Enhancing Best Analysis Selection and Parser Comparison

Aleš Horák, Vladimír Kadlec, and Pavel Smrž

Faculty of Informatics, Masaryk University Brno
Botanická 68a, 602 00 Brno, Czech Republic
E-mail: {smrz,hales,xkadlec}@fi.muni.cz

Abstract. This paper discusses methods enhancing the selection of a "best" parsing tree from the output of natural language syntactic analysis. It presents a method for cutting away redundant parse trees based on the information obtained from a dependency tree-bank corpus.
The effectivity of the enhanced parser is demonstrated by results of inter-system parser comparison. The test were run on the standard evaluation grammars (ATIS, CT and PT), our system outperforms the referential implementations.

1 Introduction

The total number of atoms in the Universe is estimated to be 10^{90}. The average number of parsing trees per input sentence strongly depends on the background grammar and thence on the language. There are natural language grammars producing at most hundreds or thousands of parsing trees but also highly ambiguous grammar systems producing enormous number of results.

Ambiguity on all levels of representation is an inherent property of natural languages and it also forms a central problem of natural language parsing. A consequence of the natural language ambiguity is a high number of possible outputs of a parser that are represented by labeled trees.

For example, a grammar extracted from the Penn Treebank and tested on a set of sentences randomly generated from a probabilistic version of the grammar has on average 7.2×10^{27} parses per sentence according to Moore's work [1]. Such a mammoth extent of result is also no exception in parsing of Czech [2] due to free word order and rich morphology of word forms whose grammatical case cannot often be unambiguously determined.

A traditional solution for these problems is presented by probabilistic parsing techniques [3] aiming at finding the most probable parse of a given input sentence. This methodology is usually based on the relative frequencies of occurrences of the possible relations in a representative corpus.

In the following text, we present an acquisition of training data for the best analysis selection. The underlying mechanism is based on the pruning constraints that automate the process of transformation of a dependency tree-bank corpus.

The results are then compared to running times of a referential parsing system. The comparison indicates that our system is fully able to compete with the best current parsers.

2 Best Analysis Selection

First, in order to be able to exploit the data from PDTB, we have supplemented our grammar with the dependency specification for constituents. Thus the output of the analysis can be presented in the form of pure dependency tree. In the same time we unify classes of derivation trees that correspond to one dependency structure. We then define a canonical form of the derivation to select one representative of the class that is used for assigning the edge probabilities.

This technique enables us to relate the output of our parser to the PDTB data. However, the profit of exploitation of the information from the dependency structures can be higher than that and can run in an automatically controlled environment. For this purpose, we use the mechanism of *pruning constraints*. A set of strict limitations is given to the syntactic analyser, which passes on just the compliant parses. The constraints can be either supplied manually for particular sentence by linguists, or obtained from the transformed dependency tree in PDTB.

The transformation is driven by guidelines specified by linguists. These guidelines relate the following information:

– *afun* — analytical function attribute from PDTB 1.0
– *term* — corresponding nonterminal or preterminal from the metagrammar
– *mtag* — morphological tag constraint
– *lexit* — lexical item constraint

The automatic procedure for generating the pruning constraints then successively tries to match the analytical function attribute in the input sentence with the records in the transformation guidelines. Each match found is then checked for agreement in the particular morphological tag and lexical item according to the given criteria (currently a pattern matching based on text regular expressions). If all required fields comport with the guidelines, the corresponding subtree is chosen as the specified nonterminal or preterminal from the metagrammar. The syntactic analysis with the pruning constraints applied then prunes those parsing trees from the resulting chart that do not contain the requested nonterminal or preterminal in that position.

If more than one records in the guidelines match, the first match is applied. This mechanism allows to prefer the most specific records to the general ones, which differ in the lexical item constraint or the morphological tag constraint only, used e.g. in the differentiation of various adverbial types:

#	*afun*	*term*	*mtag*	*lexit*
	`Adv`	`np`	`k1`	
	`Adv`	`adv`		

#	*afun*	*term*	*mtag*	*lexit*
	`Sb`	`np`	`k1`	
	`Sb_Ap`	`np`		
	`Obj`	`np`		
	`Atr`	`modif`	`k2`	
	`AuxP`	`pn`		

Table 1. Simplified example of transformation guidelines.

The process of transformation guidelines preparation is divided into several steps to assure the consistency of acquired pruning constraints. After every change, the results are checked against a testing set of input sentences and the differences are reported to the user for arbitration.

The integration of the pruning constraints obtained automatically through the mechanism of transformation guidelines has shown to be very efficient. The tedious work of the training data acquisition for the best analysis selection algorithm has been substantially facilitated. Examples of the reduction are displayed in the following table:

sentence #	# words	# analyses	# pruned analyses	reduced to (%)
00214	30	3112504044	2146560	0.07
00217	3	2	2	100
00287	12	56	4	7
00308	7	10	6	60
00486	6	1	1	100
00599	35	44660	4872	11
00612	25	2369063760	1048896	0.04
00842	17	409920	6336	1.5

The table presents examples of sentences which were randomly chosen from the set of 1000 sentences analysed first without pruning constraints and then with automatically generated pruning constraints.

The average percentage of reduction on all the tested sentences has achieved 30 % (see Figure 1). The future work on the refinement of transformation guidelines will be concentrated on further reduction of the number of automatically pruned analysis trees.

3 Parser Comparison

The effectivity comparison of different parsers and parsing techniques brings a strong impulse to improving the actual implementations. Since there is no other generally applicable and available NL parser for Czech, we have compared the running times of our syntactic analyser on the data provided at

Fig. 1. The dependency of the reduction (%) of the number of resulting analyses on the number of words in the input sentence

`http://www.cogs.susx.ac.uk/lab/nlp/carroll/cfg-resources/`. These web pages resulted from discussions at the Efficiency in Large Scale Parsing Systems Workshop at COLING'2000, where one of the main conclusions was the need for a bank of data for standardization of parser benchmarking.

3.1 HDddm Parsing Technique

The parsing technique of our system is based on the head driven approach with improvements regarding the process of confirmation of viable hypotheses. The HDddm (head driven with dependent dot move) parsing technique refers to the fact that the move of one "dot" in the head driven parsing step is dependent on the opposite move of the other one.

The head of a grammar rule is a symbol from the right hand side. For example, the second nonterminal (`np`) is denoted as the head symbol in the following grammar rule

```
np -> left_modif np
  head($2)
```

The epsilon rule has a special head symbol ϵ. The edge in the head driven parser is a triplet $[A\rightarrow \alpha_\bullet\beta_\bullet\gamma, i, j]$, where i, j are integers, $0 \leq i \leq j \leq n$ for n words in the input sentence and $A\rightarrow \alpha\beta\gamma$ is a rule in the input grammar. The direction of the parsing process does not move unidirectionally from left to right, but it starts at the head of the grammar rule.

The parsing algorithm can be summarized by the following schema, where the symbol G stands for the input grammar with a set of rules P and the root symbol S, a_1, ..., a_n are input words (preterminals):

Initialisation phase

1. for each $p \in P \mid p = A \rightarrow \epsilon$ add edges $[A \rightarrow {\bullet}{\bullet}, 0, 0]$, $[A \rightarrow {\bullet}{\bullet}, 1, 1]$, ..., $[A \rightarrow {\bullet}{\bullet}$, n, n$]$ to the chart.
2. for each $p \in P \mid p = A \rightarrow \alpha \underline{a_i} \beta$ (a_i is the head of the rule) add edge $[A \rightarrow \alpha_{\bullet} a_{i \bullet} \beta$, i-1, i$]$ to the chart.

Iteration phase

1. if edge E in the chart is in the form $[A \rightarrow {}_{\bullet}\alpha_{\bullet}, j, k]$, then for each edge:
   $[B \rightarrow \beta_{\bullet}\gamma_{\bullet}A\delta, i, j]$ in the chart, create edge $[B \rightarrow \beta_{\bullet}\gamma A_{\bullet}\delta, i, k]$.
   $[B \rightarrow \beta A_{\bullet}\gamma_{\bullet}, k, l]$ in the chart, create edge $[B \rightarrow \beta_{\bullet}A\gamma_{\bullet}, j, l]$.
2. if E is in the form $[B \rightarrow \beta_{\bullet}\gamma_{\bullet}A\delta, i, j]$, then for each edge $[A \rightarrow {}_{\bullet}\alpha_{\bullet}, j, k]$ in the chart, create edge $[B \rightarrow \beta_{\bullet}\gamma A_{\bullet}\delta, i, k]$.
3. if E is in the form $[B \rightarrow \beta A_{\bullet}\gamma_{\bullet}, k, l]$, then for each edge $[A \rightarrow {}_{\bullet}\alpha_{\bullet}, j, k]$ in the chart, create edge $[B \rightarrow \beta_{\bullet}A\gamma_{\bullet}, j, l]$.
4. if E is in the form $[A \rightarrow \beta_{\bullet}\gamma_{\bullet}a_{j+1}\delta, i, j]$, then create edge $[A \rightarrow \beta_{\bullet}\gamma a_{j+1 \bullet}\delta, i, j{+}1]$.
5. if E is in the form $[A \rightarrow \beta a_{i \bullet}\gamma_{\bullet}, i, j]$, then create edge $[A \rightarrow \beta_{\bullet}a_i\gamma_{\bullet}, i\text{-}1, j]$.
6. if E is in the form $[A \rightarrow {}_{\bullet}\alpha_{\bullet}, i, j]$, then for each rule $B \rightarrow \beta\ \underline{A}\ \gamma$ in the input grammar, create edge $[B \rightarrow \beta_{\bullet}A_{\bullet}\gamma, i, j]$ (symbol A is the head of the rule).

N.B., that the left dot in the edge cannot move leftwards until the right dot moves to the right. The parser never creates edges like $[A \rightarrow \alpha_{\bullet}\beta\underline{A}\gamma_{\bullet}\delta, i, j]$ for non empty β. This approach avoids the redundant analysis of such edges. On the other hand, the parser does not use any top-down filtering or "follow check" technique.

The efficiency of the parser depends to a considerable extent on the choice of grammar rule heads. The current positions of heads in our grammar have been chosen experimentally and they accords with the conception of the leading constituent in the traditional Czech grammars.

3.2 Running Time Comparison

The best results reported on standard data sets (ATIS, CT and PT grammars) until today are the comparison data by Robert C. Moore [1]. In the package, only the testing grammars with input sentences are at the disposal, the release of referential implementation of the parser is currently being prepared (Moore, personal communication).

The basic characteristics of the testing grammars are presented in Table 2. A detailed description of these grammars is given in the [12].

The results of the parser comparison appear in Table 3. The values in the table give the total CPU times in seconds required by the parser to completely process the test set associated with the grammar.

Since we could not run the referential implementation of Moore's parser on the same machine, the above mentioned times are not fully comparable (we assume that our tests were run on a slightly faster machine than that of Moore's

Grammar	CT	Atis	PT
Rules	24,456	4,592	15,039
Nonterminals	3,946	192	38
Terminals	1,032	357	47
Test sentences	162	98	30
Average Parses	5.4	940	more than 2^{64}
Grammar	CT	Atis	PT

Table 2. Test grammars and test sentences.

ATIS grammar, Moore's LC_3 + UTF	11.6
ATIS grammar, our system	4.19
CT grammar, Moore's LC_3 + UTF	3.1
CT grammar, our system	4.19
PT grammar, Moore's LC_3 + UTF	41.8
PT grammar, our system	17.75

Table 3. Running times comparison (in seconds)

tests). We prepare a detailed comparison, which will try to explain the differences of results when parsing with grammars of varying ambiguity level.

The longer running times on the data of the CT grammar are caused by little ambiguity of the grammar, so that our parsing technique optimized for highly ambiguous grammars cannot display its strong suits.

4 Conclusions

The methods of the best analysis selection algorithm show that the parsing of inflectional languages calls for sensitive approaches to the evaluation of the appropriate figures of merit. The acquisition of these output arranging quantities is based on a representative training data set. The method of pruning constraints described in this paper enables to automate the process of treebank corpus transformation.

The integration of the presented methods to the parsing system has no destructive impact on the efficiency of the parser. This is documented by the comparison of the running times. Our system outperforms the results of the best referential parsing system on highly ambiguous grammars, for which it is optimized.

Future directions of our research lead to improvements of the quality of training data set so that it would cover all the most frequent language phenomena. The overall efficiency of the parser will be guaranteed by supplementary filtering techniques, which are going to be implemented.

References

1. R. C. Moore. Improved left-corner chart parsing for large context-free grammars. In *Proceedings of the 6th IWPT*, pages 171–182, Trento, Italy, 2000.
2. Pavel Smrž and Aleš Horák. Large scale parsing of Czech. In *Proceedings of Efficiency in Large-Scale Parsing Systems Workshop, COLING'2000*, pages 43–50, Saarbrucken: Universitaet des Saarlandes, 2000.
3. H. Bunt and A. Nijholt, editors. *Advances in Probabilistic and Other Parsing Technologies.* Kluwer Academic Publishers, 2000.
4. S. Caraballo and E. Charniak. New figures of merit for best-first probabilistic chart parsing. *Computational Linguistics*, 24(2):275–298, 1998.
5. M. Chitrao and R. Grishman. Statistical parsing of messages. In *Proceedings of the Speech and Natural Language Workshop*, pages 263–266, Hidden Valley, PA, 1990.
6. R. J. Bobrow. Statistical agenda parsing. In *Proceedings of the February 1991 DARPA Speech and Natural Language Workshop*, pages 222–224. San Mateo: Morgan Kaufmann, 1991.
7. P. Smrž and A. Horák. Implementation of efficient and portable parser for Czech. In *Text, Speech and Dialogue: Proceedings of the Second International Workshop TSD'1999*, Pilsen, Czech Republic, 1999. Springer Verlag, Lecture Notes in Computer Science, Volume 1692.
8. J. T. Maxwell III and R. M. Kaplan. The interface between phrasal and functional constraints. In M. Rosner, C. J. Rupp, and R. Johnson, editors, *Proceedings of the Workshop on Constraint Propagation, Linguistic Description, and Computation*, pages 105–120. Instituto Dalle Molle IDSIA, Lugano, 1991. Also in Computational Linguistics, Vol. 19, No. 4, 571–590, 1994.
9. K. Pala and P. Ševeček. Valencies of Czech verbs. In *Proceedings of Works of Philosophical Faculty at the University of Brno*, pages 41–54. Brno, 1997. (in Czech).
10. P. Sgall, E. Hajičová, and J. Panevová. *The Meaning of the Sentence and Its Semantic and Pragmatic Aspects.* Academia/Reidel Publishing Company, Prague, Czech Republic/Dordrecht, Netherlands, 1986.
11. J. Hajič. Building a syntactically annotated corpus: The Prague Dependency Treebank. In *Issues of Valency and Meaning*, pages 106–132, Prague, 1998. Karolinum.
12. R. C. Moore. Time as a measure of parsing efficiency. In *Proceedings of Efficiency in Large-Scale Parsing Systems Workshop, COLING'2000*, pages 23–28, Saarbrucken: Universitaet des Saarlandes, 2000.