

# *Evaluating State-of-the-Art Treebank-style Parsers for Coh-Matrix and Other Learning Technology Environments*

CHRISTIAN F. HEMPELMANN, VASILE RUS,  
ARTHUR C. GRAESSER and  
DANIELLE S. MCNAMARA

*Institute for Intelligent Systems, Departments of Computer Science and Psychology,  
The University of Memphis, Memphis, TN 38120, USA  
e-mail: {chmplmn, vrus, a-graesser, dsmcnamr}@memphis.edu*

*(Received 1 October 2005)*

---

## **Abstract**

This paper evaluates four of the most commonly used, freely available, state-of-the-art parsers on a standard benchmark as well as with respect to a set of data relevant for measuring text cohesion, as one example of a learning technology application that requires fast and accurate syntactic parsing. We outline advantages and disadvantages of existing technologies and make recommendations. Our performance report uses traditional measures based on a gold standard as well as novel dimensions for parsing evaluation. To our knowledge, this is the first attempt to evaluate parsers across genres and grade levels for the implementation in learning technology using both gold standard and directed evaluation methods.

---

## **1 Introduction**

The task of syntactic parsing is valuable to most natural language understanding applications, e.g., anaphora resolution (Hobbs 1978; Lappin and Leass 1994) or question answering (Voorhees 2002). Syntactic parsing in its most general definition may be viewed as discovering the underlying syntactic structure of a sentence. The specificities include the types of elements and relations that are retrieved by the parsing process and the way in which they are represented. For example, Treebank-style parsers retrieve a bracketed form that encodes a hierarchical organization (tree) of smaller elements (called phrases), while Grammatical-Relations(GR)-style parsers explicitly output relations together with elements involved in the relation (subj(John,walk)).

The present paper presents an evaluation of parsers for the Coh-Matrix project (Graesser, McNamara, Louwerse and Cai 2004) at the Institute for Intelligent Systems of University of Memphis. Coh-Matrix is a text-processing tool that provides new methods of automatically assessing text cohesion, readability, and difficulty. In its present v1.1, few cohesion measures are based on syntactic information, but its

next incarnation v2.0 will depend more heavily on hierarchical syntactic information. We are developing these measures. Thus, our current goal is to provide the most reliable parser output available for them, while still being able to process larger texts in real time. The usual trade-off between accuracy and speed has to be taken into account.

In the first part of the evaluation, we adopt a constituent-based approach for evaluation, as the output parses are all derived in one way or another from the same data and generate similar, bracketed output. The major goal is to consistently evaluate some freely available state-of-the-art parsers on a standard data set and across genres on corpora typical for learning technology environments. The choice of parsers was led by familiarity of authors with them. For the parsers thus selected, we report competitiveness along an array of dimensions including performance, robustness, tagging facility, and length of input they can handle.

Next, we briefly address particular types of misparses and mistags in their relation to measures planned for Coh-Metrix 2.0. These measures are typical for NLP tasks in learning technology environments in their requirement of accurate parses for more sophisticated further processing. Coh-Metrix 2.0 measures that centrally rely on good parses include:

- *causal and intentional cohesion* (Dufty, Hempelmann, Graesser and McNamara 2005), for which the main verb and its subject must be identifiable from the parse;
- *temporal cohesion* (Hempelmann, Crossley, Graesser, Louwerson and McNamara 2005), for which the main verb and its tense/aspect must be identifiable from the parse;
- *anaphora resolution* (Rus, Dufty, Hempelmann, Graesser and McNamara 2005), for which the syntactic relations of pronoun and referent must be accurately provided by the parse.

All of these measures require complex algorithms operating on the cleanest possible sentence parse, as a faulty parse will lead to a cascading error effect.

### *1.1 Parser types*

While the purpose of this work is not to propose a taxonomy of all available parsers, we consider it necessary to offer a brief overview of the various parser dimensions. Parsers can be classified according to their general approach (hand-built-grammar-based versus statistical), the way rules in parses are built (selective vs. generative), the parsing algorithm they use (LR, chart parser, etc.), type of grammar (unification-based grammars, context-free grammars, lexicalized context-free grammars, etc.), the representation of the output (bracketed, list of relations, etc.), and the type of output itself (phrases vs. grammatical relations). Of particular interest to our work are Treebank-style parsers, i.e., parsers producing an output conforming to the Penn Treebank (PTB) annotation guidelines (Marcus, Santorini and Marcinkiewicz 1993). The PTB project defined a tag set and bracketed form to represent syntactic trees that became a standard for parsers developed/trained

on PTB. It also produced a treebank, a collection of hand-annotated texts with syntactic information, of considerable size.

Given the large number of dimensions along which parsers can be distinguished, an evaluation framework that provides both parser-specific (to understand the strength of different technologies) and parser-independent (to be able to compare different parsers) performance figures is desirable and commonly used in the literature.

### *1.2 General parser evaluation methods*

Carroll, Briscoe and Sanfilippo (1999) broadly divide evaluation methods into non-corpus- and corpus-based methods with the latter subdivided into unannotated (expert-only) and annotated (gold standard) corpus-based methods. The non-corpus method simply lists linguistic constructions covered by the parser/grammar. It is well-suited for hand-built grammars because during the construction phase the covered cases can be recorded. However, it has problems with capturing complexities occurring from the interaction of covered cases.

The most widely used corpus-based evaluation methods are: (1) the constituent-based (phrase structure) method, and (2) the dependency/Grammatical-Relations (GR)-based method. The former has its roots in the Grammar Evaluation Interest Group (GEIG) scheme (Grishman, MacLeaod and Sterling 1992) developed to compare parsers with different underlying grammatical formalisms. It promoted the use of phrase-structure bracketed information and defined Precision, Recall, and Crossing Brackets measures. The GEIG measures were extended later to constituent information (bracketing information plus label) and have since become the standard for reporting automated syntactic parsing performance. Among the advantages of constituent-based evaluation are generality (less parser specificity) and fine grain size of the measures. On the other hand, the measures used in this method are weaker than the criterion of full identity for a sentence (if the entire parse is correct a perfect score is assigned, otherwise the null score is given), and it is not clear if they properly evaluate how well a parser identifies the true structure of a sentence. Lin (1995) shows that many phrase boundary mismatches stem from differences between parsers/grammars and corpus annotation schemes. Usually, treebanks are constructed with respect to informal guidelines. Annotators often interpret them differently, which leads to a large number of different annotations for syntactically identical structures.

Dependency-based and GR-based methods are closely related. The dependency-based method relies on a dependency-graph with arcs encoding certain dependencies while the GR-based method relies on sets of parser-independent dependency relationships defined as a triplet (modifier, modiffee, label) where the label is the type of relationship. Henceforth, the two are considered a single evaluation method. It can be applied indirectly to phrase structure analyses from parsers and treebanks by mapping phrasal structures into triplets. The mapping can be done at the cost of losing certain grammatical information.

There are two major approaches to evaluate parsers using the constituent-based method. On the one hand, there is the expert-only approach in which an expert

looks at the output of a parser, counts errors, and reports different measures. We use a variant of this approach for the directed parser evaluation (see next section). Using a gold standard, on the other hand, is a method that can be automated to a higher degree. It replaces the counting part of the former method with a software system that compares the output of the parser to the gold standard, highly accurate data, manually parsed – or automatically parsed and manually corrected – by human experts. The latter approach is more useful for scaling up evaluations to large collections of data while the expert-only approach is more flexible, allowing for evaluation of parsers from new perspectives and with a view to special applications, e.g., in learning technology environments.

### 1.3 Gold standard evaluation

In the first part of this work we use the gold standard approach for parser evaluation, initially reported in an early form in elsewhere (Rus and Hempelmann 2005). The evaluation is done from two different points of view. First, we offer a uniform evaluation for the parsers on section 23 from the Wall Street Journal (WSJ) section of PTB, the community norm to report parser performance. The goal of this first evaluation is to offer a good estimation of the parsers when evaluated on identical environments (same configuration parameters and same hardware and systems software configuration for the evaluator software). We also observe the following features which are extremely important for using the parsers in large scale text processing and to embed them as components in larger systems.

- *Self-tagging*: whether or not the parser does tagging itself. It is advantageous to take in raw text since it eliminates the need for extra modules.
- *Performance*: if the number of correctly parsed sentences is 85% or higher.
- *Long sentences*: the ability of the parser to handle sentences longer than 40 words.
- *Robustness*: relates to the property of a parser to handle any type of input sentence and return a reasonable output for it and not an empty line or some other useless output.
- *Stability*: whether or not the parser can process large collections of text without crashing or freezing.
- *Resource-needy*: if the parser requires large quantities of memory or it takes a long period of time to parse average sentences making it unusable in interactive systems.

Second, we evaluate the parsers on narrative and expository texts to study their performance across the two genres. This second evaluation step will provide additional important results for learning technology projects. We use *evalb* (Sekine and Collins 1997) to evaluate the bracketing performance of the output of a parser against a gold standard. The software evaluator reports numerous measures of which we only report the two most important: labelled precision (LR), labelled recall (LR).

### 1.4 Directed parser evaluation method

For the third step of this evaluation we looked for specific problems that will affect Coh-Metrix 2.0, and presumably learning technology applications in general, with a view to amending them by postprocessing the parser output. The following four classes of problems in a sentence's parse were distinguished:

- *None*: the parse is generally correct, unambiguous, and poses no problem for Coh-Metrix 2.0.
- *One*: there was one minor problem, e.g., a mislabeled terminal or a wrong scope of an adverbial or prepositional phrase (wrong attachment site) that did not affect the overall parse of the sentence, which is therefore still usable for Coh-Metrix 2.0 measures.
- *Two*: there were two or three problems of type one, or a problem with the tree structure that affected the overall parse of the sentence, but not in a fatal manner, e.g., a wrong phrase boundary, or a mislabelled higher constituent.
- *Three*: there were two or more problems of type two, or two or more of type one as well as one or more of type two, or another general problem that made the parse of the sentence completely useless, unintelligible, e.g., an omitted sentence or a sentence split into two, because a sentence boundary was misidentified.

## 2 Evaluated parsers

### 2.1 Apple Pie

Apple Pie (AP) (Sekine and Grishman 1995) extracts a grammar from PTB v.2 in which S and NP are the only true non-terminals (the others are included in the right-hand side of S and NP rules). The rules extracted from the PTB have S or NP on the left-hand side and a flat structure on the right-hand side, for instance  $S \rightarrow NP\ VBX\ JJ$ . Each such rule has the most common structure in the PTB associated with it, and if the parser uses the rule it will generate its corresponding structure. The parser is a chart parser and factors grammar rules with common prefixes to reduce the number of active nodes. Although the underlying model of the parser is simple, it can't handle sentences over 40 words due to the large variety of linguistic constructs in the PTB.

### 2.2 Charniak's parser

Charniak developed a parser (CP) based on probabilities gathered from the WSJ part of the PTB (Charniak 1997). It extracts the grammar and probabilities and with a standard context-free chart-parsing mechanism generates a set of possible parses for each sentence retaining the one with the highest probability (probabilities are not computed for all possible parses). The probabilities of an entire tree are computed bottom-up. In Charniak (2000), he proposes a generative model based on a Markov-grammar. It uses a standard bottom-up, best-first probabilistic parser to first generate possible parses before ranking them with a probabilistic model.

### 2.3 Collins's (*Bikel's*) parser

Collins's statistical parser (CBP), described in Collins (1996, 1997), is based on the probabilities between head-words in parse trees. It explicitly represents the parse probabilities in terms of basic syntactic relationships of these lexical heads. Collins defines a mapping from parse trees to sets of dependencies, on which he defines his statistical model. A set of rules defines a head-child for each node in the tree. The lexical head of the head-child of each node becomes the lexical head of the parent node. Associated with each node is a set of dependencies derived in the following way. For each non-head child, a dependency is added to the set where the dependency is identified by a triplet consisting of the non-head-child non-terminal, the parent non-terminal, and the head-child non-terminal. The algorithm is based on CYK-style dynamic programming chart parsing.

### 2.4 Stanford Parser

The Stanford Parser (SP) (Klein and Manning 2003) uses a context-free grammar with state splits. The parsing algorithm is simpler, the grammar smaller and fewer parameters are needed for the estimation. It uses a CYK chart parser which exhaustively generates all possible parses for a sentence before it selects the highest probability tree. Here we used the default lexicalized version but an unlexicalized version of it is also available. The unlexicalized version has the advantage of generating a simpler statistical model since lexical information is not used for it. The unlexicalized version was not readily available at the time of the experiments.

## 3 Experiments and results

### 3.1 Text corpus

We performed experiments on three data sets. First, we chose the norm for large scale parser evaluation, the 2416 sentences of WSJ section 23. Since parsers have different parameters that can be tuned leading to (slightly) different results, we first report performance values on the standard data set and then use same parameter settings on the second data set for more reliable comparison.

The second experiment is on a set of three narrative and four expository texts. The gold standard for this second data set was built manually by the authors starting from CP's as well as SP's output on those texts. The four texts used initially are two expository and two narrative texts of reasonable length for detailed evaluation:

- *The Effects of Heat* (SRA Real Science Grade 2 Elementary Science): expository; 52 sentences, 392 words: 7.53 words/sentence;
- *The Needs of Plants* (McGraw-Hill Science): expository; 46 sentences, 458 words: 9.96 words/sentence;
- *Orlando* (Addison Wesley Phonics Take-Home Reader Grade 2): narrative; 65 sentences, 446 words: 6.86 words/sentence;

- *Moving* (McGraw-Hill Reading - TerraNova Test Preparation and Practice – Teachers Edition Grade 3): narrative, 33 sentences, 433 words: 13.12 words/sentence.

An additional set of three texts was chosen from the Touchstone Applied Science Associates, Inc., (TASA) corpus<sup>1</sup> with an average sentence length of 13.06 (overall TASA average) or higher.

- *Barron17*: expository; DRP=75.14 (college grade); 13 sentences, 288 words: 22.15 words/sentence;
- *Betty03*: narrative; DRP=56.92 (5th grade); 14 sentences, 255 words: 18.21 words/sentence;
- *Olga91*: expository; DRP=74.22 (college grade); 12 sentences, 311 words: 25.92 words/sentence.

We also tested all four parsers for speed on a corpus of four texts chosen randomly from the MetaMetrics corpus<sup>2</sup> of school text books, across high and low grade levels and across narrative and science texts (see Section 3.2.2).

- *G4*: 4th grade narrative text, 1,500 sentences, 18,835 words: 12.56 words/sentence;
- *G6*: 6th grade science text, 1,500 sentences, 18,237 words: 12.16 words/sentence;
- *G11*: 11th grade narrative text, 1,558 sentences, 18,583 words: 11.93 words/sentence;
- *G12*: 12th grade science text, 1,520 sentences, 25,098 words: 16.51 words/sentence.

### 3.2 General parser evaluation results

#### 3.2.1 Accuracy

The parameters file we used for *evalb* was the standard one that comes with the package. Some parsers are not robust, meaning for some input they do not output anything, leading to empty lines that are not handled by the evaluator. Those parses had to be “aligned” with the gold standard files so that empty lines are eliminated from the output file together with their peers in the corresponding gold standard files.

In Table 1 we report the performance values on Section 23 of WSJ. Tables 2 and 3 give the results for our own corpus, the former against the CP-based gold standard, and the latter against the SP-based one. Note that CP and SP possibly still score high because of this bias. However, CBP is clearly a contender despite the bias,

<sup>1</sup> This corpus contains paragraph-length sections from K-12 educational texts. A more detailed citation is unfortunately not possible. More information can be found at <http://www.tasa.com>.

<sup>2</sup> The University of Memphis is grateful to MetaMetrics ([www.metametricsinc.com](http://www.metametricsinc.com)) for making this large corpus of school textbooks available under a research agreement.

Table 1. Accuracy of parsers

Parser	Performance(LP/LR/Tagging - %)		
	WSJ 23	Expository	Narrative
Appie Pie	43.71/44.29/90.26	41.63/42.70	42.84/43.84
Charniak's	84.35/88.28/92.58	91.91/93.94	93.74/96.18
Collins/Bikel's	84.97/87.30/93.24	82.08/85.35	67.75/85.19
Stanford	84.41/87.00/95.05	75.38/85.12	62.65/87.56

Table 2. Performance of parsers on the narrative and expository text (Charniak-based GS)

File	Performance(LR/LP - %)			
	Apple Pie	Charniak	Stanford	Bikel/Collins
Barron17	41.83/41.51	94.68/96.14	72.24/86.76	86.69/84.44
Betty03	45.12/43.11	97.21/96.76	83.72/80.72	71.63/77.39
Heat	46.86/46.86	94.69/97.76	88.41/91.27	90.58/91.24
Moving	36.59/40.15	92.27/97.36	51.14/88.58	75.23/84.22
Olga91	33.45/33.92	85.37/88.13	57.84/70.94	65.61/75.20
Orlando	47.47/47.26	93.62/94.22	63.75/91.25	58.25/91.37
Plants	41.91/45.18	91.91/92.90	76.38/86.30	82.13/86.16

while AP is not.<sup>3</sup> The reported metrics are Labelled Precision (LP) and Labelled Recall (LR). Let us denote by  $a$  the number of correct phrases in the output from a parser for a sentence, by  $b$  the number of incorrect phrases in the output and by  $c$  the number of phrases in the gold standard for the same sentence. LP is defined as  $a/(a+b)$  and LR is defined as  $a/c$ . A summary of the other dimensions of the evaluation is offered in Table 4. A stability dimension accounting for the proneness of a parser to crash is not reported because we were not able to find a bullet-proof parser so far, but we must recognize that some parsers are significantly more stable than others, namely CP and CBP. In terms of resources needed, the parsers are comparable, except for AP which uses less memory and processing time. The LP/LR of AP is significantly lower, partly due to its outputting partial trees for longer sentences. Overall, CP offers the best performance.

Note in Table 1 that CP's tagging accuracy is worst among the three top parsers but still delivers best overall parsing results. This means that its parsing-only performance is slightly better than the numbers in the table indicate. The numbers actually represent the tagging and parsing accuracy of the tested parsing systems.

<sup>3</sup> AP's performance is only reported for sentences shorter than 40 words, 2,250 out of 2,416. SP is also not robust enough and the performance reported is only on 2,094 out of 2,416 sentences in section 23 of WSJ.

Table 3. *Performance of parsers on the narrative and expository text (Stanford-based GS)*

File	Performance(LR/LP - %)			
	Apple Pie	Charniak	Stanford	Bikel/Collins
Barron17	44.61/44.39	84.80/86.50	88.73/91.88	87.75/88.18
Betty03	47.93/46.22	84.33/84.72	92.17/89.69	77.42/84.42
Heat	49.63/48.31	89.22/89.78	96.53/97.01	94.29/92.46
Moving	38.94/42.74	85.58/88.12	56.73/94.78	77.88/85.71
Olga91	31.12/31.45	69.93/71.94	65.38/79.91	57.69/66.00
Orlando	44.16/50.81	78.04/89.54	67.99/96.68	57.24/90.07
Plants	41.79/46.59	78.77/83.14	80.09/90.59	79.87/85.08

Table 4. *Evaluation of parsers with respect to the criteria listed at the top of each column*

Parser	Self-tagging	Performance	Long-sentences	Robustness
AP	Yes	No	No	No
CP	Yes	Yes	Yes	Yes
CBP	Yes	Yes	Yes	Yes
SP	Yes	Yes	No	No

Nevertheless, this is what we would most likely want to know since one would prefer to input raw text as opposed to tagged text. If more finely grained comparisons of only the parsing aspects of the parsers are required, perfect tags extracted from PTB must be provided to measure performance.

Note also that the only significant difference between expository and narrative fictional texts showed up in the precision of the parses of CBP and SP, which performed better on expository text.

Table 5 shows average measures for each of the parsers on the PTB and seven expository and narrative texts in the second column and for expository and narrative in the fourth column. The third and fifth columns contain standard deviations for the previous columns, respectively. Here too, CP shows the best result.

### 3.2.2 Speed

All parsers ran on the same Linux Debian machine: P4 at 3.4GHz with 1.0GB of RAM.<sup>4</sup> AP's and SP's high speeds can be explained to a large degree by their skipping longer sentences, the very ones that lead to the longer times for the other

<sup>4</sup> Some of the parsers are also available for the Windows environment.

Table 5. *Average performance of parsers. LR = labeled recall, LP = labeled precision SD = standard deviation*

Parser	Ave. (LR/LP - %)	S.D. (%)	Ave. on Exp+Nar (LR/LP - %)	S.D. on Exp+Nar (%)
AP	42.73/43.61	1.04/0.82	42.24/43.46	5.59/5.41
CP	90.00/92.80	4.98/4.07	87.17/89.79	4.85/4.66
CBP	78.27/85.95	9.22/1.17	74.36/88.31	14.24/6.51
SP	74.14/86.56	10.93/1.28	75.88/84.42	12.66/7.11

Table 6. *Parser speed in seconds*

	G4	G6	G11	G12	Sentences Parsed
AP	144	89	144	242	619
CP	647	499	784	1406	3336
CBP	485	1947	1418	1126	4976
SP	449	391	724	651	2215
Ave.	431	732	768	856	

Table 7. *Average performance of parsers over all texts (directed evaluation)*

	Ave. (%)	S.D. (%)
AP	77.31	15.00
CP	88.69	8.86
CBP	79.82	18.94
SP	83.43	11.42

two candidates. Taking this into account, SP is clearly the fastest among the three accurate parsers, but the large range of processing times need to be heeded.

### 3.3 Directed parser evaluation results

This section reports the results of expert rating of parses for specific problems (see section 1.3). The results were produced as follows: A sentence was considered usable for further processing if the parse had no error or only one error of the least problematic type one. Table 7 shows the average percentage of sentences considered usable by these criteria that each parser produced across all seven texts. The best results are achieved by CP with an average of 88.69% output useable for Coh-Metrix 2.0. The other three candidates are clearly trailing behind, namely by between 5% (SP) and 11% (AP). CP also produces correct output most consistently at a standard deviation over the seven texts of 8.86%. The distribution of severe problems is comparable for all parsers.

Table 8. *Correlation of average performance per text for all parsers and average sentence length (directed evaluation)*

Text	perf. (%)	length (#words)
Heat	92.31	7.54
Plants	90.76	9.96
Orlando	93.46	6.86
Moving	90.91	13.12
Barron17	76.92	22.15
Betty03	71.43	18.21
Olga91	60.42	25.92

Table 9. *Specific problems by parser. PP = wrong attachment site for a prepositional phrase; ADV = wrong attachment site for an adverbial phrase; cNP = misparsed complex noun phrase; &X = wrong coordination*

	PP	ADV	cNP	&X
AP	13	10	8	9
CP	15	1	2	7
CBP	10	0	0	13
SP	22	6	3	4
Sum	60	17	13	33

As expected, longer sentences are more problematic for all parsers, as can be seen in Table 8. No significant trends in performance differences with respect to genre type, narrative (Orlando, Moving, Betty03) vs. expository texts (Heat, Plants, Barron17, Olga91), were detected (cf. also speed results in Table 6). But we assume that the difference in average sentence length obscures any genre differences in our relatively small sample.

The most common non-fatal problems (type one) involved the well-documented adjunct attachment site issue, in particular for prepositional phrases (Abney, Schapire and Singer 1999; Brill and Resnick 1995; Collins and Brooks 1995; Ratnaparkhi, Renyar and Roukos 1994) as well as adjectival phrases (Table 9).

A typical example for a wrong PP attachment from our corpus is *He would (miss (all his friends) (from the neighborhood))*, in which *from the neighborhood* is parsed as a sister to the V *miss* instead of correctly as part of the NP headed by *friends*: *He would (miss (all his friends (from the neighborhood)))*.

Similar misattachment issues for adjuncts are encountered with adverbial phrases, but they were rare in our corpus and postprocessing is assumed to provide less improvement at higher costs.

Another common problem are deverbal nouns and denominal verbs. They share surface forms leading to ambiguous part of speech assignments. In the following

example, apart from a resulting or underlying general structure misparse, the noun *surfaces* is mistagged as a denominal verb:

(S (NP (JJ Many) (NNS leaves)) (VP (VBP have) (ADJP (JJ broad)) (, ,) (ADJP (JJ flat) (S (VP (VBZ *surfaces*) (SBAR (IN that) (S (VP (VB help) (S (NP (PRP them)) (VP (VB take) (PP (IN in) (NP (NN sunlight))))))))))))))

An especially complex problem of this type has been left unaddressed and given only one POS tag, namely, *-ing* forms that are tagged as VBG. For several Coh-Metrix 2.0 measures, most obviously temporal cohesion, it is necessary to be able to distinguish gerunds from gerundives from deverbal adjectives and deverbal nouns, for example to identify verb aspect for progressive verbs to assess the succession of tenses in a text.

Problems with NP misidentification are particularly detrimental in view of the important role of NPs in Coh-Metrix 2.0 measures. This pertains in particular to the mistagging/misparsing of complex NPs and the coordination of NPs. In the following example, *help* is parsed as a noun coordinated with *roots*, rather than correctly as the head of a VP: (*NP roots also help*) (*VP take in water ...*).

Parses with fatal problems are expected to produce useless results for algorithms operating with them. Wrong coordination is another notorious problem of parsers (cf. Cremers 1993; Grootveld 1994). In our corpus we found 33 instances of miscoordination, of which 23 involved NPs. Postprocessing approaches that address these issues are currently under investigation.

#### 4 Conclusion

This paper presented the evaluation of four freely available, Treebank-style, parsers. We offered a uniform evaluation for four parsers: Apple Pie, Charniak's, Collins/Bikel's, and the Stanford parser. A novelty of this work is the evaluation of the parsers along new dimensions such as robustness and across genre, in particular narrative and expository. For the latter part we developed a gold standard for evaluating parsers with selected narrative and expository texts from the TASA corpus. No significant effect, not already captured by variation in sentence length, could be found here.

Another novelty is the evaluation of the parsers with respect to particular error types that are anticipated to be problematic for further processing of the resulting parses. The results of this directed evaluation confirmed the automated parser evaluation. But the output of both methods does not correlate at all, so that neither method covers the ground of the other and both contributed to the overall evaluation of the parsers.

Overall, from the general and directed evaluations, if not the speed testing, Charniak's parser emerged as the most successful candidate among the tested parsers to be integrated where learning technology requires syntactic information from real text in real time. It needs to be heeded, of course, that parsers not evaluated here may outperform this candidate. Also noted should be that Charniak has recently improved his parser further (Charniak and Johnson 2005).

## References

- Abney, S., Schapire, R. E. and Singer, Y. (1999) Boosting applied to tagging and PP attachment. *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 38–45.
- Brill, E. and Resnik, P. (1995) A rule-based approach to prepositional phrase attachment disambiguation. *Proceedings of the 15th International Conference on Computational Linguistics*, Kyoto, Japan.
- Carroll, J., Briscoe, E. and Sanfilippo, A. (1999) Parser evaluation: current practice. *Evaluation of Natural Language Processing Systems: Final Report*, pp. 140–150. EC DG-XIII LRE EAGLES Document EAG-II-EWG-PR.1.
- Charniak, E. (1997) Statistical parsing with a context-free grammar and word statistics. *Proceedings of the 14th National Conference on Artificial Intelligence*, Menlo Park: AAAI/MIT Press.
- Charniak, E. (2000) A Maximum-Entropy-inspired parser. *Proceedings of the North-American Chapter of Association for Computational Linguistics*, Seattle, WA.
- Charniak, E. and Johnson, M. (2005) Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pp. 173–180. Ann Arbor, MI.
- Collins, M. (1996) A new statistical parser based on Bigram lexical dependencies. *Proceedings of the 34th Annual Meeting of the ACL*, Santa Cruz, CA.
- Collins, M. (1997) Three generative, lexicalised models for statistical parsing. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, Madrid, Spain.
- Collins, M. and Brooks, J. (1995) Prepositional phrase attachment through a backed-off model. *Proceedings of the Third Workshop on Very Large Corpora*, Cambridge, UK.
- Cremers, C. (1993) *On Parsing Coordination Categorially*. Leiden University.
- Dufty, D., Hempelmann, C. F., Graesser, A. C. and McNamara, D. S. (2005) Automatic assessment of causal and intentional cohesion in Coh-Metrix. Technical Report, Institute for Intelligent Systems, University of Memphis.
- Graesser, A. C., McNamara, D. S., Louwerse, M. M. and Cai, Z. (2004) Coh-Metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, and Computers* **36**(2): 193–202.
- Grishman, R., MacLeod, C. and Sterling, J. (1992) Evaluating parsing strategies using standardized parse files. *Proceedings of the Third Conference on Applied Natural Language Processing*, pp. 156–161.
- Grootveld, M. (1994) *Parsing Coordination Generatively*. Leiden University.
- Hempelmann, C. F., Crossley, S., Graesser, A. C., Louwerse, M. M. and McNamara, D. S. (2005) Automatic assessment of temporal cohesion in Coh-Metrix. Technical Report, Institute for Intelligent Systems, University of Memphis.
- Hobbs, J. (1978) Resolving pronoun references. *Lingua* **44**: 144–151.
- Klein, D. and Manning, C. (2003) Accurate unlexicalized parsing. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan.
- Lappin, S. and Leass, H. J. (1994) An algorithm for pronominal anaphora resolution. *Computational Linguistics* **20**(4): 535–561.
- Lin, D. (1995) A dependency-based method for evaluating broad-coverage parsers. *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 1420–1427.
- Marcus, M. P., Santorini, B. and Marcinkiewicz, M. A. (1993) Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* **19**(2): 313–330.
- Ratnaparkhi, A., Renyar, J. and Roukos, S. (1994) A maximum entropy model for prepositional phrase attachment. *Proceedings of the ARPA Workshop on Human Language Technology*.
- Rus, V. and Hempelmann, C. F. (2005) Across-genres and empirical evaluation of state-of-the-art treebank-style parsers. *Proceedings of the 2nd Language and Technology Conference*, Poznan, Poland.

- Rus, V., Dufty, D., Hempelmann, C. F., Graesser, A. C. and McNamara, D. S. (2005) Implementing Anaphora Resolution in Coh-Metrix. Technical Report, Institute for Intelligent Systems, University of Memphis.
- Sekine, S. and Grishman, R. (1995) A corpus-based probabilistic grammar with only two non-terminals. *Proceedings of the International Workshop on Parsing Technologies*, pp. 216–223.
- Sekine, S. and Collins, M. (1997) *EvalB: a bracket scoring program*. URL: <http://www.cs.nyu.edu/cs/projects/proteus/evalb/>
- Voorhees, E. (2002) Overview of the TREC 2002 question answering track. *Proceedings of the 11th Text Retrieval Conference (TREC 2002)*.