

## Chapter 18

# DEPENDENCY-BASED EVALUATION OF MINIPAR

Dekang Lin

*Department of Computing Science*

*University of Alberta*

*Edmonton, Alberta, Canada T6G 2H1*

lindek@cs.ualberta.ca

**Abstract** In this paper, we first present a dependency-based method for parser evaluation. We then use the method to evaluate a broad-coverage parser, called MINIPAR, with the SUSANNE corpus. The method allows us to evaluate not only the overall performance of the parser, but also its performance with respect to different grammatical relationships and phenomena. The evaluation results show that MINIPAR is able to cover about 79% of the dependency relationships in the SUSANNE corpus with about 89% precision.

**Keywords:** English, Parser Evaluation, Dependency, Susanne Corpus

## 1. INTRODUCTION

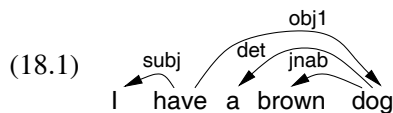
With the emergence of broad-coverage parsers, quantitative evaluation of parsers becomes increasingly important. It is generally accepted that quantitative parser evaluation should be conducted by comparing the parser-generated parse trees (we call them **answers**) with manually constructed parse trees (we call them **keys**). A commonly used method (Black et al., 1991) is to compare the phrase boundaries between answers and keys. (Lin, 1995; Lin, 1998) argued that there are many problems with this approach. For example, a single attachment error may lead to multiple incorrect phrase boundaries. More seriously, nonsensical parses may get rather high phrase-boundary scores. We proposed a dependency-based evaluation scheme in which both the answers and keys were converted into dependency trees and the evaluation scores were computed according to the set of dependency relationships in the answers and the keys.

In this paper, we present the results of using such a scheme to evaluate MINIPAR, which is a descendent of PRINCIPAR (Lin, 1993; Lin, 1994). MINIPAR adopted some of the ideas in the Minimalist Program (Chomsky, 1995), such as bare phrase structure and economy principles.

## 2. DEPENDENCY-BASED PARSER EVALUATION

A dependency relationship (Hays, 1964; Hudson, 1984; Mel'čuk, 1987; Bömová et al., this volume) is an asymmetric binary relationship between a word called **head** (or governor, parent), and another word called **modifier** (or dependent, daughter). Dependency grammars represent sentence structures as a set of dependency relationships. Normally the dependency relationships form a tree that connects all the words in a sentence. A word in the sentence may have several modifiers, but each word may modify at most one word. The root of the dependency tree does not modify any word. It is also called the head of the sentence.

For example, (18.1) is a dependency structure of a sentence. The head of the sentence is “have”. There are four pairs of dependency relationships, depicted by four arcs from heads to modifiers.



We use a list of tuples to represent a dependency tree. Each tuple represents a node in the dependency tree and has the following format

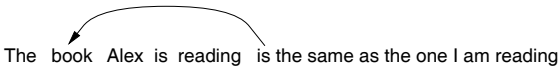
(word category [head] [relationship])

where:

- word is the word represented by the node,
- category is the lexical category of word,
- head specifies which word is modified by word, and
- relationship is a label assigned to the dependency relationship, for example, subj (subject), adjn (adjunct), cmlpl (complement), spec (specifier), *etc.*

The fields `head` and `relationship` are optional. The `head`-field consists of a position indicator followed by a word. The position indicator takes one of the following values: {<, >, <<, >>, <<<, ...}. If the `head`-field is “<...<<sub>n</sub> w”, the head of word is the *n*th occurrence of *w* to the right of the modifier. The meaning of “>...><sub>n</sub> w” is similarly defined in the opposite direction.

For example, in the following sentence:

(18.2)  The book Alex is reading is the same as the one I am reading

the word “book” modifies the second “is” to its right. Its head-field in the dependency tree should be: “<< is”.

The dependency tree in (18.1) is represented by the following tuples:

(18.3)

Modifier	Category	Head	Type
I	N	< have	subj
have	V		
a	Det	< dog	spec
brown	Adj	< dog	adjn
dog	N	> have	comp

## 2.1 Evaluation Measures and Procedure

Once the answer and the key are both represented as dependency trees, we can compare them on a word-by-word basis and calculate two scores:

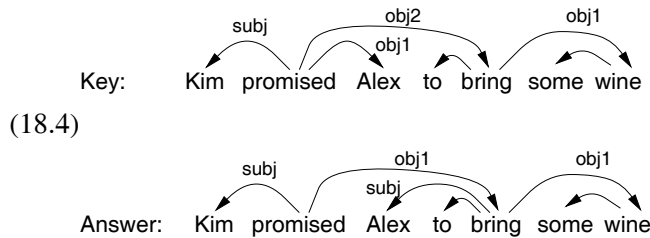
**recall:** the percentage of dependency relationships in the key that are also found in the answer.

**precision:** the percentage of dependency relationships in the answer that are also found in the key.

Table 18.1 shows the dependency-based evaluation result for the key-answer pair in (18.4). Except for the word “Alex”, all the other words are assigned the same head in the answer as in the key. Therefore, both the precision and recall of the answer are 5/6.

Table 18.1. Dependency-based evaluation of the structures in (18.4)

word	Key			word	Answer			correct
	cate	head	rel		cate	head	rel	
Kim	N	< promised	subj	Kim	N	< promised	subj	yes
promised	V			promised	V			
Alex	N	> promised	obj1	Alex	N	< bring	subj	no
to	Aux	< bring	aux	to	Aux	< bring	aux	yes
bring	V	> promised	obj2	bring	V	> promised	obj1	yes
some	Det	< wine	spec	some	Det	< wine	spec	yes
wine	N	> bring	obj1	wine	N	> bring	obj1	yes



The above evaluation ignores the labels assigned to the dependency relationships. This is analogous to ignoring constituency labels in phrase boundary based evaluations (Black et al., 1991). If both the answer and the key use the same set of dependency labels or a mapping between equivalent labels can be established, labels can be incorporated into the evaluation algorithm by requiring not only that a word modify the same word in the answer as in the key, but also that the dependency relationships have the same or equivalent labels. For example, if the labels were taken into account in Table 18.1, the precision and recall of the answer would be 4/6.

When both the answer and the key are full parses, each of them have  $N - 1$  dependencies, where  $N$  is the number of words in the sentence. That means that the precision and recall value will be the same. Therefore, in our earlier proposal (Lin, 1995), the ratio between the number of correct dependencies and the number of words was adopted as the evaluation metric. However, a number of situations exist in which the number of dependency relationships in the answer may not be the same as in the key:

- Due to differences in tokenization, the answer and key may be segmented into different numbers of words. For example, under one scheme, a hyphenated word is treated as a single token. In another, the hyphens and the word they separate are regarded as individual tokens.
- When we selectively evaluate different aspects of a parser, only a subset of the dependencies participates in the evaluation.
- Although it is safe to assume that the key is always a fully connected dependency tree, the answer may consist of several fragments of dependency trees.

The precision and recall measures are more flexible under these circumstances.

### 3. EVALUATION OF MINIPAR WITH SUSANNE CORPUS

#### 3.1 MINIPAR

MINIPAR is a principle-based English parser (Berwick et al., 1991). Like PRINCIPAR (Lin, 1993), MINIPAR represents the grammar as a network.

where the nodes represent grammatical categories and the links represent types of syntactic (dependency) relationships. The grammar network consists of 35 nodes and 59 links. Additional nodes and links are created dynamically to represent subcategories of verbs. MINIPAR employs a message passing algorithm that essentially implements distributed chart parsing. Instead of maintaining a single chart, each node in the grammar network maintains a chart containing partially built structures belonging to the grammatical category represented by the node. The grammatical principles are implemented as constraints associated with the nodes and links.

The lexicon in MINIPAR is derived from WordNet (Miller et al., 1990). With additional proper names, the lexicon contains about 130K entries (in base forms). The lexical entry of a word lists all possible parts of speech of the word and its subcategorization frames (if any). The lexical ambiguities are handled by the parser instead of a tagger. Like chart parsers, MINIPAR constructs all possible parses of an input sentence. However, it outputs a single parse tree with the highest ranking. Although the grammar is manually constructed, the selection of the best parse tree is guided by the statistical information obtained by parsing a 1GB corpus with MINIPAR.

## 3.2 SUSANNE

The SUSANNE corpus (Sampson, 1995; Sampson, this volume) is a subset of the Brown Corpus of American English. It contains parse trees of 64 of the 500 texts in the Brown Corpus of American English. The texts are evenly distributed over the following four Brown genre categories:

- A press reportage;
- G belles lettres, biography, memoirs, etc.;
- J “learned” (technical and scholarly prose);
- N adventure and Western fiction.

While the SUSANNE corpus is not large enough for the purpose of statistical training, it is much larger and much more diverse than the testing corpora used in previous evaluation of broad coverage parsers (e.g., Magerman, 1995; Collins, 1996; Collins, 1997). Compared with other possible alternatives such as the PennTreebank (Marcus et al., 1993), the SUSANNE corpus offers several advantages:

- The PennTreebank is made up solely of Wall Street Journal sentences. The SUSANNE corpus contains a wide variety of texts, including news articles, letters and biographies, scientific and technical writing, and fiction.
- The SUSANNE corpus puts more emphasis on precision and consistency, which are more important properties than quantity for parser evaluation purposes.

SUSANNE corpus, as well as the Penn TreeBank II, not only labels the categories of phrases, but also the functional roles, such as subjects, direct/indirect objects, and temporal adjuncts. For example, if a phrase is labeled  $Ns:s$ , it means that the phrase belongs to the category  $Ns$  and its role is the logical subject. This information can be used to determine how well a parser recognizes predicate-argument relationships, or how good it is at prepositional phrase attachment.

### 3.3 Evaluation Results

Table 18.2 summarizes the results of evaluating MINIPAR with the SUSANNE corpus. The second and the third column in Table 18.2 are the number of sentences in each category and their average lengths. Since each category contains roughly the same number of words, there are fewer sentences in categories with longer sentences. The precision, recall and the error rate are calculated by summing up the total number of errors and the total number of dependency relationships in the corpus or a subcorpus, rather than by averaging the measures for individual sentences.

Table 18.2. Results of evaluating MINIPAR

Category	no. of sentences	avg. length	precision	recall	avg. words per second*
A	1614	22.12	88.60	81.81	292
G	1525	23.74	88.76	77.15	335
J	1398	25.27	88.28	79.95	339
N	2566	13.92	88.54	75.29	319
All categories	7103	20.15	88.54	78.58	320

\*on a Pentium II 300 with 128 MB memory.

In the development of MINIPAR, as well as PRINCIPAR, we only used news articles and example sentences from linguistics textbooks to test and debug the parser and the grammar. It is not surprising that MINIPAR performed visibly better on category A than on the others. It is also worth noting that the greatest differences in performance were in recall, while the precision remains almost constant across different categories. One possible explanation for this is that the lower recall values of categories G, J and N (compared with Category A) are due to less complete coverage of linguistic phenomena in those categories. However, when the structure of a sentence is indeed covered by the parser, the parser has similar likelihood of generating correct dependency relationships as in category A.

Another interesting observation is that even though the sentences in category N (fiction) are much shorter than the sentences in other categories, MINIPAR's

recall in this category is lower than in other categories. This seems to contradict the intuition that shorter sentences are easier to parse.

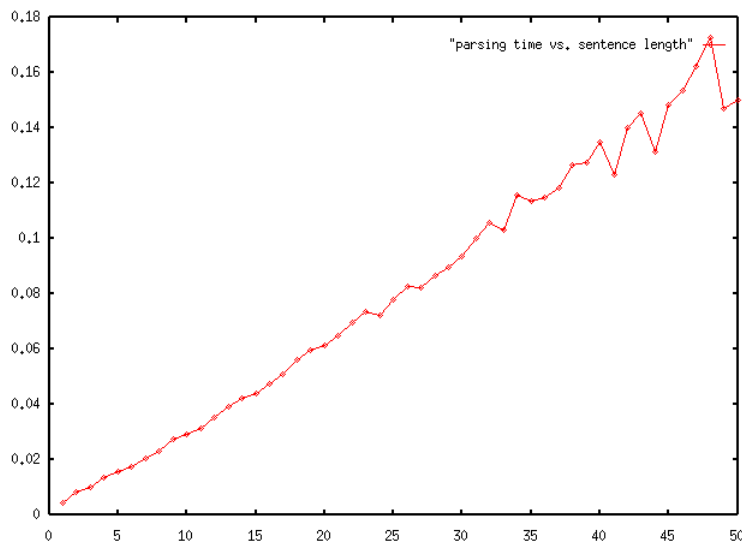


Figure 18.1. Sentence length vs. parsing time (in seconds)

The timing results in Table 18.2 shows that MINIPAR is very efficient. It is able to parse about 300 words per second on a Pentium II 300 with 128M memory. In contrast, Magerman's statistical parser (Magerman, 1995) processes about 1.86 words per second on a SGI R4400 with 160M memory. His result is obtained by limiting sentence length to 40 words, whereas we parsed all the sentences in the SUSANNE corpus, in which the longest sentence contains 215 words.

Figure 18.1 plots the average parsing time for sentences with a given length (from 1 to 50). Surprisingly, the parsing time looks linear to the number of words in the sentence. The linear parsing time can also be observed in Table 18.2.

#### 4. SELECTIVE EVALUATION

An advantage of the dependency-based evaluation method is that it allows us to evaluate the parser performance with respect to specific types of dependency relationships, such as predicate-argument, or prepositional attachment. This can be achieved by comparing a subset of the words in the answer and the key, instead of all the words.

Assuming that the dependency relationships are labeled, we can use the labels to select a subset of the dependency relationships. Consider the example

in Table 18.1. The labels “obj1” and “obj2” are assigned to the dependency relationships between verbs and their first and second objects. If we only include a word in the evaluation if the dependency between the word and its parent is labeled “obj1” or “obj2”, the precision and recall values will tell us how the parser performed on verb-object relationships.

(18.5)

Key			Answer		
head	modifier	label	head	modifier	label
promised	Alex	obj1			
promised	bring	obj2	promised	bring	obj1
bring	wine	obj1	bring	wine	obj1

For example, by restricting the dependency relationships to those with “obj1” or “obj2” label, only the dependency relationships in (18.5) are selected from the dependency relationships in Table 18.1. Therefore, the recall and precision of the verb-object dependency in Table 18.1 are 66.7% and 100% respectively. If a dependency in the answer is only considered to be correct if it has the same label as the head, the recall will be 33.3% and the precision will be 50%. Similarly, if we score the subject relationships in Table 18.1, the recall is 100%, but the precision is 50%.

Table 18.3 summarizes the evaluation results of different types of dependency relationships. The second and third column in Table 18.3 contain the total numbers of dependency relationships belonging to the given type in the MINIPAR outputs and in the SUSANNE corpus respectively.

Table 18.3. Evaluation of different types of dependency relations

Dependency Type	Total in Answer	Total in Key	Precision	Recall
Subject	10328	11917	88.7%	78.4%
Complement	8557	7031	87.9%	71.8%
PP attachment	15629	16344	77.9%	72.2%
Relative Clause	1588	1135	51.8%	55.8%
Conjunction	3404	4810	67.1%	49.6%

Table 18.3 shows that the dependency relationships that involve systematic attachment ambiguities, such as PP attachments, relative clauses and conjunctions have much lower precision and recall than predicate-argument relationships. This is because MINIPAR resolves attachment ambiguities by “Minimal Attachment” and “Right Association.” However, many of the ambiguities can only be resolved on semantic grounds.

The selective evaluation can be carried out further by considering a parser’s performance with respect to a word. For example, Table 18.4 shows evaluation scores for the ten most frequent prepositions.



Table 18.4. Prepositional attachments

Prep	Total in Answer	Total in Key	Precision	Recall
of	4227	4636	95.2%	91.8%
in	2533	2684	70.5%	60.7%
to	1025	1353	73.0%	69.3%
for	930	1059	68.7%	64.4%
with	892	824	74.6%	66.7%
on	745	662	68.1%	60.1%
by	746	697	80.3%	71.8%
at	610	561	67.0%	58.7%
from	597	532	77.8%	69.5%
as	332	362	53.9%	58.1%

While most of the prepositions have similar precision and recall, the score for “of” is much better than others and the score for “as” is significantly lower than others. The errors involving “of” can be classified into the following categories:

- **Extrapolation:** MINIPAR does not handle extrapolation of prepositional phrases. For example, the head of “of” in (18.6) is “murder” in SUSANNE, but is “1935” in MINIPAR output.
 

(18.6) the murder in 1935 of her son-in-law
- **Quotation marks:** MINIPAR fails to parse a sentence when there is a quotation mark after a preposition. For example,
 

(18.7) In the area of “community health services” ...
- **Clausal complement:** MINIPAR also fails when the complement of a preposition is a wh-clause. For example,
 

(18.8) a body of legal principle which by and large was made up of what Western nations could do in the world arena
- **Conjunction:** MINIPAR generates a different tree than the dependency tree derived from SUSANNE analysis when a conjunction is modified by a prepositional phrase. An example is shown in Figure 18.2. Such differences are due to a difference in representation rather than genuine parser errors. We did not deliberately transform the SUSANNE analysis

of [NP and NP of NP] into a different dependency structure than MINIPAR outputs. It is simply a result of our translation rules that deals with conjunctions and prepositional phrase attachments. While it is not hard to modify the translation rules to reconcile such a difference, the large number of such differences makes it hard to fix all of them.



Figure 18.2. Different analysis of conjunctions in SUSANNE and MINIPAR.

Selective evaluation may also be carried out if the nodes in constituency trees are also labeled with grammatical roles. However, it may still be necessary to identify the head node in a constituent, so that a constituent in parser outputs may be considered to match a constituent in the key even if they have different spans. For example, the verb-object relation in

(18.9)

Ans: [S [NP-SUBJ I] [VP saw [NP-OBJ a man] with a telescope]]

Key: [S [NP-SUBJ I] [VP saw [NP-OBJ a man with a telescope]]]

should be considered to be correct even though the object NPs in the answer and key have different spans. By identifying the head in every constituent, we implicitly transform the constituency trees into dependency trees.

## 5. RELATED WORK

The main obstacle in parser evaluation is that different grammatical theories and different parsers may have different notions of what constitutes a correct parse. The best one can do is to compare a common factor, i.e., information that is encoded in all types of parse trees. In our approach, the common factor is the set of dependency relationships in the answer and the key. In the phrase boundary based evaluation methods the common factor is the set of phrase boundaries identified by parse trees.

The phrase boundaries in the answer and the key are treated as two sets (A and K respectively). The phrase boundary based evaluation computes the following measures:

**Recall:** The percentage of phrase boundaries in the key that are also found in the answer ( $\frac{|A \cap K|}{|K|}$ ).

**Precision:** The percentage of phrase boundaries in the answer that are also found in the key ( $\frac{|A \cap K|}{|A|}$ ).

Suppose (18.10a) is the key and (18.10b) is the answer. They each contain six phrases. Five of the phrases in the key are also found in the answer and five of the phrases in the answer are also found in the key. Therefore, the precision and recall are both 5/6.

(18.10)

- a. [Kim [promised [Alex] [to bring [some wine]]]]
- b. [Kim [promised [Alex [to bring [some wine]]]]]

Table 18.5. Dependency-based evaluation of structures in Figure 18.3

word	Key			word	Answer			correct
	cate	head	rel		cate	head	rel	
a	Det	< screen	det	a	Det	< bib	det	no
bib	N	< summary	nn	bib	N	< appeared	subj	no
summary	N	< screen	nn	summary	N	< screen	subj	yes*
screen	N	< appeared	subj	screen	V	> summary	rel	no
appeared	V			appeared	V			

\*assuming labels are ignored

Compared with the recall/precision of phrase boundaries, the recall/precision measures of dependency relationships are intuitively more meaningful. The phrase boundaries themselves are not used directly in semantic interpretation. Getting the phrase boundaries right is not even a necessary condition for semantic interpretation. Consequently, it is hard to predict how missing or spurious phrase boundaries affect semantic interpretation. On the other hand, since semantic dependencies are embedded in syntactic dependencies, the semantic interpretation process should be more sensitive to the percentage of correct syntactic dependencies than the percentage of correct phrase boundaries.

Consider the two parse trees “Answer” and “Key” in Figure 18.3. The answer treats the word “screen” as a verb and “summary screen” as a relative clause modifying the noun “BIB.” The answer is obviously a very poor parse of the sentence. It is unlikely that the sentence can be interpreted correctly on the basis of this analysis. However, according to the phrase-boundary scores proposed in (Black et al., 1991), the answer has 100% recall and 75% precision. In contrast, Table 18.5 shows the dependency-based evaluation of the two parses in Figure 18.3. Both the recall and precision of the answer are 25%, which are appropriate for a very bad parse.

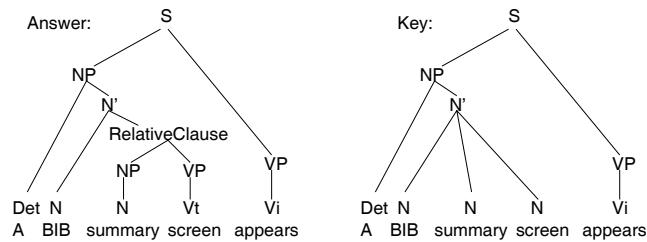


Figure 18.3. Two parse trees of “A BIB summary screen appears”

More recently, (Carroll et al., 1999) proposed a parser evaluation scheme similar to (Lin, 1995). Instead of comparing unlabeled dependency trees, Carroll *et al.* extracted relational tuples that represent a set of labeled dependency relationships between content words. One advantage of their scheme is the possibility of evaluating the deep-level grammatical relationships in structures involving ‘moved’ elements, such as passive and relative clauses.

## 6. CONCLUSIONS

We presented a dependency-based evaluation of a principle-based broad-coverage parser, called MINIPAR. The dependency-based method offers several advantages over previous methods that rely on the comparison of phrase boundaries. By evaluating a selected subset of dependencies, we can measure parser performance with respect to different types of grammatical relationships, or different types of lexical properties. Our evaluation shows that MINIPAR is able to achieve about 89% precision and 79% recall.

## References

- Bömová, A., Hajič, J., Hajicová, E., Hladká, B. (2003). The Prague Dependency Treebank: A Three level Annotation Scenario. In Abeillé, Anne (ed.) *Treebanks: Building and Using Syntactically Annotated Corpora*, Kluwer Academic Publishers, this volume.
- Berwick, R. C., S. P. Abney, C. Tenny (eds.) (1991). *Principle-Based Parsing: Computation and Psycholinguistics*. Kluwer Academic Publishers.
- Black, E., S. Abney, D. Flickenger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, T. Strzalkowski (1991). A Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars. *Proceedings of Speech and Natural Language Workshop*. p. 306–311.

- Carroll, J., G. Minnen, T. Briscoe (1999). Corpus Annotation for Parser Evaluation. *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora*. Bergen, Norway.
- Chomsky, N. (1995). *The Minimalist Program*. MIT Press.
- Collins, M. J. (1996). 'A New Statistical Parser Based on Bigram Lexical Dependencies'. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*. Santa Cruz, CA, p. 184–191.
- Collins, M. J. (1997). Three Generative, Lexicalized Models for Statistical Parsing. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*. Madrid, Spain, p. 16–23.
- Hays, D. (1964). Dependency theory: a formalism and some observations. *Language* **40**, p. 511–525.
- Hudson, R. (1984). *Word Grammar*. Oxford, England: Basil Blackwell Publishers Limited.
- Lin, D. (1993). Principle-based Parsing without Overgeneration. *Proceedings of ACL-93*. Columbus, Ohio, pp. 112–120.
- Lin, D. (1994) PRINCIPAR—An Efficient, Broad-coverage, Principle-based Parser. *Proceedings of COLING-94*. Kyoto, Japan: p. 482–488.
- Lin, D. (1995) A Dependency-based Method for Evaluating Broad-coverage Parsers. *Proceedings of IJCAI-95*. Montreal, Canada, p. 1420–1425.
- Lin, D. (1998). A Dependency-based Method for Evaluating Broad-Coverage Parsers. *Journal of Natural Language Engineering*, p. 97–114.
- Magerman, D. (1995). Statistical Decision-Tree Model for Parsing. *Proceedings of ACL-96*. Cambridge, MA, p. 276–283.
- Marcus, M. P., B. Santorini, M. A. Marcinkiewicz (1993). Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics* **19**(2), p. 313–330.
- Mel'čuk, I. A. (1987). *Dependency syntax: theory and practice*. Albany: State University of New York Press.
- Miller, G. A., R. Beckwith, C. Fellbaum, D. Gross, K. J. Miller (1990). 'Introduction to WordNet: An on-line lexical database'. *International Journal of Lexicography* **3**(4), p. 235–244.
- Sampson, G. (2003). Thoughts on two decades of drawing trees. This volume.
- Sampson, G. R. (1995). *English for the Computer*. Oxford University Press.

