Discovering Grammatical Relations in Czech Sentences

Aleš Horák and Pavel Rychlý

Natural Language Processing Centre, Faculty of Informatics, Masaryk University
{hales,pary}@fi.muni.cz

Abstract. The syntactic parser synt developed at NLP Centre, Faculty of Informatics, Masaryk University, can provide as one of its possible outputs a list of dependency relations discovered in the analysed sentence. In the paper, we present the result of codification and translation of the (rather technically labeled) dependency relations from synt to linguistically significant relations.
The resulting relations are demonstrated by means of Word Sketches (WS), where the new relations are compared with traditional WS relations from WS grammar.

Key words: syntactic analysis; Word Sketches; dependency; grammatical relations

1 Introduction

Syntactic parsing techniques provide various kinds of information, where the most frequent possibility is a list of syntactic trees. The trees are expressed in the respective formalism being it a dependency tree [1], a phrasal tree [2,3] or a phrase structure tree [4].

The `synt` parser internally works with phrasal trees (in the form of packed shared forest – chart), but it is able to build a dependency graph using specific dependency actions in its meta-grammar.

In this paper we discuss the work of using the dependency relations obtained by synt to build new Word Sketches over new big Czech corpus named CZES. We have used two different systems for the dependency relations discovery – the standard Sketch Grammar approach based on regular expressions, and dependency relations obtained by means of full syntax parsing of Czech. We give a detailed description of the various features of the Sketch Engine in relation to the Czech language.

2 The Sketch Engine

The Sketch Engine is a sophisticated corpus query system. In addition to the standard corpus query functions such as concordances, sorting, filtering,

Petr Sojka, Aleš Horák (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2009, pp. 5–??, 2009.

it provides *word sketches*, one page summaries of a word's grammatical and collocational behaviour by integrating grammatical analysis.[1]

Based on the grammatical analysis, the Sketch Engine also produces a distributional *thesaurus* for the language, in which words occurring in similar settings, sharing the same collocates, are put together, and *sketch differences*, which specify similarities and differences between near-synonyms. The system is implemented in C++ and Python and designed for use over the web.

Once the corpus is loaded into the Sketch Engine, the concordance functions are available. The lexicographer can immediately use the search boxes provided, searching, for example, for a lemma specifying its part of speech.

We must note here that the quality of the output of the system depends heavily on the input, i.e. the quality of tagging and lemmatization is not always satisfactory. According to the sources of some parts of the CZES corpus, the texts can contain misspelled words and neologism, which are tagged by the *guesser* module of the tagger.

On the results page the concordances are shown using KWIC view. With VIEW options it is possible to change the concordance view to a number of alternative views. One is to view additional attributes such as POS tags or lemma alongside each word.

3 Word Sketches and the CZES corpus

Word sketches are the distinctive feature of the Sketch Engine. Word sketches are one-page automatic, corpus-based summaries of a word's grammatical and collocational behaviour. Word sketches improve on standard collocation lists by using a grammar and parser to find collocates in specific grammatical relations, and then producing one list of subjects, one of objects, etc. rather than a single grammatically blind list.

In order to identify a word's grammatical and collocational behaviour, the Sketch Engine needs to know how to find words connected by a grammatical relation. For this to work, the input corpus needs to be parsed or at least POS tagged.

If the corpus is parsed, the information about grammatical relations between words is already embedded in the corpus and the Sketch Engine can use this information directly. A modification of this method was used to handle output of a syntactic parser. If the corpus is POS-tagged but not parsed, grammatical relations can be defined by the developer within the Sketch Engine using a Sketch Grammar.

---

[1] The Sketch Engine prefers input which has already been lemmatized and POS tagged. If no lemmatized input is available it is possible to apply the Sketch Engine to word forms which, while not optimal, will still be a useful lexicographic tool.

3.1 Czech Sketch Grammar

In this model, grammatical relations are defined as regular expressions over POS-tags. For example, a grammatical relation specifying the relation between a noun and a pre-modifying adjective looks like this.

```
=modifier
2:"A.*" 1:"N.*"
```

The first line, following the `=`, gives the name of this grammatical relation. The `1:` and `2:` mark the words to be extracted as first argument (the keyword) and second argument (the collocate).

The result is a regular expression grammar which we call a Sketch Grammar. It allows the system to automatically identify possible relations of words to the keyword. These grammars are of course less than perfect, but given the errors in the POS-tagging, this is inevitable however good the grammar. The problem of noise is mitigated by the statistical filtering which is central to the preparation of word sketches.

The first version of the Czech Sketch Grammar was created in the early stage of the Sketch Engine development [5]. It was prepared for the "Prague" tag-set used in the Czech National Corpus. We have adopted the grammar to match the Brno annotation.

When the corpus is parsed with the grammar, the output is a set of tuples, one for each case where each pattern matched. The tuples comprise (for the two-argument case), the grammatical relation, the headword, and the collocate, as in the third column in the table. This work is all done on lemmas, not word forms, so headword and collocate are lemmas.

The Czech Sketch Grammar generates about 46 million triples (dependences) from the 85 million token corpus.

3.2 Dependency Relations from Syntactic Parser

The Czech syntactic parser `synt` [2,6] is developed in the Natural Language Processing Centre at Masaryk University. The parsing system uses an efficient variant of the head driven chart parsing algorithm [7] together with the meta-grammar formalism for the language model specification. The advantage of the meta-grammar concept is that the grammar is transparent and easily maintainable by human linguistic experts. The meta-grammar includes about 200 rules covering both the context-free part as well as context relations. Contextual phenomena (such as case-number-gender agreement) are covered using the per-rule defined contextual actions. The meta-grammar serves as a basis for a machine-parsable grammar format used by the actual parsing algorithm – this grammar form contains almost 4,000 rules.

Currently, the `synt` system offers a coverage of more than 92 percent of (common) Czech sentences[2] while keeping the analysis time on the average of 0.07s/sentence.

[2] measured on 10,000 sentences from the DESAM corpus [8].

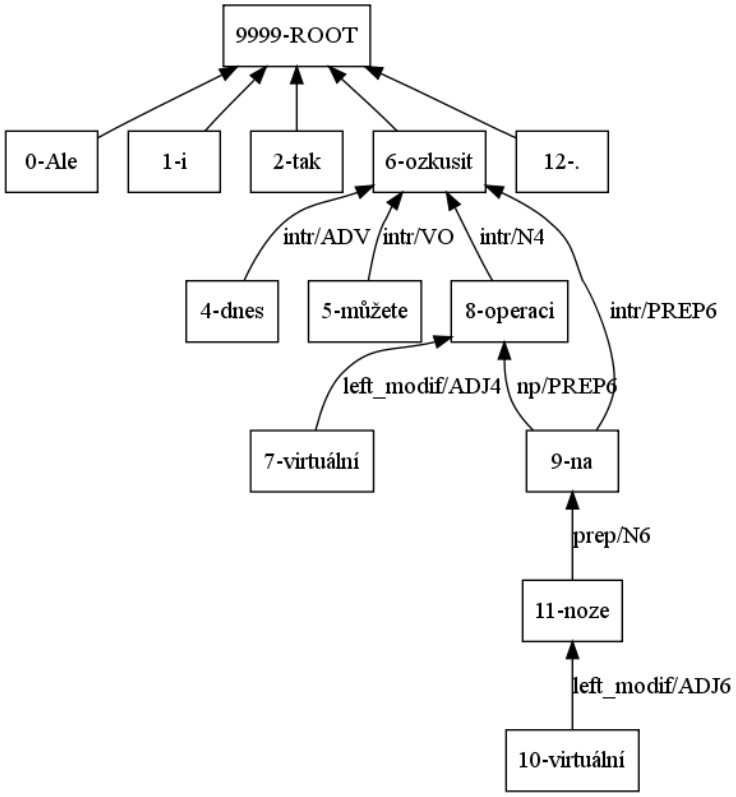

Fig. 1. An example of `synt` dependency graph output for the sentence "*Ale i tak už dnes můžete ozkusit virtuální operaci na virtuální noze.*" (Even so you can today try virtual operation on a virtual leg).

Besides the standard results of the chart parsing algorithm, `synt` offers additional functions such as partial analysis (shallow parsing) [9], effective selection of *n*-best output trees [7], chart and trees linguistic simplification [10], or extraction of syntactic structures [11]. All these functions use the internal chart structure which allows to process potentially exponential number of standard derivation trees still in polynomial time.

Apart from the common generative constructs, the metagrammar includes feature tagging actions that specify certain local aspects of the denoted (non-)terminal. One of these actions is the specification of the head-dependent relations in the rule—the `depends()` construct:

```
/* černá kočka (black cat) */
np → left_modif np
    depends($2,$1)
/* třeba (perhaps) */
```

```
part → PART
    depends(root,$1)
```

In the first rule, `depends($2,$1)` says that (the head of) the group under the `left_modif` non-terminal depends on (the head of) the `np` group on the right hand side. In the second example, `depends(root,$1)` links the `PART` terminal to the root of the resulting dependency tree. The meta-grammar allows to assign *labels* to parts of derivation tree, which can be used to specify dependencies "crossing" the phrasal boundaries. The `synt` system thus allows to process even *non-projective phenomena*, which would otherwise be problematic within a purely phrasal approach.

The relational `depends` actions sequentially build a graph of dependency links between surface tokens. Each call of the action adds a new edge to the graph with the following information about the *dependent* group:

1. the non-terminal at the top of the group (`left_modif` or `np` in the example above),
2. the pre-terminal (word/token category) of the *head* of the group, i.e. the single token representing the group, and
3. the grammatical case of the head/group, if applicable.

An example list of such dependency relations for a corpus sentence "*Ale i tak už dnes můžete ozkusit virtuální operaci na virtuální noze.*" (Even so you can today try virtual operation on a virtual leg) may look like this:

from	*label*	*to*	*from*	*label*	*to*
0	part/PART	2	5	intr/VO3	6
1	part/PART	2	2	intr/ADV	6
7	left_modif/ADJ4	8	4	intr/ADV	6
10	left_modif/ADJ6	11	8	intr/N4	6
11	prep/N6	9	9	intr/PREP6	6
9	np/PREP6	8			

The corresponding dependency graph of this sentence is depicted in Figure 1.

We can see that the information in these relations contains more details that come from the parsing process. However, not all details bring the same amount of linguistic adequacy – e.g. distinguishing `left_modif/ADJ4` and `left_modif/ADJ6` does not bring any new information,[3] whereas `intr/N1` links to verbs where the dependent group is a subject and `intr/N4` lists objects in accusative.

Within the experiment of parsing the CZES corpus (about 4 million sentences), we have obtained more than 52 millions of dependency relations, out of which about 4 thousands were distinct relations in one direction. We have provided translation and simplification for the obtained relations in both directions (e.g. `left_modif/ADJ6` and `Rleft_modif/ADJ6`) with the resulting names corresponding to linguistically adequate terms like `subj` or `obj4`. An example of the resulting Word Sketch is displayed in Figure 2.

[3] It just says that the collocation *adjective+noun* was in accusative or locative.

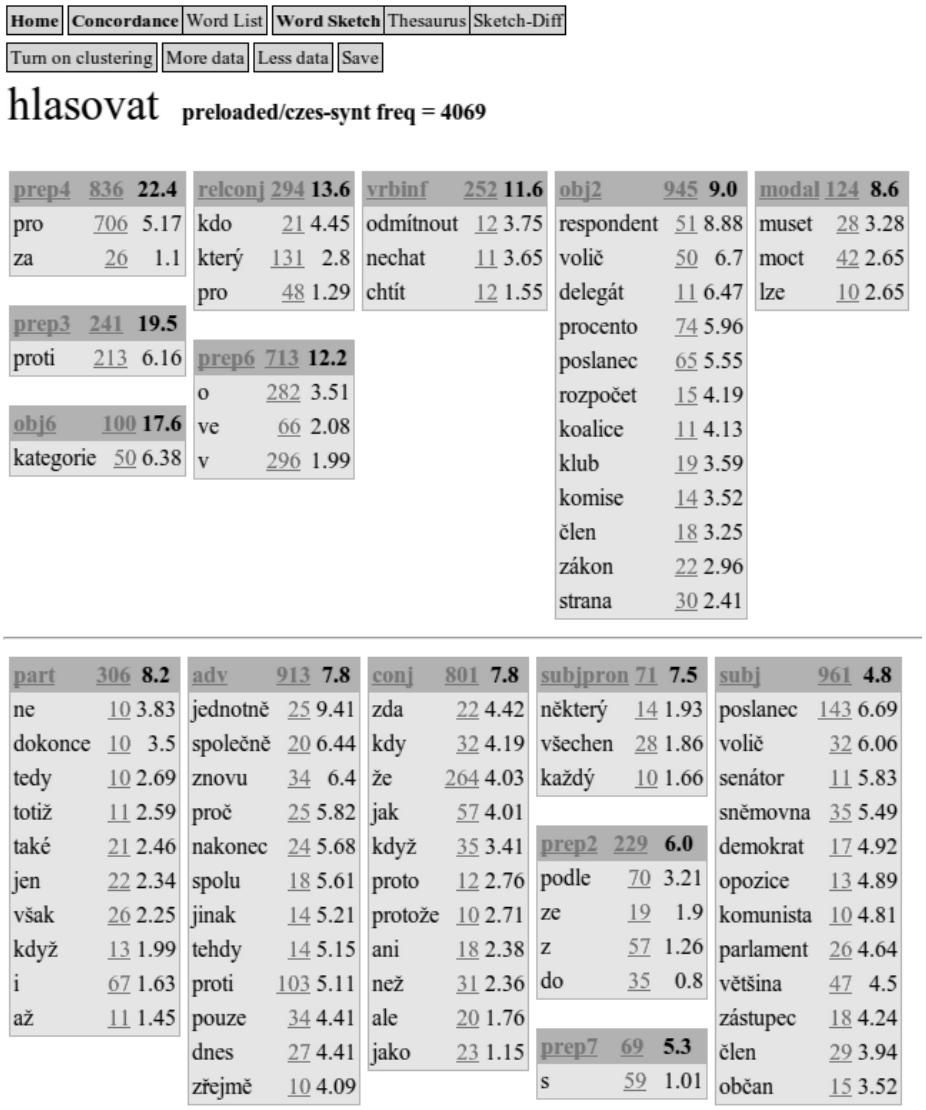

Fig. 2. Word sketch for the word "hlasovat" (to poll).

3.3 Thesaurus

Once the corpus has been parsed and the tuples extracted, we have a very rich database that can be used in a variety of ways.

We can ask "which words share most tuples", in the sense that, if the database includes both $\langle gramrel, w_1, w\rangle$ and $\langle gramrel, w_2, w\rangle$ (for example $\langle$subj, hlasovat, poslanec$\rangle$ and $\langle$subj, rozhodovat, poslanec$\rangle$), then we can say that w_1 and w_2 share a triple. A shared triple is a small piece of evidence that two words are similar. Now, if we go through the whole lexicon, asking, for each pair of words, how many triples do they share, we can build a 'distributional thesauruses', which, for each word, lists the words most similar to it (in an approach pioneered in [12,13]). The Sketch Engine computes such a thesaurus. A thesaurus entry for *hlasovat* obtained from the standard Sketch Grammar starts with:[4]

- vyslovit (pronounce), učinit (make), vyjádřit (express), schválit (authorize), podpořit (support)
- rozhodovat (decide), volit (vote), zvolit (select)
- zasedat (sit), shodnout (agree), zasednout (sit)
- diskutovat (discuss), sejít (meet), vystupovat (stand out)
- zastávat (perform)

The same thesaurus entry computed with the dependency relations obtained from syntactic parsing looks like:

- diskutovat (discuss), shodnout (agree)
- rozhodovat (decide), souhlasit (consent), usilovat (aspire), uvažovat (consider), přistoupit (accede), prosadit (enforce)
- volit (vote)
- uspět (succeed), odejít (leave), sedět (sit)
- vyslovit (pronounce)
- kandidovat (stand)
- sáhnout (clutch)

The main synonym *hlasovat* stays the same, but other similar words are grouped in different order. Evaluation of these two approaches, however, needs further studies from both grammarian and lexicographer's point of view.

4 Conclusion

We have presented the use of dependency relations obtained by full syntax analysis for building the list of Word Sketch relations and for the construction of automatic thesaurus.

Within the future work, the resulting different linguistic presentation of corpus Word Sketches will be evaluated by linguistic experts.

[4] The words are grouped according to the thesaurus score.

Acknowledgements

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 407/07/0679.

References

1. McDonald, R.: Discriminative learning and spanning tree algorithms for dependency parsing. Ph.D. thesis, University of Pennsylvania (2006).
2. 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.
3. Horák, A., Holan, T., Kadlec, V., Kovář, V.: Dependency and Phrasal Parsers of the Czech Language: A Comparison. In: Proceedings of the 10th International Conference on Text, Speech and Dialogue, Pilsen, Czech Republic (2007) accepted for publication.
4. Torisawa, K., Nishida, K., Miyao, Y., Tsujii, J.: An HPSG parser with CFG filtering. Natural Language Engineering **6**(01) (2000) 63–80.
5. Kilgarriff, A., Rychlý, P., Smrž, P., Tugwell, D.: The Sketch Engine. Proceedings of Euralex (2004) 105–116 `http://www.sketchengine.co.uk`.
6. Horák, A.: The Normal Translation Algorithm in Transparent Intensional Logic for Czech. Ph.D. thesis, Masaryk University (2002).
7. Horák, A., Kadlec, V., Smrž, P.: Enhancing Best Analysis Selection and Parser Comparison. In: Lecture Notes in Artificial Intelligence, Proceedings of TSD 2002, Brno, Czech Republic, Springer Verlag (2002) 461–467.
8. Pala, K., Rychlý, P., Smrž, P.: DESAM – Annotated Corpus for Czech. In: Proceedings of SOFSEM '97, Springer-Verlag (1997) 523–530.
9. 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.
10. Kovář, V., Horák, A.: Reducing the Number of Resulting Parsing Trees for the Czech Language Using the Beautified Chart Method. In: Proceedings of the 3rd Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics, Poznan, Poland (2007) 433–437.
11. Jakubíček, M., Horák, A., Kovář, V.: Mining Phrases from Syntactic Analysis. In: Proceedings of TSD 2009, Springer-Verlag (2009).
12. Grefenstette, G.: Explorations in automatic thesaurus discovery. Springer (1994).
13. Lin, D.: Automatic Retrieval and Clustering of Similar Words. In: Conference on Computational Linguistics (COLING-ACL). (1998) 768–774.