses, the parser suggests labels for nodes and edges. Reliable and unreliable assignments are distinguished, and the annotator is prompted for confirmation or correction of the unreliable assignments.

3. Sample Annotation

Below, we present a typical example of the interactive, semi-automatic annotation process supported by our annotation tool, Annotate, and the tagger/parser running in the background. The parser is viewed as a black box throughout the presentation in this section. The internal processes of the parser when applied to the same sentence as below are described in section 5.

After startup and selection of a corpus to be modified, Annotate displays the current sentence in its main window². Since the sentence is not yet part-of-speech tagged, clicking the right mouse button calls the external part-of-speech tagger, which assigns the tags shown in figure 1.

The assignment of the tag VAINF (for "verb, auxiliary, infinitive") for *werden* has been judged unreliable by the parser, so the annotator is prompted for confirmation or correction. Since VAINF is indeed the correct part-of-speech tag, the annotator presses Enter to confirm it. Clicking the right mouse button again now calls for the external parser to suggest a phrase (see figure 2).

All label assignments for this phrase are reliable, so no further interaction with the annotator is needed. Calling

¹For availability, please check <http://www.coli.uni-sb.de/sfb378/negra-corpus/>

²The English translation of the sample sentence in the following figures is shown for demonstration purposes only and is not part of the normal annotation.

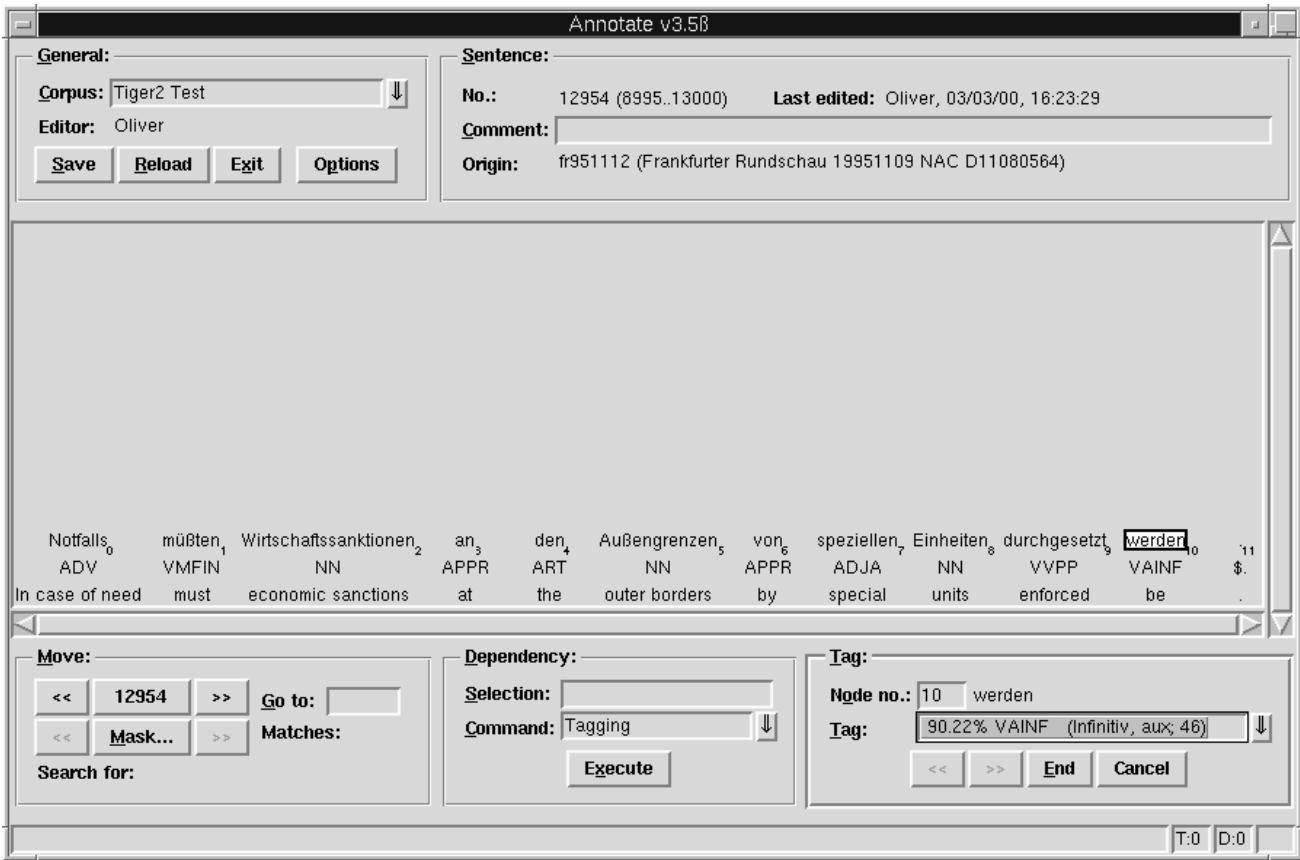


Figure 1: Only the part-of-speech tag VAINF needs to be confirmed

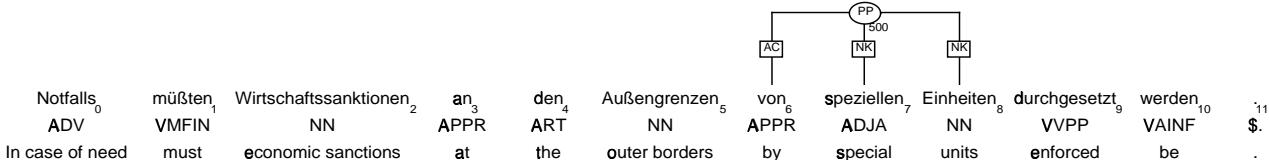


Figure 2: A correct phrase hypothesis suggested by the parser

the parser again, it suggests a new phrase (501), which is also correct. The next suggestion (phrase 502; cf. figure 3) is incorrect, since the PP *an den Außengrenzen* is not a modifier of the NP *Wirtschaftssanktionen*.

The annotator therefore rejects this phrase hypothesis with a simple mouse click. Since the parser now knows that *Wirtschaftssanktionen* and the PP *an den Außengrenzen* do not form an NP, it will not suggest this phrase again. Instead, it comes up with a different hypothesis, namely to group the two PPs and *durchgesetzt* into a VP, as shown in figure 4.

One of the edge labels is unreliable and needs to be confirmed. After calling the parser two more times, the syntactic structure is complete (cf. figure 5). Since the syntactic structure is as desired and does not need any further manipulation, the annotator switches to the next sentence, and the process described above is repeated.

4. The Parser

Annotate interacts with a parser running in the background to produce syntactic structures on-line. It sends a

partial annotation to the parser, which returns one new element. The user can either accept or reject this new element.

The parser is based on the part-of-speech tagger TnT (Brants, 2000)³ and *Cascaded Markov Models* (Brants, 1999). Each layer of the syntactic structure is represented by a separate Markov Model, starting with the part-of-speech layer. Higher layers are a generalization of the part-of-speech tagging layer. For the purpose of annotation, the tagger and the parser both run in interactive mode.

First, a sequence of part-of-speech tags is generated on-line for a given sentence. In order to facilitate error detection, the tagger generates alternative tags together with their probabilities $P(t_1)$ and $P(t_2)$, giving rise to a reliability measure. If the distance (their quotient) is large, the best tag is simply added to the annotation. If their distance is small, the human annotator is asked for confirmation. We choose a threshold θ and classify the best tag t_1 :

$$\text{reliable, if } \frac{P(t_1)}{P(t_2)} \geq \theta; \text{ unreliable, if } \frac{P(t_1)}{P(t_2)} < \theta.$$

³cf. <http://www.coli.uni-sb.de/~thorsten/tnt>

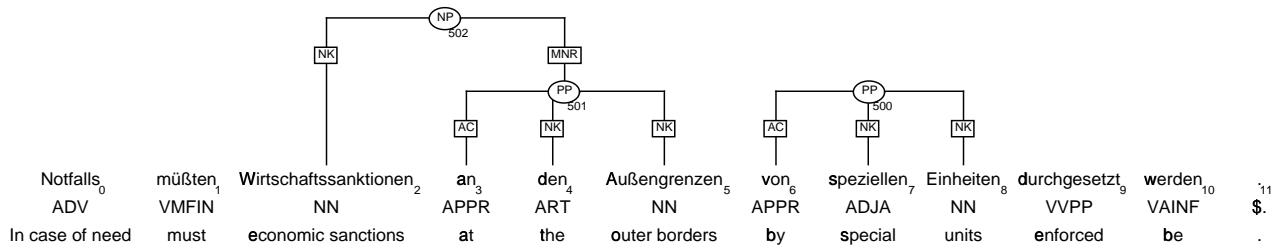


Figure 3: The suggested NP is incorrect

Figure 4: The VP suggested by the parser is correct but one of the edge labels needs to be confirmed

θ is chosen empirically such that the expected accuracy of all reliable cases is above 99%.

For part-of-speech tagging in the NEGRA corpus, approx. 85% of the tags are classified as reliable with $\theta = 100$. These have an accuracy of 99.2%. Such a level of accuracy is usually very hard to achieve even for human annotators. The remaining 15% are classified as unreliable with an accuracy of 83.0%. The human annotator mainly needs to concentrate on these 15%, which speeds up the process enormously.

Having finished the part-of-speech annotation, it is sent to the parser. The parser builds a lattice with all phrase hypotheses for layer 1 of the resulting structure. With the help of a Markov Model, the best phrase hypothesis is selected and presented to the annotator (please see (Brants, 1999) for details on the selection process). If the user accepts the phrase, all hypotheses that are not compatible with this phrase are removed from the lattice. If the user rejects it, this phrase is removed from the lattice. Probabilities are re-

calculated, and again the best hypothesis is displayed to the annotator.

For a given annotation with n structural layers, $n+1$ lattices are generated and processed using Cascaded Markov Models. At each point, the phrase with the highest probability is displayed. The suggestion of the parser is correct in approx. 70% of the cases. This low number is partly due to crossing branches in the corpus which cannot be handled by the parser. Nevertheless, the human annotator needs to take care of less than a third of all phrases.

The program guides the human annotator through the syntactic structure. We prefer this incremental, bottom-up approach over the immediate generation of complete structures and subsequent error correction by a human because our approach is much faster to perform and much less error-prone.

Labels for grammatical functions (edge labels) are inserted using Markov Models (Brants et al., 1997). Basically, the same technique as for part-of-speech tagging is

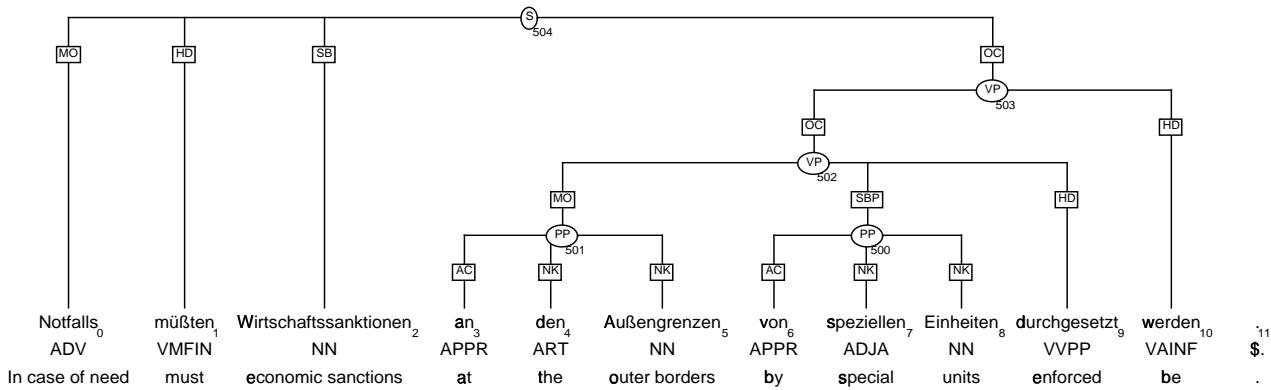


Figure 5: Completed syntactic structure

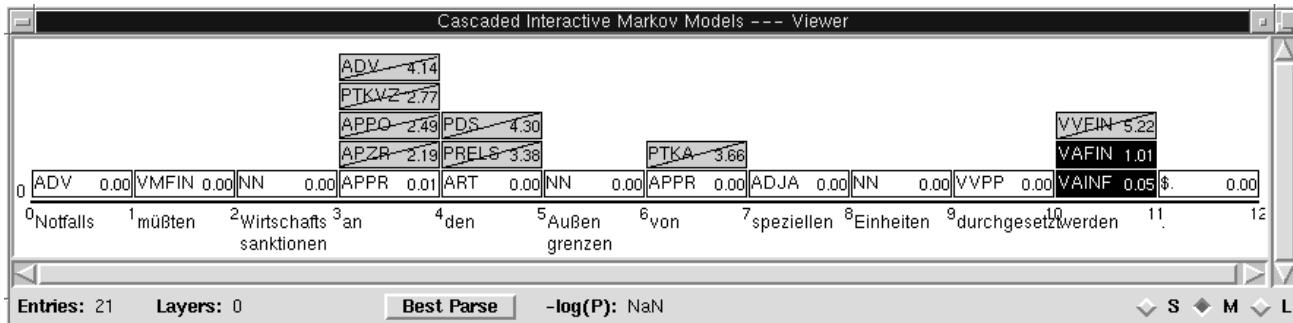


Figure 6: Generation of part-of-speech tags.

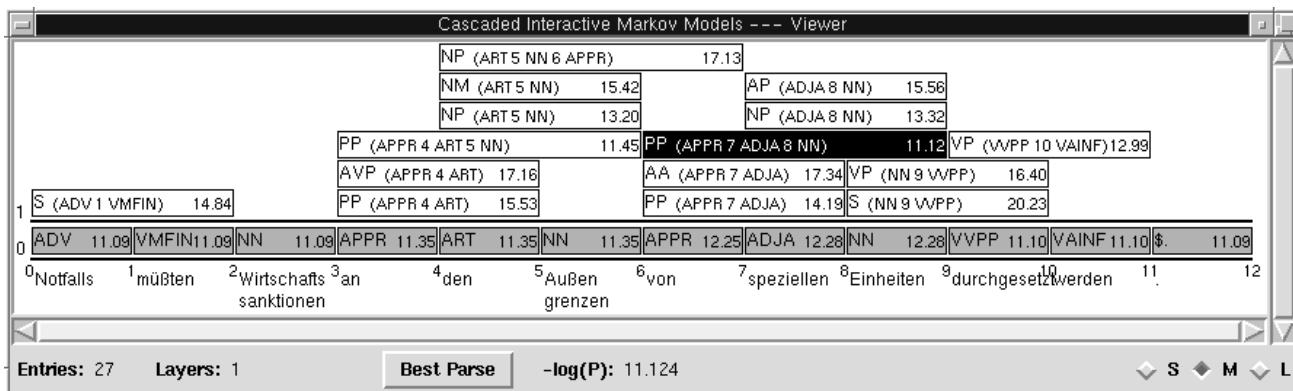


Figure 7: First hypothesis (black), alternatives (white), and previously confirmed elements (grey).

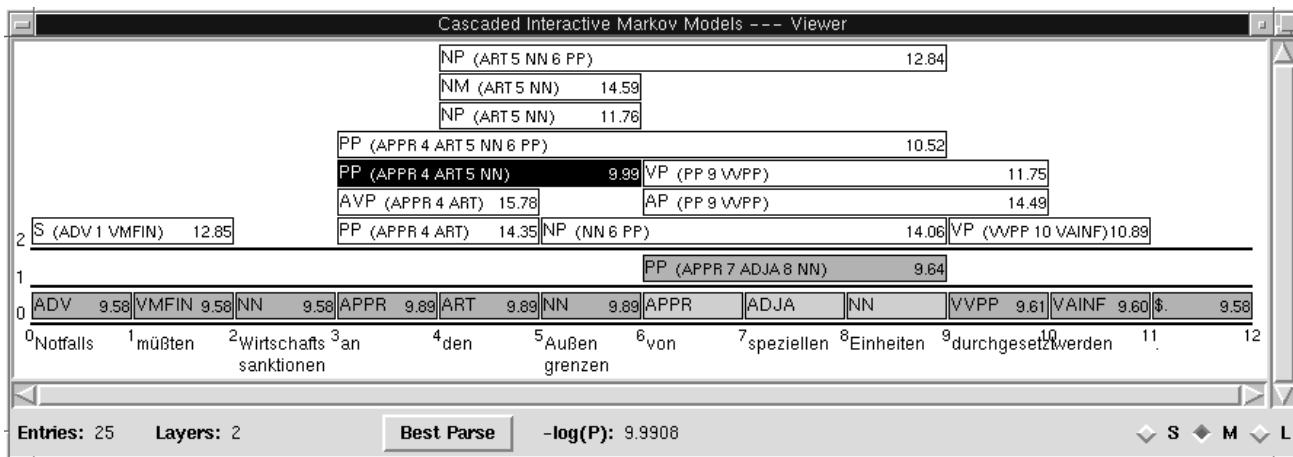


Figure 8: Second hypothesis.

employed. Instead of assigning tags to words, we assign grammatical functions to tags and node labels. The main difference is that each phrase type is processed with a separate Markov model. As in the case of part-of-speech tagging, a reliability measure is employed in order to significantly reduce the work by concentrating on unreliable tags. 91% of the assignments are classified as reliable, with an accuracy of 99.6%; the remaining 9% are classified as unreliable with an accuracy of 79.2%.

5. Sample Annotation — Parser Internals

We now present the internal part of the parsing process, i.e., hypothesis selection using cascaded Markov models. Section 3 showed the parts that are displayed to the human annotator, the lattices presented in this section are usually invisible.

Hypotheses are organized in lattices which are filtered by Markov models. Each layer is processed by a different model. Each element in the lattice receives a forward-backward probability (γ probability). The following figures show the negative logarithms (base 10) of these probabilities⁴.

First, part-of-speech tags are generated by the model for layer 0. The system inserts a tag automatically if a word gives rise to only one hypothesis or if alternative tags have a low probability. These *reliable* assignments are based on the distance $\log_{10} P(\text{best}) - \log_{10} P(\text{alternative}) \geq 2$. If the value is less than 2, the assignment is regarded as *unreliable* and highlighted⁵.

Figure 6 shows reliable tags in white, alternatives of reliable tags (with very low probabilities) in grey and dashed, and unreliable tags in black. In this case, only the tag for *werden* is unreliable.

After confirmation of the tag VAINF for *werden*, we proceed with the structural annotation. The main difference between layer 0 (for part-of-speech tagging) and higher layers (for phrases) is that at layer 0, all hypotheses span exactly one word, while at higher layers, hypotheses may span several words. When the annotator presses the mouse button, all phrase hypotheses according to a context-free grammar (generated from previously annotated sentences) enter the lattice. Forward-backward probabilities are calculated. The phrase with the highest probability is selected and displayed to the annotator.

Grammatical functions (edge labels) for a suggested phrase are generated in a very similar way as part-of-speech tags (cf. figure 6). A set of Markov models assigns labels to part-of-speech and non-terminal nodes. Each phrase type (dominating the processed tags and nodes) is represented by a separate Markov model. Alternatives are generated internally, and labels for which an alternative within a pre-defined threshold exist are marked for confirmation by the human annotator.

Figure 7 shows elements that were previously selected or confirmed by the annotator in grey. This is the existing

⁴The probabilities are not normalized. Normalization is not necessary for parsing since we are interested in rankings, not in absolute probabilities.

⁵This value is empirically determined and corresponds to a factor of 100 (see section 4).

annotation upon which the next phrase is built. Context-free hypotheses are shown in white, the best of these is marked black (the value $-\log(\gamma) = 11.12$ is smaller than all others). This PP is presented to the annotator. The PP is confirmed by the annotator, so all hypotheses that are not compatible with this choice are removed (₆PP₈; ₆AA₈; ₇NP₉; ₇AP₉; ₈S₁₀; ₈VP₁₀).

Furthermore, new hypotheses on top of the new PP are generated and enter the lattice (cf. figure 8).

The highest layer now is layer 2, so the Markov model for layer 2 processes the lattice. Fixed elements are shown in grey, new elements are white, and the best new element, which is displayed to the annotator, is marked black. The best hypothesis in this case is the PP *an den Außengrenzen*, which is correct and therefore confirmed.

All incompatible hypotheses are removed, new hypotheses enter the lattice. Layer 2 is still the highest layer.

The NP suggested next (cf. figure 9) is not correct. After rejection, this NP and all hypotheses with the same span are removed (₂S₆; ₂NP₆; ₂VP₆). The remaining hypotheses are again processed by the Markov model, as shown in figure 10.

The best phrase now is the VP that spans the input from position 3 to position 10. This hypothesis is correct and therefore confirmed. In the next step, the next VP is built (cf. figure 11).

And finally, the S node is suggested and confirmed (cf. figure 12). This completes the annotation.

6. Conclusions

The presented incremental bottom-up process always builds on previously confirmed phrases. This has two advantages for the annotation process. First, the annotator is guided through the structure and therefore looks at each phrase; and second, the parser can immediately exploit corrections made by the annotator.

This semi-automatic process facilitates a very rapid annotation. A trained annotator needs on average 50 seconds/sentence (approx. 1,300 tokens/hour) for part-of-speech plus structural annotation in the NEGRA corpus. This is the fastest structural annotation reported in the literature.

The tools are not restricted to a particular tagset or annotation scheme. It is easy to train the tagger and parser on a small amount of annotated data and to subsequently apply them to annotating according to the new scheme. Using the newly annotated data for training improves the parser's performance and thereby constitutes a bootstrapping approach. In addition to their use for creating the NEGRA corpus, the tools are used for the annotation of the Verbmobil corpora, consisting of transcribed German (Stegmann et al., 1998), English, and Japanese dialogues. Furthermore, trained versions for the Penn Treebank (Marcus et al., 1993) and the Susanne corpus (Sampson, 1995) exist.

To sum up, our annotation tools facilitate comfortable, customizable and fast semi-automatic part-of-speech and structural syntactic annotation. The tools run under Solaris and Linux and are freely available to universities and similar institutions for research purposes⁶.

⁶cf.

<http://www.coli.uni-sb.de/sfb378/>

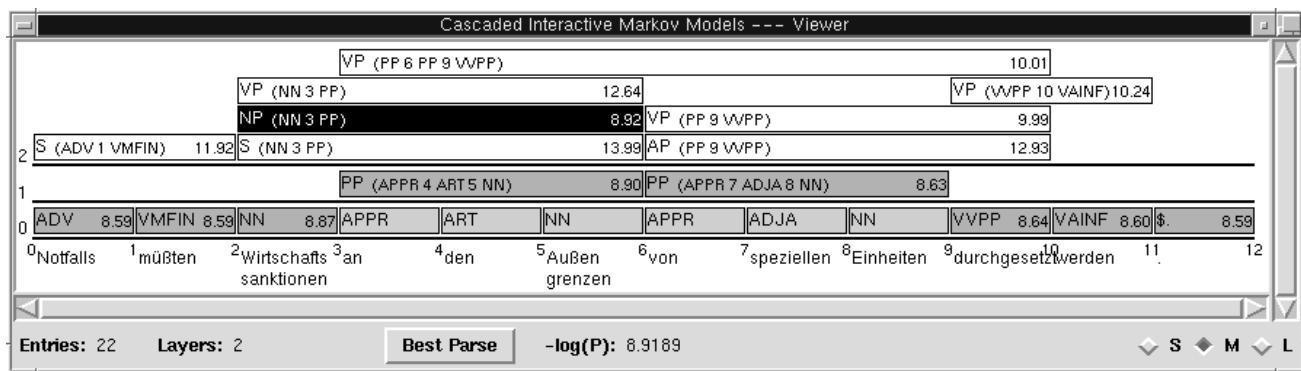


Figure 9: Third hypothesis.

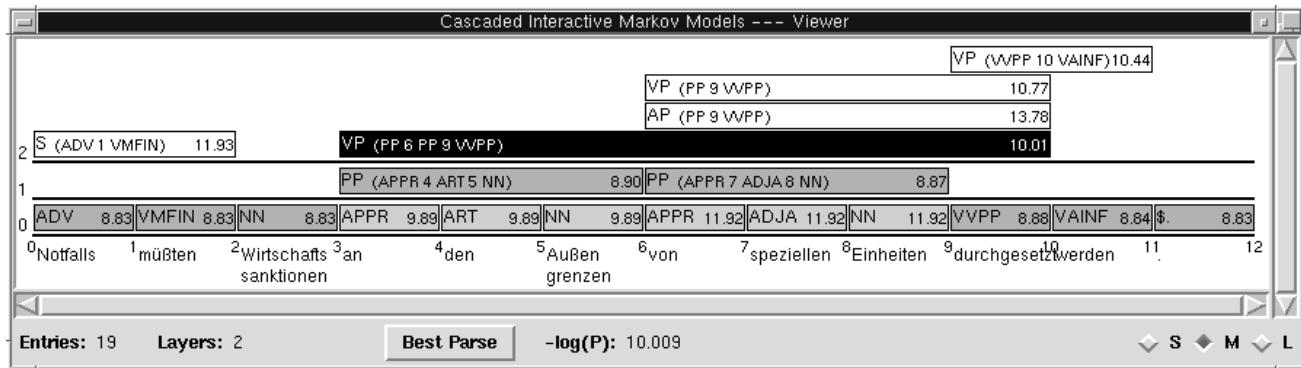


Figure 10: Fourth hypothesis.

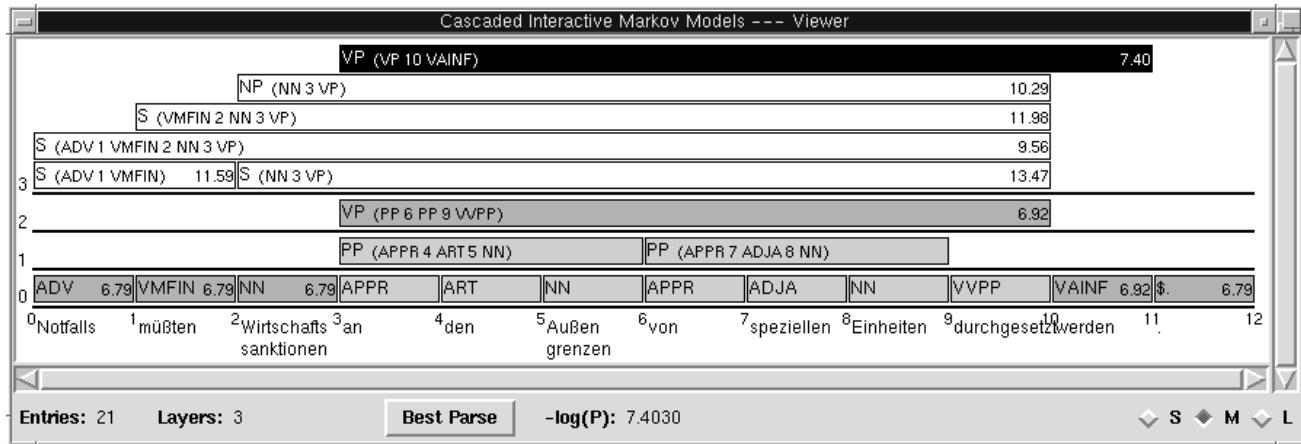


Figure 11: Fifth hypothesis.

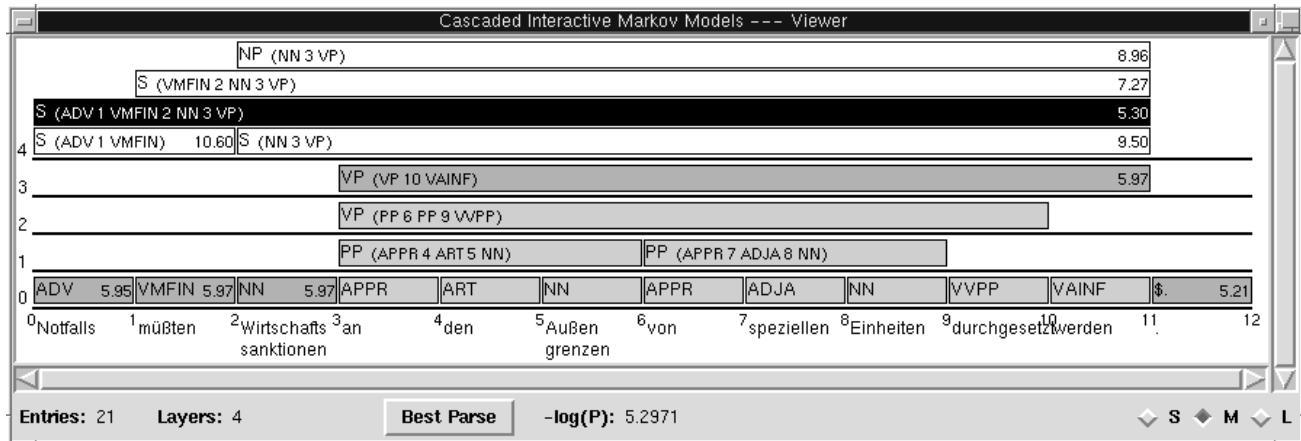


Figure 12: Sixth and final hypothesis.

Acknowledgements

The work on the annotation tools was carried out in the DFG Sonderforschungsbereich 378 *Resource-Adaptive Cognitive Processes*, project C3 *Concurrent Grammar Processing*, and is now continued in the DFG Project TIGER *Linguistic Annotation of a German Corpus*.

We would like to thank the numerous annotators, who have used and are using these tools, for their work and for valuable suggestions for improvements.

7. References

- Brants, Thorsten, 1999. Cascaded Markov models. In *Proceedings of 9th Conference of the European Chapter of the Association for Computational Linguistics EACL-99*. Bergen, Norway.
- Brants, Thorsten, 2000. TnT – a statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing ANLP-2000*. Seattle, WA.
- Brants, Thorsten, Wojciech Skut, and Brigitte Krenn, 1997. Tagging grammatical functions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing EMNLP-97*. Providence, RI, USA.
- Marcus, Mitchell, Beatrice Santorini, and Mary Ann Marcinkiewicz, 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- Sampson, Geoffrey, 1995. *English for the Computer*. Oxford: Oxford University Press.
- Skut, Wojciech, 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.
- Stegmann, Rosemary, Heike Schulz, and Erhard Hinrichs, 1998. Stylebook for the German treebank in VerbMobil. VerbMobil report, Seminar für Sprachwissenschaft, Universität Tübingen, Germany.