Punctuation Detection with Full Syntactic Parsing

Miloš Jakubíček and Aleš Horák

Faculty of Informatics
Masaryk University
Botanická 68a, 602 00 Brno
Czech Republic
xjakub@fi.muni.cz, hales@fi.muni.cz

Abstract. The correct placement of punctuation characters is in many languages, including Czech, driven by complex guidelines. Although those guidelines use information of morphology, syntax and semantics, state-of-art systems for punctuation detection and correction are limited to simple rule-based backbones. In this paper we present a syntax-based approach by utilizing the Czech parser `synt`. This parser uses an adapted chart parsing technique for building the chart structure for the sentence. synt can then process the chart and provide several kinds of output information. The implemented punctuation detection technique utilizes the synt output in the form of automatic and unambiguous extraction of optimal syntactic structures from the sentence (noun phrases, verb phrases, clauses, relative clauses or inserted clauses). Using this feature it is possible to obtain information about syntactic structures related to expected punctuation placement. We also present experiments proving that this method makes it possible to cover most syntactic phenomena needed for punctuation detection or correction.

1 Introduction

Incorrect usage of punctuation characters such as comma, semi-colon, or dash has always been a common mistake in written texts, especially in languages with such complex (and strict) guidelines for punctuation placement as in the case of the Czech language [1]. It has been shown that mistakes in punctuation (21 % of all errors) represent the second most frequent category of mistakes in Czech – the first position is occupied by stylistics with 23 % [2].

It is no wonder that automatic detection and correction of punctuation for Czech is still an open task and the state-of-art systems usually lack both precision and recall around 50 % [3], as they use only a small-to-medium set of rules matching the simplest guidelines for punctuation. It is obvious that such methods do not cover phenomena on syntactic or semantic levels and that such approach cannot be easily adapted to do so.

In this paper we focus on superseding the deficiencies of current punctuation detection systems with analysis on the syntactic level by using `synt`, a powerful

© A. Gelbukh (Ed.)
Special issue: Natural Language Processing and its Applications.
Research in Computing Science 46, 2010, pp. 335-343

Received 27/11/09
Accepted 16/01/10
Final version 10/03/10

and feature-rich syntactic parser for Czech (more on the parser in Section 2). The main reason of the technique is that since the parse is able to parse (i. e. *recognize*) punctuation in the input, it also might be able to fill in (i. e. *generate*) missing punctuation. We show that with relatively small set of post-processing rules this method achieves significantly better results than the current systems (as described in [3]).

The synt parser provides several options of output information based on the packed *chart structure* containing the resulting analyses. Besides standard enumeration of all phrasal or dependency trees, synt can compute the optimal decomposition of the input text to selected syntactic structures such as noun, verb or prepositional phrases, clauses etc. The extraction of structures (described in details in Section 3) allows us to obtain all the necessary syntactic information needed for filling in the punctuation into the given sentence.

The structure of this papers is as follows: in Sections 2 and 3 we briefly describe the synt parser and the extraction of structures, then in Section 4 we explain how we adapt it and use it for punctuation detection and finally we present results of our work in Section 5.

1.1 Related Work

Since the Czech rules for placing punctuation are so complicated, this topic has been addressed by several authors, however, with partial success only. There are two commercially available products which try to tackle this task *Grammaticon* [4] and *Grammar Checker* [5], which is included in the Czech localisation of MS Word text editor. A comparison of both systems has been made by Pala in 2005 [3] showing that both of them lack especially recall. A proof-of-concept system has been also shown in [6] trying to use Inductive Logic Programming to solve this task.

The main problem of current solutions is that they are designed as rather simple rule-based systems with a set of (hard-coded or inducted, context-free or context-sensitive) rules trying to describe the placement of Czech punctuation. Although many of the principles for placing punctuation have syntactic background, none of them applies full syntactic parsing for this task. In this paper we utilize the Czech parse synt and show that a syntax-based approach has promising results.

2 The synt Parser

The Czech parser synt [7,8] has been developed in the Natural Language Processing Centre at Masaryk University. It performs a head-driven chart-type syntactic analysis based on the provided context-free grammar with additional contextual tests and actions. For easy maintenance, this grammar is edited in the form of a metagrammar (having about 200 rules) from which the full grammar can be automatically derived (having almost 4,000 rules). The per-rule defined contextual actions are used to cover phenomena such as case-number-gender agreement.

In recent measures [9, p. 77] it has been shown that synt accomplishes a very good coverage (above 90 %) but the analysis is highly ambiguous: for some sentences even millions of output syntactic trees can occur. There are several strategies developed to cope with the ambiguity: first, the grammar rules are divided into different priority levels which are used to prune the resulting set of output trees. Second, every resulting chart edge has a ranking value assigned from which the ranking for the whole tree can be efficiently computed in order to sort the trees and output N best trees while keeping it in polynomial time.

The synt parser contains (besides the chart parsing algorithm itself) many additional functions such as maximum optimal coverage of partial analyses (shallow parsing) [10], effective selection of n-best output trees [11] or chart and trees beautification [12]. The punctuation detection technique uses the function of extraction of syntactic structures [13], which is described in detail in the next section.

3 Extraction of Phrase Structures

Usually, a derivation tree is presented as the main output of syntactic parsing of natural languages, but currently most of the syntactic analysers for Czech lack precision. However there are many situations in which it is not necessary and sometimes even not desirable to require such derivation trees as the output of syntactic parsing, may it be simple information extraction and retrieval, transformation of sentences into a predicate-arguments structure or any other case, in which we rather need to process whole syntactic structures in the given sentence, especially noun, prepositional and verb phrases, numerals or clauses. Moreover, so as not to deal with the same problems as with the tree parser output, we need to identify these structures unambiguously.

The phrase extraction functionality in synt enables us to obtain syntactic structures from the analysed sentence that correspond to a given grammar nonterminal in a number of ways. For the purpose of phrase extraction, the internal parsing structure of synt is used, the so called *chart*, a multigraph which is built up during the analysis holding all the resulting trees. An important feature of chart is its polynomial size [7, p. 133] implying that it is a structure suitable for further effective processing[1] – as the number of output trees can be exponential to the length of the input sentence, processing of each tree separately would be otherwise computationally infeasible.

However, as the algorithm works directly with chart and not trees, it is very fast and can be used for processing massive amount of data in a short time, even if we are extracting many structures at once (see Table 1 for details).

The output of the extraction is shown in the following two examples of extracting clauses:

[1] By processing the chart we refer to the result of the syntactic analysis, i.e. to the state of the chart after the analysis.

- **Example 1.**
  Input:
  Muž, který stojí u cesty, vede kolo.
  (A man who stands at the road leads a bike.)
  Output:
  ```
  [0-9): Muž , , vede kolo
  ```
  (a man leads a bike)
  ```
  [2-6): který stojí u cesty
  ```
  (who stands at the road)
- **Example 2.**
  Input:
  Vidím ženu, která drží růži, jež je červená.
  (I see a woman who holds a flower which is red.)
  Output:
  ```
  [0-3): Vidím ženu ,
  ```
  (I see a woman)
  ```
  [3-7): která drží růži ,
  ```
  (who holds a flower)
  ```
  [7-10): jež je červená
  ```
  (which is red)

4 Punctuation Detection

The main idea of using the syntactic parser with extraction of structures for punctuation detection is as follows: if the parser has anyway to expect (i. e. match and parse) the actual presence of the punctuation using its grammar rules, we may try to modify those grammar rules in such a way that enables[2] the punctuation detection even if the punctuation is (by mistake) not present in the sentence, and then extract the related (empty) punctuation nonterminals as described in

[2] by allowing empty productions in the grammar

Table 1. Results of detection on a sample set of 500 sentences.[†]

Step	Precision	Recall
Adding ϵ-rules	35.37 %	20.12 %
Further grammar modifications	69.43 %	57.31 %
Matching coordinations	**82.27 %**	**84.76 %**
Sentences	500	
Average time needed per sentence	0.65 s	

[†] The precision and recall were measured across the whole sentence set.

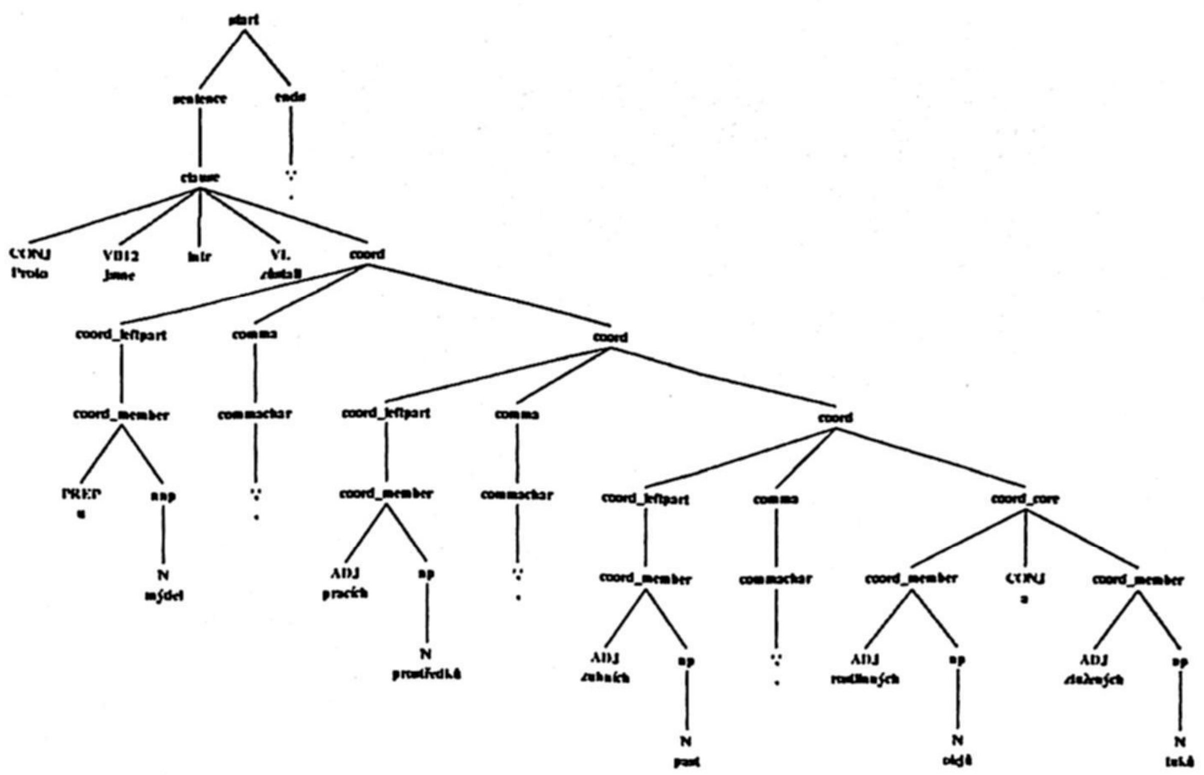


Fig. 1. Example tree for an input sentence *Proto jsme zůstali u mýdel, pracích prostředků, zubních past, rostlinných olejů a ztužených tuků* (Thus we continued with soaps, detergents, tooth pastes, vegetable oils and hardened fats).

the previous section. Since the extraction of the punctuation basically represents a projection from the chart structure to the sentence surface structure, the missing punctuation can be identified as a post-analysis step.

The difference between parsing a present punctuation mark and detecting a missing punctuation is displayed in Figures 1 and 2: while the first one represents a regular derivation tree for a sentence containing punctuation, the other one shows an almost identical derivation tree that, however, was produced from a sentence in which no punctuation (except the trailing full stop) was present, but the parser still deduced the correct placement of comma nonterminals. The whole process of punctuation detection is then demonstrated in Figure 3.

As the first step, we have modified the relevant grammar punctuation rules to allow empty productions.[3] Even this trivial change in the grammar led to a recall of $< 20\,\%$ (see Table 1 for details), therefore we further analysed the grammar rules and improved them in order to better fit the purpose of punctuation detection.

The grammar modifications mainly focused on improving the ability to parse (and hence also detect) punctuation in common Czech sentences (especially be-

[3] I. e. for each rule covering some punctuation marks, e. g. `comma` $\rightarrow$ `","`, we added a rule allowing the empty production: `comma` $\rightarrow \epsilon$.

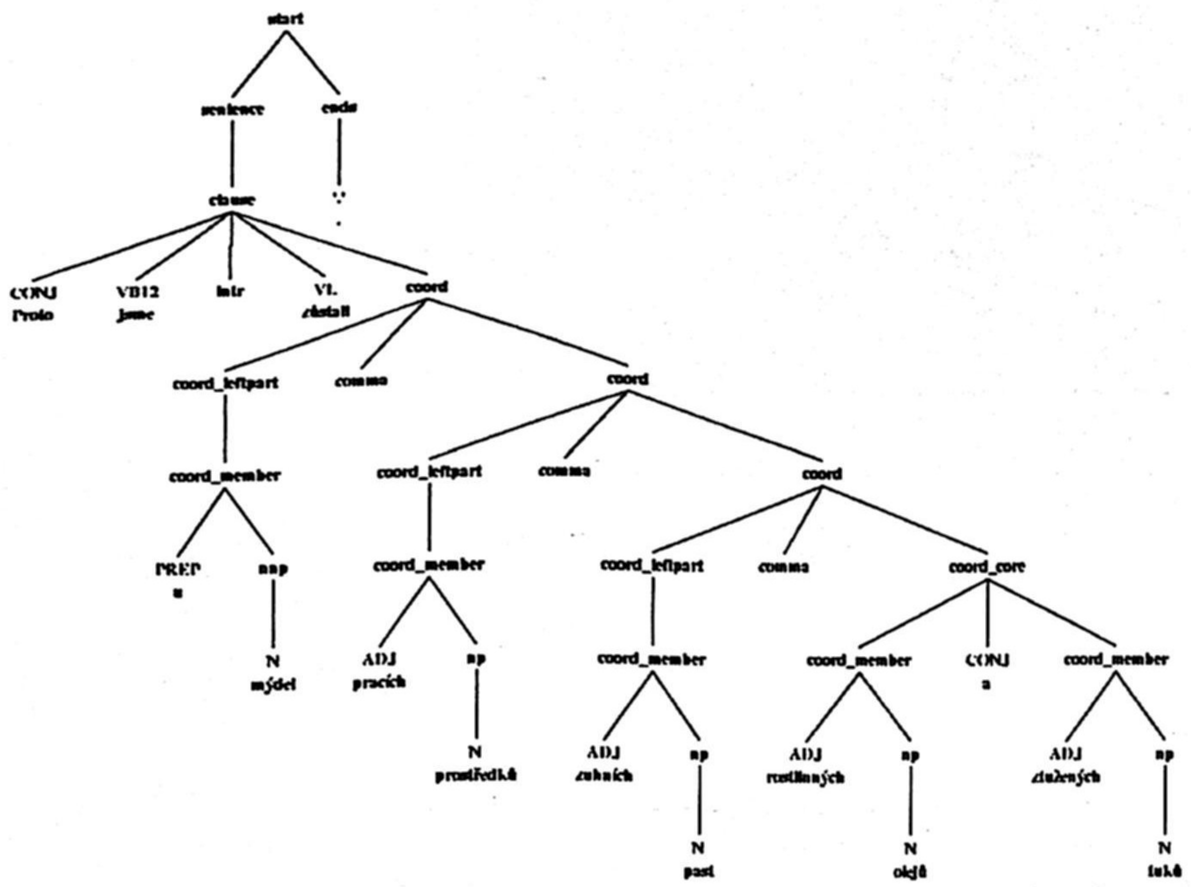


Fig. 2. Example tree for the same sentence from which punctuation has been removed.

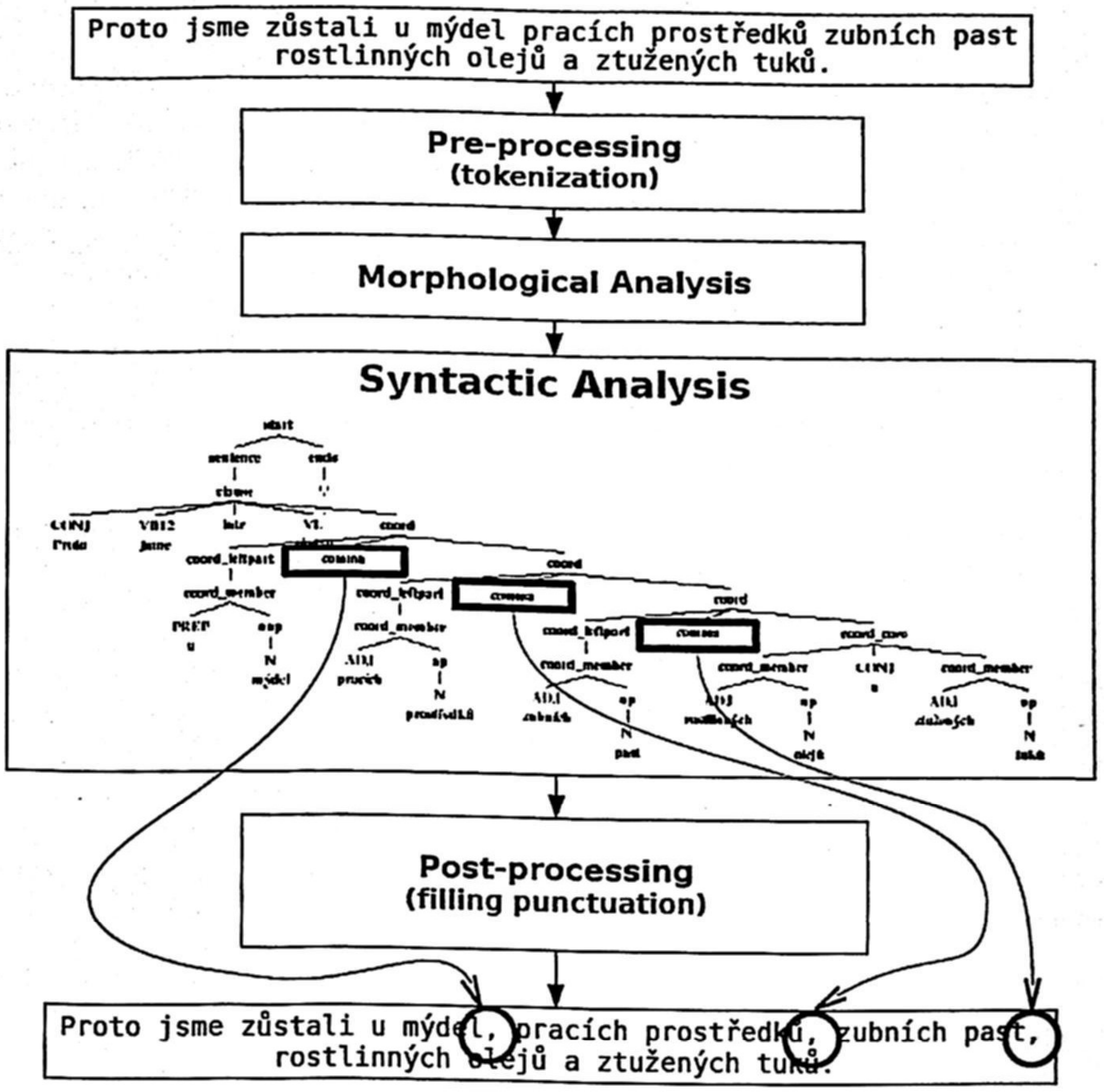
tween clauses, relative clauses and conjunctions), the recall increased to almost 60 %.

Finally we paid special attention to coordinations of phrases where punctuation plays an important role.[4] Detecting correct placement of punctuation marks in coordinations required new grammar rules to distinguish coordinations (in which all members have to agree in grammatical case) from common noun or prepositional phrases (where no such grammatical restrictions are applied) as shown in the example derivation tree with coordinations (Figure 2). The resulting recall as well as precision increased to more than 80 %.

4.1 Evaluation and Results

For the purpose of evaluation of the described method, 500 randomly chosen sentences from the manually annotated DESAM corpus [14] have been used. Firstly, we removed all punctuation marks from the given sentence, then we filled in the punctuation using the enhanced parser and finally we compared

[4] In general, the punctuation here distinguishes the meaning of the coordinations. This however requires semantic information (see Section 4.2), therefore we concentrated to syntactic phenomena in coordinations.



Proto jsme zůstali u mýdel pracích prostředků zubních past rostlinných olejů a ztužených tuků.
Pre-processing (tokenization)
Morphological Analysis
Syntactic Analysis
Post-processing (filling punctuation)
Proto jsme zůstali u mýdel, pracích prostředků, zubních past, rostlinných olejů a ztužených tuků.

Fig. 3. Overview of the punctuation detection process.

it with the original sentence. The presented results confirm the importance of coordinations for solving this problem, because they contain many punctuation characters as is indicated by the increase in recall by almost 30 %. Also, as can be seen from the value of precision, the coordinations can be precisely analysed on the syntactic level.

4.2 Problems and Limitations

As mentioned above, the guidelines for placing punctuation in Czech take information not only from morphology and syntax, but also from semantics – such phenomena cannot be covered by a pure syntax-based system. Using semantic information for punctuation detection is necessary in order to distinguish e. g.:

- *coordinations from gradually extending attributes*
  In Czech, coordinations are written with punctuation, but gradually extending attributes without it, e.g. *Je to velký starý drahý dům* (It is a big old expensive house), but *Vidím velký, střední a malý dům* (I see a large, medium and small house) Currently, we consider a noun phrase sequence to be a coordination only if it contains a coordination core as its part, consisting from a phrase followed by a conjunction and another phrase[5], i.e. *střední a malý* (medium and small) in the previous sentence. However, the coordination core does not need to be present in some (stylistically determined) situations.
- *sequences of subordinating clauses*
  If considering the sentence *Petr si uvědomil, že Pavel zavřel okno[,] a šel domů* (Peter realized that Paul closed the window[,] and went home), the presence of the comma determines whether it was Peter or Paul who went home.

Obviously, for proper detection of comma placement in the first example, it would be necessary to have the knowledge of the meaning of the sentences or at least of some parts of the sentence – moreover, we would need to know the relation between the meanings: to know that *large, medium, small* belong to the same semantic class while *big, old, expensive* does not. It turns out that to tackle the problem, large ontologies for Czech would be needed which are unfortunately not available.

The situation in the second example is even worse: we would need large context, anaphora resolution and logical inference to deduce the proper placement of the punctuation and even then, the situation might be just ambiguous from the available information.

5 Conclusions and Future Directions

In this paper we have presented an efficient method for punctuation detection in common sentences by employing full syntactic parsing. Although all measures were related to Czech, we believe that the technique might be easily generalized for other languages with complicated punctuation rules provided that they have the necessary language resources and tools.

The proposed method has already achieved significantly better results than the state-of-art systems and we believe that it might be further improved by supplying additional information which could overcome its current limitations described in the previous section. In particular, we plan to use the Czech WordNet [15] to distinguish coordinations and gradually extending attributes, and search for other semantic resources that may help us by improving this method.

Furthermore, an application consisting of an OpenOffice.org [16] grammar checker extension is being developed for practical testing of the punctuation detection technique and for making it available to a broad scope of users.

[5] Both phrases must also agree in case.

Acknowledgments

This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and in the National Research Programme II project 2C06009 and by the Czech Science Foundation under the project P401/10/0792.

References

1. Hlavsa, Z., et al.: Akademická pravidla českého pravopisu (Rules of Czech Orthography). Akademia, Praha (1993)
2. Pala, K., Rychlý, P., Smrž, P.: Text corpus with errors. In: Text, Speech and Dialogue: Sixth International Conference, Berlin, Springer Verlag (2003) 90–97
3. Pala, K.: Pište dopisy konečně bez chyb – Český gramatický korektor pro Microsoft Office. Computer **13–14** (2005) 72 CPress Media.
4. Lingea s. r. o.: Grammaticon. `http://www.lingea.cz/grammaticon.htm` (2003)
5. Oliva, K., Petkevič, V., Microsoft s.r.o.: Czech grammatical checker. `http://office.microsoft.com/word` (2005)
6. Pala, K., Nepil, M.: Checking punctuation errors in czech texts. [online, quoted on 2009-11-20]. Available from: `http://nlp.fi.muni.cz/publications/neco2003\_pala\_nepil/neco2003\_pala\_nepil.rtf` (2003)
7. Horák, A.: The Normal Translation Algorithm in Transparent Intensional Logic for Czech. PhD thesis, Faculty of Informatics, Masaryk University, Brno (November 2001)
8. Kadlec, V., Horák, A.: New Meta-grammar Constructs in Czech Language Parser synt. In: Lecture Notes in Computer Science, Springer Berlin / Heidelberg (2005)
9. Kadlec, V.: Syntactic Analysis of Natural Languages Based on Context-Free Grammar Backbone. PhD thesis, Faculty of Informatics, Masaryk University, Brno (September 2007)
10. Ailomaa, M., Kadlec, V., Rajman, M., Chappelier, J.C.: Robust stochastic parsing: Comparing and combining two approaches for processing extra-grammatical sentences. In Werner, S., ed.: Proceedings of the 15th NODALIDA Conference, Joensuu 2005, Joensuu, Ling@JoY (2005) 1–7
11. Kovář, V., Horák, A., Kadlec, V.: New Methods for Pruning and Ordering of Syntax Parsing Trees. In Proceedings of Text, Speech and Dialogue 2008. In: Lecture Notes in Artificial Intelligence, Proceedings of Text, Speech and Dialogue 2008, Brno, Czech Republic, Springer-Verlag (2008) 125–131
12. Kovář, V., Horák, A.: Reducing the Number of Resulting Parsing Trees for the Czech Language Using the Beautified Chart Method. In: Proceedings of 3rd Language and Technology Conference, Poznań, Wydawnictwo Poznańskie (2007) 433–437
13. Jakubíček, M.: Extraction of syntactic structures based on the czech parser synt. In: Proceedings of Recent Advances in Slavonic Natural Language Processing 2008, Brno, Czech Republic, Masaryk University (2008) 56–62
14. Pala, K., Rychlý, P., Smrž, P.: DESAM – Annotated Corpus for Czech. In: Proceedings of SOFSEM '97, Springer-Verlag (1997) 523–530
15. Pala, K., Hlaváčková, D.: Derivational Relations in Czech WordNet. In: Proceedings of the Workshop on Balto-Slavonic Natural Language Processing 2007, Praha, Czech Republic, The Association for Computational Linguistics (2007) 75–81
16. OpenOffice.org Community: Openoffice.org. `http://www.openoffice.org` (2009)