TTS – A Treebank Tool Suite

Aoife Cahill*, Josef van Genabith*

* Dublin City University
Glasnevin, Dublin 9, Ireland
{acahill, josef}@compapp.dcu.ie

Abstract

Treebanks are important resources in descriptive, theoretical and computational linguistic research, development and teaching. This paper presents a treebank tool suite (TTS) for and derived from the Penn-II treebank resource (Marcus et al, 1993). The tools include treebank inspection and viewing options which support search for CF-PSG rule tokens extracted from the treebank, graphical display of complete trees containing the rule instance, display of subtrees rooted by the rule instance and display of the yield of the subtree (with or without context). The search can be further restricted by constraining the yield to contain particular strings. Rules can be ordered by frequency and the user can set frequency thresholds. To process new text, the tool suite provides a PCFG chart parser (based on the CYK algorithm) operating on CFG grammars extracted from the treebank following the method of (Charniak, 1996) as well as a HMM bi-/trigram tagger trained on the tagged version of the treebank resource. The system is implemented in Java and Perl. We employ the InterArbora module based on the Thistle display engine (LTG, 2001) as our tree grapher.

1. Introduction

Treebanks are parse annotated text corpora. They are important resources in descriptive, theoretical and computational linguistic research, development and teaching. Treebanks are used as data repositories in syntactic research (Sampson, 2001) and grammar extraction, training and ("gold-standard") evaluation resources in statistical parsing and tagging (Charniak, 1993).

More recently, treebanks are used to (semi-) automatically generate "higher-level" syntactic and semantic representations such as e.g. dependency structures (Collins, 1996) and, in our own research, Lexical-Functional Grammar (LFG) (Kaplan and Bresnan, 1982) f(unctional) structures (attribute-value structures) (Frank, 2000), (Sadler et al, 2000), (Cahill et al, 2002).

In our research (Sadler et al, 2000), (Cahill et al, 2002) we automatically annotate treebank trees with LFG feature-structure information in terms of annotation principles and/or annotation algorithms. The annotation principles/algorithms work on local (sub-) tree configurations of depth one, i.e. CFG rules (see (Frank, 2000) for a more general approach that involves tree fragment rewriting).

Treebank trees tend to involve a large number of CFG rules. The Penn-II treebank (Marcus et al, 1993) features more than 19K distinct CFG rule types extracted from a corpus of about 1000K words of parse-annotated text from the Wall Street Journal. In our automatic annotation work, we derive annotation principles based on the most frequent rule types such that for each syntactic category (S,NP,VP,..) the token occurrences of the rule types expanding the category cover more than 80% of the overall token occurrences of the rules. The annotation principles are applied automatically to the treebank trees and thereby also cover less frequent local trees (CFG rules) and allow us to generate good quality, higher level f-structures automatically from the treebank trees (for more details see (Cahill et al, 2002)).

In order to support the development of the annotation principles and algorithms we need a treebank tool that, for each CFG rule type occurrence in the treebank trees, allows inspection of the trees containing the rule, the local trees rooted by the rule and the terminal yields (with or without left and right context) of the rule.

In this paper we present a treebank tool suite (TTS) for and derived from the Penn-II treebank resource (Marcus et al, 1993). The tools include treebank inspection and viewing options which support search for CF-PSG rule tokens in the treebank, graphical display of complete trees containing the rule instance, display of the subtree rooted by the rule instance and display of the yield of the subtree (both with or without context).). The search can be further restricted by constraining the yield to contain particular strings. Rules can be ordered by frequency and the user can set frequency thresholds.

To process new text, the tool suite provides a PCFG chart parser (based on the CYK algorithm) operating on CFG grammars extracted from various versions of the treebank (e.g. with and without automatically annotated f-structure information) following the method of (Charniak, 1996) as well as a HMM bi-/trigram tagger trained on the tagged version of the treebank resource.

It turns out that the TTS suite developed in our automatic annotation project can also be used for

- treebank development (either manually or semi-automatically using the tagger and PCFG parser followed by a manual inspection and correction phase);
- treebank inspection as a graphical interface to the Penn-II resource for teaching and research;
- as a grammar development tool for inspection, profiling and proofing of the coverage and application of large scale CFG grammars and their output on a corpus.

Below, we first give an overview of the functionality of the TTS treebank inspection tool. The tool supports search and display of treebank information based on CFG rule and terminal yields. We then describe the HMM based tagger and PCFG parser components in the TTS system. Finally, we discuss further work and conclude.

2. CFG Rule Based Treebank Search and Display

Given a treebank resource such as Penn-II, it is straightforward to extract the CFG grammar rules embodied in the treebank trees (Charniak, 1996). For the 1000K word Wall Street Journal section more than 19K rule types are extracted.

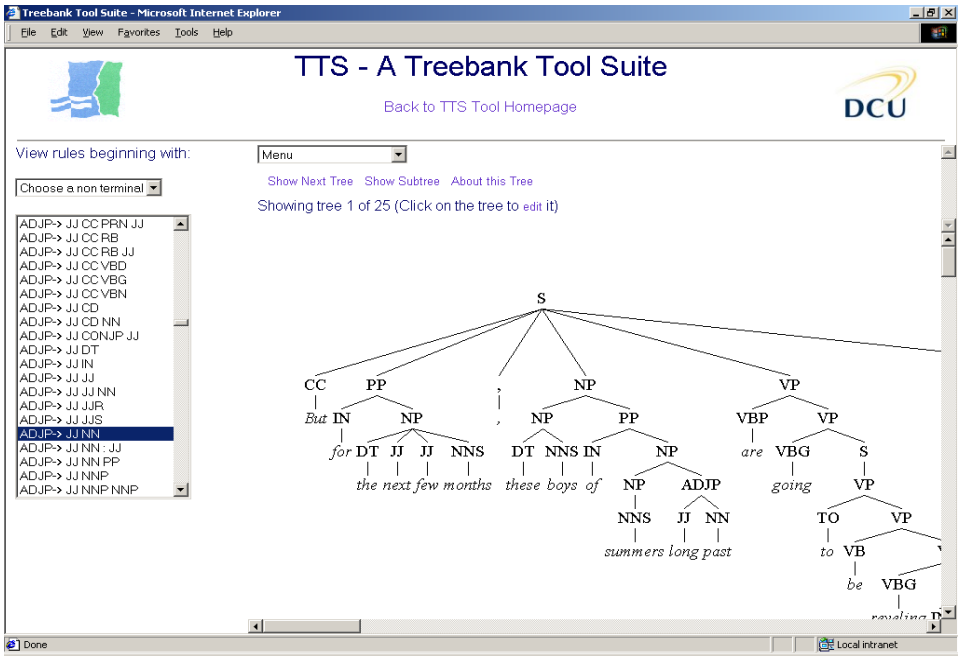
Given a particular CFG rule type, we would like to be able to retrieve and view

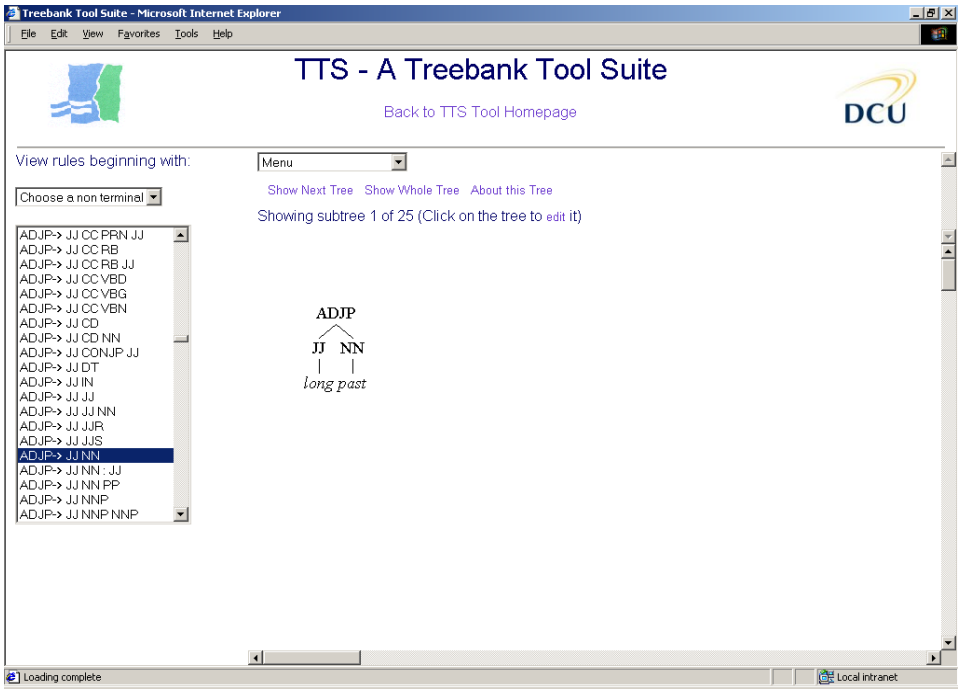
- trees containing instances of the rule,
- subtrees rooted by the rule,
- the terminal yields of the rule (the strings covered by the rule) with our without left/right context

in the treebank.

In order to achieve this, we preprocess the treebank to index extracted rule types with rule token occurrences in treebank trees. This allows efficient retrieval during runtime, an important end user requirement.

The user selects a rule and a presentation format (trees, sub-trees or yields with or without context). If there are multiple solutions, for efficiency reasons, the TTS tool will retrieve and present such in consecutive sets of 50.





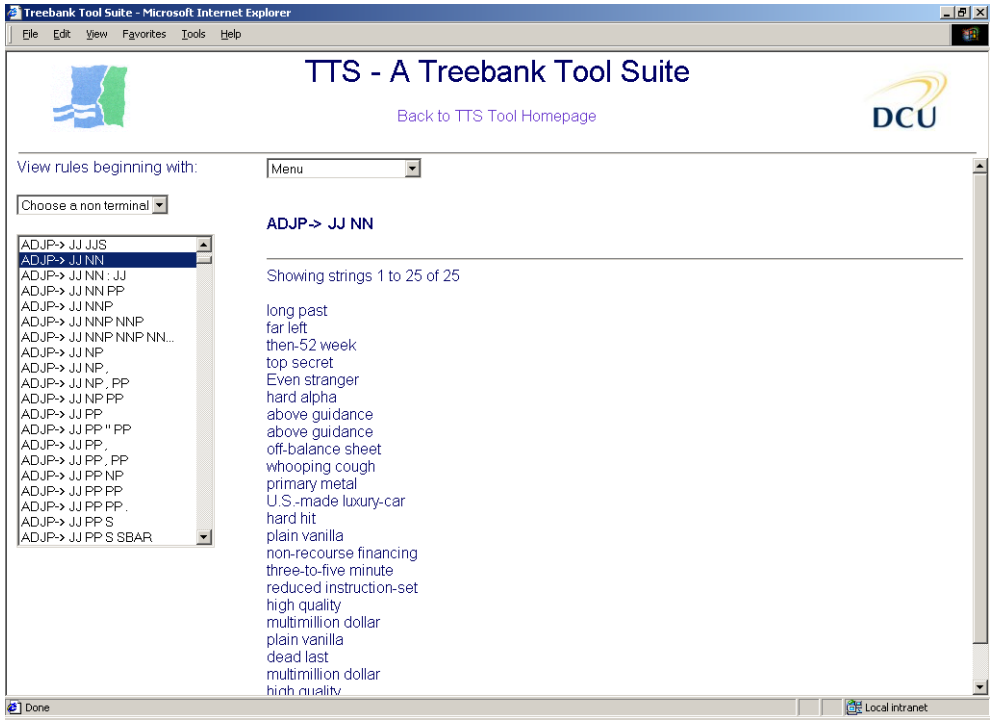


Figure 3. TTS displaying the yield of the rule ADJP-> JJ NN

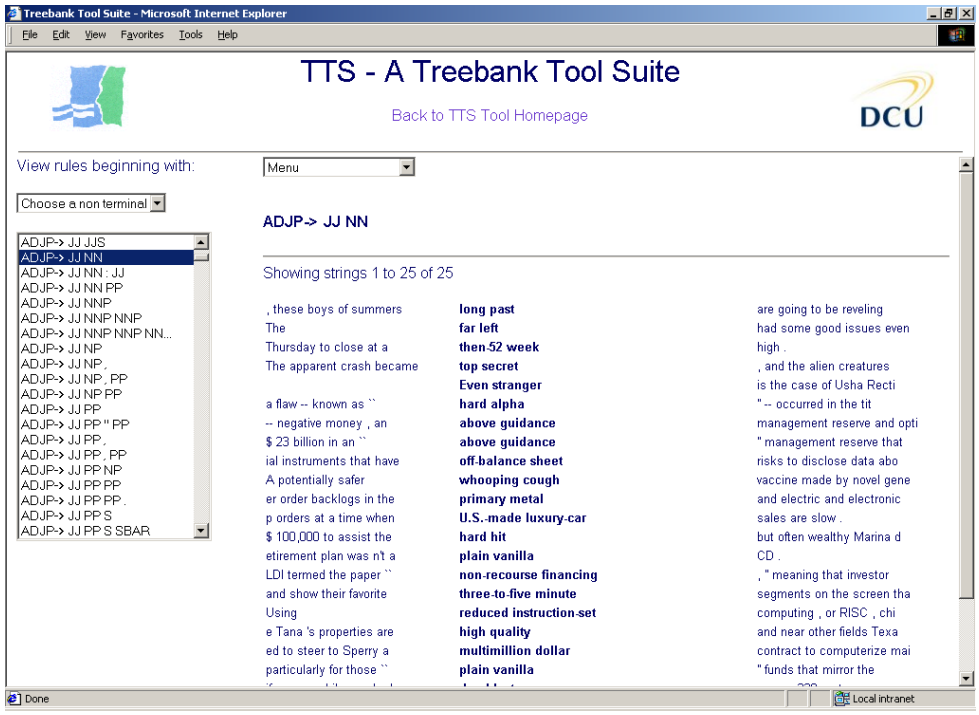


Figure 4. TTS displaying the yield(in context) of the rule ADJP-> JJ NN

Certain CFG rules occur more than several thousand times in the corpus and exhaustive retrieval of all solutions before presenting the first 50 solutions would be time consuming and involve unacceptable response times.

The yield with/without context option effectively turns the TTS tool into a "yield in context" (YIC - KWIC) application. In contrast to e.g. Richard Pito's `tgrep`, the TTS tool does not support arbitrary tree fragment (of variable depth) searches. However, we can often approximate such searches by using a TTS search option involving a CFG rule together with a terminal yield constraint. In order to select trees exhibiting subject control constructions, e.g., a user can specify a VP -> VBD NP S rule and require that `'promise|promises|promised|promising'` be an element of the yield of the rule

This search query generates the following response:

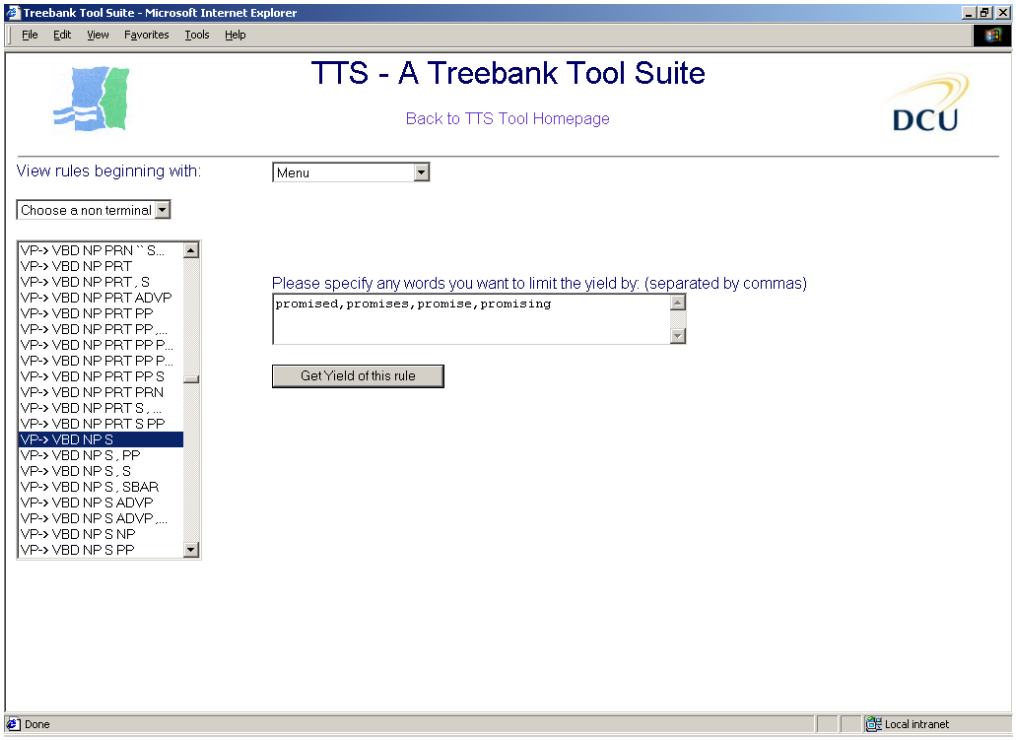


Figure 5. TTS limiting the rule VP→ VBD NP S

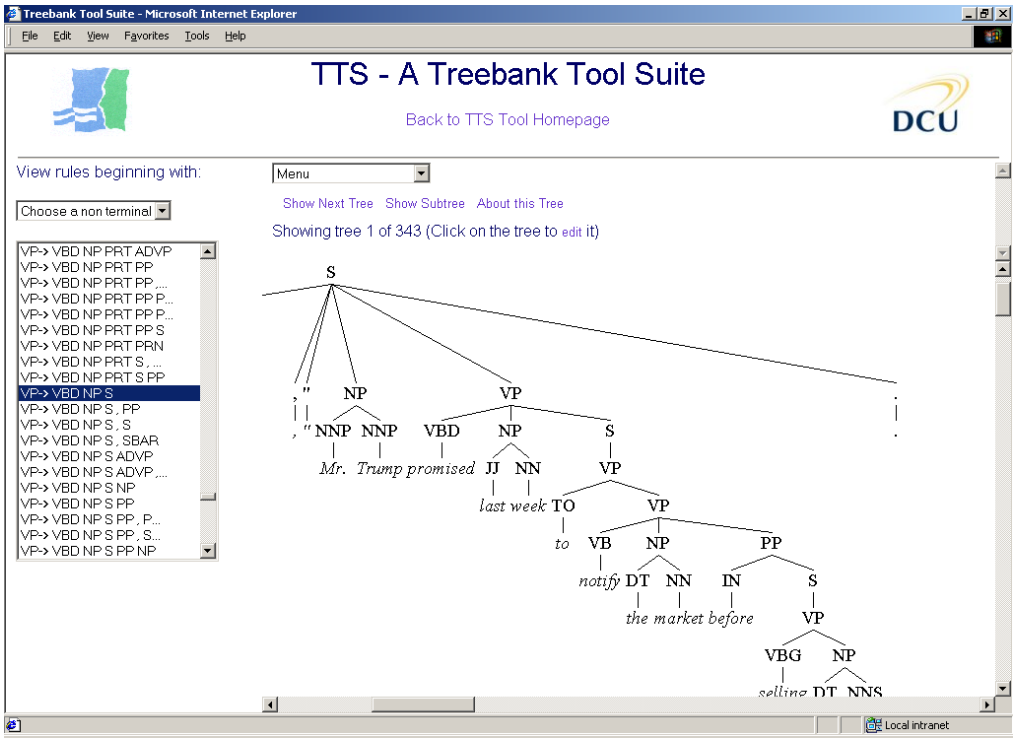


Figure 6. TTS displaying the trees of the limited yield of the rule VP→ VBD NP S

CFG Rules can be ordered by frequency and the user can set frequency thresholds. In our current implementation the CFG rule search and tree display option works with a number of versions of the Penn-II treebank and extracted grammars.

The first is the original version of the Penn-II treebank (with the functional annotations provided by the original annotators).

The second is a version of the Penn-II treebank pre-processed for PCFG (probabilistic context free grammar) parsing (Charniak, 1996). In this version we strip all functional annotations provided by the treebank annotators (PP-TMP becomes PP etc.) and keep only the monadic CFG categories. In addition, we eliminate unary, empty (epsilon) productions and trees containing FRAG constituents. The CFG rule set extracted is of size 19K.

The third version is pre-processed for PCFG parsing similar to above but this time we preserve the functional annotations provided by the original treebank annotators. The CFG rule set extracted is of size 29K.

The fourth version is identical to the third version but this time (in addition to the functional annotations present in the original treebank) carries all the f-structure (attribute-value structure) annotations generated by our automatic annotation algorithm (Cahill et al, 2002). The CFG rule set extracted is of size 37K.

3. Treebank Based HMM Tagger and PCFG parser.

In addition to providing an interface to existing or pre-processed versions of the Penn-II treebank, TTS provides tools for analysing new text. These tools include a HMM tagger and a PCFG parser, both trained on the Penn-II treebank resource.

The HMM tagger (Charniak, 1993) is available as both a bigram and a trigram tagger. For a sequence of words, the bigram tagger maximises the probability of the tag sequence:

$$P(t_i,...t_n \mid w_i,...w_n) = \prod_{i=1}^{n} P(t_i \mid w_{i-1}, t_{i-1}) P(w_i \mid t_i)$$

The PCFG parser is based on the CYK algorithm (Collins, 1999) and returns the best parse. Rule probabilities are calculated for the pre-processed treebank resource using simple maximum likelihood estimators. The probability of a rule is simply the number of times the rule occurs in the corpus divided by the number of occurrences of all rules expanding the same left hand side.

$$P(LHS \longrightarrow RHS_j) = \frac{\#(LHS \longrightarrow RHS_j)}{\sum_{i=1}^{n} \#(LHS \longrightarrow RHS_i)}$$

Using CFG independence assumptions, the probability of a tree is the product of the probabilities of the rules in the tree.

$$P(TREE) = \prod_{i=1}^{n} P(Rule_i)$$

Both the tagger and the parser can be used in stand alone mode (you can tag text and you can parse already tagged text) or interfaced (the tagger is used to tag raw text and the tagged version is then parsed by the parser). The tagger and the parser can be used in semi-automatic corpus/treebank development with manual correction of the output.

4. Conclusions and Further Work

The TTS tool suite is designed to provide a query and display interface to (variants of) the Penn-II treebank resource or to resources generated with the tagger and PCFG parser provided. CFG rules can be clicked and the desired display (containing trees, subtrees rooted by rule, yield, yield in context) is generated. Search can be refined by constraining the rule terminal yield to contain particular strings. CFG Rules can be ordered by frequency and the user can set frequency thresholds. The tool set provides an HMM tagger and a probabilistic parser to process new text. The HMM tagger is available in both bi- and trigram versions. Bi- and trigram transitions as well as

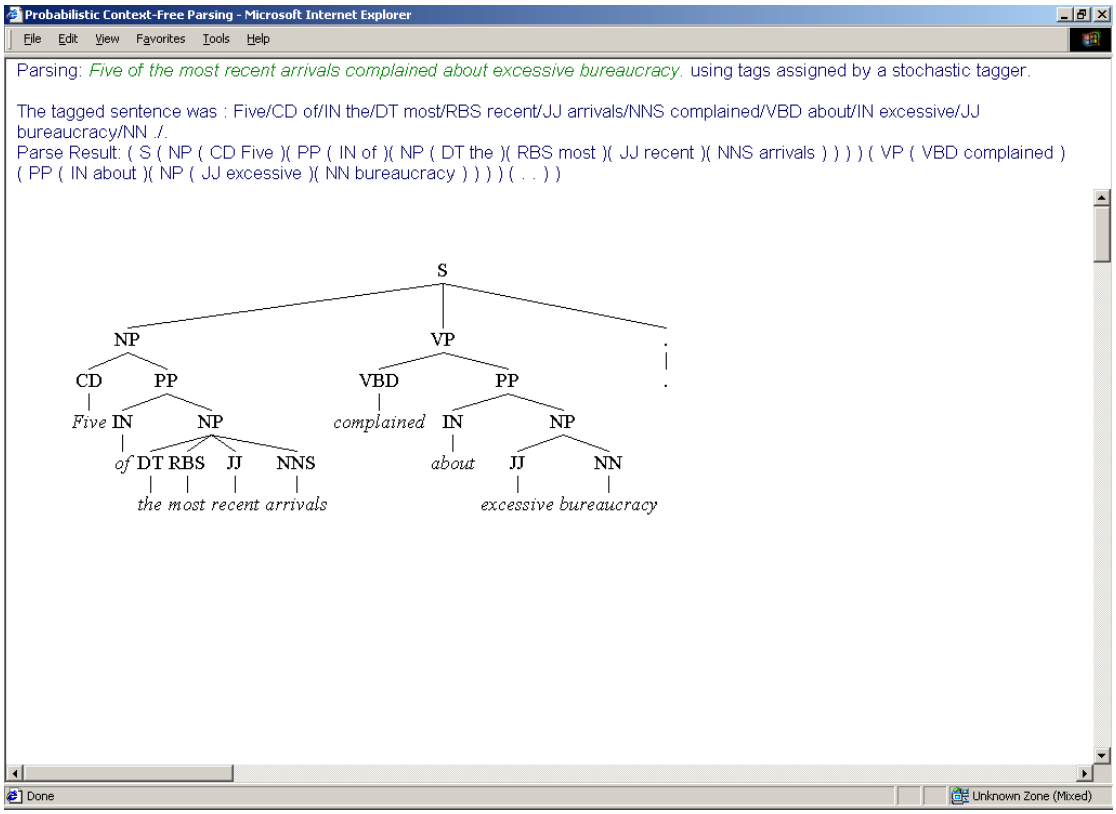


Figure 7. TTS displaying the results of a parse of new text

lexical probabilities are computed from the tagged version of the Penn-II resource. The probabilistic CFG chart parser is based on the CYK algorithm and operates on CFG rule sets and lexical entries extracted from the Penn-II treebank.

TTS can support quality control in treebank and computational grammar development. A treebanker or grammar developer can query CF-PSG rules and the system retrieves and displays complete trees which contain instances of the rule from manually annotated text in the treebank or from the output generated by the tagger and the PCFG parser. Alternatively, the system is able to display local subtrees (including their yield) headed by rules queried. As a third possibility, the system can display terminal strings (the yield of rules queried) with or without context and thus provides some of the functionality of a basic concordancing tool triggered by CF-PSG rule rather than string query. This enables a treebanker or grammar developer to proof, profile and control the coverage and application of CFG rules in a parse annotated corpus. TTS can be used in a data based teaching scenario to give students access to graphical representations of parse annotated text. Students can explore rule types and search the parse annotated corpus for rule token applications.

A web-based demo version of the CF-PSG rule query and display part of the TTS system can be accessed with Java enabled browsers at

http://www.compapp.dcu.ie/~acahill/ica/

The system interface, indexing, retrieval, tagger and parser components are implemented in Java and Perl. To display retrieved trees we use the InterArbora module based on the Thistle display engine provided by the Language Technology Group in Edinburgh (LTG, 2001).

In future work we hope to be able to port the TTS system to other treebanks such as the SUSANNE corpus (Sampson, 1995) and to make the system available as freeware to interested users.

5. References

Cahill, A., McCarthy, M., van Genabith, J., and Way, A. 2002. Automatic Annotation of the Penn II Treebank with LFG F-Structure Information. In LREC 2002 Workshop on Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Language Data, Las Palmas, Canary Islands, Spain, June 1st 2002.

Charniak,E. 1993. Statistical Language Learning, MIT Press

Charniak,E. 1996. Treebank Grammars. Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96), MIT Press, pp.1031-1036

Collins, M. 1996 A New Statistical Parser Based on Bigram Lexical Dependencies 34th Annual Meeting of the Association of Computational Linguistics, Santa Cruz, CA, pp184-191.

Collins, M. 1999 Head-driven Statistical Models for Natural Language Parsing. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Frank,A. 2000. Automatic F-Structure Annotation of Treebank Trees, LFG-2000, 19 July - 20 July 2000, CSLI Publications, Stanford, CA, http://www-csli.stanford.edu/publications/

Kaplan R.M, Bresnan, J. 1982. Lexical Functional Grammar. The Mental Representation of Grammatical Relations, MIT Press, pp.173-281

The Language Technology Group, 2001. 2, Buccleuch Place, Edinburgh EH8 9LW, UK. http://www.ltg.ed.ac.uk/index.html (20/11/01)

Marcus,M., Santorini,B and Marcinkiewicz, M.A. 1993 . Building a Large Annotated Corpus of English: The Penn Treebank. Computational Linguistics, 19(2); pp.313-330

Sadler,L., van Genabith, J. and Way, A. 2000. Automatic F-Structure Annotation from the AP Treebank; LFG-2000, 19 July – 20 July 2000, CSLI Publications, Stanford,CA, http://www-csli.stanford.edu/publications/

Sampson, G. 1995. English for the Computer: The SUSANNE Corpus and Analytic Scheme, Clarendon Press, Oxford.

Sampson, G. 2001. Empirical Linguistics, Continuum International, London and New York.