RASLAN 2009
Recent Advances in Slavonic Natural Language Processing

muni
PRESS

Masaryk University
`http://nlp.fi.muni.cz/raslan/2009/`

P. Sojka, A. Horák (Eds.)

RASLAN 2009

Recent Advances in Slavonic Natural Language Processing

Third Workshop on Recent Advances in Slavonic Natural Language Processing, RASLAN 2009
Karlova Studánka, Czech Republic,
December 4–6, 2009
Proceedings

Masaryk University
Brno 2009

Proceedings Editors

Petr Sojka
Faculty of Informatics, Masaryk University
Department of Computer Graphics and Design
Botanická 68a
CZ-602 00 Brno, Czech Republic
Email: sojka@fi.muni.cz

Aleš Horák
Faculty of Informatics, Masaryk University
Department of Information Technologies
Botanická 68a
CZ-602 00 Brno, Czech Republic
Email: hales@fi.muni.cz

Katalogizace v knize – Národní knihovna ČR

RASLAN 2009 (3. : Karlova Studánka, Česko)

RASLAN 2009 : Recent Advances in Slavonic Natural Language Processing :
third workshop on ..., Karlova Studánka, Czech Republic,
December 4-6, 2009 : proceeedings / P. Sojka, A. Horák (eds.).
- 1st ed. - Brno : Masaryk University, 2009. - VIII+135 s.

ISBN 978-80-210-5048-8

81’322 * 004.82/.83:81’322.2
- computational linguistics
- natural language processing
- proceedings of conferences
- počítačová lingvistika
- zpracování přirozeného jazyka
- sborníky konferencí

410 - Linguistics [11]
006.3 - Artificial intelligence [23]
81 - Lingvistika. Jazyky [11]
004.8 - Umělá inteligence [23]

ISBN 978-80-210-5048-8

Preface

This volume contains the Proceedings of the RASLAN (RASLAN 2009), co-organized by the the Center of Natural Language Processing at the Faculty of Informatics, Masaryk University and held on December 4th–6th 2009 in Karlova Studánka, Sporthotel Kurzovní, Jeseníky, Czech Republic.

The RASLAN Workshop is an event dedicated to exchange of information between research teams working on the projects of computer processing of Slavonic languages and related areas going on in the Centre. RASLAN is focused on theoretical as well as technical aspects of the project work, presentations of verified methods are welcomed together with descriptions of development trends. The workshop also serves as a place for discussion about new ideas. The intention is to have it as a forum for presentation and discussion of the latest developments in the the field of language engineering, especially for undergraduates and postgraduates affiliated to the NLP Center at FI MU.

Topics of the Workshop include (but are not limited to):

* text corpora and tagging
* syntactic parsing
* sense disambiguation
* machine translation, computer lexicography
* semantic networks and ontologies
* semantic web
* knowledge representation
* applied systems and software for NLP

RASLAN 2009 offers a rich program of presentations, short talks, technical papers and mainly discussions. A total of 17 papers were accepted, contributed altogether by 24 authors. Our thanks go to the Program Committee members and we would also like to express our appreciation to all the members of the Organizing Committee for their tireless efforts in organizing the Workshop and ensuring its smooth running. In particular, we would like to mention the work of Pavel Rychlý, Aleš Horák and Dana Hlaváčková. The TEXpertise of Petr Sojka resulted in the speedy and efficient production of the volume which you are now holding in your hands. Last but not least, the cooperation of Masaryk University as publisher of these proceedings, and of tribun.eu as printer is gratefully acknowledged.

Brno, December 2009 Karel Pala

Table of Contents

Part I

Morphological Analysis

Morphology-Aware Spell-Checking Dictionary for Esperanto

Marek Blahuš

Faculty of Informatics, Masaryk University
xblah@fi.muni.cz

Abstract. The article describes the process of constructing a spell checker for the Esperanto language and its implementation as a dictionary (i.e. an affix file and a word list) for the Hunspell spell-checking engine. In comparison to existing solutions, the chosen approach takes note of morphologically complex words, which are common in Esperanto due to its agglutinative nature, and applies a set of rules describing allowed morpheme compounds, along with semantic classification of all involved word roots. The result has been tested with a user community and is presently being incorporated into the *OpenOffice.org* office suite.

Key words: morphology; spelling; spell-checker; Esperanto

1 Introduction

The ease of electronic publishing is having a negative influence on the overall quality of texts, and the lack of accurate proofreading has had en especially grave impact on minority languages such as Esperanto [1]. Automated spell checking plays an essential role in helping the user produce quality texts.

There are several spell checking dictionaries for Esperanto [2], the most universal of which is that by Pokrovskij [3]. His solution, however, takes little note of Esperanto's rich morphology and is thus unable to recognize valid compounds such as "kaf·o·muel·il·o" ("coffee grinding machine") or "mal·sam·ras·an·oj" ("members of a different race").

In this paper, we describe a new spell checking dictionary for Esperanto, originally developed as a Bachelor thesis in the Natural Language Processing Centre at the Masaryk University [2]. Unlike the existing solution, it assigns each morpheme in the word list a set of semantic attributes and uses those in rules describing even complex Esperanto compounds. This has been made possible by the use of Hunspell [4], a modern spell-checking framework. Integration of the new dictionary in *OpenOffice.org* is also briefly discussed.

2 Esperanto Morphology

Esperanto has an agglutinative morphology[1] based on *roots* and *lexical and grammatical affixes*.[2] The order of affixes around a root is important, since affixes modify the entire stem they are attached to. By means of compounding, stems may be joined together, either directly or with an *epenthetic* vowel.[3] Most words require at least one grammatical suffix on their end, by means of which part of speech and grammatical categories are expressed, but there are some roots that may lack it, such as the numeral "kvar" ("four"). Thus, structure of an Esperanto word may be described by the following regular expression:

$$(LexAfx^{*} \cdot Root \cdot LexAfx^{*} \cdot Epent?)^{*} \cdot LexAfx^{*} \cdot Root \cdot LexAfx^{*} \cdot GramAfx?$$

Evidently, not all strings matched by this expression are existing Esperanto words. The two following sections describe an attempt at eliminating the unexisting words. This is particularly important, as failing to do so would cause a significant drop in the spell checker's recall, since they often coincide with misspelings of existing words.

3 Word List

It has been suggested by prominent Esperanto linguists that every root has an inherent meaning, and that roots maybe grouped in classes according to their semantic characteristics. Wennergren [5, chapter 37.1] listed several such classes, such as *people, tools*, or *activities*, along with a couple of sample roots for each of them. He also pointed out that class membership of a root may have an influence on the set of possible word forming processes it can enter. This directly affects the productivity of affixes, as he indicates for instance in his description of the prefix "bo-" (parallel to the English suffix "-in-law") by stating that it may be used only with roots expressing family relationships.

Inspired by the classification sketched by Wennergren, we analyzed his descriptions of the behavior of all the 10 prefixes and 31 suffixes, and as to be able to fulfil the root class conditions imposed by each of them, we inferred a system which encompasses a total of 15 classes. They are shown in Table 1. Each root may member in a number of classes, but some classes are mutually exclusive (such as A, I and O) and some classes are actually subclasses of others (e.g. $F \subset P \subset O$). Altogether, there are 85 possible membership combinations.

Later on, we extracted 16,780 Esperanto roots from the electronic version of the PIV dictionary [6] and designed a system for their automatic semantic classification which determines the membership of each root in each class.

[1] With the exception of suffixes *-ĉj-* and *-nj-* used in affectionate forms of proper names and family relationships, whose presence has a truncating effect on the root, e.g. "patro" ("father") → "paĉjo" ("daddy").

[2] A constructed language, Esperanto has been designed so to decrease its user's memory load – by featuring affixes such as the prefix "mal-" for antonyms, e.g. "pez·a" ("heavy") → "mal·pez·a" ("light").

[3] This is being done due to euphony or if the inherent part of speech of the preceding root needs to be changed. Grammatical affixes *-o-*, *-a-*, *-i-* or *-e-* are used as such a link. See the compounds in Introduction for example.

Table 1. Semantic classification of roots

class	description
A	attribute roots, having the **a**-ending in their base word form
B	animals ("**b**estoj" in Esperanto)
C	**c**ommon gender in animals and persons
F	**f**emale gender in persons
I	action roots, having the **i**-ending in their base word form
J	place roots, producing adverbs of spatial meaning ("e**j**oj" means "places")
K	plants ("**k**reskaĵoj" in Esperanto)
L	antonym-producing roots, which accept the prefix "ma**l**-"
M	**m**ale gender in animals and persons
N	**n**umbers (numerals and several other roots expressing amount)
O	object roots, having the **o**-ending in their base word form
P	**p**ersons
T	**t**ransitive roots, producing transitive verbs
V	words which may appear without a grammatical suffix ("**v**ortetoj" means "little words")
Y	famil**y** relationships

Various linguistic resources such as corpora, specialized vocabularies and closed categories word lists are used in this step, some of which are listed in [2]. To search them and combine the results, a *Bash* script employing utilities from the *textutils* package has been used.

Sometimes, enumerating the roots that member in a class is not straightforward, as in the case of the L class. In such cases, corpus search for the prospective prefix-root combination has been conducted to prove or disprove its actual use. But as many valid words do not appear in corpora, probably not all members of the L class can be identified in this way. Other limitations, such as the insufficient size of some specialized vocabularies used, may also negatively influence the result.

4 Affix Rules

Within his ESPSOF project, Witkam [7] has produced a list of approximately 33,000 morpheme-segmentated Esperanto words, based on words appearing in the PIV and manually adjusted by him. We used this list to represent actual language use and inferred from it rules concerning allowed morpheme combinations within Esperanto words.

In the beginning, grammatical affixes were stripped and all roots within the words were identified, of which there may be several in a word, since Witkam's list includes also compounds. This has revealed that 39 % (12,970) of the words consist of a single root and no lexical affixes, 18 % (6,005) of them are a compound of two roots and 4 % (1,386) are a compound of two roots linked by an epenthetic vowel "o". The remaining 38 % (12,639) are words containing lexical affixes, the most frequent *pattern* being a root followed by the "aĵ" suffix.[4] In total, 632 various word patterns have been discovered.

Later, all the roots were classified using the system described in the previous section, and every time all the words matching each of the discovered *patterns* (i.e. combinations of root placeholders and concrete lexical affixes) were examined at once. This examination has been done manually, using just common sense of a fluent Esperanto speaker, and its goal was to discover similarities among the roots that fell into the same place of the pattern, with the ultimate goal to replace this set of roots by a much smaller set of root classes. Based on the morphological assumptions made above, it should be possible to perform such an abstraction without excluding any existing words neither introducing words that do not exist (but it's probable that many existing words not present in Witkam's dictionary were included in this step).

Result of the automatic root classification and manually conducted examination and abstraction of the morphological patterns that emerged was a set of rules such as

[BKP]·"id"

meaning that the suffix "id"[5] may be attached after a root from either the B, K or P class, producing words like

"kat·id·o" ("kitten") from "kat·o" ("cat")

"kverk·id·o" ("oak offspring") from "kverk·o" ("oak")

"reĝ·id·o" ("prince") from "reĝ·o" ("king").

5 Implementation

In order to implement the designed spell checker as a dictionary (i.e. a word list and an affix file) for Hunspell, we had to find a workaround for Hunspell's very limited capabilities of working with regular expresssions. Currently, only the asterisk and the question mark operators are supported, of which only the question mark is of direct use for us – the optionary epenthetic vowel may be expressed by it. In the very frequent case when roots from several possible classes may occupy certain positions, the regular expression had to be split and separate expression had to be created, explicitly stating each of the possibilities.

This, along with word compounding, has led to dramatic increase of the number of regular expressions that form the affix file. Although some partial remedies have been found, such as grouping common sequences of classes into one virtual class of morpheme compounds, the resulting affix file has a size of

[4] This suffix is used to denote a concrete manifestation of the root, such as "manĝ·aĵ·o" ("meal") from "manĝ·i" ("to eat"). [5] This suffix is used to denote offspring or descendant of the root object, such as "hund·id·o" ("puppy") from "hund·o" ("dog").

37,155 rules, which slows down the spell-checking process, but fortunately not to any really remarkable extent yet. This could be avoided once a newer version of Hunspell would support the plus operator in its regular expressions.

The created Hunspell dictionary may be used for spell checking in software packages such as *Mozilla Firefox* or *OpenOffice.org*. For this, it needs to be provided with some additional information (such as meta information and license agreement) and packed up in form of an extension file for the particular application. Particular emphasis has been put on integrating the spell checker in *OpenOffice.org*, since a new Esperanto localization of this office suite is being prepared and the developed dictionary could become its official spell checker. For this, a dedicated subcomponent called "spellcheck" has been recently set up in *OpenOffice.org*'s bugtracking system *Issuetracker*, where users may submit their comments on the functionality of the dictionary.

6 Conclusion

In the article, we have described the process of developing a new spell-checking dictionary for Esperanto, with consideration of the language's word building system. We have developed a system of classes that reflect important semantic properties of word roots, as well as an automatic classification mechanism. We have inferred rules that make use of this class system to describe morphological structures of existing Esperanto words, and we have implemented these rules in form of a dictionary for the Hunspell framework.

Tests performed on computer transcription of an Esperanto-language literature manuscript[6] have shown a 19 % decrease (from 206 to 167) of misspelling recall in comparison with Pokrovskij's dictionary, which we consider an unpleasant side effect of extending the dictionaries morphological capabilities (what produces valid forms that coincide with misspellings) that, on the other hand, has caused a decrease in the amount of false positives (by 10 %, from 1565 to 546 unique words).

Future tasks include research on balancing the spell checker's precision and recall in order to achieve maximum user satisfaction, as well as further testing the developed solution with users and performing the necessary administrative steps to integrate the new dictionary as an automatically installed part of the Esperanto distribution of *OpenOffice.org*.

Acknowledgements. This study has been partially supported by the grants "Language helper for Esperanto" of the Esperantic Studies Foundation and MUNI33/212008 of the Faculty of Informatics of the Masaryk University and by the Ministry of Education of CR within the National Research Programme II project 2C06009.

[6] "Dívka na vdávání" by Miloslav Švandrlík, in Esperanto translation "Edzinigebla knabino" by Josef Vondroušek. 10,400 unique words. 52,700 words in total.

References

1. Petrović Lundberg, Sonja et al: Language helper for Esperanto – a project proposal. Esperantic Studies Foundation (2007).
2. Blahuš, Marek: A Spell Checker for Esperanto. Bachelor thesis, Faculty of Informatics, Masaryk University, Brno (2008).
3. Pokrovskij, Sergio: Vortaro por ISpell, 3.4 (2008) `http://www.esperanto.mv.ru/Download/ispell/ispelleo.tar.bz2`.
4. Németh, László: Hunspell : open source spell checking, stemming, morphological analysis and generation under GPL, LGPL or MPL licenses, 1.2.8 (2008) `http://hunspell.sourceforge.net/`.
5. Wennergren, Bertilo: Plena manlibro de Esperanta gramatiko. El Cerrito : ELNA, 2005. 696 pp. ISBN 0939785072. `http://bertilow.com/pmeg/`.
6. Plena ilustrita vortaro de Esperanto 2005. Editor Gaston Waringhien. Paris : SAT, 2005. 1265 pp. ISBN 2950243282.
7. Witkam, Toon: ESPSOF (Esperanto-Softvaro). Versio 0.8 (2008) `http://www.espsof.com/`.

Yet Another Formalism for Morphological Paradigm

Marek Grac

Natural Language Processing Centre, Faculty of Informatics
Masaryk University, Brno
xgrac@fi.muni.cz

Abstract. Morphology is one of the few areas in the natural language processing where computers are good enough. Different approaches lead to different problems. For Slavonic languages rules and statistical methods are commonly used. Rule based methods are more precise but tend to fail when parsing unknown words. Hybrid technologies with statistical methods helps to solve this problem. It is also possible to solve this problem by extending existing rule-based resources. These resources can be used also for other linguistic research. This paper presents new formalism which is closer to human understanding of natural language morphology and its application in extending morphological dictionary.

Key words: morphology; morphological analysis

1 Introduction

Morphology was first area in natural language processing reaching maturity. Tokenization, splitting running text into tokens, is difficult problem for Arabic or Chinese but not for Slavonic languages. Morphological analysis was first real problem for them. Simple solutions suitable for English with its simple morphological system shown to be ineffective for Slavonic languages. For them morphological paradigm, contains information about lemma and possible word forms, needs higher level of abstract formalism.

2 Suffix Based Formalism

Rule based system for Slavonic languages are popular and widely used (e.g. [1]). These systems use different sets of grammatical tags and are not used for different languages. Most of them use suffix grammar to model existing paradigm. This approach is suitable for Slavonic languages because they use mostly postfix morphology with limited prefix morphology (verb and adjection negation *ne*, superlatives *naj*. Formalisms uses only few basic operations that can be performed at the end of word e.g. add/remove character. Such simple formalism helps us to create morphological analyzers that look simple and can be very fast (tens of thousands analysis per second). Disadvantage of this approach is that we have to define large number (hundreds) of patterns. In [2] we can find several tricks how to partially reduce their numbers in particular cases but it won't help too much.

3 Linguists Defined Formalism

Morphology is interested not only for computer linguists but also for traditional linguists. Our attempts to formalize patterns described in linguistics resources ended with puzzled results. Authors of these books wrote them for readers with their language experience, so explained patterns uses words like 'sometimes' or 'mainly'. Ambiguity of these patterns is problem even for non-expert readers. This formalism when one pattern is described on several pages is not suitable for computers as we cannot parse them correctly. Main benefit of this approach is that we ends in small amount of patterns which are distinguishable from each other.

4 Yet Another Formalism

Our attempts to formalize knowledge in monographies about morphology lead us to create formalism which will be closer to traditional patterns but still unambiguous. In SBF we define pattern as a set of suffixes with tags. LDF extends this because it tends to use semantic characteristics or etymology. Unfortunately these information are not easily accessible for machine usage. We have found that only suitable feature of LDF patterns for us is condition constraining lemma. Usually lemma has to have defined suffix (this is covered by SBF, too) and after removing this suffix, it is possible to generate new conditions for this form. For Slovak and Czech language we found out that they belongs to these group:

– on N-th position is character X *b, ch*
– on N-th position is character which belongs to class X (e.g. soft consonant *š*, long vowel *á*
– on N-th position ends long/short syllable *domáci, cudzí*

Using set of conditions from previous list can rapidly reduce number of lemma which can potentionaly be part of pattern. Problems which does not have to be solved in SBF arise. Some characters contain more then one letter (e.g. ch) and others are specific for one languages (e.g. ř, dž). Also splitting word into character can be ambiguous (e.g. viachlas does not contain character ch). Also syllables can be different across languages. Splitting word into syllables is not a trivial problem. For our purpose we don't care about boundaries and we just have to found core of syllable and decide if it is short or long.

Each pattern tries to describe every accepted word form for lemma. Such word form can be divided in various ways. In SBF generation rules usually contain of what should be removed and suffix which will be added. We are following this simple method but generalize it a bit. Each word form consist of prefix, base and suffix. Each pattern can define several bases so in pattern generation we just point to them using identificators. Using several bases is based to LR (also applied in Slovak morphological database [3]). As we want do remove ambiguity, algorithm of creating base from lemma have to be disambiguate.

Algorithm for creating base can be part of lemma requirements and then it apply only to lemma which successfully based requirerements. Or it can be part of pattern directly if it will be applied to every lemma belonging to pattern. We are aware that extensive usage of bases can result in having empty suffixes. Usually for Czech or Slovak, we use one or two bases for nouns and up to five of them for verbs. Those bases can be created by operations. Several of them are language independent e.g. remove and add character on/to N-th position (operations: chop and append). It is possible to create user defined operations but it is not necessary in most of the cases.

Next part of pattern generation consists of rules. Each word form is defined with its prefix, pointer to base and suffix. It is possible to generate several word forms with same tag. Concatenating prefix, base and suffix does not have to result in final word form. In some cases we want to polish it a bit. For Slovak we want quite often to shorten last syllable if previous one is long e.g. domácí -> domáci. Such filters can be applied directly to word form. Second possibility are filters that are generally valid across language, they can be defined for language itself and does not have to be mentioned in patterns e.g. in Slovak medved' + e -> medvede.

In some cases we want to add a special pattern to lemma. We distinguish two such cases. First case is when we know that given word form can be used as lemma with defined pattern. Example of such case in Slovak is generation of deverbalism plávať -> plávanie (pattern vysvedčenie). Second case is different because in some cases we want to use pattern of lemma. Such case is best shown on pattern negation which adds prefix ne- and then copy existing pattern (we will have just one negation for each PoS).

5 Langusta Framework

In project Langusta we attempted to write an implementation of previous formalism. We have decided to use Java programming language and free BSD license. Result of our work is framework which consist of language independent parts which are inherited by language specific parts.

Defining new Slavonic language is very simple. We have to begin with alphabet definition which contains all characters and classes to which they belong. Second step is to modify general word to characters splitter to cover cases when there is an ambiguity. After these two steps we are able to start writing patterns. Usually after just few of them we will realize which operations and filters will be usefull to write. They have to follow appropriate interface and in the language definition we will add those Java objects to identificators used in pattern definiton. This part usually takes few days. Last and usually the more complex and time consuming part is to write pattern definition for flective parts of speech.

We are in a process of creating set of morpholigical patterns but preliminary results are promising. Our tool based on Trdlo framework allow us to work very fast 200–300 lemma/hour because it shows expert only those patterns that are acceptable for given lemma. It means that even if we have around 50 verb patterns for Slovak in usual case only 2 or 3 them are shown.

6 Conclusion

In this paper we presented a new approach for formalisation of morphological pattern. Benefits of such approach where explained. Architecture of framework Trdlo was described and preliminary results are promising. In the future we would like to finish writing patterns for Slovak and Czech. Having such resource for several languages can be very usefull for comparative linguists. After we will annotate enough data we are going to try to use more automatic methods to determine pattern according to data found in non-tagged corpora.

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.

References

1. Sedláček, R.: Morfologický analyzátor češtiny. Master's thesis, FI MU, Brno (1999).
2. Garabik, R.: Levenshtein Edit Operations as a Base for a Morphology Analyzer. Computer Treatment of Slavic and East European Languages. Ed. R. Garabík. Bratislava: Veda (2005) 50–58.
3. Benko, V., Hašanová, J., Kostolanský, E.: Morfológia podstatných mien. Počítačové spracovanie prirodzeného jazyka. Pedagogická fakulta UK, Bratislava (1998).

Fast Morphological Analysis of Czech

Pavel Šmerk

Faculty of Informatics, Masaryk University
Botanická 68a, CZ-60200 Brno, Czech Republic
smerk@mail.muni.cz

Abstract. This paper presents a new Czech morphological analyser which takes an advantage of Jan Daciuk's algorithms for minimal deterministic acyclic finite state automata. The new analyser is six times faster than the current analyser `ajka` concerning the proper analysis, i.e. returning possible lemmata and tags for a given word form, but for some other related tasks is the difference even bigger.

Key words: morphological analysis; Czech language

1 Introduction

In the last year our current Czech morphological analyser `ajka` [4,5] have started to be used in two large projects. One of its new "users" is Seznam.cz[1], the first and biggest Czech internet portal and web search engine. The other one is Masaryk University Information System[2] which services not only needs of the second largest Czech university, but also a nation-wide registries of graduate theses[3] and of colloquial and other smaller college works[4] which allow full-text search and detection of plagiarism.

It has turned out that `ajka` is too slow to satisfy the new performance requirements. We have developed a new morphological analyser `majka` which uses the same language data as `ajka`, but the analyser itself as well as the format of the data are completely new. Both are described in Section 2 and in Section 3 some performance comparison with `ajka` is shown.

2 Data

The new morphological analyser `majka` is an implementation of the approach proposed in [6]. The data are simply a list of all combinations of a recognized input and corresponding outputs of the analyser, where pairs of two words are encoded as pairs formed by the first word and a difference between the words. For example, in the following part of data for word form → lemma + tag analysis

[1] `http://www.seznam.cz/` [2] `http://is.muni.cz/?lang=en` [3] `http://theses.cz/?lang=en`
[4] `http://odevzdej.cz`, the website is only in Czech

```
klouček:A,k1gMnSc1
klоučka:Cek,k1gMnSc2
klоučka:Cek,k1gMnSc4
```

the colon is a delimiter between the possible inputs and corresponding outputs and the letters A and C as the first and the third letters of the alphabet mean "to get the lemma delete n-1 (i.e. 0 or 2, respectively) last characters from the word form and then attach the rest of the string (i.e. empty string or *ek*, respectively)". Then the word form *klouček* will be analyzed as a lemma *klouček* with a morphological tag *k1gMnSc1*[5] and a word form *kloučka* as a lemma *klouček* with morphological tags *k1gMnSc2* and *k1gMnSc4*[6].

Such a list is then represented as a minimal deterministic acyclic finite state automaton using Jan Daciuk's algorithms for incremental building of minimal DAFSAs [1]. This representation dramatically reduces the size of the data (some particular figures can be seen later in Table 2). The lookup is then very simple: if the analysed string concatenated with the delimiter is found in the automaton, then each possible remaining path to a final state of the automaton encodes one of possible analyses.

It means that there is no "real" analysis as a sophisticated algorithm above some grammar model or a system of paradigms, but whole analysis is only a simple — and therefore fast — dictionary search.

3 Performance Comparison

Results of a performance comparison of the analysers `ajka` and `majka` are presented in the Table 1. The comparison was done on the first one million words from the SYN2000 corpus [7] which is a part of the Czech National Corpus[7].

Table 1. Results of comparison of the old analyser `ajka` and the new analyser `majka`

	size of data in MB		time in seconds		
	`ajka`	`majka`	`ajka`	`majka`	ratio
morphological analysis	3.1	4.4	18.22	2.88	**6.3**×
lemmatisation		4.0	16.76	1.57	**10.7**×
all word forms		6.1	55.33	8.42	**6.6**×
restoration of diacritics		3.3	8698.80	1.61	**5403**×

The measured time is a "wall clock" time as was reported by a unix command `time`. All times are averages of three runs. Outputs of the analysers were allways redirected to `/dev/null` to measure only a CPU time and not waits for a hard disk etc.

[5] *little boy* in nominative form [6] *little boy* in genitive and accusative form

[7] `http://www.korpus.cz/english/`

It should be noted that, in the contrary of what a reader might expect, the extreme difference in the speed of the diacritics restoration is the least surprising for us, because this task is implemented very poorly in `ajka`.

The outputs of the old and new analyser are not quite identical: there are still some minor differences or even bugs on either side (mainly in an analysis of compound words), which will be addressed in the near future, but none of these differences can notably affect the overall performance.

As is obvious from the previous section, the new analyser `majka` has to have separate dictionary for each task — unlike `ajka`, which has only one common data, and even smaller. It is a kind of a tax for the speedup, but this several megabytes difference is not a problem for present-day computers.

The Table 2 shows the extent of the compression of dictionaries. For other languages, Kowaltowski [3] reports 0.25 byte per one word-tag-lemma entry for Brazilian Portuguese and Daciuk [1] reports less then 0.15 byte per one word-lemma-tag entry for German and ca. 30 times better compression rate compared to gzip on that data. The presented results show similar or better compression for the Czech data as well

Table 2. Statistical information on the data files of the new analyser `majka`

type of dictionary	# of entries	size of entries	size of dict.	bytes/entry
word (diacritics restor.)	13,609,590	186,154,068	3,263,374	0.240
word → lemma	14,101,767	239,578,702	4,042,839	0.287
word → lemma, tag	80,303,929	2,477,786,062	4,353,616	0.054
word → all word forms	957,464,060	19,993,465,213	6,105,429	0.006

In both tables, there are only tasks, which can be directly handled by `ajka`. Besides, we have also dictionaries for generation all word forms from lemma (which is a “subset” of word → all word forms) and for tasks lemma (or any word form) → word forms + tags, and lemma (or any word form) + tag → word forms, but these tasks are not supported by `ajka`.

4 Conclusions and Future Work

According to Gelbukh and Sidorov [2], the designer of a morphological analyzer for an inflective language has the following choice:

- either generate all word forms and build a system with a large dictionary and a very simple ‘analysis" (just searching) algorithm,
- or build a system with a much smaller dictionary of stems with information about possible endings, but with some more sophisticated algorithm (analysis through generation, in particular).

For the inflective languages they strongly suggest the second option, because it allows to use a grammar model almost directly taken over from traditional grammars, which are oriented mainly toward generation. Our results clearly show that they are wrong. The first approach is better even for the inflective languages, because the analyser remains simple and therefore the analysis runs fast (but of course the simplicity of any program is a value in itself concerning a long term maintenance and development). Apparently, using our approach the designer of a morphological analyser is absolutely free regarding the choice of a suitable grammar model, because there are no constraints on how and from what sources one can generate the dictionary data.

The presented results are only preliminary as the new analyser `majka` is still under an intensive development. We expect that the final data files will be smaller: at least in some cases, as e.g. the generation of all word forms, we are aware of particular inefficiencies regarding the format of data. We believe that there is also a potential of some performance improvements, so that the final version will be even faster.

Acknowledgements This work has been partly supported by the Academy of Sciences of the Czech Republic under the project 407/07/0679 and by the Ministry of Education, Youth and Sports within the National Research Programme II project 2C06009 and project LC536.

References

1. Jan Daciuk. 1998. *Incremental Construction of Finite-State Automata and Transducers, and their Use in the Natural Language Processing.* Ph.D. dissertation, Technical University of Gdańsk, Poland.
2. Alexander Gelbukh and Grigori Sidorov. 2003. *Approach to Construction of Automatic Morphological Analysis Systems for Inflective Languages with Little Effort.* In: Computational Linguistics and Intelligent Text Processing. Proc. CICLing-2003. LNCS 2588, Springer-Verlag, pp. 215–220.
3. Tomasz Kowaltowski, Cláudio L. Lucchesi, and Jorge Stolfi. 1998. *Finite Automata and Efficient Lexicon Implementation.* Technical Report IC-98-02, University of Campinas, São Paulo.
4. Radek Sedláček and Pavel Smrž. 2001. *A New Czech Morphological Analyser* `ajka`. In Proceedings of the 4th International Conference TSD 2001. LNCS 2166, Springer-Verlag, pp. 100–107.
5. Radek Sedláček. 2004. *Morphematic analyser for Czech.* Ph.D. dissertation, Faculty of Informatics, Masaryk University, Brno.
6. Pavel Šmerk. 2007. *Morphemic analysis: A Dictionary Lookup Instead of Real Analysis.* Petr Sojka, Aleš Horák (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2007. Masaryk University, Brno. pp. 77–85.
7. Czech National Corpus – SYN2000. 2000. Institute of the Czech National Corpus, Praha. Accessible at: `http://www.korpus.cz`.

Domain Collocation Identification

Jiří Materna

Centre for Natural Language Processing
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00, Brno, Czech Republic
`xmaterna@fi.muni.cz`
`http://nlp.fi.muni.cz`

Abstract. In this paper we present a new method of automatic collocation identification. Collocation is an important relation between words, which is widely used, among others, in information retrieval tasks. Over the last years, many methods of automatic collocation acquisition from text corpora have been proposed. The approach described in this paper differs from the others in focusing on domain collocations. By the domain collocation we mean a collocation which is specific for a relatively small set of documents related to the same topic. The proposed method has been implemented and used in a real information retrieval system. Comparing to the common non-domain approach, the precision of the system has increased significantly.

Key words: collocation; domain; information retrieval

1 Introduction

Lexical collocations are an important phenomenon in many natural language processing tasks like computational lexicography [1], word sense disambiguation [2], machine translation [3], information extraction [4], etc. In this work we focus on their exploitation in information retrieval. In our information retrieval system, there is a need for identifying collocations in queries in order to treat them as single units.

Let's have a look at a simple query *finanční úřad v Karlových Varech (tax office in Karlovy Vary)*. Combinatorially, there are many ways how to parse it but, in our point of view, the correct one is only *(finanční úřad) v (Karlových Varech)*. It means that the identified collocation (in this case *finanční úřad* and *Karlových Varech*) should not be split.

Collocation is an expression consisting of two or more associated words or tokens. Unfortunately, there is no formal linguistic definition. For us, the collocation is understood as an *n*-gram of tokens whose co-occurrence in a large text corpus is statistically outstanding. There are many statistical measures usable to detect collocations in corpora. Most of them are based on classical mathematical statistics (t-score, chi-square) or information theory (mutual information). All these methods are widely used and explored in many applications but they suffer from the following disadvantages:

– The association scores are strongly influenced by the size of the corpus. Thus, the score values acquired from different corpora are not comparable and even the maximum or minimum values of the score may be different.
– The association scores are suitable for identifying global collocations but they are not convenient for domain-specific collocations, which are relevant only for a small set of documents related to the same topic.

The presented paper deals mainly with the second problem and is structured as follows. In the next section we will introduce the most commonly used statistical methods of collocation identification in text corpora. In Sectioni 3 we will focus on a new method of domain collocation acquisition. In this section we will also discuss advantages and disadvantages of the proposed method.

2 Statistical Approaches to Collocation Identification

Over the last years, many methods of automatic collocation acquisition from large text corpora have been proposed. All association scores which are subjects of this research use only word frequency characteristics. The simplicity and the ease of use belong to the most important advantages of the statistical approach. In order to ensure maximum readability of the paper we will only consider collocations consisting of two tokens but the described methods are universal. In the rest of the paper the following notation will be used:

– $f(t)$ – number of occurrences of term t in the whole corpus;
– $f(t_1, t_2)$ – number of co-occurrences of terms t_1, t_2 (by the co-occurrence we mean that the tokens occur in the corpus directly one after another);
– n – number of all tokens in the corpus.

T-score

This measure uses classical statistic approach based on Student's t-test [5]. The association score is defined as:

$$T(t_1, t_2) = \frac{f(t_1, t_2) - \frac{f(t_1)f(t_2)}{n}}{\sqrt{f(t_1, t_2)}}$$

MI-score

This measure comes from information theory and corresponds to the quantity of information given by the occurrence of one term about occurrences of another one. The mutual information association score is defined as:

$$MI(t_1, t_2) = \log_2 \frac{f(t_1, t_2)n}{f(t_1)f(t_2)}$$

*MI*²-score

Mutual information score is a useful measure but it is strongly influenced by the frequency of tokens. To reduce this disadvantage same heuristics have been proposed [6]. One of the most popular is MI^2-score:

$$MI^2(t_1, t_2) = \log_2 \frac{f(t_1, t_2)^2 n}{f(t_1) f(t_2)}$$

Dice score

Dice score identifies pairs with a particularly high degree of lexical cohesion (i.e. those with nearly total association) [7]:

$$D(t_1, t_2) = \frac{2f(t_1, t_2)}{t_1 + t_2}$$

logDice score

Dice score gives a good association score but the problem is that the values are usually very small numbers. This is solved by the logDice score [8]:

$$logD = 14 + \log_2 \frac{2f(t_1, t_2)}{t_1 + t_2}$$

In many application we need an universal score, which corresponds to the degree of collocability. In this work the score is called *proximity* and ranges from 0 (absolutely independent terms) to 1 (perfect collocations). To get the proximity

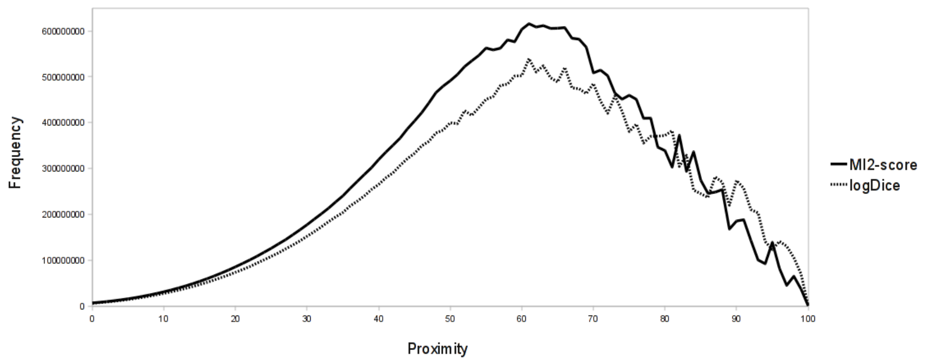


Fig. 1. Proximity distribution

from association scores described above, we need to transform the scores into interval $[0, 1]$. The conversion is done by normalizing the scores by their maximal values. The proximity distribution in a Czech web corpus (200 billions of tokens) for two best-resulting association scores is shown in Figure 1.

3 Domain Collocations

The association scores described in the previous section have some advantages – computation of their values is very fast and simple, they satisfactorily reflect collocation scores of global collocations, etc., but they also have at least one disadvantage – they are not convenient for identifying domain-specific collocations, which are relevant only for a small set of documents related to the same topic. An example of the domain collocation is *rozhodovací strom (decision tree)*, which is a strong collocation in the computer science domain but hardly in general.

The idea behind the method of identifying domain collocations is that domain collocations should be generated from domain specific sub-corpora. Nevertheless, there are many problems: what domains should be used, how to divide the corpus into domain specific sub-corpora, how to detect domain specific collocation, etc.

Probably the best way how to solve these problems is to avoid them. The solution is based on using the association scores described above with redefinition of the $f(t)$ function. Value of the $f(t)$ function is in the domain approach defined as *number of occurrences of term t in all documents containing all constituents of the investigated potential collocation*. In other words, for bigram (t_1, t_2), value of the $f(t_1)$ function is computed as a number of occurrences of t_1 in all documents containing both t_1 and t_2 at arbitrary location and vice versa. This approach has following consequences:

- Value of the $f(t)$ function is different for different bigrams. This is the source of a high computational complexity.
- Value of the domain proximity is always higher then value of the non-domain proximity for the same bigrams.
- Comparing to the non-domain approach, in the domain approach the proximity of good collocations increases rapidly, whereas proximity of non-collocations increases slightly.

Examples of proximity values for some bigrams from the corpus are shown in the Figure 2.

collocation type	**bigram**	**non-domain proximity**	**domain proximity**
global collocation	jízdní řády (timetables)	0.952	0.992
	karlovy vary	0.983	0.995
non-collocation	a ale (and but)	0.295	0.319
	zelené myšlenky (green ideas)	0.278	0.286
domain collocation	rozhodovací strom (decision tree)	0.363	0.684
	třecí síla (frictional force)	0.441	0.820

Fig. 2. Examples of proximity values.

4 Conclusions

One of the disadvantages of common association scores is a fact that some domain specific collocation cannot be identified. This work solves this problem by providing a new approach to computing term frequencies with regard to domain collocations. This method does not nearly affect scores of good general collocation and non-collocations but significantly improve proximity score in domain specific collocation. The proposed method has been tested and is being used in a real information retrieval system.

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.

References

1. Kilgarriff, A., Rychlý, P., Smrž, P., Tugwell, D.: The Sketch Engine in Practical Lexicography: A Reader. (2008) 297–306.
2. Yarowsky, D.: Word Sense Disambiguation. In: The Handbook of Natural Language Processing, New York: Marcel Dekker (2000).
3. Smadja, F., Hatzivassiloglou, V., McKeown, K.R.: Translating Collocations for Bilingual Lexicons: A Statistical Approach. Computational Linguistics (1996).
4. Lin, D.: Using Collocation Statistics in Information Extraction. In: Proceedings of the Seventh Message Understanding Conference (MUC-7. (1998).
5. Church, K.W., Gale, W.A.: Concordances for Parallel Text. In: Proceedings of the 7th Annual Conference of the UW Center for the New OED and Text Research, Oxford, UK. (1991).
6. Oakes, M.P.: Statistics for Corpus Linguistics. Edinburgh University Press, Edinburgh (1998).
7. Dias, G., Guilloré, S., Lopes, J.: Language Independent Automatic Acquisition of Rigid Multiword Units from Unrestricted Text Corpora. In: Proceedings of Traitement Automatique des Langues Naturelles (TALN), Cargèse, France. (1999).
8. Rychlý, P.: A Lexicographer-Friendly Association Score. In: Recent Advances in Slavonic Natural Language Processing. (2008).

Part II

Logic in Language

Deductive Reasoning using TIL

Martina Číhalová, Nikola Ciprich, Marie Duží, Tomáš Frydrych, and Marek Menšík

VŠB-Technical University Ostrava
17. listopadu 15, 708 33 Ostrava, Czech Republic
m.tina.cihal@gmail.com, nikola.ciprich@linuxbox.cz, marie.duzi@vsb.cz, frydrych.t@gmail.com, mensikm@gmail.com

Abstract. Transparent Intensional Logic (TIL) is a highly expressive logical system apt for the logical analysis of natural language. It operates with a single *procedural semantics* for all kinds of logical-semantic context, whether extensional, intensional or hyper-intensional, while adhering to the compositionality principle throughout. The reason why we vote for a rich procedural semantics is this. A coarse-grained analysis of assumptions yields paradoxes and puzzles, while an expressive formal system such as TIL makes it possible to build an inference machine that neither over-infers (which yields paradoxes) nor under-infers (which leads to the lack of knowledge). From the formal point of view, TIL is a hyperintensional, partial, typed lambda calculus. By way of examples we illustrate how TIL deals with particular 'puzzles' in a smooth way while adhering to Leibniz's law of substitution of identicals and to the principle of compositionality.

Key words: TIL; deductive reasoning; *TIL-Script* language; inference machine

1 Introduction

The way we understand the enterprise of logical analysis of natural language in TIL is selective. The analysis leaves aside pragmatic features of language, but makes all the logically salient features explicit and logically tractable. Yet the very name of our theory, 'Transparent Intensional Logic', is likely to strike one as being an oxymoron, like 'roaring silence'. How can there possibly be a *logic* that is *intensional* and at the same time *transparent*? Isn't any intensional logic such that it fails to heed various laws of extensional logic, such as referential transparency, substitution of identicals, and compositionality? Certainly yes, if 'intensional' is synonymous with 'non-extensional'. But 'intensional' may also mean—and this is the notion of intensionality germane to TIL—that the logic in question comes with a rich ontology of entities and the means to logically manipulate these entities. Due to its rich ontology of entities organized in a bi-dimensional that is ramified hierarchy of types TIL flouts none of the principles of *extensional* logic and is, insofar, an *extensional* logic.[1]

[1] Portions of this paper draw on material presented in [7], in particular Sections 2.6 and 2.7.

TIL operates with a single *procedural semantics* for all kinds of logical-semantic context, whether extensional, intensional or hyper-intensional. It means that it explicates the meaning of an expression as an abstract procedure encoded by the expression. Such procedures are rigorously defined as TIL constructions and we assign them to expressions as their context-invariant meanings. From the formal point of view, TIL is a hyper-intensional, partial, typed λ-calculus. Hyper-intensional, because the terms of the TIL formal language in which constructions are encoded are interpreted as procedures (generalized algorithms) rather than their products; partial, because the primitive notion of TIL is a function understood as a partial mapping that assigns to each element of its domain *at most* one element of its range; and typed, because all the entities of TIL ontology, including constructions, receive a type.

Yet such an expressive system is often characterized as being logically or computationally intractable. In our opinion, it is not the language in which a problem is encoded what can be intractable, but the *problem* itself. Problems are easy or difficult to solve. For instance, it is a well known fact that though the problem of logical validity in propositional logic is decidable, it is not computationally tractable. In Stephen Cook's famous [3] the theorem is proved that the satisfiability problem is NP-complete, and the tautology problem is co-NP-complete. This means that by a commonly accepted conjecture, these problems are regarded as computationally intractable. And it is also a well-known fact that as a consequence of Gödel's incompleteness theorem, the problem of logical validity in the first-order predicate logic is not even algorithmically decidable. Does it mean that we should reduce our reasoning to computationally tractable sub-systems of propositional logic? Certainly not. Though a great expressive power is inversely proportional to an easy implementation of a suitable deductive system, we need to know *what* should be solved prior to seeking plausible methods of the problem in question solving.

Moreover, there is another strong reason to vote for an expressive, fine-grained semantics. A coarse-grained analysis of assumptions yields paradoxes and puzzles. On the other hand, a rich formal system such as TIL makes it possible to build an inference machine that in principle neither over-infers (which yields paradoxes) nor under-infers (which leads to the lack of knowledge). However, there are two problems connected with TIL deduction system.

First, since the system is hyper-intensional, which means that the meaning of an expression is a *construction* specified by the analyzed expression, rather than the product of the construction denoted by the expression, we must strictly distinguish between constructions and their products. This amounts for distinguishing a context in which a construction is used to produce an entity (if any), and the *hyper-intensional* context in which the construction itself is only *mentioned* as an object of predication. And if the construction is used, we must distinguish the context in which it is *used intensionally* and the context in which it is *used extensionally*. If the former, then the so constructed *function* is an object of predication; and if the latter, the *value* of the constructed function is an object of predication. Only then can we specify rules of deduction for extensional, intensional as well as hyper-intensional context.

The second problem is *partiality*. The primitive notion of TIL is function rather than relation. And it is a brute fact that we need to work with *properly* partial functions, i.e. functions that are undefined at some arguments. The problem crops up when a properly partial function is applied to an argument at which the function does not return any value. Traditional formal lambda-calculi avoid this problem by simply excluding non-denoting terms as meaningless. Yet, this is not a plausible solution for the semantics of natural language. Though, for instance, the present King of France does not exist (the office is vacant) this does not mean that the term 'King of France' is meaningless. If it were, we could not reasonably and truly assert that the King of France does not exist. Similarly mathematicians had to know the meaning of 'the greatest prime' prior to proving that there is no greatest prime. Thus we cannot avoid application of a properly partial function to an argument. Only we have to take into account that the operation can fail to produce a product.[2]

The goal of the paper is not to introduce the TIL inference machine in full. Instead, by way of examples, we illustrate how TIL deals with particular 'puzzles' in a smooth way while adhering to Leibniz's law of substitution of identicals and the principle of compositionality in all kinds of context. TIL constructions are assigned to semantically unambiguous expressions as their *context-invariant* meanings. What differs dependently on the context in which the analyzed expression is used are *logical operations* applied to a meaning constituent rather than the assigned meaning itself.

The paper is organized as follows. The next Section 2 presents basic principles and definitions of TIL. In Section 3 we first characterize the three kinds of context in which a construction can occur, and then we illustrate by examples the unique way how TIL operates in these contexts. Concluding Section 4 sums up the results and outlines further research.

2 TIL in brief

Since Frege's pioneering paper [8] logicians and semanticists have striven to define so-called *structured meanings* that would comply with the principles of compositionality and universal referential transparency. Various adjustments of Frege's semantic schema have been proposed, shifting the entity named by an expression from the extensional level of atomic (physical/abstract) objects to the intensional level of abstract objects such as sets or functions/mappings. Yet natural language is rich enough to generate expressions that talk neither about extensional nor intensional objects. Propositional attitudes are notoriously known as the hard cases that are neither extensional nor intensional, as Carnap

[2] To this end we introduce a generally valid rule of β-reduction 'by value'. The rule has been precisely defined in [7], Section 2.6. Roughly, the idea resembles lazy evaluation of functional programming languages. We first check whether the construction is proper and only then substitute the Trivialization of the constructed entity for the formal parameter x. See also [4] where this substitution method is applied to anaphora pre-processing.

in [1] characterized them. It has become increasingly clear since the 1970s that we need to individuate meanings more finely than by possible-world intensions, and the need for *hyperintensional* semantics is now broadly recognised. Our position is a plea for such a semantics, which takes expressions as encoding *algorithmically structured procedures* producing extensional/intensional entities (or lower-order procedures) as their products. This approach—which could be characterized as being informed by an *algorithmic* or *computational turn*—has been advocated by, for instance, Moschovakis in [10]. Yet much earlier, in the early 1970s, Tichý introduced his notion of *construction* and developed the system of Transparent Intensional Logic, as presented in [11] and [12].

Constructions, as well as the entities they construct, all receive a type. The ontology of TIL is organized in an infinite, bi-dimensional hierarchy of types. One dimension is made up of non-constructions, i.e., entities unstructured from the algorithmic point of view. The other dimension of the type hierarchy is made up of structured, higher-order constructions which construct lower-order entities. Thus our definitions are inductive, and they proceed in three stages. First, we define the simple types of order 1 comprising non-constructions. Then we define constructions and, finally, the ramified hierarchy of types.

Definition 1 *(Types of order 1)* Let B be base, i.e., a collection of non-empty sets.

1. Every member of B is a *type of order 1 over B*.
2. Let $\alpha, \beta_1, \ldots, \beta_m$ be *types of order 1*. Then the set $(\alpha\beta_1 \ldots \beta_m)$ of partial functions with values in α and arguments in $\beta_1, \ldots, \beta_m$, respectively, is a *type of order 1 over B*.
3. Nothing is a *type of order 1 over B* unless it so follows from (1) and (2). □

The choice of the base depends on the area and language we happen to be investigating. When investigating purely mathematical language, the base can consist of, e.g., two atomic types; o, the type of truth-values, and v, the type of natural numbers. When analyzing an ordinary natural language, we use the *epistemic base* which is a collection of four atomic types, o, ι, τ, ω, where o= $\{\mathbf{T}, \mathbf{F}\}$ is the set of truth-values, ι is the universe of discourse (members: individuals), τ is the set of real numbers (or of time moments) and ω is the logical space, the set of possible worlds.

Definition 2 *(intensions and extensions)*; (PWS) *intensions* are entities of type $(\beta\omega)$: mappings from possible worlds to some type β. The type β is frequently the type of the *chronology* of α-objects, i.e., mapping of type $(\alpha\tau)$. Thus α-intensions are frequently functions of type $((\alpha\tau)\omega)$, abbreviated as '$\alpha_{\tau\omega}$'. *Extensions* are entities of a type α where $\alpha \neq (\beta\omega)$ for any type β. □

Examples of frequently used intensions are: *Propositions* (denoted by declarative sentences) are of type $o_{\tau\omega}$; *properties of individuals* (usually denoted by nouns or intransitive verbs like 'is a student', 'walks') are of type $(o\iota)_{\tau\omega}$; binary *relations-in-intension* between individuals are of type $(o\iota\iota)_{\tau\omega}$;[3] *individual offices/roles* (cf.

[3] Since *function* rather than relation is a primitive notion of TIL, we model *sets* and relations by their characteristic functions. Thus, for example, the set of prime numbers is a function of type $(o\tau)$ that associates any number with **T** or **F** according as the given number is a prime.

Church's individual concepts, usually denoted either by superlatives like 'the highest mountain' or terms with built-in uniqueness, like 'The President of the USA') are of type $\iota_{\tau\omega}$. *Expressions* which denote non-constant intensions (i.e. functions that take different values in at least two world-time pairs) are *empirical*.

Quantifiers $\forall^{\alpha}, \exists^{\alpha}$, are extensions, viz. type-theoretically polymorphous functions of type(s) $(o(o\alpha))$ defined as follows: The *universal quantifier* $\forall^{\alpha}$ is a function that associates a class C of α-elements with **T** if C contains all elements of the type α, otherwise with **F**. The *existential quantifier* $\exists^{\alpha}$ is a function that associates a class C of α-elements with **T** if C is a non-empty class, otherwise with **F**. The *singulariser* $Sing^{\alpha}$ is a partial type-theoretically polymorphic function of type(s) $(\alpha(o\alpha))$ that associates a class C with the only α-element of C if C is a singleton, otherwise the function $Sing^{\alpha}$ is undefined. We will often use the abbreviated notation '$\forall x\ A$', '$\exists x\ A$' and '$\iota x\ A$' instead of '$[{}^{0}\forall^{\alpha} \lambda x\ A]$', '$[{}^{0}\exists^{\alpha} \lambda x\ A]$', '$[{}^{0}Sing^{\alpha} \lambda x\ A]$', respectively, when no confusion can arise.

When claiming that constructions are algorithmically structured, we mean the following. A construction C consists of one or more particular steps, or *constituents*, that are to be individually executed in order to execute C. The objects a construction operates on are not constituents of the construction. Just like the constituents of a computer program are its sub-programs, so the constituents of a construction are its sub-constructions. Thus on the lowest level of non-constructions, the objects that constructions work on have to be supplied by other (albeit trivial) constructions. The constructions themselves may occur not only as constituents to be executed, but also as objects that still other constructions operate on. Therefore, one should not conflate *using* constructions as constituents of compound constructions and *mentioning* constructions that enter as input/output objects into compound constructions. Mentioning is, in principle, achieved by using atomic constructions. A construction C is atomic if it does not contain any other construction as a used sub-construction (a 'constituent' of C) but C. There are two atomic constructions that supply entities (of any type) on which compound constructions operate: *Variables* and *Trivializations*. *Compound* constructions, which consist of other constituents than just themselves, are *Composition* and *Closure*. *Composition* is the instruction to apply a function to an argument in order to obtain its value (if any) at the argument. It is *improper*, i.e., does not construct anything, if the function is not defined at the argument. *Closure* is the instruction to construct a function by abstracting over variables in the ordinary manner of the λ-calculi. Finally, higher-order constructions can be used once or twice over as constituents of constructions. This is achieved by a fifth and sixth construction called *Execution* and *Double Execution*, respectively.

Definition 3 *(construction)*

1. The *Variable* x is a **construction** that constructs an object O of the respective type dependently on a valuation v; it v-constructs O.
2. *Trivialization*: Where X is an object whatsoever (an extension, an intension or a construction), ^{0}X is the **construction** of *Trivialization*. It constructs X without any change.

3. The *Composition* $[XY_1 \dots Y_m]$ is the following **construction**. If X v-constructs a function f of a type $(\alpha\beta_1 \dots \beta_m)$, and $Y_1, \dots, Y_m$ v-construct entities $B_1, \dots, B_m$ of types $\beta_1, \dots, \beta_m$, respectively, then the Composition $[XY_1 \dots Y_m]$ v-constructs the value (an entity, if any, of type α) of f on the tuple-argument $\langle B_1, \dots, B_m \rangle$. Otherwise the *Composition* $[XY_1 \dots Y_m]$ does not v-construct anything and so is v-improper.
4. The *Closure* $[\lambda x_1 \dots \lambda x_m Y]$ is the following construction. Let $x_1, x_2, \dots, x_m$ be pairwise distinct variables v-constructing entities of types $\beta_1, \dots, \beta_m$ and Y a construction v-constructing an α-entity. Then $[\lambda x_1 \dots \lambda x_m Y]$ is the **construction** λ-Closure (or *Closure*). It v-constructs the following function $f/(\alpha\beta_1 \dots \beta_m)$. Let $v(B_1/x_1, \dots, B_m/x_m)$ be a valuation identical with v at least up to assigning objects $B_1/\beta_1, \dots, B_m/\beta_m$ to variables $x_1, \dots, x_m$. If Y is $v(B_1/x_1, \dots, B_m/x_m)$-improper (see 3), then f is undefined on $\langle B_1, \dots, B_m \rangle$. Otherwise the value of f on $\langle B_1, \dots, B_m \rangle$ is the α-entity $v(B_1/x_1, \dots, B_m/x_m)$-constructed by Y.
5. The *Execution* ^1X is the **construction** that either v-constructs the entity v-constructed by X or, if X v-constructs nothing, is v-improper.
6. The *Double Execution* ^2X is the following **construction**. Let X be any entity; the Double Execution ^2X is v-improper (yielding nothing relative to v) if X is not itself a construction, or if X does not v-construct a construction, or if X v-constructs a v-improper construction. Otherwise, let X v-construct a construction X' and X' v-construct an entity Y. Then ^2X v-constructs Y.
7. Nothing is a **construction**, unless it so follows from (1) through (6). □

Notation and abbreviations:

– 'X/α' means that the object X is (a member) of type α;
– '$X \rightarrow_v \alpha$' means that the type of the object v-constructed by X is α. We use '$X \rightarrow \alpha$' if what is v-constructed does not depend on a valuation v.
– We will standardly use the variables $w \rightarrow_v \omega$ and $t \rightarrow_v \tau$;
– If $C \rightarrow_v \alpha_{\tau\omega}$, the frequently used Composition $[[C\,w]\,t]$, the intensional descent of the α-intension v-constructed by C, will be written as 'C_{wt}'.
– When using constructions of truth-value functions, namely $\wedge$ (conjunction), $\vee$ (disjunction) and $\supset$ (implication) of type (ooo), and $\neg$ (negation) of type (oo), we often omit Trivialisation and use infix notion.
– When using identity relations $=^\alpha /(o\alpha\alpha)$, we often omit the superscript α and use infix notation, whenever no confusion arises.

As mentioned above, constructions themselves are objects and thus also receive a type. Only it cannot be a type of order 1, because a construction cannot be of the same type as the object it constructs. Constructions that construct entities of order 1 are *constructions of order 1*. They belong to a *type of order 2*, denoted by '$\star_1$'. This type $\star_1$, together with atomic types of order 1, serves as the base for the following induction rule: any collection of partial mappings, type $(\alpha\beta_1 \dots \beta_n)$, involving $\star_1$ in their domain or range is a *type of order 2*. Constructions belonging to the type $\star_2$, which identify entities of order 1 or 2, and partial mappings involving such constructions, belong to a *type of order 3*; and so on *ad infinitum*.

The definition of the ramified hierarchy of types decomposes into three parts. First, simple types of order 1 were already defined by Definition 1. Second, we define constructions of order n, and third, types of order $n + 1$.

Definition 4 *(Ramified hierarchy of types)* Let B be base.
$\boldsymbol{T_1}$ (types of order 1)–defined by Definition 1.
$\boldsymbol{C_n}$ (constructions of order n)

1. Let x be a variable ranging over a type of order n. Then x is a *construction of order n over B*.
2. Let X be a member of a type of order n. Then ${}^0X$, ${}^1X$, ${}^2X$ are *constructions of order n over B*.
3. Let X, $X_1 \dots X_m (m > 0)$ be constructions of order n over B. Then $[XX_1 \dots X_m]$ is a *construction of order n over B*.
4. Let $x_1 \dots x_m, X (m > 0)$ be constructions of order n over B. Then $[\lambda x_1 \dots \lambda x_m\, X]$ is a *construction of order n over B*.
5. Nothing is a construction of order n over B unless it so follows from $\boldsymbol{C_n}$ (1)-(4).

$\boldsymbol{T_{n+1}}$ (types of order $n + 1$)
Let $*_n$ be the collection of all constructions of order n over B. Then

1. $*_n$ and every type of order n are types of order $n + 1$.
2. If $0 < m$ and $\alpha, \beta_1, \dots, \beta_m$ are types of order $n + 1$ over B, then $(\alpha\beta_1 \dots \beta_m)$ (see $\boldsymbol{T_1}$ 2) is a type of order $n + 1$ over B.
3. Nothing is a type of order $n + 1$ over B unless it so follows from T_{n+1} (1) and (2). □

So much for the philosophy and basic definitions of TIL.

3 The outline of TIL calculus

In this section we deal with the deduction system based on TIL. We are not going to define the calculus entirely, since it would be beyond the scope of this paper. Instead we informally explain particular rules as they are valid in the three kinds of context and illustrate them by examples.

3.1 Three kinds of context

As mentioned above, constructions are full-fledged objects that can be not only used to construct an object (if any) but also serve themselves as input/output objects on which other constructions (of a higher-order) operate. This is so, because expressions of natural language, when used in a communicative act, can be used in three different ways. True, expressions are always used to express their *meaning* explicated in TIL as a *construction*. But when using an expression E, its meaning C can occur with three different suppositions:

1. The meaning C is not used to identify an object about which something is predicated; rather, C itself is an object of predication within another expression E' of which E is a sub-expression. We will say that the meaning C (and thus also the expression E) occurs *hyper-intensionally*.
2. The meaning C is used to identify an object that is a function F (possibly a 0-ary one, which is a function without arguments). Now again there are two possible suppositions:
   (a) The function F itself is an object of predication; in this case we say that the meaning C (and thus also the expression E) is used *intensionally*.
   (b) The value of F is an object of predication; in this case we say that the meaning C (and thus also the expression E) is used *extensionally*.

Note that the notions 'intensionally' and 'extensionally' are used here in a broader sense than in possible-world semantics. Whenever confusion might arise, we will explicitly say PWS-intension. Using medieval terminology, we will also talk about *de dicto* and *de re* supposition, in case that a construction of a *PWS-intension* occurs *intensionally* and *extensionally*, respectively.

Thus we must distinguish between the context of *mentioning* a construction hyper-intensionally as an input/output object on which another construction operates and *using* a construction as a constituent of another construction in two different ways, either intensionally or extensionally. However, there is another complication here. A higher context is *dominant* over a lower one. It means that if a meaning C occurs extensionally as a constituent of another construction C' which in turn occurs intensionally (as a constituent of some D), then C occurs in C' (as well as in D) intensionally. And if a meaning C is used extensionally or intensionally as a constituent of another construction C' which in turn occurs hyper-intensionally (as mentioned in some D), then C occurs in C' (as well as in D) hyper-intensionally.

The three kinds of context are specified as follows[4]:

1. *Hyperintensional context*: the sort of context in which a construction is not used to v-construct an object. Instead, the construction itself is an argument of another function; the construction is just mentioned.
   Examples: Consider the sentence "Charles is solving the equation $2 + x = 7$". When Charles is looking for the solution of '$2 + x = 7$', he is not looking for the number 5. And though the solution of '$2 + x = 7$' is the same as, e.g., of '$13 - x = 8$', it does not follow that Charles is solving the latter equation. Thus the meaning of the solution of '$2 + x = 7$' is only mentioned here. It is predicated of this very meaning that Charles is striving to find the object constructed by this meaning. When evaluating the truth-conditions of this sentence, we do not solve the equation, it is Charles' matter.
   For another example, consider the sentence "Charles believes that the President of Finland is elected directly by public but does not believe that the Head of state of Finland is elected by public". Suppose that 'the President of

[4] An exact definition is out of the scope of this paper. See, however, [7], in particular Chapters 2.6 and 2.7.

Finland' and 'the head of state of Finland' denote one and the same office.[5] Then if 'the President of Finland' and 'the head of state of Finland' were used extensionally or intensionally, the sentence would be a contradiction. Yet it is not. Thus both the expressions, or rather their meanings, occur here hyper-intensionally.

2. *Intensional context*: the sort of context in which a construction C is used to v-construct a function but not a particular value of the function, and C does not occur within another hyperintensional context.
   Example: "Sinus is a periodical function". The object of predication is here the entire function *sinus* rather than its particular value. Thus the meaning of 'sinus' is used intensionally here, and the analysis comes down to $[{}^0Periodical\,{}^0Sinus]; Periodical/(o(\tau\tau)); Sinus/(\tau\tau)$.
   For a non-mathematical example, consider the sentence "The President of Finland is elected for the period of six years". The object of predication is the office of the president, that is the function of type $\iota_{\tau\omega}$. The sentence predicates of this office (rather than of its contingent holder, if any) that it has the property of being eligible for the period of six years. The sentence is true even if the office is vacant; its truth is established by the Finnish constitution. Hence the meaning of 'the President of Finland' occurs here with *de dicto* supposition.
3. *Extensional context*: the sort of context in which a construction C of a function is used to construct a particular value of the function at a given argument, and C does not occur within another intensional or hyperintensional context.
   Example: "$\sin(\pi) = 0$" expresses the Composition $[[{}^0Sinus\,{}^0\pi] = {}^00]$, where ${}^0Sinus$ occurs extensionally; the Composition is used to construct the value of the sinus function at the argument π of type τ.
   For a non-mathematical example, consider the sentence "The President of Finland is the first female holder of the office". Now the property of being the first female holder of the office is not predicated of the entire office, but of its present holder, i.e. the value of type ι of the function of type $\iota_{\tau\omega}$. Hence the meaning of 'the President of Finland' occurs here extensionally, with *de re* supposition.

To avoid a misconception, we want to stress that the specification of particular contexts in which a construction occurs, does not involve a reference shift or even a meaning shift, as Frege proposed. Our analysis is *anti-contextualistic*. This is to say that the meaning of an unambiguous expression is the same in all the contexts. Indeed, why should the meaning of 'the President of Finland' as used in the sentence "The President of Finland is a female" be different from the meaning of this very same expression as used in "Charles believes that the President of Finland is a female"? What is dependent on the context in which one and the same meaning occurs is the way we logically manipulate the respective construction. This we are going to demonstrate in the next section.

[5] This is a simplification, because it is true that the President of Finland is the Head of state of Finland, but if Finland were for instance a kingdom, then the head of state would be a king or a queen. Thus the Head of state of Finland is a requisite of the presidential office. Yet this minor simplification is irrelevant here.

3.2 Substitution and Leibniz's Law

The extensional, intensional and hyperintensional occurrences of constructions were introduced in order to define valid inference rules for TIL in its capacity as a hyperintensional logic of partial functions. Once the difference between mentioning and using a construction, and the difference between using a construction either intensionally or extensionally, have both been defined, the specification of the rules is smooth sailing. They can be formulated as follows.

Improperness (non-existence). A construction C v-constructing an entity of a type α can be v-improper only due to a constituent D occurring *extensionally* in C. This is to say that improperness stems from using Composition, which is the procedure of applying a function f to an argument; either f has a *value gap*, or Composition C does not obtain an argument to operate on because some of the constituents of C are v-improper. In this way *partiality is strictly propagated upwards*.

Existence. If a construction C is v-proper then all its constituents D_i occurring extensionally are v-proper as well. In other words, the respective values of the functions constructed by these constituents *exist*.

Leibniz's law of substitution.

Extensional context. A collision-less replacement of v-congruent constructions D, D' in C is valid for extensionally occurring constituents; constructions D, D' are v-congruent if they v-construct one and the same entity.

Intensional context. A collision-less replacement of equivalent constructions D, D' in C is valid for all constituents of C; constructions D, D' are equivalent if they v-construct one and the same entity for all valuations v.

Hyper-intensional context. A collision-less replacement of procedurally isomorphic constructions D, D' in C is valid for all sub-constructions of C; constructions D, D' are procedurally isomorphic if they v-construct one and the same entity for all valuations v in the same procedural way. More precisely, procedural isomorphism is defined as the transitive closure of α- and η-equivalence. For instance, the constructions ${}^0Prime$, $\lambda x[{}^0Prime\, x]$, $\lambda y[{}^0Prime\, y]$, $\lambda z[\lambda x[{}^0Prime\, x]\, z]$, are procedurally isomorphic, while $\lambda x[[{}^0Card \lambda y[{}^0Divide\, y\, x]] = {}^02]$ is only equivalent to them; it does construct the set of primes, but does so in a non-isomorphic manner.

Moreover, for v-proper constituents occurring extensionally, the classical extensional rules of inference (as for instance those of a sequent calculus) are valid.

To illustrate the rules, we are now going to analyze some of the examples adduced in the previous section.

Example 1 (hyper-intensional context) "Charles is solving the equation $2 + x = 7$". As always, we begin with assigning types to the objects that receive mention in the analyzed sentence: $Charles/\iota$; $Solve/(o\iota\star_n)_{\tau\omega}$; $2, 7/\tau$; $x \rightarrow \tau$. When solving the equation, Charles wants to find out which set (here a singleton) is constructed by the Closure $\lambda x[^0\!=[^0\!+{}^0\!2\,x]\,{}^0\!7]$. Thus he is related to the Closure itself rather than its product, a particular set. Otherwise the seeker would be immediately a finder and Charles' solving would be a pointless activity. The analysis comes down to

$$\lambda w\lambda t[^0Solve_{wt}\,{}^0Charles\,{}^0[\lambda x[^0\!=[^0\!+{}^0\!2\,x]^0\!7]]]. \tag{1}$$

Thus the following argument is *invalid*:

$$\frac{\begin{array}{c}\text{“Charles is solving the equation 2 + x = 7”}\\ \text{“The solution of 2 + x = 7 is equal to the solution of 13 - x = 8”}\end{array}}{\text{“Charles is solving the equation 13 - x = 8”}}$$

This is revealed by the analysis:

$$\frac{\begin{array}{c}\lambda w\lambda t[^0Solve_{wt}\,{}^0Charles\,{}^0[\lambda x[^0\!=[^0\!+{}^0\!2\,x]^0\!7]]]\\ \lambda x[^0\!=[^0\!+{}^0\!2\,x]^0\!7]=\lambda y[^0\!=[^0\!-{}^0\!13\,y]^0\!8]\end{array}}{\lambda w\lambda t[^0Solve_{wt}\,{}^0Charles\,{}^0[\lambda y[^0\!=[^0\!-{}^0\!13\,y]^0\!8]]]}$$

The construction $[\lambda x[^0\!=[^0\!+{}^0\!2\,x]\,{}^0\!7]]$ occurs in the first premise *hyper-intensionally*. Thus a substitution *salva veritate* is valid here only for *procedurally isomorphic* constructions. Yet the second premise guarantees only the *equivalence* of the two Closures; they construct the same set of numbers, but in a non-isomorphic way.

On the other hand, the Trivialization $^0[\lambda x[^0\!=[^0\!+{}^0\!2\,x]\,{}^0\!7]]$ is a constituent used in (1) intensionally. It can never be v-improper, and the following argument is valid:

$$\frac{\begin{array}{c}\text{“Charles is solving the equation 2 + x = 7”}\\ \lambda w\lambda t[^0Solve_{wt}\,{}^0Charles\,{}^0[\lambda x[^0\!=[^0\!+{}^0\!2\,x]^0\!7]]]\end{array}}{\begin{array}{c}\text{“There is something Charles is solving”}\\ \lambda w\lambda t\exists c[^0Solve_{wt}\,{}^0Charles\;c]\end{array}}$$

The variable c is ranging over $\star_1$.
Proof. Let $Proper/(o\star_n)$ be the class of constructions that are not v-improper for any valuation v. Then in any world w at any time t the following steps are truth-preserving:

$$\begin{array}{rl}[^0Solve_{wt}\,{}^0Charles\,{}^0[\lambda x[^0\!=[^0\!+{}^0\!2\,x]^0\!7]]] & \text{assumption}\\ [^0Proper_{wt}\,{}^0[\lambda x[^0\!=[^0\!+{}^0\!2\,x]^0\!7]]] & \text{the rule of improperness}\\ \exists c[^0Solve_{wt}\,{}^0Charles\;c] & \text{existential generalisation}\end{array}$$

Example 2 (intensional context) “Charles wants to be The President of Finland”.
Types. $Charles/\iota$; $Want\_to\_be/(o\iota_{\tau\omega})$; $President\_of/(\iota\iota)_{\tau\omega}$; $Finland/\iota$

$$\lambda w\lambda t[^0Want\_to\_be_{wt}\,{}^0Charles\;\lambda w\lambda t[^0President\_of_{wt}\,{}^0Finland]]. \tag{2}$$

The Closure $\lambda w\lambda t[^0President\_of_{wt}\,{}^0Finland]$ occurs intensionally, i.e. with *de dicto* supposition, because it is not used in (2) to v-construct the holder of the office (particular individual, if any). Thus the following argument is *invalid*:

$$\frac{\begin{array}{c}\text{“Charles wants to be the President of Finland”}\\ \text{“The President of Finland is the first female holder of the office”}\end{array}}{\text{“Charles wants to be the first female holder of the office”}}$$

The analysis reveals the invalidity of the argument:

$$\frac{\begin{array}{c}\lambda w\lambda t[{}^0Want\_to\_be_{wt}\,{}^0Charles\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]]\\ \lambda w\lambda t[{}^0\!=\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}\\ [{}^0First_{wt}\,\lambda x\,[[{}^0Female_{wt}\,x]\wedge[{}^0\!=\,x\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}]]]]\end{array}}{\begin{array}{c}\lambda w\lambda t[{}^0Want\_to\_be_{wt}\,{}^0Charles\\ \lambda w\lambda t[{}^0First_{wt}\lambda x\,[[{}^0Female_{wt}\,x]\wedge[{}^0\!=\,x\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}]]]\end{array}}$$

Additional types: $x \to \iota$; $Female/(o\iota)_{\tau\omega}$; $First/(\iota(o\iota))_{\tau\omega}$: the function that selects from the set of individuals the only individual that is the first one at a given $\langle w, t\rangle$ of evaluation.

The argument is obviously invalid, because $\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]$ occurs in the first premise with *supposition de dicto*, i.e. intensionally, while the second premise guarantees only v-congruence of this Closure with $\lambda w\lambda t[{}^0First_{wt}\lambda x\,[[{}^0Female_{wt}\,x]\wedge[{}^0\!=\,x\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}]]]$, i.e. a contingent co-occupancy of the two offices, rather than equivalence needed for a valid substitution.

Example 3 (extensional context) "The President of Finland is watching TV". The analysis of this sentence comes down to the Closure

$$\lambda w\lambda t[{}^0Watch_{wt}\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}\,{}^0TV] \tag{3}$$

Additional types: $Watch/(o\iota\iota)_{\tau\omega}$; TV/ι. The meaning of 'the President of Finland' occurs with *de re* supposition in (3), i.e. extensionally. Thus we can apply the extensional rules that are also known as *two principles de re*. They are the *Principle of existential presupposition* and the *Substitutivity of co-referential expressions*. The following arguments are valid:
Argument 1:

$$\frac{\text{“The President of Finland is watching TV”}}{\text{“The President of Finland exists”}}$$

$$\frac{\lambda w\lambda t[{}^0Watch_{wt}\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}\,{}^0TV]}{\lambda w\lambda t[{}^0Exist_{wt}\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]]}$$

Argument 2:

$$\frac{\begin{array}{c}\text{“The President of Finland is watching TV”}\\ \text{“The President of Finland is Tarja Halonen”}\end{array}}{\text{“Tarja Halonen is watching TV”}}$$

$$\frac{\begin{array}{c}\lambda w\lambda t[{}^0Watch_{wt}\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}\,{}^0TV]\\ \lambda w\lambda t[{}^0\!=\,\lambda w\lambda t[{}^0President\_of_{wt}\,{}^0Finland]_{wt}\,{}^0Halonen]\end{array}}{\lambda w\lambda t[{}^0Watch_{wt}\,{}^0Halonen\,{}^0TV]}$$

Here are the proofs:
(*Ad Argument 1*) First, existence is here a property of an individual office rather than of some non-existing individual, whatever that would mean. Thus we have $Exist/(o\iota_{\tau\omega})_{\tau\omega}$. To prove the validity of the argument, we define $Exist/(o\iota_{\tau\omega})_{\tau\omega}$ as the property of an office of being occupied at a given world/time pair:

$${}^0Exist =_{of} \lambda w\lambda t\lambda c[{}^0\exists\lambda x[x = c_{wt}]],\ \text{i.e.,}\ [{}^0Exist_{wt}\,c] =_o [{}^0\exists\lambda x[x = c_{wt}]]$$

Types: $\exists/(o(o\iota))$: the class of non-empty classes of individuals; $c \rightarrow_v \iota_{\tau\omega}$; $x \rightarrow_v \iota$; $=_o\ /(ooo)$: the identity of truth-values; $=_{of}\ /(o(o\iota_{\tau\omega})_{\tau\omega}(o\iota_{\tau\omega})_{\tau\omega})$: the identity of individual-office properties.

Let $=_i\ /(o\iota\iota)$ be the identity of individuals, $Empty/(o(o\iota))$ the singleton containing the empty set of individuals and $Improper/(o\star_1)_{\tau\omega}$ the property of constructions of being v-improper in a given $\langle w,t\rangle$-pair, the other types as above. Then in any $\langle w,t\rangle$ the following proof steps are truth-preserving:

$[{}^0Watch_{wt}\ \lambda w\lambda t[{}^0President\_of_{wt}\ {}^0Finland]_{wt}\ {}^0TV]$	assumption
$\neg[{}^0Improper_{wt}\ {}^0[[\lambda w\lambda t[{}^0President\_of_{wt}\ {}^0Finland]]_{wt}]]$	def. of Composition
$\neg[{}^0Empty\ \lambda x[x =_i [\lambda w\lambda t[{}^0President\_of_{wt}\ {}^0Finland]]_{wt}]]$	obvious from the prev. step
$[{}^0\exists\lambda x[x =_i [\lambda w\lambda t[{}^0President\_of_{wt}\ {}^0Finland]]_{wt}]]$	existential generalisation
$[{}^0Exist_{wt}\ [\lambda w\lambda t[{}^0President\_of_{wt}\ {}^0Finland]]]$	by def of Exist

(*Ad Argument 2*) substitution:

$[{}^0Watch_{wt}\ \lambda w\lambda t[{}^0President\_of_{wt}\ {}^0Finland]_{wt}\ {}^0TV]$	assumption
$[{}^0{=}\ \lambda w\lambda t[{}^0President\_of_{wt}\ {}^0Finland]_{wt}\ {}^0Halonen]$	assumption
$[{}^0Watch_{wt}\ {}^0Halonen\ {}^0TV]$	substitution of identicals

Note that if the President of Finland does not exist, then neither "The President of Finland is watching TV" nor the negated "The President of Finland is *not* watching TV" have any truth-value. This is due to compositionality and the extensional rule for existence. In both sentences 'the President of Finland' occurs extensionally. Thus if one of these sentences (either positive or negative one) is True, the President of Finland exists. As a consequence, if the president does not exist, then *neither* of these sentences is true. Hence, both sentences have *no* truth value, which is just the *Principle of exitential presupposition*: the existence of the president is not only entailed but also presupposed.[6]

4 Conclusion

Partiality, as we know all too well, is a complicating factor. Yet we are convinced that logic should assist in unearthing the objective structures underlying the expressions of a given language. In order to reflect 'gaps in reality' faithfully (i.e., to obtain a counterpart of Bolzano's *Gegenstandslosigkeit*), TIL adopts *properly partial functions* and *improper constructions*. In short, part of the task of a logician must be to adequately model the semantic features of (fragments of) a given language even at the cost of incurring technical complications. This explains why we are not going to join the game of playing fast and loose with existing logical symbols in order to define new *ad hoc* connectives and 'entailment relations' so as to either preserve or invalidate this or that commonly accepted law. Instead, we deploy methods that overcome these technical complications and are at the same time in full accordance with the principles of TIL as outlined in this paper.

[6] This is valid in case 'the President of Finland' is the topic of the sentence about which the property of watching TV (focus) is predicated. Yet there is another reading with 'TV' occurring as the topic and 'president of Finland' occurring in the focus clause. Then the existence of the president is only entailed. For details on Topic-Focus ambiguities see [6] and also [9].

Yet the theoretical specification of particular rules is only the first step. When making these features explicit we keep in mind an *automatic* deduction that will make use of these rules. To this end we currently develop a computational variant of TIL, the functional programming language *$\mathcal{T}$IL-Script* (see [2]). The direction of further research is clear. We are going to continue the development the *$\mathcal{T}$IL-Script* language in its full-fledged version equivalent to TIL calculus.

Acknowledgements. This research has been supported by the Grant Agency of the Czech Republic, projects GACR 401/07/0451, 'Semantization of Pragmatics' and 401/09/H007 'Logical Foundations of Semantics', and also by the internal grant agency of FEECS VSB-TU Ostrava - IGA 22/2009, 'Modeling, simulation and verification of software processes'.

References

1. Carnap, R.: *Meaning and Necessity*. Chicago: Chicago University Press, 1974.
2. Ciprich, N., Duží, M. and Košinár, M.: The *$\mathcal{T}$IL-Script* language. In *Information modelling and Knowledge Bases XX*, Y. Kiyoki, T. Tokuda, H. Jaakola, X. Chen, N. Yoshida (eds.), Amsterdam: IOS Press, pp. 166–179, 2009.
3. Cook, S. A.: The Complexity of Theorem-Proving Procedures. In: *STOC '71*: Proceedings of the 3rd annual ACM symposium on Theory of computing. New York, NY, USA, pp. 151–158, ACM Press, 1971.
4. Duží, M.: Semantic pre-processing of anaphoric references. In Proceedings *RASLAN 2007*, P. Sojka and A. Horák (eds.), Masaryk University Brno, pp. 43–56, 2007.
5. Duží, M.: Intensional logic and the irreducible contrast between de dicto and de re, *ProFil 5*, pp. 1–34. 2004, Retrievable at `http://profil.muni.cz/01_2004/duzi_de_dicto_de_re.pdf`
6. Duží, M.: Topic-Focus articulation from the semantic point of view. In *Computational Linguistics and Intelligent Text Processing*, ed. A. Gelbukh, pp. 220–232, Springer, LNCS 5449, 2009.
7. Duží, M., Jespersen B., Materna P.: *Procedural Semantics for Hyperintensional Logic; Foundations and Applications of Transparent Intensional Logic.* Series Logic, Epistemology and the Unity of Science. Springer, Berlin (forthcoming).
8. Frege G., Über Sinn und Bedeutung, *Zeitschrift für Philosophie und philosophische Kritik*, vol. 100, pp. 25–50, 1892.
9. Hajičová, E.: What we are talking about and what we are saying about it. In *Computational Linguistics and Intelligent Text Processing*, LNCS Springer Berlin/Heidelberg, vol. 4919, pp. 241–262, 2008.
10. Moschovakis, Y. N.: Sense and denotation as algorithm and value. In: J. Väänänen, J. Oikkonen (eds.), *Lecture Notes in Logic*, vol. 2, Berlin: Springer, pp. 210–249, 1994.
11. Tichý, P. *The Foundations of Frege's Logic*. Walter de Gruyter, Berlin-New York, 1988.
12. Tichý, P. (2004): *Collected Papers in Logic and Philosophy*, V. Svoboda, B. Jespersen, C. Cheyne (eds.), Prague: Filosofia, Czech Academy of Sciences, and Dunedin: University of Otago Press.

Temporal Aspects of Knowledge and Information

Marie Duží[1], Aleš Horák[2], and Pavel Materna[3]

[1] VŠB–Technical University of Ostrava
17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
marie.duzi@vsb.cz
[2] Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
hales@fi.muni.cz
[3] Institute of Philosophy of the Academy of Sciences of the Czech Republic, v.v.i.
Jilská 1, 110 00 Prague 1, Czech Republic

Abstract. The authors of the paper have proposed a collaborative research project that should start in 2010 and proceed in three successive years. The goal of the proposed project is to develop tools and mechanisms for computer-aided natural-language analysis, knowledge management and reasoning. In a broader sense, the main objective is to propose a new solution to natural language analysis and reasoning based on the procedural semantics of Transparent Intensional Logic (TIL). Primarily, our research is classified in the area of philosophical logic. Subsidiary works then fall under theoretical linguistics and computer science. Our research is focused on the temporal, modal and epistemic aspects of knowledge representation and reasoning. The main problems under our scrutiny are: TIL proof calculus for hyperintensional, partial, typed lambda calculus; analysis of tenses and temporal logic; analysis of epistemic verbs and events; analysis of anaphora references and topic-focus articulation. Concerning practical applications, we develop functional programming language TIL-Script, the computational variant of TIL. The main result we aim to achieve is computer-aided analysis of natural-language texts based on web ontologies like WordNet, FrameNet/VerbNet, VerbaLex and large text corpora.

Key words: Transparent intensional logic; TIL; temporal knowledge

1 State-of-the-art and objectives of the project

In the age of computers there is a new challenge that might be characterised as the problem of human-computer communication, or rather *human communication via computers*. Currently, communication *via* computers is mostly computer driven. Humans must be able to formulate quasi-logical queries, and the answer provided by a computer is far from being comprehensible and natural-language like. The World Wide Web (www) media became a prevailing source of *information* and *knowledge*. The most common format of web documents, HTML, focuses on the *form* of knowledge representation rather than its content.

While this format standardisation was undoubtedly a necessary and important step, the *semantic content* of a HTML document is what we should focus on now. Otherwise the human-computer communication and web-information processing are limited and do not serve the goal.

An improvement of this situation comes with the Semantic Web.[4] Semantic Web is a vision of web data defined and linked in such a way that it can be processed and understood by machines, not just by humans. Though the key idea of Semantic Web, i.e. meta-data, is an old one, still only a negligible portion of today's web pages make use of meta-data techniques. The semantic web approach currently offers well-defined static features, e.g. ontologies, but the inference mechanisms and *temporal features* are in need of scrutiny. The evolution of the web itself shows that the vision of Semantic Web cannot be realised unless intelligent automatic procedures are developed in order to analyse and transform natural-language knowledge into a Semantic Web representation and *vice versa*. The goal of our project is to fill this gap.

The goal of the proposed project is to develop tools and mechanisms for a computer-aided natural-language analysis, knowledge management and reasoning. In a broader sense, the main objective is to propose *a new solution to natural language analysis and reasoning*.

To this end we will make use of the expressive system of *Transparent Intensional Logic* (TIL).[5] The logical system we vote for makes it possible to explicitly formally specify and process all kinds of knowledge including difficult natural-language phenomena such as temporal relations, modalities, personal attitudes to other knowledge, and anaphoric references. This is due to the *procedural semantics* of TIL. From the theoretical point of view, much has been done in the area of logical analysis of natural language, as the results of the previous projects testify. Yet the problem of *tenses* and generally *temporal logic* has been rather neglected till now in TIL-like analyses, though due to numerous applications there is a pressing need for such analyses. The only TIL paper that dealt with the analysis of tenses was a pioneer paper of Pavel Tichý [12]. Thus we will focus on *temporal aspects* of knowledge representation and reasoning. Another area under scrutiny is the analysis of *context-dependent ambiguities* such as topic-focus articulation and anaphora.[6] Here we will make use of the results of *linguistic analysis*, because theoretical linguistics and logic must collaborate and walk hand in hand.

The character of the proposed project is *interdisciplinary*. It combines *logical analysis of natural language* with *theoretical linguistics* and *computer science.*

1.1 Natural Language Analysis and Knowledge representation

Natural language analysis has a long tradition, beginning with Frege, Chomsky, Carnap, Montague and many others. However, it became a hot topic as soon as the first real-world computer applications came into being. The possibility of an automatic "intelligent" processing of the huge amount of texts that people

[4] See [1,6]. [5] See, for instance, [8,9,14,15,7]. [6] See [2].

have produced is a great contribution to the way people can make use of computers. Intelligent processing techniques usually analyse natural language sentences on several consecutive levels, to name at least the most important of these, namely morphological, syntactic and semantic levels. Morphological and syntactic natural language analysis is currently well-developed and described in numerous papers. Thus we may say that the automatic techniques dealing with morphology and syntax produce plausible results (at least regarding frequently used languages). However, this is not the case of *semantic analysis* where the results are far from being satisfactory. Logical tools that are broadly applied are mostly based on the first-order logic (FOL) paradigm. FOL tools have been primarily developed for mathematics, where they are successfully applied. But natural language is too rich to be encoded by a first-order formalism.

In the last decades numerous methods of natural-language analysis and knowledge representation have been developed. The approaches range from encoding knowledge in the form of some strictly specified procedures in a programming-like language through semantic networks and frames up to statistical and probabilistic methods. However, none of these approaches is universal and fine-grained enough to make it possible to specify all the semantically salient features of natural language expressions. Moreover, there is a pressing need for a unique knowledge representation framework. *The goal of the project is to demonstrate that TIL can serve as such a fine-grained universal framework.*

Concerning temporal aspects of knowledge, we have to take into account that sentences in the present, past and future tenses have different truth-conditions. This fact has been observed by numerous logicians, and many variants of so-called *temporal logic* have been developed. These formal systems are mostly viewed as a special case of modal logic interpreted by means of Kripkean possible-world semantics. Another view of temporal logic is motivated by the logical analysis of natural language. This approach develops temporal logic as a formalization of linguistic conventions with respect to tenses, and it has been applied in particular by Arthur Prior who developed the axiomatic system of so-called *tense logic*. Prior introduced operators like P (*weak past*, corresponding to the modal operator $\Diamond$) and H (*strong past*, corresponding to $\Box$), whose intended meanings are:

$P(q)$ – "It has at some time been the case that q".

$H(q)$ – "It has always been the case that q".

Subsequently, systems of temporal logic have been further developed by computer scientists, notably Zohar Manna and Amir Pnueli, and widely used for formal verification of programs and for encoding temporal knowledge within artificial intelligence. Despite the great applicability of particular variants of tense logic in the semantics of programming languages, the systems just mentioned suffer a drawback when applied to the semantics of natural language. The drawback is their inability to adequately analyse sentences indicating a ‘point of reference’ referring to the interval when the sentence was or will be true. Such sentences come attached with a *presupposition* under which a sentence is true or false. To illustrate the problem, consider the sentences

"Tom is sick".

"Tom has been sick".

"Tom was sick throughout the year 2008".

"The Mayor of Dunedin was sick throughout the year 2008".

The first two sentences do not come with a presupposition. They ascribe to Tom the property of being sick and of having been sick, respectively. They are true or false according as Tom has the relevant property. However, the truth-conditions of the third sentence depend not only on whether Tom has the property of being sick throughout the year 2008, but also on the time at which the sentence is evaluated. If T is the time of evaluation, then the truth-conditions are specified as follows:

(a) *If* $T \leq$ December 31, 2008, 24:00, *then* ***no value*** *(fail), else (b);*
(b) *If* the whole year 2008 precedes T (i.e., $T >$ December 31, 2008, 24:00), *then if* Tom was sick at all times during 2008, *then* the value ***True***, *else* the value ***False***.

The fourth sentence is ambiguous. Its ambiguity is pivoted on the difference between *de dicto* and *de re*. On its *de re* reading, the truth-conditions are identical to (a), (b), provided there is a Mayor of Dunedin. If the office of Mayor of Dunedin is vacant, the sentence has *no* truth-value. The *de dicto* reading of the sentence requires us to evaluate the proposition *in the past* according as the office was occupied throughout the year 2008, and according as the Mayor *had been* sick throughout the year 2008. Thus it is irrelevant whether there is a Mayor at the present time. Our analysis must respect these truth-conditions. Tichý's solution in [12] is difficult to understand, because Tichý applies the *singulariser* function to a singleton typed as containing a truth-value in order to make the set fail to deliver a truth-value in case the associated presupposition is not satisfied. Tichý's analysis is analogous to what the computer scientist would call an *imperative* rather than *declarative* analysis. The downside to an imperative analysis is that it may conceal flaws that rear their head only when the analysis is applied to extreme situations. Yet there is an elegant alternative that makes use of the 'if-then-else' connective. There has been much dispute over the semantics of 'if-then-else' in the logic of computer science, and it is often said that *if-then-else* is a non-strict function that does not behave in compliance with the compositionality principle. For instance, the application of *if-then-else* to a condition P and two formulae F_1 and F_2 is not improper (by failing to produce a product) even when F_2 is improper whenever P is true. However, there is no cogent reason to settle for non-strictness. The *procedural* semantics of TIL enables us to specify a strict definition of *if-then-else* that meets the compositionality constraint. The definition of "If P then F_1, else F_2" is a procedure that decomposes into two phases. First, on the basis of the condition P, select one of F_1, F_2 as the procedure to be executed. Second, execute the selected procedure. Thus, for instance, if P is true then F_1 is executed rather than F_2, and the possible improperness of F_2 does not matter.

1.2 Inference

There are a huge amount of systems of inference developed within contemporary logic. Beginning with the systems of 'classical logic', like general resolution method or Gentzen-like systems, there are inference systems based on the so-called 'non-classical logics'.What is important from our viewpoint is that many particular logics build up their apparatus dependently on a particular problem so that the notions necessary for solving such a problem are introduced *ad hoc*, even if they could be defined in terms of a more general system. Such are the 'of-logics' ("logic of events", "epistemic logics" (i.e. logic of knowledge), "logic of questions", "deontic logic" (i.e., logic of norms) etc.).

TIL offers a unique approach, which helps to find the *meaning-driven* logical tools for solving particular problems on the basis of a homogeneous general conception. Thus in comparison with logics based on a distinct philosophy TIL can demonstrate that its philosophy yields greater expressivity. Logics that do not recognize hyper-intensions (such as procedural constructions, i.e., the great majority of logics) do not adequately solve for example inferences in the milieu of propositional attitudes. This is also the case of the most important rival logic, viz. Montague's IL.[7] There are several shortcomings of Montague's IL. Summarising briefly. First, IL fails to satisfy Church-Rosser property. The reason typically cited for the logic of IL displaying such a deviant behaviour is that the logic has been designed to reflect opacity phenomena of natural language. But this is actually a serious deficiency caused by *ad hoc* introducing the operators $^\wedge$, $^\vee$, F and P that are equipped with hidden 'ghost' variables ranging over possible worlds and times. TIL explicit intensionalisation and temporalisation makes it possible to uniquely adhere to anti-contextualism. Second, the functions of IL are restricted to total functions. But we need to work with partial functions, unless we are content with an unmanageable explosion of domains. Third, functions typically have more than one argument. Usually we are told that these $n-$ary functions can be *represented* by unary composite functions. Schönfinkel in [11] observed that there is a one-to-one isomorphic correspondence between $n-$ary functions and certain unary composite functions.However, this isomorphism breaks down when *properly* partial functions are involved, as Tichý showed in [13], see [15, pp. 467–8]. A final objection to IL is that it fails to accommodate hyperintensionality, as indeed any formal logic interpreted set-theoretically is bound to fail to. Only when we embrace an algorithmic/procedural semantics are we able to handle structured, hyperintensional meanings of natural-language expressions. And we argue that any theory of natural-language analysis needs a hyperintensional (preferably procedural) semantics in order to render synonymy in natural language accurately, as well as to adequately analyse hyperintensional (*de dicto/de re*) attitude reports, the phenomena of anaphora, tenses, presuppositions, and other hard cases.

The central notion of logical semantics, viz. *entailment*, is exactly defined in terms of constructions. Moreover, structured meanings known as TIL constructions make it possible to precisely distinguish between analytical and

[7] See [10].

logical validity of sentences as well as of arguments, and examine the *analytical content* of these. Thus the 'scandal of deduction', as Hinttikka formulates the failure of logical deduction to explain the utility of a valid argument, is easily solved away.[8]

2 Methodology and project planning

The project work will run in parallel in three cooperating and partly overlapping groups:

1. *TIL theoretical backgrounds* group
2. *TIL logical analysis* group
3. *TIL inference machine* group

The group of TIL theoretical backgrounds will concentrate on extensions and specifications of the current TIL theory for the purposes of the project. Summary of the milestones and topics of the project plan is stated in Table 2.

2.1 Logical Analysis of Natural Languages

The main goal of the part of logical analysis is *natural-language text processing*. We aim at the development of automatic facilities to process real-world texts in natural languages (in particular English and Czech) in order to obtain a machine-readable form of the formal representation of sentences. This logical representation will then serve as an initial stock of knowledge from which inferable knowledge will be computed and deduced by the TIL Inference Machine.

Our method of analysis consists of three steps:

(a) *Type-theoretical analysis*, i.e., assigning types to all the objects talked about by the analysed sentence, i.e. denoted by semantically self-contained subexpressions of the analysed expression.
(b) *Synthesis*, i.e., combining the constructions of the objects *ad* (a) in order to construct the entity denoted by the whole expression.
(c) *Type-theoretical checking*, i.e., checking the type-theoretical consistency of the resulting analysis.

Within the project we will advance the fundamentals of TIL theory in particular with respect to temporal, modal and epistemic aspects. We will develop logical modules as semantic upgrades of the currently implemented tool for the full-fledged syntactic analysis of natural languages called *synt*.[9] This tool has been developed by the NLP Centre of the Faculty of Informatics, Masaryk University in Brno within the group of A. Horák, and it has been thoroughly tested on the samples of Czech and English texts. The *synt* system will be further enhanced by the new techniques and algorithms defined by the *theoretical backgrounds* group.

[8] Cf. Cohen-Nagel's 'paradox of inference'. [9] See [4,5].

Table 1. Summary of the project planning.

Milestones/topics	*Theoretical results*	*Applications*	
		Logical analysis	Inference
1st year (2010) Simple sentences in present, past and future tenses	– specification of TIL calculus – reference corpus of TIL constructions – type classification of basic lexicon tokens	*Computer-aided analysis* of simple sentences in past, present and future tense containing selected verbs	In the scope of FOL (enriched by explicit intensionality and temporality)
2nd year (2011) Complex sentences in present, past and future tenses	– analysis of events – analysis of grammatical tenses – type classification of attitudes	*Computer-aided automatic analysis* of relative time-related subordinate sentences	In the scope of classical typed λ-calculus
3rd year (2012) Context dependencies	– inference with background and common-sense knowledge – Analysis of sentence context and discourse	*Computer-aided analysis* of complex sentences with temporal events including direct speech	TIL inference machine including partiality and hyperintensional features

The implemented system will be tested on real texts from various domains. One specific domain we have at our disposal is the domain of medicine texts. They are available at Masaryk Memorial Cancer Institute in Brno, where the system will be used for advanced question-answering techniques in cooperation with the Masaryk University NLP Centre and the Digital Enterprise Research Institute, Ireland. We will pay attention to the multilingual features of the systems. Particular subtasks of the proposed project include constructions of logical lexicons for natural language based on previous works on the subject. These lexicons include WordNet semantic networks, verb frame lexicons, rules for building the meaning representation including meta-knowledge (knowledge about knowledge).

2.2 TIL Inference Machine – Theory and Development

From the formal point of view, TIL is a partial, hyper-intensional typed lambda calculus. Pavel Tichý in [13] specified inference rules only for the pre-1988 simpler version of TIL that was based on the simple theory of types. In order to deal with agents' (*de dicto*/*de re*) hyper-intensional attitudes in an appropriate way, we need to work in the full-fledged TIL, i.e., with the post-1988 version based on the ramified theory of types. In this version constructions are full-fledged objects *sui generic*. They can be not only used to identify an object, but also mentioned as objects of predication. Thus we need and have to develop and specify the TIL *inference machine* in full. From the theoretical point of view, such a calculus specification has been presented in [3].

Concerning applications, the computational variant of TIL, viz. the functional programming language *TIL-Script*, is currently being developed. *TIL-Script* is a FIPA compliant language. The development of TIL-Script as well as ontology languages is still a work in progress. Currently we combine traditional tools and languages like OWL (Ontology Web Language), logic programming inference tools (Prolog) and FOL proof calculi (Gentzen system and natural deduction) with the full-fledged features of TIL-Script by building transcription bridges. Thus the implementation of TIL-Script inference machine proceeds in stages. In the first stage we implemented the subset of language corresponding to the expressive power of Horn clauses. Then we extend it to the full FOL inference machine. The next stage is to implement the inference machine for the subset of classical λ-calculi, and finally, the hyper-intensional features and partiality are to be taken into account.

3 Conclusions

We have described the aims and objectives of the on-coming collaborative research project of VŠB-Technical University of Ostrava, Masaryk University in Brno and the Institute of Philosophy of the Academy of Sciences of the Czech Republic, Prague. The project will concentrate on further investigations and development of both the theory and applications of the Transparent Intensional Logic (TIL) with the focus on temporal aspects of knowledge and language. We believe that this project will be another successful step on the way to full natural language semantics by means of automatic computer-based logical systems.

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, by the Czech Science Foundation under the project 401/07/0451 (Semantization of pragmatics), and by the internal grant agency of FEECS VSB-TU Ostrava – IGA 22/2009 (Modeling, simulation and verification of software processes).

References

1. Berners-Lee, T., Hendler, J. and Lassila, O. (2001): The semantic web. *Scientific American*, May 2001.
2. Duží, M. (2009): Topic-Focus Articulation from the Semantic Point of View. In *CICLing 2009*. Ed. Gelbukh Alexander, Berlin Heidelberg: Springer-Verlag LNCS, vol. 5449, pp. 220–232, 2009.
3. Duží, M., Jespersen, B., Materna, P.: *Procedural Semantics for Hyperintensional Logic – Foundations and Applications of Transparent Intensional Logic*. Series Logic, Epistemology and the Unity of Science. Springer, Berlin, forthcoming.
4. Horák, A. (2002): *The Normal Translation Algorithm in Transparent Intensional Logic for Czech*. Ph.D. thesis, Faculty of Informatics, Masaryk University, Brno, 2002.
5. Horák, A., Kadlec, V. (2005): New Meta-grammar Constructs in Czech Language Parser synt. *Lecture Notes in Artificial Intelligence*, Berlin: Springer-Verlag, 3658/2005, 1, pp. 85–92, 2005.
6. Horrocks, I. and Patel-Schneider, P.F. (2003): Three Theses of Representation in the Semantic Web. WWW2003, May 20–24, Budapest, Hungary, `http://www2003.org/cdrom/papers/refereed/p050/p50-horrocks.html`.
7. Jespersen, B. (2005): 'Explicit Intensionalisation, Anti-Acatualism, and How Smith's Murderer might Not Have Murdered Smith'. *Dialectica* Vol. 59, No. 3, 285–314.
8. Materna, P. (1998): *Concepts and Objects*, Acta Philosophica Fennica, vol 63. The Philosophical Society of Finland, Helsinki.
9. Materna, P.(2004): *Conceptual Systems.* Logos Verlag, Berlin.
10. Montague, R. (1974): 'The proper treatment of quantification in ordinary English', in: *Formal Philosophy: Selected papers of Richard Montague,* Thomason, R. (ed.), New Haven: Yale University Press, pp. 247–170.
11. Schönfinkel, M. (1924): 'Über die Bausteine der mathematischen Logik', *Mathematische Annalen*, vol. 92, pp. 305–316.
12. Tichý, P. (1980): 'The logic of temporal discourse', *Linguistics and Philosophy*, vol. 3, pp. 343–369, reprinted in Tichý (2004), pp. 373–69.
13. Tichý, P. (1982): 'Foundations of partial type theory', *Reports on Mathematical Logic*, vol. 14, pp. 52–72, reprinted in: Tichý (2004), pp. 467–80.
14. Tichý, P. (1988): *The Foundations of Frege's Logic*, Berlin, New York: De Gruyter.
15. Tichý, P. (2004): *Collected Papers in Logic and Philosophy*, V. Svoboda, B. Jespersen, C. Cheyne (eds.), Prague: Filosofia, Czech Academy of Sciences, and Dunedin: University of Otago Press.

The Saara Framework
An Anaphora Resolution System for Czech

Vašek Němčík

NLP Laboratory, Faculty of Informatics
Masaryk University Brno, Czech Republic
`xnemcik@fi.muni.cz`

Abstract. Determining reference and referential links in discourse is one of the biggest and most important challenges in natural language understanding. In particular, computing coreference classes over the set of referring expressions in text is crucial for its further syntactic and semantic processing. We present a system for automatic anaphora resolution that can be used on arbitrary texts in Czech. The article describes the individual phases of processing the input text and mentions selected issues that need to be addressed by the system.

Key words: anafora; anafora resolution; Czech language

1 Introduction

In this work, we present Saara (System for Automatic Anaphora Resolution and Analysis), a framework for anaphora resolution (AR) which is modular in many ways. Modularity in the context of AR has many obvious advantages. It allows defining various AR algorithms and using them for different purposes. Given a corpus annotated for coreference is available, it is possible to evaluate them, compare their strong and weak points and based on this knowledge, define and test more sophisticated ones. It is also possible to experiment with algorithms across genres or even languages.

In this paper, we mainly focus on utilizing the Saara framework as a part of a Natural Language Processing (NLP) system dealing with unrestricted Czech text on input. This mainly involves suitably combining it with pre-processing tools that perform the necessary linguistic analysis of the input text. These yield information required by AR algorithms to model phenomena relevant for anaphoric relations.

To our knowledge, there is currently no other AR system for Czech that can be straightforwardly used in an application setting to deal with texts that haven't been pre-processed manually. The only other AR system applicable to Czech data was presented by Linh [1], and to my knowledge, it can be used only with data manually annotated according to the three-layer formalism used within the Prague Dependency Treebank [2,3]. Our work aims at reaching a system that can used with arbitrary, unedited plain text. This addresses a crucial bottleneck in the practical applicability of NLP systems for Czech.

In the next section, we describe the linguistic pre-processing yielding the underlying linguistic analysis. Section 3 sketches the architecture of the Saara framework performing AR itself, and next, Section 4 addresses the issues relevant to the synthesis of all the modules mentioned. Finally, we present a summary of the work presented and discuss directions of future work.

2 Syntactic Analysis

Like any higher-level linguistic processing of texts, anaphora resolution within our system requires support in lower-level analysis – especially in information about morphological and syntactic structure.

As the first step, the input text is tokenized and further processed by a morphological tagger, which carries out automatic morphological analysis and disambiguation. The tools used have been developed at the NLP Center, MU Brno. Notably, the morphological tagger is based on the moprhological analyzer ajka [4] and the chunk parser VaDis [5].

The subsequent syntacic analysis is performed using the "synt" syntactic analyzer developed by at the NLP Laboratory, FI MU, Brno [6]. The parsing is carried out in a head-driven chart-parse manner with context-free rules defined in three forms: G1, G2, and G3. G1 is a meta-grammar edited by human experts, mainly taking care of the combination of phrases, especially verbal ones. The Second Grammar Form, G2 contains also a description of context actions associated with individual G1 grammar rules. These prune combinations where conditions on agreement in grammatical categories are not met. The Expanded Grammar Form, G3 contains all necessary feature agreement tests as context-free rules.

The whole process yields a number of most probable phrasal derivation trees. These are selected and ordered using statistics concerning probability of the individual analyses and semantic features, such as verb valencies.

The following section describes how this computed structure is further utilized to reach information about anaphoric relations.

3 The Saara Framework

At present, mechanisms for performing anaphora resolution are becoming integral parts of modern NLP systems. Disregarding anaphora resolution inevitably means creating a serious bottleneck within the linguistic analysis process.

For Czech, various AR algorithms have been proposed (e.g. [7,8,9]), however, due to the inavailability of suitable Czech linguistic resources, haven't been implemented. The emergence of the Prague Dependency TreeBank [2,3], which contains annotation of pronominal coreference led to the occurrence of two AR systems: the above-mentioned AČA system by Linh [1], and the Saara Framework [10] presented below.

The architecture of the Saara Framework has been greatly inpired by earlier AR systems, especially the one developed by Byron and Tetreault [11] at the University of Rochester. They emphasise the advantages of modularity and encapsulation of the system modules into layers. Themselves, they propose three layers:

– the AR layer containing functions addressing AR itself,
– the translation layer for creating data structures,
– the supervisor layer for controlling the previous layers.

The Saara Framework exhibits a very similar distinction of processing layers. There is the so-called *"markable layer"* which is used to define the actual AR algorithms. The main feature of this layer is its maximal generality. It has access only to a general discourse model consisting of the basic discourse structure, the so-called markables, representing discourse objects and a limited number of interface functions describing relationships between them. Next, there is the *"technical layer"* which describes the actual representation of the text, in the particular formalism and format used. Furher, it encompasses the implementations of the functions from the markable layer, translating their abstract idea into the terms of the formalism in question. These two layers correspond to the first two layers mentioned by Byron and Tetreault. Their *"supervisor layer"* can be thought of in Saara as of a layer of very short programs that define a sequence of pre-processing and markable-layer modules to be called, with the specification of their parameters.

The markable-layer modules, that is AR algoritms, re-implemented and available in the Saara framework, are mainly traditional algorithm based on modeling of salience:

Plain Recency is a baseline algorithm linking each anaphor to the closest antecedent candidate agreeing in morphology.

The Hobbs' Syntactic Search [12] is one of the earliest AR approaches and unlike the other algorithms mentioned here, it is furmulated as a search by traversing the syntactic trees representing the discourse.

The BFP Algorithm [13] is based on the principles of Centering theory. It models local coherence among utterances and uses this concept to suggest anaphoric links resulting in the most coherent discourse.

Activation models considering TFA[1] [7,9] have been formulated within the Praguian framework of Functional Generative Description and are based on modeling the level of activation the individual discourse objects have in the mind of the reader.

The method of combining salience factors inspired by the RAP system by Lappin and Leass [14] is based on specifying various factors that favour (or disfavour) individual referential expressions as antecedents for anaphors. Assigning appropriate weights to individual factors allows very flexible modeling of salience.

[1] TFA stands for Topic-focus articulation; similar ideas are also known as information structure, or functional sentence perspective.

Performance of AR algorithms is very difficult to evaluate. A number of metrics have been proposed to assess the correctness of AR systems numerically, however, there is a broad range of factors that bias these numbers considerably: whether errors propagated from the pre-processing are counted, whether AR is carried out on a pre-defined set of markables or the errors in detecting anaphoric and non-referential expressions are included, the precise types of anaphora addressed, genre of the text etc. For these reasons, the figures in Table 1 are given only for the purpose of comparing the individual algorithms within our framework (not our system with other systems), revealing their advantages and disandvanteges when used on same texts.

Table 1. Performance of the system in MUC-6 measures

	Precision	Recall
Plain Recency	41.78	37.28
Hajičová 1987	41.33	36.81
Hajičová, Hoskovec, Sgall, 1995	41.33	36.80
Hobbs' syntactic search	38.87	33.91
BFP Centering	52.26	39.20
Lappin and Leass' RAP	49.86	46.28

4 Anaphora Resolution over Pre-parsed Czech Text

The two preceding sections have described the application setting used to perform automatic AR over previously unprocessed Czech text.

For each sentence, the pre-processing phase yields an ordered sequence of trees given in a bracket notation – each node representing either a terminal or non-terminal phrasal node, carrying information about its morphological features and syntactic category. As the trees are sorted according to their estimated plausibility, further processing takes advantage of the first one only.

For the AR algorithms to function correctly, we need to determine certain important structures within the derivational trees of the individual sentences.

Firstly, each sentence needs to be divided into clauses. This can be done straightforwardly based on the syntactic category tags provided by the "synt" parser. Clauses are crucial in the next step of the processing, the detection of zero subjects.

Czech is a pro-drop language, meaning that subjects of clauses need not necessarily be realized phonologically. Such, so-called, zero subjects do not correspond to any token within the text and thus are obviously missing from the syntactic parse of the sentence. They need to be added, as they play a key role in textual anaphoric relations. When a sentence does not contain any nominal phrase in nominative (and does not contain a subjectless verb), a subject node is added to the beginning of the sentence, with morphological features determined based on the verbal complex of the sentence.

As a next step, referential expressions are detected within the text, based on the syntactic category tags given by the parser. This already compiles a substantial part of the discourse model allowing the AR procedures to process the text. The only necessary issue left is to define an interface between abstract phenomena considered within the AR process, and their actual representation in the given formalism. This mainly concerns determining individual syntactic roles and ordering of referential expressions within clauses. Within "synt" derivational trees, this is done heuristically using morphological features of phrases and their linear order within the sentence.

After AR is carried out using the chosen algorithms (resolution of grammatical and textual anaphora is performed separately, one after the other), the Saara framework yields a set of markables divided into equivalence classes that are induced by coreference (or other anaphoric relation in question). This data is exported into the MMAX2 XML format for the purposes of visualisation and further processing.

MMAX2 [15] is an annotation tool that can be used to store and display data of various kinds, and to annotate various phenomena in them. The annotation can be multi-layer, which means that one annotation project can encompass a number of different unrelated phenomena, or a sequence of mutually dependent ones. For AR data, we define three separate layers over text tokens:

– sentences
– clauses
– referential expressions (grouped into coreferential sets)

Each of these annotation layers computed by the Saara framework algorithms are stored within an XML file with a straightforward structure, and can be easily used for further processing.

5 Conclusions and Further Work

This article has presented a number of linguistic tools developed at the NLP Center, MU Brno, namely the "synt" syntactic analyzer and the Saara framework for automatic AR. We mentioned how the syntesis of these tools is used to carry out AR over previously unprocessed Czech texts and discussed a number of interesting issues within this process.

Our further work aims at improving the accuracy of detecting syntactic structures. This can be done by considering more shallow structures that can be computed with stronger reliability. Further, we plan to enhance the AR algorithms themselves, by employing various semantic features, such as WordNet-like semantic classes or valency data. We also plan to use Saara with English texts.

Acknowledgments This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536.

References

1. Linh, N.G.: Návrh souboru pravidel pro analýzu anafor v českém jazyce. Master's thesis, Charles University, Faculty of Mathematics and Physics, Prague (2006).
2. Hajič, J., et al.: The Prague Dependency Treebank 2.0. Developed at the Institute of Formal and Applied Linguistics, Charles University in Prague. (2005) `http://ufal.mff.cuni.cz/pdt2.0/`.
3. Kučová, L., Kolářová, V., Žabokrtský, Z., Pajas, P., Čulo, O.: Anotování koreference v pražském závislostním korpusu. Technical report, Charles University, Prague (2003).
4. Sedláček, R.: Morfologický analyzátor (čestiny). Ph.D. thesis, Fakulta informatiky Masarykovy univerzity v Brně, Brno (1999).
5. Žáčková, E.: Parciální syntaktická analýza (čestiny). Phi.D. thesis, Fakulta informatiky Masarykovy univerzity v Brně, Brno (2002)
6. Horák, A.: Computer Processing of Czech Syntax and Semantics. Librix.eu, Brno, Czech Republic (2008).
7. Hajičová, E.: Focusing – a meeting point of linguistics and artificial intelligence. In Jorrand, P., Sgurev, V., eds.: Artificial Intelligence Vol. II: Methodology, Systems, Applications. Elsevier Science Publishers, Amsterdam (1987) 311–321.
8. Hajičová, E., Kuboň, P., Kuboň, V.: Hierarchy of salience and discourse analysis and production. In: Proceedings of Coling '90, Helsinki (1990).
9. Hajičová, E., Hoskovec, T., Sgall, P.: Discourse modelling based on hierarchy of salience. The Prague Bulletin of Mathematical Linguistics (64) (1995) 5–24.
10. Němčík, V.: The Saara Framework – work in progress. In: RASLAN 2008, Recent Advances in Slavonic Natural Language Processing, Brno (2008) 11–16.
11. Byron, D.K., Tetreault, J.R.: A flexible architecture for reference resolution. In: Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL '99). (1999).
12. Hobbs, J.R.: Resolving pronoun references. In: Grosz, B.J., Spärck-Jones, K., Webber, B.L., (eds.): Readings in Natural Language Processing. Morgan Kaufmann Publishers, Los Altos (1978) 339–352
13. Brennan, S.E., Friedman, M.W., rd, C.J.P.: A centering approach to pronouns. In: Proceedings of the th25 Annual Meeting of the ACL, Stanford (1987) 155–162.
14. Lappin, S., Leass, H.J.: An algorithm for pronominal anaphora resolution. Computational Linguistics **20**(4) (1994) 535–561.
15. Müller, C., Strube, M.: Multi-level annotation of linguistic data with MMAX2. In Braun, S., Kohn, K., Mukherjee, J., eds.: Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods. Peter Lang, Frankfurt a.M., Germany (2006) 197–214.

Verb Valency Frames in Czech Legal Texts

Eva Mráková and Karel Pala

Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
`glum@fi.muni.cz`, `pala@fi.muni.cz`
`http://www.fi.muni.cz/nlp/`

Abstract. This paper deals with valency frames for selected group of Czech verbs belonging to the domain of law. Starting with the lexical database VerbaLex we propose semantic roles for these verbs and formulate their Complex Valency Frames. The lexical database VerbaLex has been developed recently at the NLP Centre FI MU and contains approximately 10 500 Czech verbs. We integrate the proposed 'law' valency frames into it.

Key words: verb valency; Czech language; legal texts

1 Introduction

Though law terms typically consist of the noun and prespositional groups and other nominal constructions, it is necessary to pay attention to the verbs occurring in the legal texts as well. The reason is the following: verbs on one hand do not always display strictly terminological nature, but on the other they are relational elements linking the terminological noun and prepositional groups together. In this respect we can take advantage of the database containing approx. 50 000 law documents and prepared in the Institute of Government and Law Czech Academy of Sciences which we cooperate with within the GACR Grant project PES – GA 407/07/0679.

The verbs from the legal documents were originally processed by the team of F. Cvrček in the Institute of Government and Law. We had received the list of 15 110 items marked as verbs from them. Then we used `ajka` [1] for further processing and obtained the following results: 4 920 items in the list were marked as passive participles - they were not further lemmatized. After manual checking we discovered that 1 611 items from them were not recognized as verbs but for example as adjectives or nouns and they were removed from the list. Thus the list of the correctly recognized verb lemmata finally comprises 10 190 items.

There is a lexical database VerbaLex [2] that has been developed recently in the NLP Centre FI MU and it contains approx. 10 500 Czech verbs with their Complex Valency Frames (CVFs) designed to capture semantic properties of every individual verb. CVFs contain information about morphosyntactic and semantic features of the verb arguments (see below).

The idea is to investigate whether and how CVFs designed for 'normal' verbs can be used also for legal verbs and what changes have to be done in semantic labeling of the verb arguments, i. e. what new semantic roles should be added to the ones already utilized in VerbaLex.

2 About VerbaLex

The lexical database VerbaLex consists of the complex valency frames (CVFs) which can be characterized as data structures (tree graphs) describing predicate-argument structure of a verb. They contain the verb itself and its arguments determined by the verb meaning; in Czech their number usually varies from one to five. The argument structure also displays the semantic preferences on the arguments. On the syntactic (surface) level the arguments are most frequently expressed as noun or pronominal phrases in one of the seven cases (in Czech) and also as prepositional cases or adverbials. An example of a complex valency frame for the verb synset *zabít, usmrtit (kill)* capturing both its general and legal meaning looks as follows:

AG<murderer:1>$_{obl}^{kdo1}$ VERB(zabít) PAT<victim:1>$_{obl}^{koho4}$ INS<instrument:1>$_{opt}^{cim7}$
–example: vrah zabil svou oběť nožem (a murderer has killed the victim with a knife).
–synonym: usmrtit
–use: prim.

The semantics of the arguments is typically expressed as belonging to a given semantic role (or deep case), which represents a general role plus subcategorization features (or selectional restrictions). Thus valency frames in VerbaLex include information about:

1. morphosyntactic (surface) information about the syntactic valencies of a verb, i.e. what morphological cases (direct and prepositional ones in highly inflected languages such as Czech) are associated with (required by) a particular verb, and also obligatory adverbials,
2. semantic roles (deep cases) that represent the integration of the general labels with subcategorization features (or selectional restrictions) required by the meaning of the verb.

The inventory of the semantic roles is partly inspired by the Top Ontology and Base Concepts as they have been defined within EuroWordNet project [3]. Thus we work with the general roles like AG, ART(IFACT), SUBS(TANCE), PART, CAUSE, OBJ(ECT) (natural object), INFO(RMATION), FOOD, GARMENT, VEHICLE and others (32). They are combined with the literals from Princeton WordNet 2.0 where literals represent subcategorization features allowing us to climb down the hypero/hyponymical trees to the individual lexical units. For example, we have complex roles like AG(person:1|animal:1) or SUBS(liquid:1) that can be used within the individual CVFs. Their number is approx. 1200.

The verbs in VerbaLex can be characterized as having general, non terminological (legal) meaning. The task then is to take legal verbs and develop the CVFs for them and integrate them into VerbaLex.

3 Some typical legal verbs

We have the following data at our disposal:

- 3,749 verbs occurring only in the legal texts,
- 3,563 verbs occurring only in the VerbaLex,
- 4,830 verbs occurring in the both resources.

We will pay attention only to the first list from which we have chosen a small group of Czech verbs with strictly legal meanings. Two other lists are left aside here. The group of the legal verbs looks as follows (numbers show frequency in legal documents, if we know it):

Table 1. Several examples of legal verbs

verb	frequency	occurs in VerbaLex
žalovat – sue	1064	0
obžalovat – charge	774	0
zažalovat – file a suit, sue	355	0
doznat se – plead guilty	895	0
přiznat se – plead guilty		0
krást – steal		+
okrást – thieve, rob		+
okrádat – steal		+
vykrást – plunder		+
vloupat se – burgle	611	0
vloupávat se		0
znásilnit – rape	585	0
znásilňovat – rape		0
prošetřovat – investigate	306	0
prošetřit – sift, investigate		0
vyšetřit – investigate		+
vyšetřovat – investigate		+
vyslýchat – interrogate		+
odsedět – serve a sentence	231	0
osahávat – grope	159	0
obvinit – accuse		+
obviňovat – accuse		+
vraždit – murder		+
zavraždit – slaughter		+
zabít – kill		+
zabíjet – kill		+

It can be seen that the verbs in the list fall into small subgroups contaning semantically close items – they are either aspect pairs (triples, if iteratives are considered) or prefixed variants. The assumption can be made for them that the individual subgroups will share the complex frames. It also has to be remarked that for some verbs we know their frequencies only for the perfective or imperfective variant but not for both. The verbs marked with + occur in VerbaLex but only some of their meanings can be considered as legal meanings. This would require more detailed analysis which is a topic for another paper.

There are less frequent verbs in the list of legal verbs that display specialized terminological meanings, for instance the following compound verbs do not occur in the corpus SYN2000 (`http://ucnk.ff.cuni.cz/syn2000.php`) at all: *spoluvinit* (co-accuse), *spoluvázat* (co-bind), *spoluzabezpečovat* (co-ensure), *spoluzaviňovat* (co-cause), *spoluzavazovat* (co-oblige), *spoluzpůsobovat* (co-cause), *spoluzpůsobit* (co-cause, aspect counterpart of the previous one), *spolužalovat* (co-sue), etc. The first member of the given compounds is and adverb *spolu (co-*, thus it will be marked in all CVFs of these verbs.

4 Roles for legal verbs

It can be observed (unpublished report of the F. Cvrček and his team), that the legal verbs co-occur with the nouns, which can be semantically charecterized as follows:

1. one word and multi-word with the autonomous legal meaning, e. g. agreement or contract,
2. nouns with possible legal meaning that follow from the context in which it is used, e. g. person,
3. nouns with clearly non-legal meaning, e. g. chloride,
4. nouns that denote subjects or agents, for instance:
   (a) legal subject such as pachatel (malefactor),
   (b) legal subject following from context, e. g. chairman,
   (c) employment, e. g. sculptor, worker
   (d) legally preferred group, e. g. pensioner,
   (e) subjects by nationality and race, e. g. Serbian, white man,
   (f) nouns with emotional and ideological connotation, such as angel, whore,
   (g) nouns denoting animals, e. g. whale, squirrel, etc.

Some of these characterizations are already included in the VerbaLex and some new should be added (see below).

5 Some CVFs for legal verbs – examples

The above mentioned semantic categories are not too far from the semantic roles as they are used in the CVFs from the VerbaLex database. Thus it can be concluded that the CVFs are suitable for the semantic description of the legal language as well. We can observe the interesting overlaps which allow us to postulate the semantic roles in legal texts. We will mention some of them which can be easily added to the present inventory of the semantic roles in the VerbaLex. This is a positive result, which confirms the assumption that though legal language displays some specific features it can be analysed with techniques and methods developed for semantic analysis of verb meanings as they occur in a non-terminological use.

In VerbaLex we work with the roles such as AG<person:1> etc., which in legal language correspond to the 'subject' mentioned above. Thus it is possible to take advantage of the roles introduced in VerbaLex and extend them with the semantic categories indicated above. In this way, for instance, we obtain labels such as AG<judge:1>, AG<employee:1> AG<plaintiff:1>, AG<prosecutor:1> AG<defendant:1>, AG<rapist:1>, AG<investigator:3>, AG<thief:1>, AG<policeman:1>, AG<murderer:1> or PAT<victim:1> and similar ones. Then CVFs for the legal verbs mentioned may look as follows (examples):

$$\text{AG}{<}\text{murderer:1}{>}^{kdo1}_{obl}\ \text{VERB(zavraždit)}\ \text{PAT}{<}\text{victim:1}{>}^{koho4}_{obl}\ \text{INS}{<}\text{instrument:1}{>}^{cim7}_{opt},$$

and similarly one of the possible frames of the verb *obžalovat* (penalize) is

$$\text{AG}{<}\text{prosecutor:1}{>}^{kdo1}_{obl}\ \text{VERB(obžalovat)}\ \text{PAT}{<}\text{defendant:1}{>}^{koho4}_{obl}\ \text{EVENT}{<}\text{crime:1}{>}^{cim7}_{obl}.$$

The mentioned verbs and their CVFs do not occur in the VerbaLex so far. Thus the frames suggested above will be integrated into it. Below, we present two more examples of the CVFs proposed for the specialized legal verbs. It can be seen that the VerbaLex notation is suitable for this purpose:

uložit trest někomu (to condemn somebody to a sentence)

$$\text{AG}{<}\text{judge:1}{>}^{kdo1}_{obl}\ \text{VERB}\ \text{PAT}{<}\text{person:1}{>}^{komu3}_{obl}\ \text{ACT}{<}\text{sentence:1}{>}^{co4}_{obl},$$

obvinit někoho z trestného činu (to accuse somebody of criminal act)

$$\text{AG}{<}^{\texttt{public}}_{\texttt{prosecutor:1}}{>}^{kdo1}_{obl}\ \text{VERB}\ \text{PAT}{<}\text{person:1}{>}^{koho4}_{obl}\ \text{ACT}{<}\text{act:1}{>}^{zceho2}_{obl}.$$

The described valency frames thus can serve as the descriptions of the meanings of 'legal' verbs – more similar examples can be easily found. For more specialized legal verbs, however, further modifications are needed that require more detailed semantic analysis.

We also have to mention semantic classes [4] of Czech verbs – they represent a sort of 'verbal' ontology. Semantic roles in the valency frames have served as a criterion for finding relevant semantic classes of (Czech) verbs. This can be also applied to law texts in their natural form.

6 Conclusions

To sum up: the goal was to show that valency frames from VerbaLex database can be appropriately applied to the semantic analysis of the legal language. The size (4,830 verbs) of the intersection mentioned above justifies the further investigation. We have already enriched the inventory of the semantic roles in VerbaLex to obtain their more detailed and exact semantic subclassification, or, in other words, their more adequate 'legal' ontology.

Then the roles can be compared with the already existing law ontologies such as the one built within the LOIS (Lexical Ontologies for Legal Information Society) project[1]. In this project, the ontology was built in the WordNet fashion. However, WordNet-like and similar ontologies are structures capturing relations between nouns and noun groups only. We are convinced that more is needed, in particular, a kind of ontology that can be characterized as 'verbal'.

Acknowledgement This research has been funded by the Czech Grant Agency under the Grant Project GA 407/07/0679.

References

1. Sedláček, R.: Morphemic Analyser for Czech. Ph.D. thesis, Faculty of Informatics, Masaryk University, Brno (2005).
2. Horák, A., Hlaváčková, D.: VerbaLex – New Comprehensive Lexicon of Verb Valencies for Czech. In: Computer Treatment of Slavic and East European Languages, Third International Seminar, Bratislava, VEDA (2005) 107–115.
3. Vossen, P., et al.: The EuroWordNet Base Concepts and Top Ontology. Technical Report Deliverable D017, EuroWordNet LE2-4003, University of Amsterdam (1998).
4. Hlaváčková, D., Khokhlova, M., Pala, K.: Semantic Classes of Czech Verbs. In: Proceedings of the IIS Conference 2009, Krakow, in print (2009).

[1] see `http://www.ittig.cnr.it/Ricerca/materiali/lois/WhatIsLOIS.htm` and also `http://nlpweb.kaist.ac.kr/gwc/pdf2006/50.pdf`

Part III

Complex Language Resources and Relations

Semantic Network Integrity Maintenance via Heuristic Semi-Automatic Tests

Tomáš Čapek

Faculty of Informatics, Masaryk University, Brno, Czech Republic
xcapek1@aurora.fi.muni.cz

Abstract. In this article we discuss issues connected with maintaining content integrity of general-purpose semantic network that is in development. Construction of a semantic network from scratch is a long process that usually requires both linguistic work done by hand and semi-automatic methods to add or translate the data which must be subsequently reviewed. In this process many systemic and/or language-specific errors may appear in the data over time. We will introduce a method to cope with this issue systematically.

Key words: semantic network; WordNet; semantic network integrity

1 Introduction

A general-purpose semantic network is a language resource for the given language, alternative to traditional dictionaries. It consists of semantic units which are connected by semantic relations, thus creating a graph-like structure or a network. The biggest semantic network to date is WordNet (PWN) [1], that has been in development since 1985 at Princeton University. It contains more than 90.000 semantic units called synsets or synonymical sets. Many WordNet-like semantic networks exist today for other languages, developed in projects such as EuroWordNet [2] or BalkaNet [3].

To create a semantic network requires a team of linguists, software support and months of work among other things. In order to save time or resources one or both methods described below get usually employed:

- Semi-automatic translation of semantic units from other, larger networks. This also refers to so-called *expand model* in EuroWordNet terminology [4]. Basically it means that we adopt the original structure of semantic units and semantic relations among them, translate each lexeme automatically via some electronic dictionary or translator system available for our language and review the data afterwards by hand. Additional language-specific and other data are subsequently added to the network, thus *expanding* it. This is generally the fastest and the most popular method when creating a new wordnet-like semantic network and PWN is the most common semantic network used as a template. This method is the also most prone to adopting and creating new errors when used as the only method.

- Manual linguistic work – also called *merge model*[1] in EuroWordNet terminology. The main focus here is to create a semantic network with independent structure or predetermined application in mind. An existing language resource can be used as the source lexicon, data in which need to be rearranged and interconnected via semantic relations to form a semantic network. This method is similar to traditional construction of dictionaries which is known to be time-consuming and expensive. It also requires a lot of linguistic introspection on part of the developers and can be outsourced so that many different people take turn in the process of adding and editing the data. As an implication semantic networks built according to this method are also prone to contain many types of inconsistencies and errors.

2 In-development Integrity Checks

There are several ways we can take to prevent errors from appearing in our network while it is still in development. However they represent additional expenses on time, resources and manpower. As evidenced in relevant Global WordNet Conference (GWC) proceedings articles [5], these additional methods have not been used more often than they have.

2.1 Corpus Evidence

When adding, checking or translating lexemes and semantic units it is important to have an appropriate corpus available as the definitive source of real-life usage of words. No two linguists have exactly the same knowledge and perspective of a language and that changes even for a single linguist over time. In this regard, corpora help to streamline and unify otherwise divergent approaches to handle linguistic data, especially those of non-frequent nature. The bigger the corpus is the better but it is also important for it to contain only relevant documents with respect to the contents of the semantic network. Unsorted pile of random documents can provide false or inaccurate evidence for the linguists thus spoiling the benefits corpora can bring to the process of development of a semantic network.

2.2 Guideline Manual

A set of instructions how to handle new or existing semantic units and relations sets the standard for people who participate in the semantic network development and who may come and go as the process goes on. It should provide basic information on issues such as: what are the criteria for a word to be lexicalized or non-lexicalized in the network; in what way to compose or

[1] EuroWordNet was a project primarily focused to create a multi-lingual semantic network based on PWN. The *merge* part of the process refers to the final stage of development when the semantic units in the newly created network are connected to their corresponding counterparts in another network, thus *merging* it into one bilingual structure.

assume definitions for semantic units; how to use notes for further work; what semantic relations are important for particular part of speech, etc. The nature of the guidelines should be dependant on the aim of the semantic network itself. The guidelines can also be described as restrictions and implemented into a software tool used for editing of semantic data.

2.3 Quality Assurance

Ideally, any new data in the network should be reviewed independently. As we have seen, there are plenty of ways how to import erroneous data into semantic networks. It may appear as self-evident but quality of semantic data is directly related to success rate of any NLP experiment that employs it or its usefulness when used as another language resource for linguistic work. If no guidelines exist for given semantic network then quality assurance may result in ad hoc fixes or random tweaks because no one knows what aspects of development were important in the past or when they may change again. Thus the quality assurance basically means a check to what extent the semantic data conform the guidelines. In that regard we can design and implement a set of automatic tests that would filter out lists of potentially erroneous semantic units for inspection, as described in the next chapter.

3 Heuristic Tests

As shown above, contrary to our best intentions, many different errors and inconsistencies may appear in our semantic network over time. These errors may become relevant when we need to use the data for our NLP experiment but don't have time and resources to fix the data directly. One way to quickly analyze the data is to design and implement a set of heuristic tests. Each test should be a formalized pattern of an error that appears multiple times within the semantic network. For example, Czech orthography allows us to use two different suffixes in words ending with *-ism* (e.g. in albinism). We can use a suffix with *s* or *z* in it – both *albinizmus* and *albinismus* are correct word forms in Czech. However, it may be useful in more than one way to use only one suffix variant consistently. In this case the test is very simple, we choose the desired variant of the suffix and let the test search each lexeme in our semantic network for the other suffix variant. On the output we get a list of candidate semantic units for review. Again, in this case the next step is very simple as there's virtually no way we could get a false positive from this test in Czech. We can simply apply all the proposed changes into the semantic network source database in batch-mode and we are done.

Most of semantic networks continue to be edited even after the main development project has ended. Once a test is implemented it is useful to have it scheduled for regular runs after a certain period of time via *cron* tool or any other scheduler software. The results automatically reported via e-mail can also help to keep the integrity of the network up-to-date at all times. Let's take a look at several more useful tests:

- **Morphology tests** In this category of tests we check for typing errors or for incorrect word forms, lemmata of which belong to the network. As a requirement we need a spell checking tool and a dictionary for our language (e.g. *ispell* [6]) but for highly inflectional languages such as Czech and other Slavonic languages it is far more useful to employ a morphological analyzer that can generate and recognize any word forms belonging to the language. If we use the *expand* model or use other means to automatically add semantic units for subsequent translation, morphology test can also filter out the data for us that has not been translated yet.
- **Syntax tests** Especially if we don't or didn't use any formal guidelines, any type of unexpected data can get into our semantic units. Usually they are various notes from the editors or redundant characters left over from automatic imports from other language resources. A simple test for non-letter characters and for high word counts in lexeme records can discover potentially erroneous semantic units. The advantage of this test is that it is much cheaper to employ than to implement a full set of syntactic restrictions directly into the software editing tool that is used to work with the data.
- **Instance test** Many cases of semantic relation pair class-instance (e.g. sea-Aegean Sea) are often marked as simple cases of hyperonymy-hyponymy in many semantic networks. To remedy this only a simple test for capitalized lexemes in semantic units is required to filter out most cases of named entities which should have their relations to their superordinate semantic unit changed to *Instance*.
- **Orphan nodes** Each part of speech has one significant semantic relation that connects all semantic units of its kind. For instance it is hyperonymy-hyponymy pair for nouns. Sometimes when new data are added to the network by hand or automatically, some of them remain unconnected thus creating orphan nodes within the network. A simple test can discover these nodes by checking each semantic unit for that particular semantic relation. If higher rate of false positives is not a problem this test can be extended to other relations as well, even if they are not supposed to interconnect every semantic unit in given category of semantic data.

Apart from the tests above many other language-specific or general tests can be designed according to particular needs of each semantic network. It should always be quicker to implement a test if we can find a pattern in the data than to do a full revision in top-down or alphabetical order.

4 Further Work

Although the heuristic tests are often very simple and quick to implement they can only cover the surface errors and inconsistencies visible on first sight. They can also help us to find various structural defects in a network such as undesired multiple inheritance, unbalanced trees or high sense number count for a lexeme but cannot offer a solution for such problems. Our further work will therefore be focused on more sophisticated methods that would allow us to tackle practical problems with ontologies, data density or domain subtrees in a semantic network.

5 Conclusion

We have discussed an issue how to create and maintain semantic data in a semantic network that would allow us to minimize the number of errors and inconsistencies on surface level of the network. We have introduced a method of simple heuristic tests that can be easily implemented and can help us to remove frequent errors in the data even when the network is still being in development and many editors may participate in it. Although the tests are not an universal remedy to all problems we can have with the semantic data their favorable cost-benefit ratio makes then a useful tool to keep the integrity of our data intact.

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.

References

1. Miller, G., Beckwith, R., Fellbaum, C., Gross, D., Miller, K., Tengi, R.: Five papers on WordNet. International Journal of Lexicography **3**(4) (1990) 235–312.
2. Vossen, P.: Eurowordnet a multilingual database with lexical semantic networks. Computational Linguistics **25**(4) (1999).
3. Tufis, D., Cristea, D., Stamou, S.: Balkanet: Aims, methods, results and perspectives. A general overview. Science and Technology **7**(1-2) (2004) 9–43.
4. Vossen, P.: Right or Wrong. Combining lexical resources in the EuroWordNet project. In: M. Gellerstam, J. Jarborg, S. Malmgren, K. Noren, L. Rogstrom, CR Papmehl, Proceedings of Euralex '96, Goetheborg, Citeseer (1996) 715–728.
5. Sojka, P., Pala, K., Smrž, P., Fellbaum, C., Vossen, P., (eds.): Proceedings of the Second International WordNet Conference—GWC 2004, Brno, Czech Republic, Masaryk University Brno, Czech Republic (2004).
6. Kuenning, G., Willisson, P., Buehring, W., Stevens, K.: International ispell. Webpage can be found at: `http://fmg-www.cs.ucla.edu/fmg-members/geoff/ispell.html`, visited on February 17th (2004).

Exploring and Extending Czech WordNet and VerbaLex

Zuzana Nevěřilová

Masaryk University, Faculty of Informatics
Botanická 68a, 602 00 Brno, Czech Republic
`xpopelk@aurora.fi.muni.cz`

Abstract. This paper presents usage of two major, linguist-made lexical resources of Czech language: WordNet and VerbaLex. First, a conversion to RDF was made. Afterwards, a Prolog program was used to analyse Czech language inputs.
In the second part of the article an extension to current VerbaLex is proposed. Possible pitfalls are discussed. In the conclusion, we emphasize the side-effect of this work: an important feedback for authors and administrators of both lexical resources.

Key words: VerbaLex; WordNet; semantic analysis; RDF; Prolog

1 Introduction

Since 2005 a database of verb valency frames is created. This database, VerbaLex [1], has form of frame-based lexical resource: it consist of verb valency frames with slots. Each slot contains two levels of semantic information:

– semantic role, such as *agent*, *patient*, *instrument*
– value restriction in form of bottommost hypernym, specified by literal and sense number in Princeton WordNet [2] (e.g. *person:1*)

Czech WordNet (CZWN) started as part of EuroWordnet [3] project in 1998 and it is still being actively developed.

VerbaLex and CZWN are two large linguist-made resources for Czech language. These resources can be and are expected to be used together thanks to the fact that in CZWN the IDs of synsets are linked to their translations in Princeton WordNet.

This article shows how these resources can be used for semantic analysis of sentences and proposes an extension that can add background knowledge to these sentences. This background knowledge is considered necessary for semantic discourse analysis [4].

For verb frame identification, semantic role assignment and subsequent inference SWI-Prolog and RDF were used.

In the experiments we deliberately omit syntactic analysis of the sentences and use only base form of nouns (singular nominative). We expect that syntactic analysis could improve the results notably. In practice intersection of our results and those of syntactic analysis will be used.

2 Data Formats and the Program

Both CZWN and VerbaLex are stored in their own formats in the form of XML. For the purpose of their connection and inference, both data sources were converted to RDF [5] (in the form of XML). The conversion was done through XSL templates, since it is portable and easy to maintain (in case of slight changes in the structure of the XMLs).

The conversion does not cover all aspects of VerbaLex nor CZWN. For the reasons of reasonable size of the data, some features such as examples, human readable definitions etc. were omitted. In VerbaLex there is no ID for a frame, but during the conversion one is added for each verb frame. The ID consists of one of the lemmata (where czech accents were replaced by capitals), sense number and frame number (generated during the conversion). The ID is in form of URI according to RDF specification [6].

After experiments with RDF reasoners, Prolog with `rdf_db` module was chosen for inference. The advantages of this solution are:

- Prolog is able to work with large data. VerbaLex comes with more than 212 000 RDF triples, CZWN with nearly 100 000.
- It is possible to insert inference rules to the program and not to the data. The most resource-consuming relation is the hyperonymy, because it is a transitive relation. Since RDF is not able to handle transitivity, it would be necessary to use some kind OWL [7] guided with enormous increase of the number of RDF triples. Hyperonymy is handled in the Prolog program and thus the number of RDF triples is final.
- With an appropriate Prolog module, web interface can be made straightforwardly.

3 Finding Semantics through Verb Frames

Since this work does not concern syntactic analysis, almost no grammatical information is available for the analysis. The input is simple: the verb and a list of nouns in their base form (singular nominative).

In our analysis of a sentence, we can identify 3 kinds of bearers of the meaning:

- nouns occuring in the sentence identify *hypernyms* occuring in the verb frame
- *semantic roles* that the nouns play
- the verb frame *structure*, especially the number, semantic role and occupancy of other slots

The output contains the ID of a verb frame and nouns of the list with their semantic roles assigned:

```
?- find_roles('přicestovat',['ministr','zastávka'],FrameID,Roles).
FrameID = 'http://nlp.fi.muni.cz/verbalex#pRicestovat_1_2',
Roles = [ (ministr, 'AG', kdo1, obl), (zastávka, 'LOC', čeho2, opt)] ;
```

The input: verb *přicestovat* (arrive) and the nouns *ministr* (minister) and *zastávka* (station). The resulting role assignment: minister as AG(ent) and nominative animate (kdo1), obl(igatory) value of the slot, station as LOC(ation) inanimate genitive (čeho2), opt(ional).

3.1 Features, Problems and Solutions

The result of the analysis brings following advantages:

- appropriate verb meaning recognition
- frame identification
- semantic roles assignment
- grammatical information (cases)

It is necessary to keep in mind that the result is a set. In the case above, this set has only one element.

Problems occuring during the analysis can be following:

- verb not found in VerbaLex. This is not expected to occur often, since VerbaLex contains 19 360 valency frames from more than 10 000 verbs [8]. But if this case occurs, the analysis brings no result.
- word from the list not found in CZWN. This occurs almost in every sentence, since CZWN is much smaller than Princeton WordNet. Moreover it does not contain proper names at all. The instant solution is to take subsets of the input set and try to assign as much nouns as possible. A long-term solution consists of improving CZWN and using other resources for proper names.
- no suitable frame for the list of words. In VerbaLex, only *common use* is encoded. In some cases, language users do not follow the common use. This occurs rarely. Most often there are words not related to the verb (e.g. parts of noun phrases) or nouns contained in adverbial phrases. Solution is again to take subsets of input set.
- no suitable hyperonym for a word. This came in sight as the most difficult problem. It seems that there is not much consensus about the bottommost hypernyms in frame slots. For example the verb *koupit* (to buy) has the OBJ(ect) slot value *goods:1*. But the object of buying can be almost every object or even animal. Thus it seems that the value of the OBJ slot should be *object:1*. In this case verb frame will not offer much information.

4 Proposed Extension of VerbaLex

VerbaLex is a frame-based lexical resource. Like other resources, such as FrameNet [9], it contains slots describing typical situations (in this case noun phrases related to the verb), with restriction on their values (in this case WordNet hypernyms).

Contrary to FrameNet, VerbaLex frames are not related together, there is no hierarchy among the frames.

According to [10] it makes sense that frame information should be inherited through type hierarchy. Frame-based representation can be also used to encode additional information not mentioned in the sentences. This underlying knowledge is believed to be very useful in interpreting language. In particular knowledge about causality is very important. Frame-based knowledge representations consist at least of:

– preconditions
– effects
– decompositions

FrameNet, as a representant of large frame-based resources, contains even more types of relations. Proposed extension rests in introducing these three relations to the frames. Prolog program was extended that it supports inference rules.

These inference rules are in form of another RDF (encoded in XML) and related to VerbaLex through RDF IDs. Only information is: type of relation (precondition, effect, decomposition), relation to another frame and mapping between the slots:

```
<proposition rdf:about="#pRicestovat_1_1_effect_1">
  <action rdf:resource="#pRicestovat_1_1"/>
  <effect rdf:resource="#nachAzet_se_1_1"/>
  <mapping>
    <map>
      <from rdf:resource="#AG"/>
      <into rdf:resource="#ENT"/>
    </map>
  </mapping>
  <mapping>
    <map>
      <from rdf:resource="#LOC"/>
      <into rdf:resource="#LOC"/>
    </map>
  </mapping>
</proposition>
```

In this piece of XML the *effect* of *přicestovat* (arrive) is to *nacházet se* (inhere). Mapping is done from AG(ent) to ENT(ity) and from LOC(ation) to another LOC(ation). Note that in the example above the grammatical change occurs on the basis of VerbaLex information. No other information is needed in the inference rule.

With these data program is able to output:

```
?- find_effect('přicestovat',['ministr','zastávka'],FrameID,Roles).
FrameID = 'http://nlp.fi.muni.cz/verbalex#nachAzet_se_1_1',
Roles = [ (ministr, 'ENT', kdo1, obl), (zastávka, 'LOC', čem6, opt)] .
```

The input: verb *přicestovat* (arrive) and the nouns *ministr* (minister) and *zastávka* (station). With the inference rule that *přicestovat* (arrive) *has the effect* of *nacházet se* (inhere): minister as ENT(ity) and nominative animate (kdo1), obl(igatory) value of the slot, station as LOC(ation) inanimate locative (čem6), opt(ional).

Result of inference brings in addition to features mentioned above following:

– new frame identification
– change of roles assignment (AG → ENT)
– change of grammatical information (čeho2 → čem6)

4.1 Problems and Solutions

Main problem of this extension is how to build effectively set of inference rules.

Proposition is to group verbs according to structure of their frames and assign rules depending on which group each verb joins.

For example: LOC(ation) slot with genitive indicates that one of role representants (either AG(ent) or PAT(ient) changes LOC(ation)). In most cases, s/he either starts or stops to be placed in that LOC(ation). Verbs fulfilling this structure are the verbs of motion [1] such as *dorazit*, *přicestovat* (arrive), *dojízdět* (commute), or the verbs of sending and carrying such as *cpát* (crowd), verbs of spatial configuration such as *klesat*, *svažovat se* (slope down).

This grouping can lead to a semi-automatically created inference rules set.

4.2 Introducing New Entities and New Roles to the Discourse

According to [10], knowledge about usual situations in which actions occur is useful for language interpretation. Moreover if these situations are defined, the knowledge reveals new objects that do not have to be mentioned, but exist in the discourse.

For example buying something involves four objects: the buyer, the seller, the object and an amount of money. Even if the money is not mentinoned in discourse, it is contained in it.

Decomposition of buying is:

– buyer gives money to seller
– seller gives object to buyer

Moreover agents in the discourse can play new roles. Every living person can be buyer or seller, but during the act of buying, AG(ent) has the role of buyer (buyer is not a new entity in the discourse, but it is a new role of the entity previously mentioned).

In future work we will concentrate on encoding these new entities and roles to inference rules so they can be used in the discourse semantic analysis.

5 Conclusion

We have introduced Prolog program that is able to analyse verb and nouns occuring in a sentence. The analysis acquire following information:

– valency frame identification
– semantic role assignment
– grammatical information

We have proposed an extension to VerbaLex that can imply new propositions. Main problem is how to build an appropriate set of rules. With this extension we can even introduce new object to the discourse or to assign new roles to the agents previously mentioned. This background knowledge is believed to be useful for language interpretation.

Side-effect of this analysis is that on corpus sentences it offers an important feedback to the authors and administrators of VerbaLex and CZWN. Namely choice of bottommost hypernym in VerbaLex slots can be checked.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536.

References

1. Hlaváčková, D.: Databáze slovesných valenčních rámců VerbaLex. Master's thesis, Masarykova univerzita, Filozofická fakulta, Ústav českého jazyka (2007).
2. Fellbaum, C.: WordNet: An Electronic Lexical Database (Language, Speech, and Communication). The MIT Press (1998).
3. Vossen, P.: EuroWordNet – a multilingual database with lexical semantic networks (1998).
4. van Dijk, T.A.: Semantic discourse analysis. In: Handbook of Discourse Analysis: Dimensions of Discourse. Volume 2., London, Academic Press (1985).
5. Beckett, D., McBride, B.: RDF/XML syntax specification (2004).
6. Lassila, O., Swick, R.R.: Resource Description Framework, (RDF) model and syntax specification (1999).
7. McGuinness, D.L., van Harmelen, F.: OWL Web Ontology Language Overview (2004).
8. Hlaváčková, D.: Počet lemmat v synsetech VerbaLexu. In: After Half a Century of Slavonic Natural Language Processing, Brno, Czech Republic, Tribun EU (2009).
9. Baker, C.F., Fillmore, C.J.: FrameNet (2009) [Online; accessed 30-July-2009].
10. Allen, J.: Natural Language Understanding (2nd ed.). Benjamin-Cummings Publishing Co., Inc., Redwood City, CA, USA (1995).

Measuring Coverage of a Valency Lexicon using Full Syntactic Analysis

Miloš Jakubíček, Vojtěch Kovář, and Aleš Horák

Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{xjakub,xkovar3,hales}@fi.muni.cz

Abstract. Recent development showed that valency information provides a great benefit in many areas of natural language processing. Building valency lexicons is however a complex and time-consuming task from both theoretical and practical points of view, since designing of the lexicon plays a crucial role in its future usability as well as its careful and considerated preparation. As for any manually created resource, it is complicated to evaluate its quality. In this paper we consider the usage of the syntactic parser `synt` for estimating the coverage of the Verbalex verb valency lexicon for Czech. For this task we extended the phrase extraction functionality of the parser, which we describe briefly. Finally we discuss our results and further development.

Key words: verb valency; syntactic analysis; lexicon coverage

1 Introduction

During the last decade researchers tried to enhance their NLP applications by supplying additional linguistic information. The usage of all kinds of resources from basic lexicons over annotated corpora to complex ontologies has proved to be necessary for further development of most computational linguistics applications. Their preparation is however very time-consuming and therefore costly since the prevailing majority of them needs to be created manually (at least partially). This raises many problems in both design and implementation of a particular resource: it ought to be carefully designed from the theoretical point of view because the opportunities to modify it automatically in the future are limited. The actual preparation also has to be addressed attentively to ensure consistency, validity and completeness of the prepared data.

It is therefore of great benefit to make any of these steps a bit easier, e. g. to move from fully manual to semi-automatic processing and to provide (semi-)automatic evaluation methods. In this paper we describe one of such steps that we have taken in the case of building the Verbalex verb valency lexicon for Czech [1]. We used a Czech parser called `synt` [2] for automatic extraction of shallow verb valencies from the DESAM corpus [3] which is manually annotated. The proposed method gives an estimation of the lexicon coverage as well as speeds up the preparation of the lexicon by providing preprocessed data for annotators and offer suitable examples for existing verb valencies.

2 Syntactic Parser `synt`

The syntactic parser `synt` [4] has been developed for several years in the Natural Language Processing Centre at Masaryk University. It performs an agenda-based head-corner chart parsing using the provided context-free grammar for Czech. For easy maintenance this grammar is recorded in the form of a metagrammar (having about 200 rules) from which the full grammar can be automatically derived (having almost 4,000 rules). Contextual phenomena (such as case-number-gender agreement) are covered using the contextual actions defined for each rule.

It has been shown that `synt` achieves a very good coverage (more than 90 % [5, p. 77]), but the analysis it provides is highly ambiguous: for some sentences even millions of output syntactic trees can occur. There are two main strategies developed to fight such ambiguity. First, the grammar rules are divided into different priority levels that are used to prune the resulting set of output trees. Second, each grammar rule has a ranking value assigned from which the ranking for the whole tree can be efficiently computed in order to select only the best trees for the output.

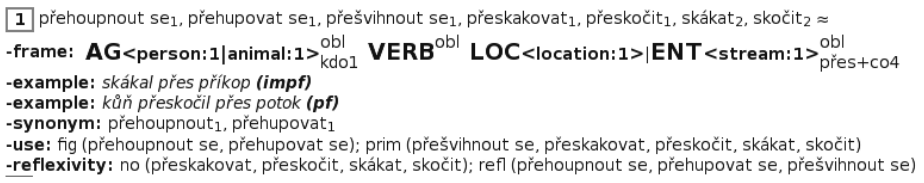
This parser also allows effective and *unambiguous* phrases extraction by using the internal parsing structure of `synt`, a so called *chart*. The chart is an acyclic multigraph which is built up during the analysis stage and contains all resulting trees. We have employed this technique in extracting shallow verb valencies as described further in this paper. Detailed description of the general extraction algorithm can be found in [6].

3 Verbalex Valency Lexicon

The Verbalex valency lexicon for Czech has been continuously developed since 2004. Currently it contains over 21,000 verb frames for more than 10,000 Czech verbs. It uses the notion of two-level annotation for the so called *complex valency frames* [7]: the first level provides shallow syntactic valencies (e. g. grammatical cases) whereas the second level contains deep semantic annotation using the *semantic roles*. Moreover, each valency frame contains a synonymic set mapped to the Czech WordNet [8]. In this paper we are concerned only with the first (syntactic) level. An example valency frame for the verb *skákat (to jump)* looks as follows:

3.1 The BRIEF Format

The Verbalex valency lexicon is stored primarily as XML documents, however the `BRIEF` format [9] can also be used for describing shallow syntactic valencies. The phrases extraction functionality of the `synt` parser mentioned above allows us to extract all possible valencies of the verb in this `BRIEF` format. Thanks to this, we can compare the output of the `synt` parser with the shallow valencies as recorded in the Verbalex lexicon. Sample output of the `synt` valency extraction in the `BRIEF` format is provided in the following example:


$\boxed{1}$ přehoupnout se$_1$, přehupovat se$_1$, přešvihnout se$_1$, přeskakovat$_1$, přeskočit$_1$, skákat$_2$, skočit$_2$ ≈

-frame: **AG**<person:1|animal:1>$^{obl}_{kdo1}$ **VERB**^{obl} **LOC**<location:1>|**ENT**<stream:1>$^{obl}_{přes+co4}$

-example: *skákal přes příkop* ***(impf)***

-example: *kůň přeskočil přes potok* ***(pf)***

-synonym: přehoupnout$_1$, přehupovat$_1$

-use: fig (přehoupnout se, přehupovat se); prim (přešvihnout se, přeskakovat, přeskočit, skákat, skočit)

-reflexivity: no (přeskakovat, přeskočit, skákat, skočit); refl (přehoupnout se, přehupovat se, přešvihnout se)

Fig. 1. Example valency frame for the Czech verb *skákat (to jump)*: it shows a nominative valency with semantic role of a `person` or an `animal` and an accusative valency with the preposition *přes (over)* and a semantic role of a `location` or `stream`.

```
; extracted from sentence: Nenadálou finanční krizi musela podnikatelka
řešit jiným způsobem .
řešit <v>hTc4a-hTc7
```
*(The businessman had to **solve** the sudden financial crisis in another way.)*
An accusative and instrumental valency has been found.
```
; extracted from sentence: Hlavní pomoc ale nacházela v dalších
obchodních aktivitách .
nacházet <v>hTc4-hTc6r{v}
```
*(However she **found** the main help in further business activities.)*
An accusative and ablativ valency with the preposition *v (in)* has been found.
```
; extracted from sentence: U výpočetní techniky se pohybuje v rozmezí od
8000 Kč do 16000 Kč .
pohybovat <v>hTc2r{u}-hTc6{v}
```
*(For information technology [it] **ranges** between 8000 Kč and 16000 Kč.)*
A genitive valency with the preposition *u (for)* and an instrumental valency with the preposition *v (in)* has been found.

4 Extraction of Shallow Valencies

As outlined above, we have extended the extraction functionality of the `synt` parser in a way that enables us to obtain various syntactic structures for the given corpus sentences. From these structures we construct simple shallow valency frames for every verb that we then compare to the valency frames available in the Verbalex lexicon. The exact extraction procedure is as follows:

1. identify clauses in the input sentence and process each of them separately,
2. in each clause, identify all prepositional and noun phrases, infinitives and selected conjunctions recorded in Verbalex (*až, že, jestli, zda, ať, aby, jak*),
3. construct a valency frame in the `BRIEF` format using the above information together with available morphological annotation,
4. for all automatically extracted valencies of a particular verb, check whether they are available in Verbalex.

In the way described above we are able to find incomplete valency frames, suggest valencies for missing verbs in Verbalex, or offer examples with most complete valency frames. The number of complete valency frames enables us

Table 1. Coverage of the Verbalex valency lexicon on the annotated DESAM corpus

indicator	covered	total	%
verb coverage	2,957	3,685	**80.24**
valency coverage	5,348	9,430	56.71
valency coverage with consideration of error analysis	5,348	6,397 [1]	**83.60**

Table 2. Error analysis of missing valencies.

indicator	number of missing valencies	%
noun phrases (*case* only valencies)	499	11.00
prepositional phrases (*preposition+case* valencies)	3,142	76.97
other (*subordinated clauses, infinitives* etc.)	491	12.03
total	4,082	100

also to determine the minimal coverage of the Verbalex lexicon with regard to the data in the DESAM corpus. The related results are shown below – note that for the purpose of this measurements, several relaxations have been performed. We matched the valency frame only against those valencies that can potentially be found by the `synt` extraction (as listed in Point 2 of the above enumeration). We also ignored the animacy denoted in the `BRIEF` format since there is no way how `synt` could obtain this information (besides the animacy of Czech masculines), i. e. the `hP` and `hT` tags have been considered to be equal and finally we also didn't differentiate between various meanings of a single verb denoted in Verbalex since there is no way how to do this on the syntactic level.

An automatic extraction of valencies performed in the way described above has one obvious drawback: with the available syntactic information we are generally not able decide whether a noun or prepositional phrase associated with a verb is an obligatory valency (argument) or a non-obligatory adjunct (modifier) usually expressing time, place or manner (in most cases with a preposition). Therefore we performed a manual error analysis of a random sample of 200 potential prepositional valencies which have not been found (representing over 75 % from all missing valencies) in the Verbalex lexicon. It revealed that a vast majority (193, i. e. 96.5 %) of those valencies were actually adjuncts (deliberately not listed in the Verbalex lexicon).

Thus our results provided in Table 1 below consist of three different measurements. First, we show the coverage of the lexicon on a verb-only level (i. e. whether there is an entry for a verb in the lexicon). Second, we show the raw results of valency matching (performed on the verbs available in the lexicon). In the end we give an estimation of the valency coverage by extrapolating our error analysis results on missing prepositional valencies to the whole lexicon. Based on this measurement, we estimate the minimum coverage of the Verbalex lexicon on valency level to be 83.60 %.

[1] The extrapolated value from the error analysis has been computed as: $9430 - 0.965 \cdot 3142$ *(number of all potential valencies − relative frequency of adjuncts in the sample · number of all missing prepositional valencies).*

5 Conclusion

In this paper we described the involvement of the Czech parser synt in developing and evaluating of the Verbalex valency lexicon. We consider the demonstrated method as well as the underlying technique (extraction of phrases) to be universal and easily applicable to similar tasks in the future. It should be also mentioned that the relation between synt and Verbalex is symbiotic in many aspects: we can not only improve Verbalex by synt, but also enhance the parser with the help of Verbalex, as it has been proposed in [10]. Our results support the evidence that verb valencies play an integral role in Czech syntax and should be further investigated.

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.

References

1. Horák, A., Pala, K.: Building a large lexicon of complex valency frames. In: Proceedings of the FRAME 2007: Building Frame Semantics Resources for Scandinavian and Baltic Languages, Lund University, Sweden, Tartu, Estonia (2007) 31–38.
2. 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, Springer Verlag (2007) 76–84.
3. Pala, K., Rychlý, P., Smrž, P.: DESAM – Annotated Corpus for Czech. In: Proceedings of SOFSEM '97, Springer-Verlag (1997) 523–530.
4. Kadlec, V., Horák, A.: New meta-grammar constructs in czech language parser synt. In: Lecture Notes in Computer Science, Springer Berlin / Heidelberg (2005).
5. Kadlec, V.: Syntactic analysis of natural languages based on context-free grammar backbone. Ph.D. thesis, Faculty of Informatics, Masaryk University, Brno (2007).
6. Jakubíček, M., Horák, A., Kovář, V.: Mining phrases from syntactic analysis. In: Proceedings of the 12th International Conference on Text, Speech and Dialogue, Pilsen, Czech Republic, Springer Verlag (2009) 124–130.
7. Pala, K., Horák, A.: Can complex valency frames be universal? In: RASLAN 2008, Brno, Masarykova Univerzita (2008) 41–48.
8. Pala, K., Smrž, P.: Building Czech wordnet. In: Romanian Journal of Information Science and Technology, Romanian Academy (2004) 79–88.
9. Pala, K., Ševeček, P.: The valencies of Czech words. In: Sborník prací FFBU, Brno, Masarykova univerzita (1997) 41–54.
10. Hlaváčková, D., Horák, A., Kadlec, V.: Exploitation of the Verbalex verb valency lexicon in the syntactic analysis of Czech. In: Proceedings of Text, Speech and Dialogue 2006, Brno, Springer-Verlag (2006) 85–92.

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, 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].

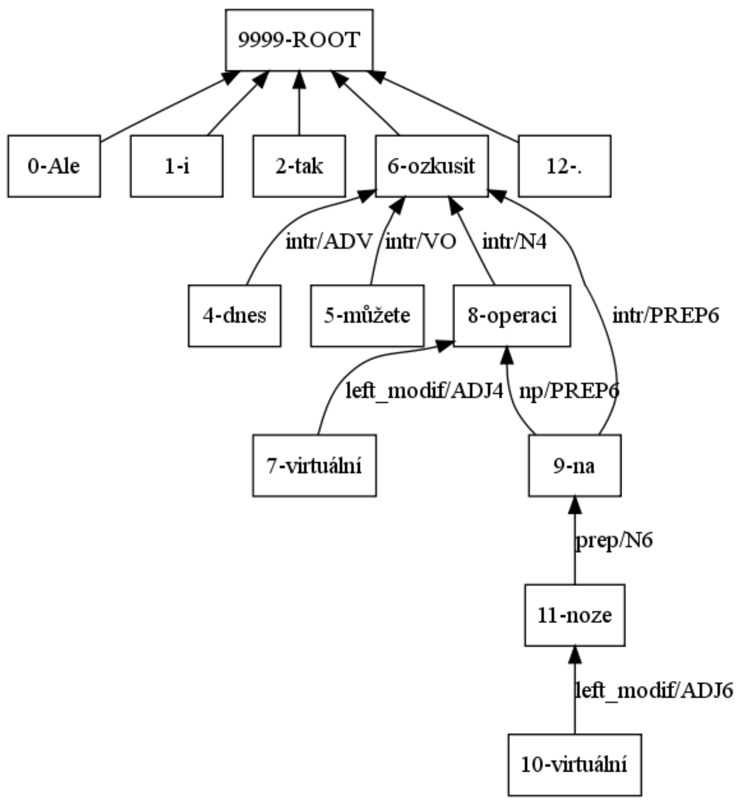


Fig. 1. An example of `synt` dependency graph output for the sentence "*Ale i tak už dnes můžete ozkusit virtuální operaci na virtuální noze.*" (Even so you can today try virtual operation on a virtual leg).

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:

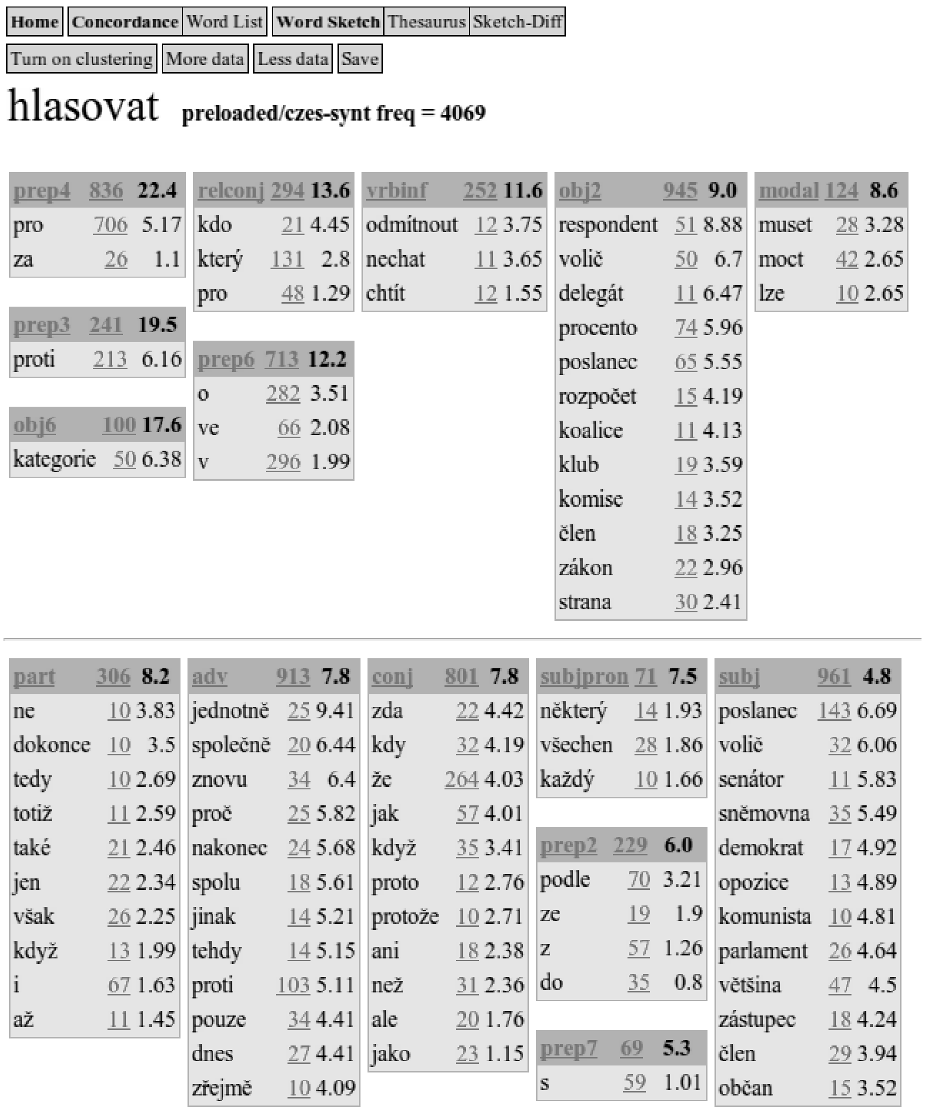
from	*label*	*to*
0	`part/PART`	2
1	`part/PART`	2
7	`left_modif/ADJ4`	8
10	`left_modif/ADJ6`	11
11	`prep/N6`	9
9	`np/PREP6`	8
5	`intr/VO3`	6
2	`intr/ADV`	6
4	`intr/ADV`	6
8	`intr/N4`	6
9	`intr/PREP6`	6

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.



hlasovat **preloaded/czes-synt freq = 4069**

prep4	836	**22.4**
pro	706	5.17
za	26	1.1

prep3	241	**19.5**
proti	213	6.16

obj6	100	**17.6**
kategorie	50	6.38

relconj	294	**13.6**
kdo	21	4.45
který	131	2.8
pro	48	1.29

prep6	713	**12.2**
o	282	3.51
ve	66	2.08
v	296	1.99

vrbinf	252	**11.6**
odmítnout	12	3.75
nechat	11	3.65
chtít	12	1.55

obj2	945	**9.0**
respondent	51	8.88
volič	50	6.7
delegát	11	6.47
procento	74	5.96
poslanec	65	5.55
rozpočet	15	4.19
koalice	11	4.13
klub	19	3.59
komise	14	3.52
člen	18	3.25
zákon	22	2.96
strana	30	2.41

modal	124	**8.6**
muset	28	3.28
moct	42	2.65
lze	10	2.65

part	306	**8.2**
ne	10	3.83
dokonce	10	3.5
tedy	10	2.69
totiž	11	2.59
také	21	2.46
jen	22	2.34
však	26	2.25
když	13	1.99
i	67	1.63
až	11	1.45

adv	913	**7.8**
jednotně	25	9.41
společně	20	6.44
znovu	34	6.4
proč	25	5.82
nakonec	24	5.68
spolu	18	5.61
jinak	14	5.21
tehdy	14	5.15
proti	103	5.11
pouze	34	4.41
dnes	27	4.41
zřejmě	10	4.09

conj	801	**7.8**
zda	22	4.42
kdy	32	4.19
že	264	4.03
jak	57	4.01
když	35	3.41
proto	12	2.76
protože	10	2.71
ani	18	2.38
než	31	2.36
ale	20	1.76
jako	23	1.15

subjpron	71	**7.5**
některý	14	1.93
všechen	28	1.86
každý	10	1.66

prep2	229	**6.0**
podle	70	3.21
ze	19	1.9
z	57	1.26
do	35	0.8

prep7	69	**5.3**
s	59	1.01

subj	961	**4.8**
poslanec	143	6.69
volič	32	6.06
senátor	11	5.83
sněmovna	35	5.49
demokrat	17	4.92
opozice	13	4.89
komunista	10	4.81
parlament	26	4.64
většina	47	4.5
zástupec	18	4.24
člen	29	3.94
občan	15	3.52

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 ⟨subj, hlasovat, poslanec⟩ and ⟨subj, rozhodovat, poslanec⟩), 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.

Part IV

Text Annotation and Classification

Applying Word Sketches to Russian*

Maria Khokhlova[1,2]

[1] Faculty of Philology and Arts, St. Petersburg University, Russia
[2] Institute for Linguistic Studies, St. Petersburg, Russia

Abstract. The paper describes work on writing a Russian Sketch grammar for the system Sketch Engine. The objective of such a system is to provide lexicographers with sufficient lexical material and tools for getting information about a word's collocability and to generate lists of the most frequent phrases for a given word, and then to classify them for appropriate syntactic models. The system will give information about a word's collocability on concrete dependency models, and will generate lists of the most frequent phrases for a given word for various grammatical models.

Key words: Word Sketches; Russian

1 Introduction

The system known as Sketch Engine was developed by British and Czech scholars (A. Kilgarriff, P. Rychlý, H. Pomikálek; [1]). The Sketch Engine combines approaches of both traditional linguistics (e.g. syntactic models) and statistics. It is widely used by scholars when compiling grammars and dictionaries (Oxford University Press, Cambridge University Press, Collins, Macmillan etc.). It was developed for a number of languages (English, Irish, Spanish, Italian, German, Portuguese, Slovene, French, Czech, Chinese, Japanese). However, there is no such a system for the Russian language. Sketch Engine is a corpus tool which takes as input a corpus of any language and corresponding grammar patterns and which generates word sketches for words of that language. Word sketches are one-page automatic, corpus-based summaries of a word's grammatical and collocational behaviour [2,3]. One can understand word sketches as typical phrases determined on the one hand by syntax that restricts words' collocability in a given language and on the other hand by probability closely related to word usage.

2 Methods of Corpus Linguistics and Collocations

Corpora are vital tools for linguistic studies and solution for applied tasks. The application of corpora methods to the analysis of lexical collocability enables

* This project was supported by the Russian Humanitarian Scientific Fund (grant number # 07-04-00161a).

to write grammars and compile dictionaries of a new type, dictionaries of collocations, idioms etc. The issue of collocability is highly important in modern linguistics. The investigation of collocability is closely connected to the study of syntagmatics as a deeper level of lexical relations. With arrival of text corpora and corpus linguistics lexicographers and other linguists have gained an opportunity to look at big collections of word usage. Corpora not only help to study lexical units in context but also to get data on word frequency, frequency of lexemes, grammatical categories, their collocability etc.

Although the above mentioned corpora opportunities are very useful, there is a need of another kind of software for further improvement of linguistic research as it is impossible to process huge amount of linguistic data manually. It can be described as an additional system between a corpus and its users (linguists) which can process significant language data.

The problem of syntagmatic relations is one of the most notorious in linguistics. There are various concepts of collocation and ways of how to extract collocations. Statistical methods for data treatment are widely used in corpus linguistics. Our intention is to study statistical methods of collocation extraction in comparison with the traditional (semantic) methods. That's why we chose the system Sketch Engine as a platform for implementing this task. Other software for processing corpus data (various corpus managers etc.) does not provide such features.

Nowadays there are several ways in statistics to calculate coherence of collocation parts, to highlight the most important ones. There are different measures based on calculation of words' "closeness" in a text, namely, MI (mutual information), t-score, log-likelihood, z-score, chi-square. They are based on comparison of frequencies registered for pairs of words in a real corpus material with independent (relative) frequencies. And statistically significant deviations of real frequencies from hypothetical probabilities are being searched. But formulas for different measures more often than not produce elevated numbers for word frequency, length of word window etc. As a result, they extract not only set phrases but free phrases as well as lexical items of the same semantic fields. The association measures do not take into account grammatical relations between tokens either. Besides, the statistical methods give significant results when they are based on representative corpora. Thus it is a need in such corpora that often lack.

3 Building Syntactic Models of Phrases in Russian

3.1 Corpus Building

The first preparatory stage of the project consisted in collecting texts to build a corpus of Russian. Originally we had a test corpus of letters of N.V. Gogol' [4], a famous Russian writer (1809–1852). This corpus contained about 0.5 mln tokens. As far as we know there isn't any work on extracting collocations on such a material (Russian texts of the XIXth century). The Russian language of the XIXth

century is notable for syntactic constructions that are different from modern ones. During this work (described in [5]) we have shown that methods presented can be effectively used for studying the authors' language and writing authors' dictionaries, for revealing collocability of words in different styles or within the given time period.

Afterwards we decided to make a number of corpora that reflect various language styles. They are fiction (about 10 mln tokens), scientific texts (about 0.5 mln tokens), news (about 5 mln tokens; journalistic genre), and texts of "common" style from the Internet (subcorpus of 10 mln tokens, this only corpus was compiled by S.A. Sharoff). This proportion can be seen as a strange one but we speak only about first steps in this project. Further work will be done on increasing corpora (their volume and number). This choice was motivated by a number of reasons. First of all to obtain better results we need to have quite similar texts (time period, genre etc.). Secondly, texts should be homogeneous (inside one corpus), have similar structure to give more statistical "weight" to its phrases (as their probability will be higher). The issue of corpus composition is a crucial one in linguistics, but we do not intend to discuss it here for lack of space and it wasn't our goal to compile corpora in this "narrow" scientific sense.

Then these texts were uploaded to the Sketch Engine where they were automatically processed and morphologically lemmatized and annotated by the program TreeTagger [6]. The Sketch Engine input format, often called "vertical" or "word-per-line", is as defined at the University of Stuttgart in the 1990s and widely used in the corpus linguistics community. Each token (e.g., word or punctuation mark) is on a separate line and where there are associated fields of information, typically the lemma and a POS-tag; they are included in tab-separated fields. Structural information, such as document beginnings and ends, sentence and paragraph mark-up, and meta-information such as the author, title and date of the document, its region and its text type, are presented in XML-like form on separate lines [7].

3.2 Word Sketch Grammar

The Sketch Engine needs to know how to select words that are connected by grammatical relations, i.e. that can be possibly collocations. That's why a scholar has to write a set of rules that describe grammatical relations that exist between words (word pairs). Strictly speaking, grammatical relations are defined as regular expressions over part-of-speech tagging.

During the second stage we investigated various sets of rules for different languages (English, Czech, Slovak etc), made a comparison of differences in the Russian and Czech syntax relevant to word sketches and then wrote grammatical rules that take into account syntactic constructions of the Russian language based on the morphologically tagged corpus in terms of grammar of Sketch Engine. This grammar represents itself a collection of definitions that allow the system to automatically identify possible relations of words to the keyword. On the basis of these rules and statistical measures it generates tables with word sketches for a keyword.

While writing rules we used regular expressions and query language IMS Corpus Workbench. The system searches for tags which correspond to word forms. For example, tag *Ncfpnn* means common noun *(Nc)* female gender *(f)* plural *(p)* noun case *(n)*: «Эти /P---pn/ этот перспективы /Ncfpnn/ перспектива и /C/ исвязаны/Afp-p-s/ связанный». After slashes there are a POS-tag and lemma. Below there is an example of grammatical rules for the phrases *"adjective+noun"*:

```
*DUAL
=a_modifier/modifies
  2:"A....n." (([word=","]|[word="и"]|[word="или"]) [tag="A....n."])0,3 1:"N...n."
  2:"A....g." (([word=","]|[word="и"]|[word="или"]) [tag="A....g."])0,3 1:"N...g."
  2:"A....d." (([word=","]|[word="и"]|[word="или"]) [tag="A....d."])0,3 1:"N...d."
  2:"A....a." (([word=","]|[word="и"]|[word="или"]) [tag="A....a."])0,3 1:"N...a."
  2:"A....i." (([word=","]|[word="и"]|[word="или"]) [tag="A....i."])0,3 1:"N...i."
  2:"A....l." (([word=","]|[word="и"]|[word="или"]) [tag="A....j."])0,3 1:"N...l."
```

Above mentioned rules take into account all such phrases, e.g. nouns and adjectives in the same case with conjunctions «и» ("and"), «или» ("or"), comma or adjectives between them within the distance of 3 words. The numeral 1 stands for a keyword (for instance, *1:"N...n."*) and the numeral 2 indicates a collocate (for instance, *2:"A...n."*). For example, «лучшие /Afp-pnf/ хороший помощники /Ncmpny/ помощник», «печатный /Afpmsnf/ печатный текст /Ncmsnn/ текст», «яркие /Afp-pnf/ яркий мысли/Ncfpnn/ мысль», «сегодняшний /Afpmsaf/ сегодняшний день /Ncmsan/ день», «благоприятные /Afp-paf/ благоприятный условия /Ncnpan/ условие», «потенциальным /Afp-pdf/ потенциальный возможностям /Ncfpdn/ возможность», «стандартным/Afp-pdf/ стандартный кредитам /Ncmpdn/ кредит». Here are several examples of relations between words: =subject/subject_of («собака лает» / "the dog is barking") =object/object_of («принять решение» / "make a decision") =a_modifier/modifies («крепкий чай» / "strong tea") Originally these rules were written on the basis of existing rules for English and Czech [3]. Then we have written the second variant of word sketches rules within the approach of Vladimir Benko (oral paper presented at Mondilex workshop in Bratislava, April 2009) [8] for the Slovak National Corpus [9]. Its distinctive feature is that these rules describe all phrases found in a corpus. For example, "verb + any word" (see below):

```
=Verb X/X Verb
    2:[tag="V.*"] 1:[tag!="SENT"]
    1:[tag!="SENT"] 2:[tag="V.*"]
```

The second line means that there will be found all phrases for any word (if it isn't a punctuation mark that has its own tag in the corpus) with a verb. The rule in the third line describes the same phrases but a verb is to the right of a keyword.

It should be remarked that this approach has its advantage as word sketches are generated for any word (because very often morphological ambiguity or mistakes of automatic tagging prevent from giving objective results).

In the theory of information retrieval there are two notions – "precision" and "recall". Precision means the percentage of documents returned that are relevant, i.e. in case of words it's the percentage of correct collocations compared to all phrases given. Recall is the fraction of the documents that are relevant to the query (that are successfully retrieved), i.e. the fraction correct collocations between all the collocations. Let's consider the following example. If our word sketch for *"tea"* contains only *"strong"* and *"green"*, it has 100% precision, since all the collocates given are correct, but low recall, since there are many other collocates it does not give. Using these terms we can say that the first approach (the first variant of rules) gives higher precision while the second one higher recall.

3.3 Word Sketch Tables

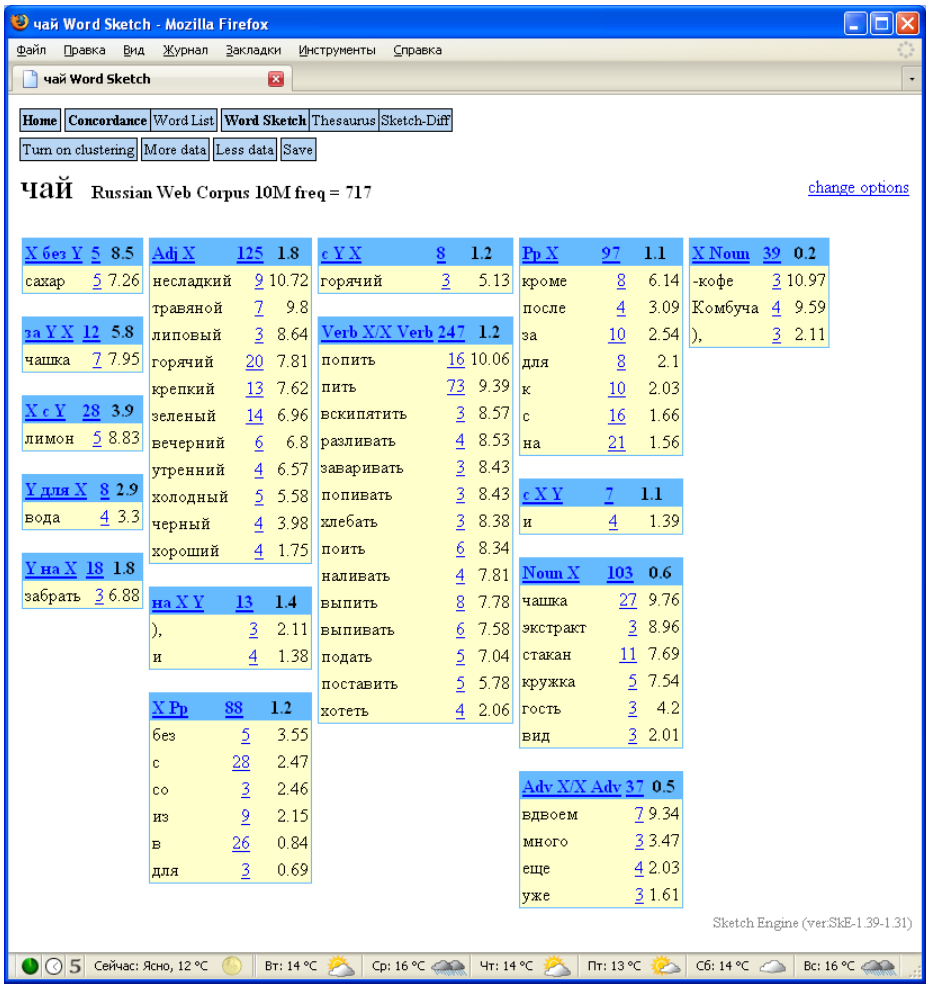
Table 1 shows word sketch for the Russian word «чай» ("tea"). The blue heading of each small table has the name of the grammatical relation between words. X stands for the keyword, whereas Y signifies a collocate. In the column *"Adj X"* (the model "adjective + keyword") we find typical qualifying adjectives (that can be applied to other nouns too), several set phrases, and also terms (they are true for English too): «несладкий» ("non-sweet"), «травяной» ("herbal"), «липовый» ("lime leaf"), «горячий» ("hot"), «крепкий» ("hot"), «зеленый» ("green"), «вечерний» ("evening"), «утренний» ("morning"), «холодный» ("cold"), «черный» ("black"), «хороший» ("good"). As for the column "Verb X/X Verb" (the model "verb + keyword / keyword + verb") here we also find collocates that are inherent for the word "tea" in Russian. They are «пить» or «попить» ("to drink"), «вскипятить» ("to boil"), «разливать» ("to pour"), «заваривать» ("to brew"), «хлебать» ("to gulp"), «подать» ("to serve"). The user can choose various options for the display of the word sketches. Collocates can be ranked according to the raw frequency of the collocation, or according to its salience score [10]. The user can set a frequency threshold so low-frequency collocations are not shown, or click a button for "more data" or "less data". They can go to the related concordance by clicking on the hit-count for a collocation.

3.4 Word Sketch Differences

Once the word sketch grammar is written this information is used in other Sketch Engine feature, namely, Word Sketch Differences. This feature shows for two semantically related words their behaviour (what they do have in common and in what differ). This information is presented in the form of multicolored diagrams. Such summary offers both common collocates that share the comparing pair and also collocates that are inherent only for one word in this pair. Synonymous words tend to share some of the collocates but not all.

Table 2 shows word sketch differences for the Russian words «большой» ("big") and «крупный» ("large"); the number of tokens for «большой» is 7593, for «крупный» is 1997. The compared two words are on each end of the multicolored scale. The yellow color shows common collocates (as we can see this part is the biggest one), the green one denotes collocates for the word «большой», and the pink one indicates collocates for the word «крупный». Each table has

Table 1. Word sketch for the Russian word «чай» (“tea”)


чай Word Sketch - Mozilla Firefox
Файл Правка Вид Журнал Закладки Инструменты Справка
чай Word Sketch
Home Concordance Word List Word Sketch Thesaurus Sketch-Diff
Turn on clustering More data Less data Save
change options
Сейчас: Ясно, 12 °C Вт: 14 °C Ср: 16 °C Чт: 14 °C Пт: 13 °C Сб: 14 °C Вс: 16 °C

чай Russian Web Corpus 10M freq = 717

X без Y	5	8.5
сахар	5	7.26

за Y X	12	5.8
чашка	7	7.95

X с Y	28	3.9
лимон	5	8.83

Y для X	8	2.9
вода	4	3.3

Y на X	18	1.8
забрать	3	6.88

Adj X	125	1.8
несладкий	9	10.72
травяной	7	9.8
липовый	3	8.64
горячий	20	7.81
крепкий	13	7.62
зеленый	14	6.96
вечерний	6	6.8
утренний	4	6.57
холодный	5	5.58
черный	4	3.98
хороший	4	1.75

на X Y	13	1.4
),	3	2.11
и	4	1.38

X Pp	88	1.2
без	5	3.55
с	28	2.47
со	3	2.46
из	9	2.15
в	26	0.84
для	3	0.69

с Y X	8	1.2
горячий	3	5.13

Verb X/X Verb	247	1.2
попить	16	10.06
пить	73	9.39
вскипятить	3	8.57
разливать	4	8.53
заваривать	3	8.43
попивать	3	8.43
хлебать	3	8.38
поить	6	8.34
наливать	4	7.81
выпить	8	7.78
выпивать	6	7.58
подать	5	7.04
поставить	5	5.78
хотеть	4	2.06

Pp X	97	1.1
кроме	8	6.14
после	4	3.09
за	10	2.54
для	8	2.1
к	10	2.03
с	16	1.66
на	21	1.56

с X Y	7	1.1
и	4	1.39

Noun X	103	0.6
чашка	27	9.76
экстракт	3	8.96
стакан	11	7.69
кружка	5	7.54
гость	3	4.2
вид	3	2.01

Adv X/X Adv	37	0.5
вдвоем	7	9.34
много	3	3.47
еще	4	2.03
уже	3	1.61

X Noun	39	0.2
-кофе	3	10.97
Комбуча	4	9.59
),	3	2.11

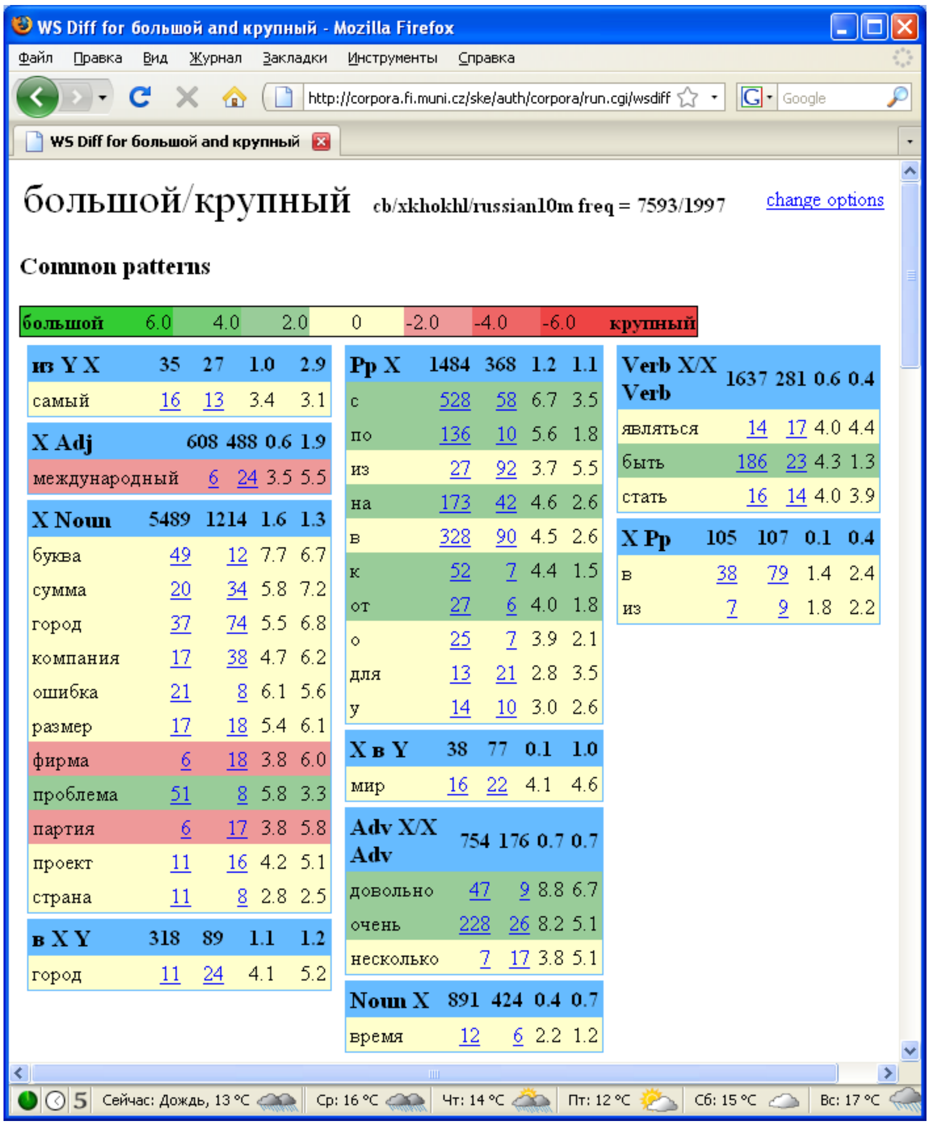
Sketch Engine (ver:SkE-1.39-1.31)

five columns: a collocate, a collocate’s frequency for the first word, a collocate’s frequency for the second word, and statistical measures (in this case it’s salience, computed for the collocate and the word).

4 Results

There is a question of corpus volume. For example, we know that different association measures extract different collocations but here one can’t see differences between results obtained by a number of statistical measures, it means that collocates will be quite the same. This problem arises from low

Table 2. Word sketch differences for the Russian words «большой» (“big”) and «крупный» (“large”)


WS Diff for большой and крупный - Mozilla Firefox

http://corpora.fi.muni.cz/ske/auth/corpora/run.cgi/wsdiff

большой/крупный cb/xkhokhl/russian10m freq = 7593/1997 change options

Common patterns

большой	6.0	4.0	2.0	0	-2.0	-4.0	-6.0	крупный

из Y X	35	27	1.0	2.9
самый	16	13	3.4	3.1

X Adj	608	488	0.6	1.9
международный	6	24	3.5	5.5

X Noun	5489	1214	1.6	1.3
буква	49	12	7.7	6.7
сумма	20	34	5.8	7.2
город	37	74	5.5	6.8
компания	17	38	4.7	6.2
ошибка	21	8	6.1	5.6
размер	17	18	5.4	6.1
фирма	6	18	3.8	6.0
проблема	51	8	5.8	3.3
партия	6	17	3.8	5.8
проект	11	16	4.2	5.1
страна	11	8	2.8	2.5

в X Y	318	89	1.1	1.2
город	11	24	4.1	5.2

Pp X	1484	368	1.2	1.1
с	528	58	6.7	3.5
по	136	10	5.6	1.8
из	27	92	3.7	5.5
на	173	42	4.6	2.6
в	328	90	4.5	2.6
к	52	7	4.4	1.5
от	27	6	4.0	1.8
о	25	7	3.9	2.1
для	13	21	2.8	3.5
у	14	10	3.0	2.6

X в Y	38	77	0.1	1.0
мир	16	22	4.1	4.6

Adv X/X Adv	754	176	0.7	0.7
довольно	47	9	8.8	6.7
очень	228	26	8.2	5.1
несколько	7	17	3.8	5.1

Noun X	891	424	0.4	0.7
время	12	6	2.2	1.2

Verb X/X Verb	1637	281	0.6	0.4
являться	14	17	4.0	4.4
быть	186	23	4.3	1.3
стать	16	14	4.0	3.9

X Pp	105	107	0.1	0.4
в	38	79	1.4	2.4
из	7	9	1.8	2.2

frequencies of words and phrases. As was pointed above we are going to work on further corpus data increase.

A number of problems arise from errors in morphological annotation as: 1) every punctuation mark has its own tag (so it should be excluded in the sketch grammar); 2) parts of compound nouns also have different lemmata that is why in sketch tables we can find only one part of such words as a collocate; 3) usual mistakes of annotation, e.g. homonyms or homographs, mistakes in assigning the correct case or number; 4) mistakes in assigning correct lemmata (it is especially the case while annotating texts of the last centuries or, vice versa, of modern period with lots of neologisms).

The evaluation of the results obtained suggests that the word sketch mechanism is a useful tool for selecting the most significant collocations that are often not presented in dictionaries.

5 Conclusion

We believe that the present project may contribute to the theoretical studies of the Russian language (at the borderland between lexicography and syntax) as well as to the solution of a number of practical issues.

Further development of this mechanism of collocation extraction is closely related to writing more exact grammatical rules (that will be based on syntactically parsed corpus), more corpus data etc. Most errors in the word sketches result from errors in lemmatisation and POS-tagging. We are currently explore alternative tools for automatic morphological annotation. Manual morphological disambiguation can be seen as a possible solution for the problem of reducing errors of annotation. But this work is labour- and time-consuming and unfortunately can be applied only to a small part of a corpus.

Also there is a question of further sketch grammar improvement. New variant of the sketch grammar should be based on compilation of various grammars of the Russian language (Russian Academy Grammar [11] etc.).

The results of the research project are of practical value, as the information about a word's collocability is not often reflected in dictionaries and other reference books. The data about words' syntagmatic behaviour may find an extensive use in various fields of linguistics, such as in: dictionary compiling, language learning and teaching, translation (including machine translation), phraseology, information retrieval etc.

Acknowledgement I cordially thank my friends and colleagues from the NLP laboratory of the Masaryk University (Brno, Czech Republic) for their help and support.

References

1. Sketch Engine project: `http://www.sketchengine.co.uk`
2. Kilgarriff, A., Rychlý, P., Smrž, P., Tugwell, D. (2004). The Sketch Engine. In: Proceedings of EURALEX-2004, 105–116.
3. Rychlý, P., Smrž, P. Manatee, Bonito and Word Sketches for Czech. (2004). In: Trudy mezhdunarodnoy konferentsii "Korpusnaja lingvistika-2004": Sbornik dokladov. St.-Petersburg, 324–334.

4. Gogol, N.V. Polnoye sobraniye sochineniy: [V 14 t.]. (1937–1952). Moscow – St.-Petersburg, 1937–1952. T. X-XIV.
5. Khokhlova, M., Zakharov, V. Corpus-based analysis of lexico-grammatical patterns (on the corpus of letters of N.V. Gogol). (2009). In: Proceedings of the Fifth International Conference "Computer Treatment of Slavic and East European Languages", Bratislava, Slovakia, 25–27 November 2009. Bratislava. (in print)
6. TreeTagger: `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger`
7. Documentation to the Sketch Engine: `http://trac.sketchengine.co.uk`
8. Benko, V. Word Sketches for the Slovak National Corpus [Oral presentation at Mondilex workshop]:
`http://korpus.juls.savba.sk/~mondilex/programme3.pdf`
9. Slovak National Corpus: `http://korpus.sk`
10. Rychlý, P. A Lexicographer-Friendly Association Score. (2008) In: Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2008. Brno, 6–9.
11. Russkaja grammatika. (1980) Toma I, II. Moscow. (AG-80).

Prague Dependency Treebank Annotation Errors
A Preliminary Analysis

Vojtěch Kovář and Miloš Jakubíček

Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{xkovar3, xjakub}@fi.muni.cz

Abstract. This paper presents a basic analysis of syntactic annotation errors and inconsistencies in the Prague Dependency Treebank, the biggest corpus of Czech with manual syntactic annotation. The corpus is used for developing and testing of many syntactic analysers of Czech and the problems in the annotation have an essential impact on the evaluation of the quality of these parsers and the results of precision measurements. We identify some of the basic annotation problems and in some cases, we outline possible solutions.

Key words: error in text; annotation; Prague Dependency Treebank

1 Introduction

The Prague Dependency Treebank (PDT, [1]) forms the only big source of manually annotated Czech syntactic data. Currently, this corpus contains about two million tokens annotated in three layers – morphological, analytical and tectogrammatical – and it is of great use in the process of developing and testing syntactic analysers of Czech.

However, there is a large number of inconsistencies and errors in the data which makes using the corpus quite problematic and questionable. These flaws result from various reasons ranging from insufficient annotation guidelines and apparent mistakes to shortcomings of the annotation as it has been formalised. Furthermore, it is not clear what the percentage of the wrong annotation is and how it can affect the measurements that use the PDT as the gold standard data used for training and testing various algorithms.

In this paper, we present a preliminary analysis of errors and inconsistencies in a PDT sample. We describe some problems in annotation that were revealed during the work with this sample, try to figure out the sources of the particular problems and suggest possible solutions. We also estimate the overall percentage of error in our sample and, assuming that the sample is representative, in the whole corpus.

2 The Prague Dependency Treebank

The Prague Dependency Treebank has been developed according to the tradition of the Prague linguistic school – it uses the formalism of Functional Generative

Description [2,3]. The annotation consists of three layers – morphological, analytical and tectogrammatical.

Within the scope of this paper, we are interested in the analytical layer only. This part of annotation contains the description of the dependency syntax in form of labeled dependency trees (acyclic oriented graphs over the input tokens). Furthermore, we will deal just with the structure of the trees, not the functional classification of the particular edges that is recorded as edge labels. This classification is not so critical and most parsers also do not label their outputs.

In the whole text, we refer to the current version of the corpus, PDT 2.0.

2.1 The Sample

For training and testing purposes, the PDT data is divided into 10 parts – 8 of them are provided for training, 2 are dedicated to testing.

For the purposes of this paper, we used the beginning of the first training set, *train-1* that was previously used by development of the *SET* parsing system [4]. Most of the examples come from the first 60 sentences of this testing set since these sentences were checked many times during the parser development and we know them very well.

3 Error Analysis

In this section, we present examples of errors, inconsistencies and other problems in our sample and briefly discuss various aspects of these problems.

3.1 Random Errors

The first group of problems we met during the parser development were apparent errors in the annotation. Such errors cannot be explained as inconsistencies or flaws of the annotation formalism, they are just random defects created by the human annotators. The existence of such random errors is unavoidable in all human annotated data and it must be presumed that anything done by humans, including the annotators, can and will be erroneous to some extent. However, every effort should be made to keep the number of annotator errors as low as possible.

As an example, we show the beginning of the sentence #00040 (see Figure 1). There are two apparent problems:

- The dependency *Většinu* ← *fax* (*Most* ← *fax*). There is no reason for this markup, the phrase *jako fax* (*as fax*) clearly belongs to the phrase *jako výkonnou kopírku* (*as an efficient copier*) – these two phrases should be joined in a coordination.
- The coordination in the top level of the tree. Previously mentioned coordination should be marked instead of this one and the whole structure should depend on the verb *používat* (*use*).

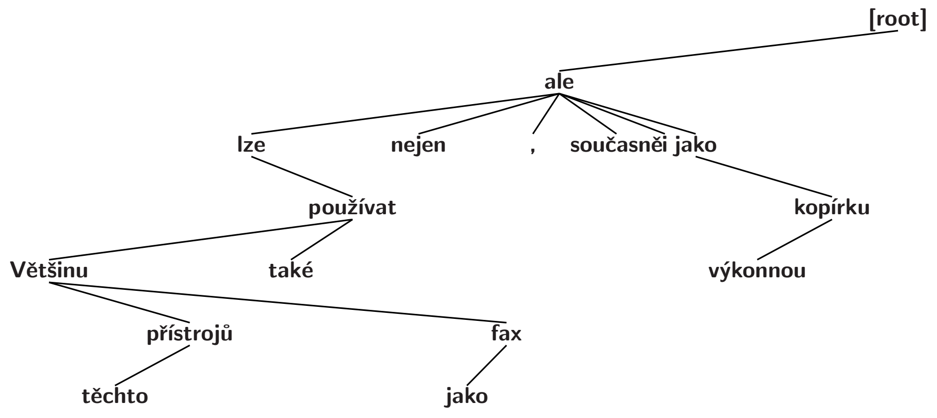


Fig. 1. The first part of the sentence #00040 as recorded in the PDT: *Většinu těchto přístrojů lze také používat nejen jako fax, ale současně i jako výkonnou kopírku...* (*Most of these devices can be used not only as fax but in the same time also as an effective copier…*)

We can see that such quite simple mistakes can globally change the structure of the tree (which might be also seen as a disadvantage of the dependency annotation formalism) and usage of sentences with such errors is very problematic in every possible application.

3.2 Inconsistencies

Inconsistencies in the annotation seem to be a bigger problem than random errors. They occur systematically in the corpus and are very common. Although an extensive manual for annotators is provided [5] to avoid these problems, there are still many language phenomena that are not described clearly enough or are even not described at all. The creativity of annotators then creates more annotation variants for a single phenomenon. According to our estimations, about 30 or 40 % sentences contain one of the phenomena that are marked inconsistently in some of the sentences.

Our first example of inconsistency shows annotation of punctuation in parts of item lists. Both sentences in Figures 2 and 3 contain an asterisk that has definitely the same meaning in both cases. However, the two annotations differ. No matter which of these variants is correct, in case of short sentences as in our two examples, the edge adjacent to the asterisk stands for 20 or 25 percent of sentence annotation.

The first example was rather technical and could be basically solved by some automatic or semi-automatic procedures. More serious inconsistencies can be found in annotation of frequent linguistic phenomena, e.g. passive verb forms. As illustrated in Figures 4, 5 and 6, this phenomenon is marked in various ways in the corpus.

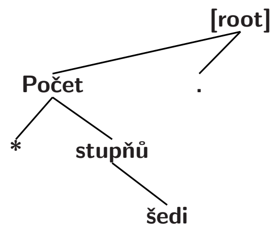


Fig. 2. The sentence #00048 as recorded in the PDT: * *Počet stupňů šedi* (* *Number of levels of grey*)

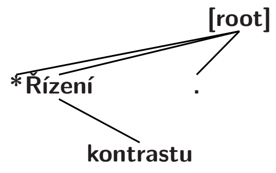


Fig. 3. The sentence #00053 as recorded in the PDT: * *Řízení kontrastu* (* *Contrast control*)

In Figure 4, the auxiliary verb *je* (*is*) is at the top of the phrase and the complement (*jaké*) depends on the participle form of the verb. In Figure 5, the auxiliary verb is still on the top of the phrase but the complements of the predicate depend on the auxiliary verb. Finally, in the third example (Figure 6), the participle is on the top of the phrase and all the rest including the auxiliary verb depends on the participle.

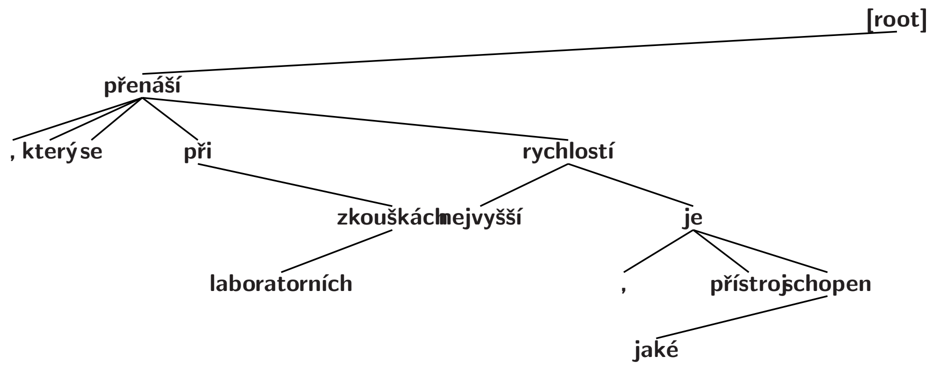


Fig. 4. Part of the he sentence #00012 as recorded in the PDT: *..., který se přenáší při laboratorních zkouškách nejvyšší rychlostí, jaké je přístroj schopen...* (*..., that is tranferred in laboratory tests with the highest speed that is the device able...*)

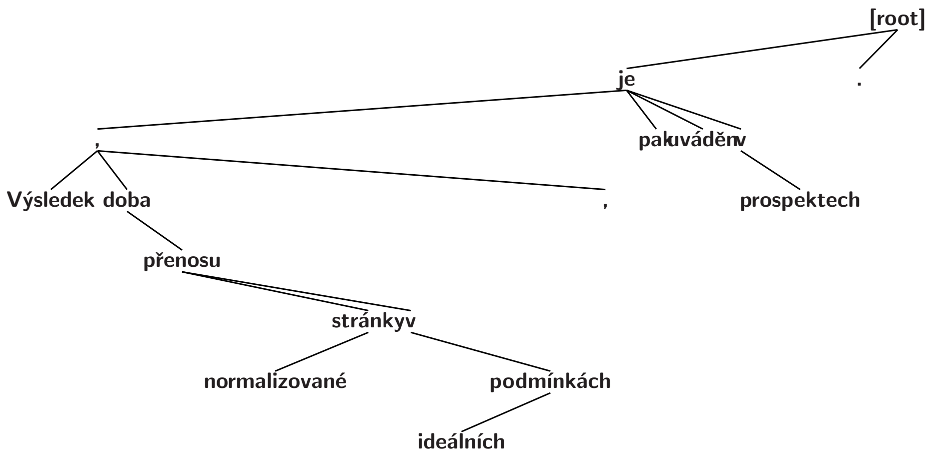


Fig. 5. The sentence #00013 as recorded in the PDT: *Výsledek, doba přenosu normalizované stránky v ideálních podmínkách, je pak uváděn v prospektech.* (*The result, the transfer time of the normalized page in ideal conditions, is then reported in brochures.*)

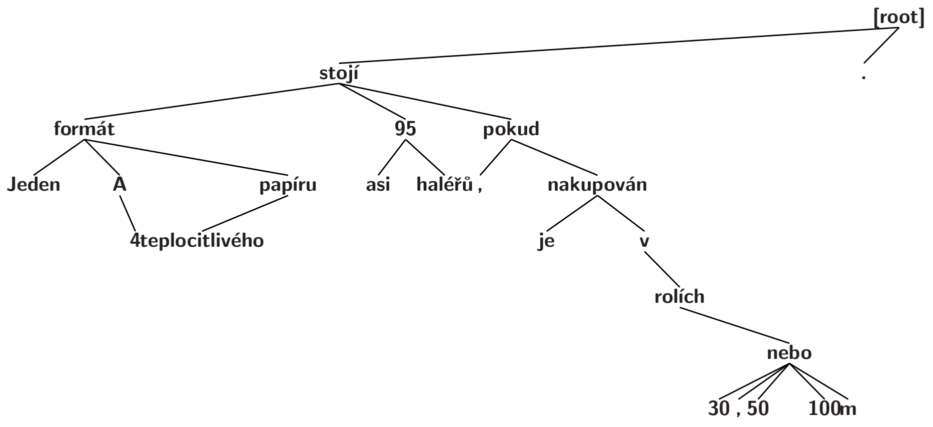


Fig. 6. The sentence #00028 as recorded in the PDT: *Jeden formát A 4 teplocitlivého papíru stojí asi 95 haléřů, pokud je nakupován v rolích 30, 50 nebo 100 m.* (*One page A 4 of the thermosensitive paper costs approximately 95 hellers, if it is bought in reels 30, 50 or 100 m.*)

This inconsistency in annotating the predicate structure is very painful since this structure determines the shape of the whole clause. For instance, in the process of parser developing and testing, developers (or training algorithms) have to search the most frequent annotation pattern for this case so that the parser has maximum precision against the data. However, they are doomed to fail when trained on such inconsistent data.

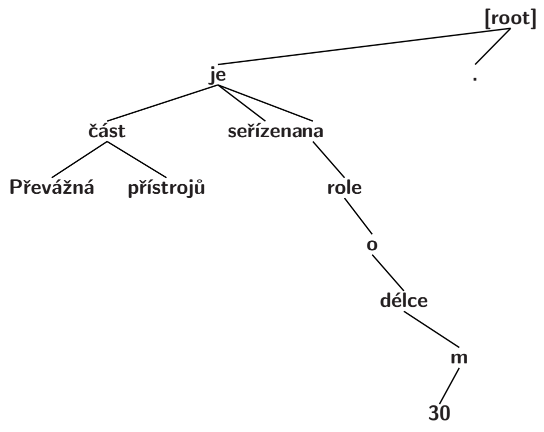


Fig. 7. The sentence #00030 as recorded in the PDT: *Převážná část přístrojů je seřízena na role o délce 30 m.* (*Most devices are adjusted for reels of length 30 m.*)

Another similar problem is annotation of phrases with numerals. Though they are well covered in the annotation manual that we previously mentioned, annotation of many sentences does not respect the instructions. An example is shown in Figure 7 (the "m" token should depend on the numeral in this case, according to the annotation manual).

3.3 The "Dirty" Cases

Although the problems showed above are the most remarkable ones, we are still far from a complete error list. In some cases, it is not clear if the particular annotation is a mistake or an intention.

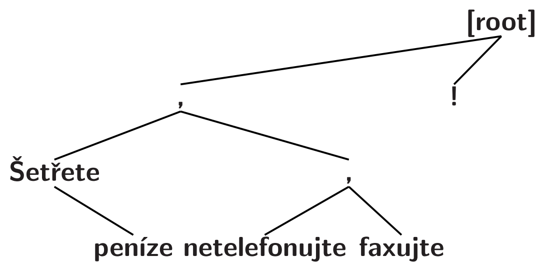


Fig. 8. The sentence #00004 as recorded in the PDT: *Šetřete peníze, netelefonujte, faxujte!* (*Save money, do not phone, fax!*)

This is the case shown in Figure 8 – it is not clear why the coordination is structured in this particular way. There might be a doubtful semantic hint that members of the segment *netelefonujte, faxujte* (*do not phone, fax*) has closer relationship with each other than with the first phrase in the sentence, however, it is a question if any possible parser could reveal that hint. In our opinion, such cases should be annotated in the most straightforward way possible – as a flat coordination of three verbs. Unfortunately, the difference between the structured and the flat coordination markup in this case is more than 50 percent, which is to be taken as a flaw of the annotation formalism.

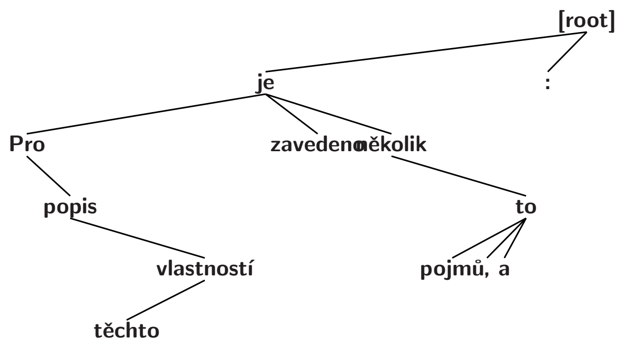


Fig. 9. The sentence #00047 as recorded in the PDT: *Pro popis těchto vlastností je zavedeno několik pojmů, a to:* (*For description of these properties, some terms are introduced, as follows:*)

In the last example (Figure 9), there is another strange annotation example in the end of the sentence – the phrase *pojmů, a to*. The structure of this phrase does not seem to have any rational basis, it just needs to fit into the dependency format *somehow*. It is even not clear, why these words should belong to one phrase.

4 On Parser Evaluation

As we have already mentioned above, even if big effort is made to eliminate annotator errors, some of them will still remain – and we dare to predict that the number of errors grows with the size of the data in a non-linear way. This raises a question whether treebanks represent a good way for measuring parser quality at all. Besides treebank consistency issues, there is more evidence which makes the use of treebanks for parser evaluation questionable: often the evaluation is significantly influenced by the different formalisms, annotations and last (but definitely not least!) by different linguistic insights and opinions. Finally, there have been recently proposed application-driven evaluation techniques for parsers (see e. g. [6]) which we believe to continue becoming more widely used in the parsing community. A detailed discussion of this topic is however outside of the scope of this paper.

5 Conclusion

We have described the main problems in the PDT annotation that we met during the process of parser development. We have presented examples of the selected problems and also showed the fact that in some cases, one simple mistake or inconsistency can lead to structural changes in the sentence annotation and significantly affect the results of parser tests and development. This can be considered as a negative feature of the dependency annotation formalism in general.

The total number of errors in the annotation in our sample is not clear because there is not a good characterization of what is an error. However, according to our estimations, the difference between the current state and the correctly and consistently annotated sentences would be 5 to 10 percent. This quite a big number may be one of the reasons why the current parsers of Czech are not so successful as parsers for English or German [7], although they have been intensively developed.

In the future, we want to perform more thorough critical analysis of the errors in the PDT corpus annotation and propose some automatic and semi-automatic methods leading to their elimination. We will also propose possible changes in the formalism and in the precision metrics used in the process of developing and testing syntactic parsers.

Acknowledgements This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536.

References

1. Hajič, J.: Building a syntactically annotated corpus: The Prague Dependency Treebank. In: Issues of Valency and Meaning, Prague, Karolinum (1998) 106–132.
2. Sgall, P.: Generativní popis jazyka a česká deklinace (Generative Description of the Language and the Czech Declension). Academia, Prague, Czech Republic (1967).
3. Sgall, P., Hajičová, E., Panevová, J.: The Meaning of the Sentence and Its Semantic and Pragmatic Aspects. Academia/Reidel Publishing Company, Prague, Czech Republic/Dordrecht, Netherlands (1986).
4. Kovář, V., Horák, A., Jakubíček, M.: Syntactic analysis as pattern matching: The SET parsing system. In: Proceedings of the 4th Language & Technology Conference, Poznań, Poland (2009).
5. Hajič, J., Panevová, J., Buráňová, E., Urešová, Z., Štěpánek, J., Pajas, P., Kárník, J.: Anotace na analytické rovině – Návod pro anotátory (2005) `http://ufal.mff.cuni.cz/pdt2.0/doc/manuals/cz/a-layer`.
6. Miyao, Y., Sagae, K., Saetre, R., Matsuzaki, T., Tsujii, J.: Evaluating contributions of natural language parsers to protein-protein interaction extraction. Bioinformatics **25**(3) (2009) 394–400.
7. Baumann, S., Brinckmann, C., Hansen-Schirra, C., et al.: The MULI project: Annotation and analysis of information structure in German and English. In: Proceedings of the LREC 2004 Conference, Lisboa, Portugal (2004).

Classification of Errors in Text

Jan Bušta, Dana Hlaváčková, Miloš Jakubíček, and Karel Pala

Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{xbusta,ydana,xjakub,pala}@fi.muni.cz
http://nlp.fi.muni.cz/

Abstract. This paper presents two classifications of errors in Czech texts. As a basic resource we use the corpus (Chyby – Errors) which has been continuously developed from 1999–2000 ([1]). The corpus text contains various kinds of errors such as spelling, typographical, grammatical, semantic, lexical, and stylistic ones. They have been corrected manually and annotated according to the classification of errors (annotation scheme) developed for this purpose. For the annotation we implemented a tool named WinCorr.
We mention the first annotation scheme and discuss the second one which has been designed recently to obtain more adequate description of the errors occurring in texts. We also discuss the principles on which both classifications are based.

Key words: errors in text; classification of errors

1 Introduction

In any text written by humans there always occur errors in spelling, grammar, semantics, style and typography. Not only this: *if humans correct errors in texts, they are not able to remove all of them*. That is why publishing houses and editorial boards have to employ readers and proof-readers whose task is to find errors in texts, correct them and finally produce printed texts of the best possible quality.

At present the prevailing majority of texts is produced on computers, which in turn are used for typesetting, storing and dissemination via Internet. No wonder that there is also a strong tendency to use computers to correct texts and remove errors from them. Programs called (spelling, grammar, style) *checkers* have appeared, and they allow us to correct some well recognised errors in texts. In some respects, they are more reliable than humans and are able to remove errors of some types completely.

The existing checkers are in some respects quite limited. Thus to be able to analyse all kinds of errors occurring in natural language texts, it is necessary to have a collection of texts (in our case in Czech) containing all kinds of errors. Therefore we decided to build a text corpus that contains various spelling, grammatical, style, semantic, typographical (and possibly other) errors and annotate them in the corpus text. The corpus with annotated errors is named **Chyby** ([1]).

In this paper we briefly report on building the Czech corpus Chyby and on how errors have been marked and annotated with the help of the tools (programs) developed particularly for this purpose. There are two of them, the old one is WinCorr [2], the new one is OOCorr [3] (see below).

2 Why the Chyby Corpus?

At a first glance, it might seem that standard general corpora such as BNC [4] or the Czech National Corpus [5], could serve reasonably well for our purposes. After closer inspection of the texts from these resources it appears that the general corpora mostly contain texts that already have been proof-read and corrected (newspaper texts or fiction etc.). They still contain some errors as mentioned above, but their number is rather small since the worst ones have been removed. However, if we watch humans in the process of producing texts spontaneously we observe a different picture. The number of errors in such texts is quite high and some of them are quite severe.

Thus we turned to *spontaneous texts (s-texts)*. These texts were generated by students at FI MU, who take in their Bachelor studies a subject called *Elements of Style*. During the course they have to write two kinds of texts: an essay and an introduction to their Bachelors theses, each of them comprising approx. 600–700 words. The submitted texts have been corrected manually by four teachers and returned to the students who have to prepare final corrected versions of their texts and annotate the marked errors electronically using a program developed for this purpose (WinCorr, OOCorr, see below). The corrected and annotated texts have been used for creating the corpus Chyby by means of the corpus manager *Bonito/Manatee* [6]). At present, the size of the Chyby is approx. 500,000 word forms.

The nature of the texts delivered by the students is in accordance with our idea of s-texts: the number of errors and their types can be considered representative enough. In some cases, the texts are not well written and in our view they contain a large percentage of errors. In 650 words it is sometimes possible to find about 30 bad errors, though not all errors are regularly related to the individual word forms. For example, they involve changing word order, deleting and substituting whole lines or even paragraphs.

3 How to Classify Errors in Text?

The starting point for our classifications of errors in texts and the annotation scheme based on it are the Rules of Czech Orthography [7] and their electronic version [8], an official reference manual published by the Institute of Czech Language, Czech Academy of Sciences. It describes the basic principles of Czech orthography, which in comparison with English are much more phonetically oriented, although they are governed by a number of historical rules as well, especially in what concerns of inflection. The Rules also contain the punctuation rules, which reflect the syntactic segmentation of Czech sentences, e. g. main and subordinate clauses are typically separated by commas on both sides and commas have to be placed before or after some conjunctions as well. In this respect Czech punctuation is somewhat complicated. This is the reason for a large percentage of the punctuation errors in the students' texts.

As a whole the Rules represent a reference manual based on the empirical rules, the majority of which can be characterised as deterministic (we estimate this amount at approx. 80 %). We have found it reasonable to start with the Rules and in combination with the data obtained from the Chyby, work out a more complete and formal description of the errors occurring in Czech texts together with their detailed categorisation. As far as we know, there is no general classification of errors that may occur in the texts. However, on the Web one can find reports and papers about grammar checkers and their development where overviews of the main types of errors can be found, see e. g. [9,10] or [11].

4 S-texts and Errors in Them

In agreement with rules of Czech orthography ([7]) the errors were originally classified in the following way:

- spelling,
- morphology,
- syntax (grammar),
- punctuation,
- lexical and semantic choice,
- style,
- typography.

This classification contains some subgroups and was used in the tool WinCorr [2]. We have been using it since 2002. During this time it served its purpose decently. However, there appeared various problems (e. g. it cannot handle the new ISO-standardized ODT document format which is used more and more extensively by our students). Thus we decided to revise the tool and develop a new and, hopefully, a more adequate one.

There are also several reasons for designing a new classification, though we are aware that it is possible to design an infinite number of them. We mention here the following points that we have been considering in the revision of the first error classification:

- the classification contains items that are overlapping. This, in our view, can hardly be avoided but it can be minimized. We are approaching it in the new classification.
- some of the spelling errors can be characterized as rather formal. These are mostly errors that can be discovered by a spelling checker.
- there are errors that on one hand can be characterized as spelling ones, but on the other hand also as grammatical (morphosyntactic) ones.
- a special group represents semantic and lexical errors. Their nature is not formal and can be discovered and corrected by humans only.

– the same can be said about stylistic errors though they grow from the language form,
– frequency considerations.

The new classification of errors as they occur in s-texts:

– spelling errors
  • obvious typing errors (recognizable by a spelling checker)
  • other typing errors (*i*/*y*, *s*/*z*, that cannot be recognized by a checker)
  • inflectional noun endings
  • syntactic (valencies, verbs–NPs, adjectives–NPs, agreement in NPs, NP–verbs)
  • capital letters, lowercases
  • compounds (mostly adverbial)
– punctuation (comma, colon, semicolon, dot, triple dot)
  • constituents (usually types of coordination)
  • clauses (relative clause, subject, object, adverbial, coordination)
– lexico-semantic errors
  • MWEs or sentences with broken meaningfulness
  • omitted or missing words
  • incorrect use of the possesives (*svůj*, *váš*,. . .)
  • incorrect choice of lexical items
– stylistic errors
  • incorrect register (colloquial, archaic, slang)
  • repeated expressions (demonstratives, adverbs, particles)
  • cumulation of the nouns ending with *-ní*
  • passive vs. reflexive passive
  • incorrect word order
  • clumsy expressions (MWEs, sentences)
  • too long sentences
– typographical errors
  • local errors: spaces (in acronyms), hyphens, inverted commas, brackets, one character consonant prepositions, incorrect characters
  • overall document layout: incorrect document structure, wrong paragraphization and hyphenation, orphans and widows, rags and rivers
  • incorrect choice of visualization means: inappropriate typeface, font properties or typesetting combination, low readability of text, wrong disposition of non-text items etc.

5 Annotation Scheme and Tags

In the previous section we indicated what kinds of errors we distinguish and want to annotate in the corpus Chyby. The next step is the design of the annotation scheme, which allows us to mark the errors and their types in the corpus text.

The original annotation scheme developed for the Chyby distinguishes the following types of errors:

- *Spelling*, errors which can be relatively well recognised in the texts and tools exist for their recognition (spelling checkers).
  Example: ***s**kouška* instead of correct *zkouška (examination)* or *standar**t**ní* instead of *standar**d**ní (standard).*
  tag: `errtype=prav-pism`,
- *Typographical* errors consist in the incorrect use of various characters such as inverted commas, hyphens, placement of spaces, or single letter consonant prepositions at the ends of lines, etc.
  Example: *4 MB* instead of *4MB*,
  tag: `errtype=prav-mez`,
- *Morphological* and *syntactic* errors consist in using wrong endings in the inflected words (nouns, adjectives, pronouns, numerals, verbs and adverbs). There is, in fact, overlapping between those two types of errors, because the wrong ending (morphological error) causes an error in grammatical agreement on the syntactic level.
  Example: the incorrect ending in the noun group *dvě**mi** způsoby (in two ways).* Similarly the agreement of subject and verb is violated in the cases like *ženy š**li*** instead of *ženy š**ly** (women went).*
  tag: `errtype=ms-nom`
- Clear *syntactic* errors consist in using incorrect verb valencies. Czech verbs in their valency frames strictly require concrete cases as complements, e. g. verb *zabít (to kill)* requires subject in nominative, object in accusative and if the instrument of killing is mentioned it has to be expressed by instrumental case.
  Example: in *Cizinec zabil chlapci nože. (The stranger killed boy knives)* the cases are used incorrectly. Only *Cizinec zabil chlapce nožem (The stranger killed the boy with knife)* is the correct use of the valency frame for *zabít (to kill).*
  `tag: errtype=ms-val`
- *punctuation* errors follow from missing or incorrect placement of commas or other delimiters (!, ?, ;) in the sentences. In Czech, punctuation rules reflect the syntactic structure of the sentence, commas typically separate the main and subordinate clauses and are obligatory, especially with some conjunctions. The frequency of the punctuation errors in the Chyby is consequently quite high.
  Example: *Student ví že musí složit zkoušku. (The student knows that he has to pass the exam.)* The missing comma in front of *že* has to be inserted *Student ví**,** že musí složit zkoušku.*
  `tag: errtype=intp-pvety`
- *semantic (lexical)* errors include cases where expressions are incorrectly used, causing violation of semantic meaning fullness.
  Example: *rektor fakulty (Rector of the Faculty* – the correct expression is *děkan fakulty (Dean of the Faculty)*
  `tag: errtype=sem-slovo`
- *stylistic* errors represent a collection of the various violations such as inappropriate use of colloquial slang or jargon expressions, archaic or too informal words, repetitions of some expressions within a relatively short context (up

to five sentences). As stylistic errors we also classify the repetitions of some words (*také (also)*) in short contexts, superfluous use of demonstrative pronouns (determiners), abundant use of passive constructions, long chains of noun groups, especially the prepositional ones, and ambiguous uses of anaphoric pronouns, i.e. errors in co-reference. We have developed detailed subclassification of stylistic errors but here we show only two groups related to the substandard uses of some expressions.
Example 1: incorrect slang expression *spakovaný soubor* instead of *komprimovaný soubor (compressed file)*
`tag: errtype=styl-subst,`
Example 2: archaic form of the infinitive *nalézti* as opposed to the standard form *najít (to find)*
`tag: errtype=styl-nadst`

The final format is an XML application. The `<corr>` elements are used for error annotation.

6 Tools for Tagging Errors in the Texts

The tagging of errors is a tedious task which we have tried to make as simple as possible. Each student is responsible for his/her own document and his/her final course grade is partly based on the quality of the tagging of previous errors in the essay. It corresponds to the level of comprehension of each particular grammatical phenomenon.

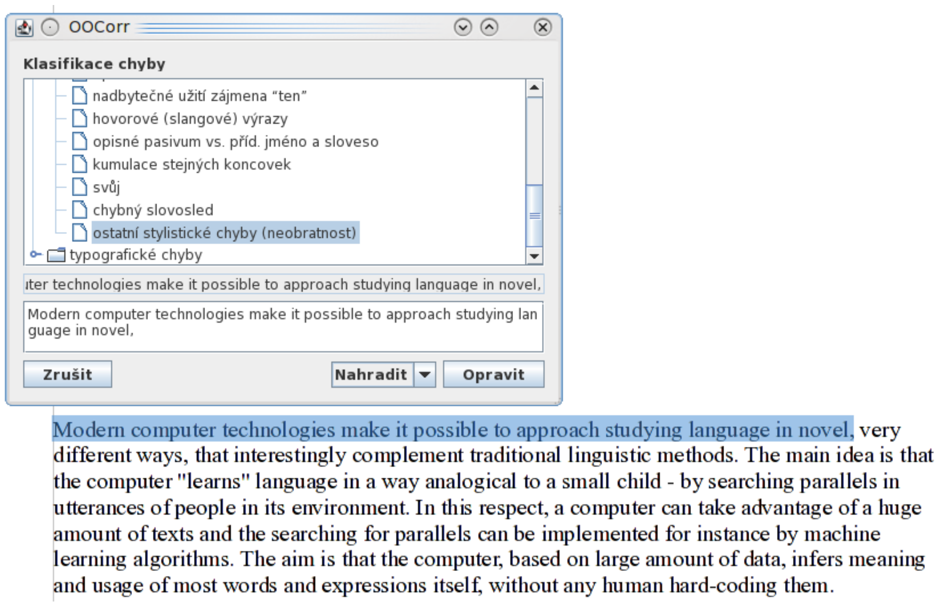
Our first tool developed for this purpose (WinCorr) was implemented as a standalone text editor for the RTF document format. It has the advantage of being fully in control of users behaviour. On the other hand, it has a relatively poor functionality, it is not multiplatform and restricts the users in their choice of a text editor. Besides, there was also a set of Microsoft Word macros implementing similar functionality, which, however, had similar disadvantages and maintenance problems.

The new OOCorr application implements the functionality of a simple corrector as an extension in the environment of the OpenOffice.org Writer text editor system. This enables users to employ an arbitrary text editor for writing their texts provided that it is able to store the document in one of wide range of document formats supported by OpenOffice.org Writer (e. g. DOC, ODT, RTF, HTML etc.). Moreover, it benefits from the multiplatformity and rich functionality of the OpenOffice.org system which is being continuously developed and enhanced.

The function of this program is demonstrated by the screenshots in Figures 1, 2 and 3, where the error marking process is shown.

7 New Annotation Scheme/Error Classification

Since the time the original corpus Chyby has been built, the error classification was stable even if there were some problematic places. Decisions on how to

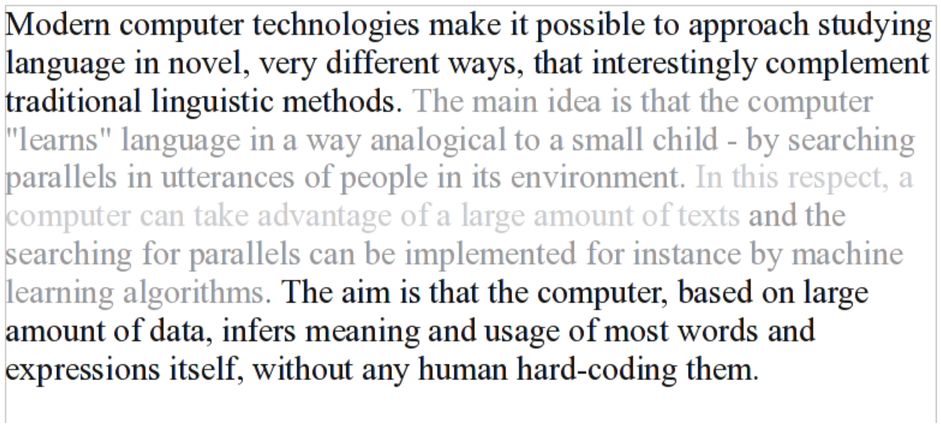

OOCorr
Klasifikace chyby
nadbytečné užití zájmena "ten"
hovorové (slangové) výrazy
opisné pasivum vs. příd. jméno a sloveso
kumulace stejných koncovek
svůj
chybný slovosled
ostatní stylistické chyby (neobratnost)
typografické chyby
Zrušit
Nahradit
Opravit

ıter technologies make it possible to approach studying language in novel,

Modern computer technologies make it possible to approach studying lan guage in novel,

Modern computer technologies make it possible to approach studying language in novel, very different ways, that interestingly complement traditional linguistic methods. The main idea is that the computer "learns" language in a way analogical to a small child - by searching parallels in utterances of people in its environment. In this respect, a computer can take advantage of a huge amount of texts and the searching for parallels can be implemented for instance by machine learning algorithms. The aim is that the computer, based on large amount of data, infers meaning and usage of most words and expressions itself, without any human hard-coding them.

Fig. 1. Marking the text and choosing the error classification.



Modern computer technologies make it possible to approach studying language in novel, very different ways, that interestingly complement traditional linguistic methods. The main idea is that the computer "learns" language in a way analogical to a small child - by searching parallels in utterances of people in its environment. In this respect, a computer can take advantage of a large amount of texts and the searching for parallels can be implemented for instance by machine learning algorithms. The aim is that the computer, based on large amount of data, infers meaning and usage of most words and expressions itself, without any human hard-coding them.

Fig. 2. OOCorr allows you to mark included errors; they are colored for better recognition.

classify a specific error have not been self-evident in some cases. There were two (or more) possibilities for correct classification.

Our wish is to simplify the classification, so that all the errors can be placed into one case only. That leads to building a new classification for error annotation.

– *Spelling (simple)*, error recognizable by a checker (`errtype=preklep`);

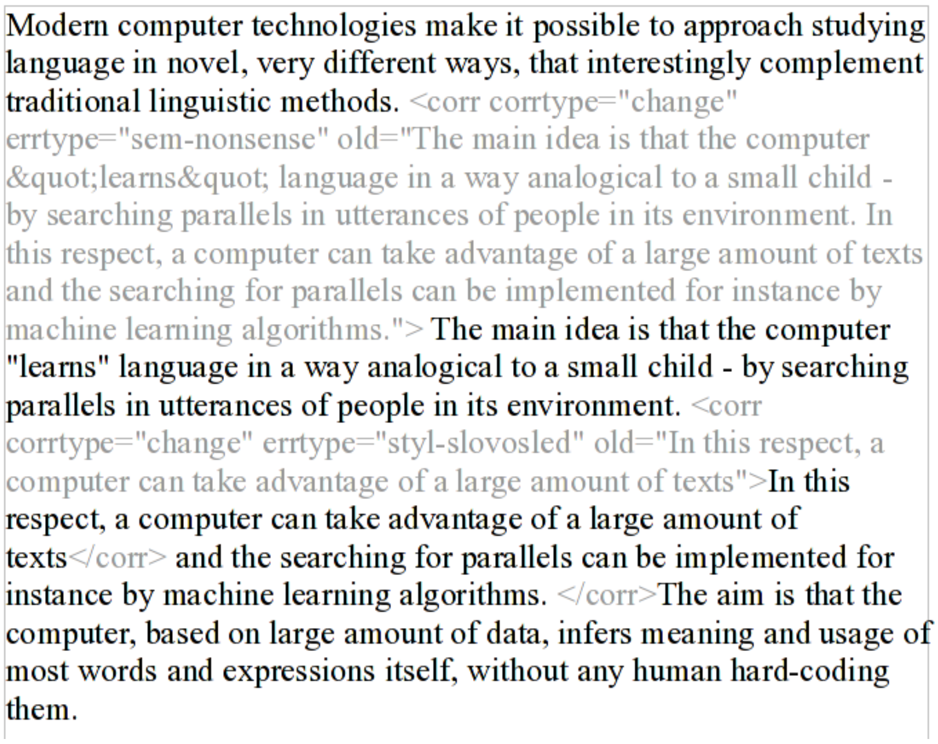

Modern computer technologies make it possible to approach studying language in novel, very different ways, that interestingly complement traditional linguistic methods. <corr corrtype="change" errtype="sem-nonsense" old="The main idea is that the computer &quot;learns&quot; language in a way analogical to a small child - by searching parallels in utterances of people in its environment. In this respect, a computer can take advantage of a large amount of texts and the searching for parallels can be implemented for instance by machine learning algorithms."> The main idea is that the computer "learns" language in a way analogical to a small child - by searching parallels in utterances of people in its environment. <corr corrtype="change" errtype="styl-slovosled" old="In this respect, a computer can take advantage of a large amount of texts">In this respect, a computer can take advantage of a large amount of texts</corr> and the searching for parallels can be implemented for instance by machine learning algorithms. </corr>The aim is that the computer, based on large amount of data, infers meaning and usage of most words and expressions itself, without any human hard-coding them.

Fig. 3. The principle of the OOCorr is a feature of OpenOffice.org Writer, which is suited to handle "hidden text style". In this style the information about the error, its type and way of correction is saved.

– *Orthography*: error can not be recognized by a checker: punctuation error (`errtype=prav-interp`, *i/y* in specified words (`errtype=prav-iy`), *s/z* in specified words (`errtype=prav-sz`), using the right form of pronouns (`errtype=prav-mneme`), capital letters and lowercase (`errtype=prav-mala-velka`), composites (`errtype=prav-sprezky`), foreign words (`errtype=prav--prejata`), other orthographic errors (`errtype=prav-jine`);

– *Typography*: hyphen and dash (`errtype=typo-spojovnik`), hyphenation (`errtype=typo-delenislov`), division of text into paragraphs (`errtype=typo--odstavce`), spaces(`errtype=typo-mezery`), prepositions at the end of rows (`errtype=typo-predlozky`), brackets, quotes, overall layout and graphical outlook (`errtype=typo-jine`);

– *Morpho-syntactic errors*: morphologically wrong form (`errtype=synt-morf`), error in agreement (`errtype=synt-shoda`), verb and noun valencies (`errtype=synt-vazba`), possessives related errors (`errtype=synt-zamena`), other (`errtype=synt-jine`);

– *Lexico-semantic errors*: non meaningful expression (`errtype=sem-vyraz`), nonsensical or untrue steatment (`errtype=sem-nonsense`), and other semantics(`errtype=sem-jine`);
– *Stylistic*: word repeating (`errtype=styl-opakovani`), redundant use of demonstratives (`errtype=styl-ten`), using slang expressions (`errtype=styl--hovor`), cumulation of the same noun endings (`errtype=styl-koncovky`), incorrect stylistic word order (`errtype=styl-slovosled`), and other stylistic mistakes (`errtype=sytl-jine`).

8 The Differences

The new system is not error annotation specification dependent. There is a possibility to change the classification without changing the program just using another XML error definition file where all needed information is provided. The usage of more than one classification for different purposes is allowed as well as different language-specific settings.

To provide possibility of nesting errors (in up to three levels), the corr tag has been changed. Currently this tag is specified as a pair XML tag which means that, as opposed to the previous version, the *words* attribute which defined the length of the corrected text is not necessary anymore.

```
<corr errtype='string' corrtype='string' old='old text'>
new repaired text
</corr>
```

Fig. 4. New concept of the *corr* tag.

Simplified classification (only six main categories) helps us to build better corpus. It will be faster and easier for students to decide how to annotate their errors. Of course, precise annotation is crucial for getting accurate statistics from the corpus.

9 Results Based on the New Error Classification

At present we are not able to offer a detailed comparison of the Chyby with a standard corpus like DESAM [12] to see what differences exist in the distribution of the errors.

It is not surprising that the most frequent errors in the Chyby are stylistic ones (see Figure 5). The reason for this lies in the fact that the creators of the texts in the Chyby are students who are learning how to write. However, it is also true that the principles of good writing belong to the most neglected issues in the Czech high schools.

Error Group	count	%
Spelling (simple)	2,347	13.04
Morpho-syntactic	1,689	9.39
Spelling (other)	867	4.82
Lexico-semantics	2,536	14.09
Punctuation	3,837	21.32
Stylistic	4,184	23.25
Typography	2,165	12.03
unsorted	371	2.06
Total	**17,996**	100.00

Fig. 5. Error classification group statistics in the Chyby corpus.

The formal nature of stylistic errors is not very thoroughly explored even though they can be reliably identified in the texts. However, attempts to build a formal recognition procedure for them have been successful only partially.

The second most frequent error type is punctuation. Its high frequency is caused by the relative complexity of the Czech punctuation orthography rules and by the fact that the students do not possess the necessary writing skills at this level. The lexical and semantic errors also display a high frequency (3rd in order) for the same reasons. Recognition procedures for them, however, do not exist so far and they can be processed only manually.

10 Conclusions

In this paper we describe a Czech text corpus (Chyby) containing various kinds of errors – spelling, typographical, grammatical, style, lexical, etc. Resources for the Chyby come from the student's texts, reviews and essays written for the subject *Elements of Style*. They are corrected by the teachers and returned to the students who tag the marked errors and insert the respective corrections electronically into their texts. In this way the annotated corpus has been created.

The classification of the errors as they occur in the Chyby and the annotation scheme is presented together with the description of the tools used for inserting the tagged errors into the texts. The new tool developed for this purpose is OOCorr [3].

The present size of the corpus Chyby is approx. 500,000 word forms. It can be seen that the most frequent errors are stylistic ones – 23.25 %, followed by punctuation errors – 21.32 %, and lexical errors – 14.09 %.

The building of the Chyby and the analysis of the errors in the texts is a part of the larger project in the NLP Laboratory at FI MU whose goal is:

– to explore all types of errors that occur in the spontaneous texts,
– depending on the frequency and nature of the errors, to analyse whether effective procedures for an automatic correction can be designed,

– experiments not reported here (to be published in another paper) have already been performed so as to formulate an algorithm for automatic correcting punctuation errors using full parsing,
– to better map the area of stylistic errors and estimate what error detection rules can be developed in this respect for Czech texts.

Acknowledgement This research has been supported by Czech Ministry of Education, under the Grant LC536 and NPV 2C06009.

References

1. Pala, K., Rychlý, P., Smrž, P.: Text corpus with errors. In: Text, Speech and Dialogue: Sixth International Conference, TSD 2003, Berlin, Springer Verlag (2003) 90–97.
2. Kukačka, M.: Correcting errors in WinCorr. (Student Project at the Laboratory of Natural Language Processing, Faculty of Informatics, Masaryk University, Brno, Czech Republic) (2000).
3. Moravec Jaroslav: Korekturní rozšíření pro OpenOffice.org (bakalářská práce). Faculty of Informatics, Masaryk University, Brno (2009).
4. Burnard, L., ed.: Users Reference Guide for the British National Corpus. Oxford University Computing Service (1995).
5. Kocek, J., Kopřivová, M., Kučera, K., eds.: Český národní korpus – úvod a příručka uživatele (Czech National Corpus – Introduction and Users Guide). FF UK – ÚČNK (2000).
6. Rychlý, P., Smrž, P.: Manatee, Bonito and word sketches for Czech. In: Proceedings of the Second International Conference on Corpus Linguisitcs, Saint-Petersburg, Saint-Petersburg State University Press (2004) 124–132.
7. Hlavsa, Z., et al.: Akademická pravidla českého pravopisu (Rules of Czech Orthography). Akademia, Praha (1993).
8. ÚJČ AV ČR, FI MUNI: Internetová jazyková příručka. [online] `http://prirucka.ujc.cas.cz/` (2008).
9. Carlberger, J., Domeij, R., Kann, V., Kuntsson, O.: A Swedish grammar checker. `http://citeseer.nj.nec.com/305098.html` (2000).
10. Wei, Y.H., Davies, G.: Do grammar checkers work? `http://www.camsoftpartners.co.uk/euro96b.htm` (2002).
11. Negrilo, A.D., Fernandéz-Domínguez, J.: Error tagging systems for learner corpora. RESLA – Spanish Journal of Applied Linguistics **16** (2006) 83–102.
12. Pala, K., Rychlý, P., Smrž, P.: DESAM – Annotated Corpus for Czech. In: Proceedings of SOFSEM '97, Springer-Verlag (1997) 523–530.

Problems of Machine Translation Evaluation

Vít Baisa

NLP Laboratory,
Faculty of Informatics, Masaryk University,
Brno, Czech Republic
xbaisa@fi.muni.cz

Abstract. In this article we deal with general aspects of machine translation evaluation. We describe several commonly used methods of the evaluation and discuss their problems and shortcomings. Then we outline a few thoughts and ideas which try to solve mentioned problems and stand behind a design of a new method of machine translation evaluation.

Key words: machine translation; evaluation

1 Introduction

The main goal of an evaluation of machine translation (MT) is to compare different MT systems. Since we have a lot of them at our disposal (Google Translate, Yahoo Babel fish, SYSTRAN Translator, PC Translator,. . .) we want to be able to say which one is the best. The second, and often neglected, goal is to measure quality of translation of an arbitrary sentence.

What does a good, appropriate translation mean? There isn't the only definition but it is quite natural to expect that translation t of a sentence s should preserve the meaning of s and that t is understandable enough. The former requirement is called *accuracy* (sometimes *adequacy*) of translation whereas the latter is called *intelligibility* (sometimes *fluency*).

It is tricky to make a metric for these two features. In a human evaluation, seven or five degree scales are typically used (from the worst to the best translations), several aspects of translation are rated and, in spite of being influenced by subjectivity of human evaluators, this method is considered as the best approach. Unfortunately, it costs a lot of money and it takes a lot of time to employ people and let them manually evaluate thousands of sentences. The aim of automatic evaluation methods is to eliminate these shortcomings of the human evaluation, however, at the expense of losing exactness in the sense of correlation with the human evaluation. In other words, we look for a method which will approximate the human evaluation best.

2 Paradox of an evaluation

This searching is made difficult by a significant factor. Let us call this factor *paradox of an evaluation*. It has two points of view.

The first point could be expressed in this way: an evaluator of any translation must have better knowledge about both source and resultant language than its translator has. Otherwise he (the evaluator) would never be able to detect a single mistake of the translator.

The second point results from the first point: it is impossible to make an automatic universal evaluation method which would be better than MT itself. If we had such method at our disposal it would be quite easy to use for instance a genetic algorithm. A process of translation would start with random strings (random sentences) and in each generation it would evaluate newly evolved sentences with the evaluation method and choose adepts for the next generation. At the end we would obtain a sentence which would be as good as the evaluation method. But the algorithm could then serve as a new MT which is in contradiction with presumption.

Thus, if we want to make a versatile evaluation method, we are always limited by the paradox. The only way how to avoid the paradox is to make a non-versatile evaluation method as authors of following methods do.

3 Evaluating methods – BLEU and the others

In this section we will describe several commonly used MT evaluation methods. All these methods use referential translations of source sentences from a test set. These translations are prepared manually and, in most cases, there are several different translations of a single source sentence from test set made by several different translators.

A little more formally: we have a source sentence s, a set of its referential translations $R = \{r_i\}$ and a candidate translation c translated by a MT system from s. Evaluation methods use the candidate translation c and all referential translations r_i.

3.1 BLEU, [1] and NEVA, [2]

BLEU is the oldest evaluation method and that is probably why it is considered to be a standard. BLEU tries to find out what sentences c and r_i have in common employing n-grams. Very simply said, BLEU counts matched uni-, bi-, tri- and quadrigrams between c and r_i.

Since it is supposed that a good candidate translation should be approximately as long as a referential translation, BLEU introduces penalty for brevity. The shorter c is the worse resulting score it obtains. On the contrary, c which is longer than referential translations is implicitly penalized by n-gram matching itself.

Because BLEU fails on evaluation of short sentences, authors of the next method NEVA slightly altered BLEU's formula to achieve robustness even on short sentences. In the other aspects NEVA is very similar to BLEU.

3.2 WAFT, [2] and TER, [3]

The next two methods use *edit distance* between c and R instead of n-grams.

The method WAFT defines edit distance between c and R as minimum number of deletions, substitutions and insertions of words needed to turn c into one of $r_i \in R$ (the closest one, in the sense of edit distance, is taken and evaluated). Once edit distance is computed and normalized it serves as the evaluation of c.

Another method TER uses almost the same formula as WAFT but it works with (continuous) sequences of words. Thus we can shift (but not delete and insert) two neighbouring words in one edit step whereas in WAFT we need two edit steps to do it.

3.3 METEOR, [4]

The last method stands a bit apart from the others since it uses (as the only one) synonyms in process of an evaluation. METEOR tries to map words (unigrams) from c onto words from r_i (for all i). A word w_c can be mapped onto a word w_{r_i} if $w_c = w_{r_i}$ or w_c is synonym of w_{r_i}. METEOR exploits WordNet to be able to work with synonyms.

The whole process of the evaluation is more complex but isn't so important for us. Important thing is that, thanks to synonyms, authors of METEOR achieved higher correlation with human evaluation requiring less referential translations then the others methods.

For more details on described methods and for examples see References.

4 Problems and shortcomings of described methods

4.1 Problems concerning n-gram matching

The main idea behind usage of n-grams in machine translation evaluation is that a good candidate translation is supposed to be similar to a referential (manually prepared and thus sufficiently proper) translation. It obviously holds but what about good candidate translations differing from all referential translations? The idea strongly depends on amount of referential translations. It is evident that the more referential translations we have the higher score in the evaluation we obtain.

Unigram matching corresponds with accuracy: if we find a word in c and the same word in r_i it is probably well translated word. N-gram matching corresponds with intelligibility: human translation r_i has always high intelligibility and the longer part of r_i we match in c the higher intelligibility of c can be expected.

It has also been shown in [5] that a high score (as a result of a method which uses n-grams) probably indicates a good translation but a low score is not necessarily an indication of a poor translation.

4.2 Problems concerning edit distance

Methods using edit distance don't take relevancy of an edit step (mistake) into account at all. But it is obvious that there are a lot of possible types of mistakes differing in relevancy. Especially for morphologically rich languages, a small alteration on character level (at the end of a word as for Czech language) has smaller impact on accuracy of translation than a change of a whole word despite both are considered as one undistinguishable edit step.

Simple example proves it. The sentence *Petr mít velký červený kniha*, consisting only of Czech lemmas, has relatively high accuracy and even quite hight intelligibility (depending on a source sentence, of course, in this case s = *Peter has a big red book.*). But it would require four substitutions to change it into one of possible referential translations e.g. *Petr má velkou červenou knihu*.

Another problem concerning edit distance is complexity of computing edit distance in general.

4.3 A source sentence matters

None of presented methods takes a source sentence s into account. Since the manually prepared referential translations are supposed to be semantically equivalent (or very close) to their counterpart in a source language and also well formed, we can omit s. But not in general: it is not such a mistake to translate s wrong if s is not well structured, indeed.

One could admit that, simply, there aren't such cases of badly formed source sentences but let us consider a machine translation between *minor* languages M_1 and M_2 which requires usage (especially in statistical machine translation) of a transfer language T (typically English language). A MT system translates a sentence s_1 in language M_1 to a sentence s_t in language T but it can (and it does) make mistakes. When it translates s_t to s_2 in M_2, mistakes accumulate and the total translation is worse than the two partial translations.

So that, in this case, a quality of s_t should be taken into account for more accurate evaluation of the total translation.

5 Possible treatment

5.1 A language model: s and c

Sometimes, as we have shown, a source sentence s matters. The most straightforward way to check a quality of s is to engage language models. There are many publications about language models so we outline our idea directly.

We check every uni-, bi-, tri-, ... n-grams in s. In ideal case, the whole s would be covered by a language model. In general, the more n-grams of s are covered by the language model the better quality (in the sense of intelligibility) of s should be expected.

Checking of intelligibility of a candidate translation is much more important. The main thought behind this step is that if c is a good translation of s then c should be well formed sentence.

5.2 Semantic matching

The idea of semantic matching between s and c is similar to METEOR's mapping of words between c and r_i. The distinction is that s and c differ in languages. Thus we must exploit bilingual dictionaries. WordNet can also be used thanks to its ILI (interlingua index).

This approach brings other problems into process. Let us consider this example: $s =$ *He has a new key* and $c =$ *Má nový klíč*. It is hard task to determine a proper counterpart of a word w_s from s to a word w_c from c: both w_s and w_c can have several different meanings and we must choose the proper pair: *key* vs. *klíč* (a key for locking), *klávesa* (a key on a keyboard), *tónina* (pitch of a voice). Moreover w_s can have none counterpart: *a* (an article) and several words from s can have a single counterpart in c: *He has* and *má*.

5.3 Putting it together

Question is: what is more important – intelligibility or accuracy of a translation? It isn't easy to answer it but the goal of putting accuracy (semantic matching) and intelligibility (language model checking) together is to balance both aspects and, at the same time, dealing with intelligibility of s. Obviously the better accuracy of the translation and intelligibility of c are the better quality of the translation should be expected and, on the contrary, the worse intelligibility of s is the worse quality of translation should be expected.

6 Future work

We plan to implement all of mentioned features into a new method of MT evaluation with working name LAMENT (LAnguage model and Meaning based Evaluation of machiNe Translation). The method should prove or falsify a hidden hypothesis: if it is possible to divide MT evaluation into two parts – to separate checking of intelligibility of a candidate translation (with help of language models) from matching words between a source sentence and a candidate translation. And, at the same time, provide a sufficient correlation with human evaluation not requiring any referential translations.

7 Conclusion

We have concisely described several commonly used methods of MT evaluation and commented their problems and shortcomings. Since these methods use manually prepared referential translations they could be regarded as semiautomatic methods. They simplified a process of developing new MT systems remarkably: if we include a new rule into our MT system we can instantly check out its impact on performance of the system. Despite these benefits they aren't suitable for an evaluation of arbitrary translations since they aren't versatile.

It may seem, after having mentioned the paradox of an evaluation, that developing of an universal MT evaluation methods is waste of time. But MT systems and MT evaluation methods are strongly interconnected therefore thinking of these methods helps us also with understanding of machine translation and with understanding of natural language in general. That is why it is worth dealing with them.

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.

References

1. Papineni, K., Roukos, S., Ward, T., and Zhu, W.: *A method for automatic evaluation of machine translation.* In: Proceedings of the 40th Annual Meeting on Association For Computational Linguistics. 2002.
2. Forbsom, E.: *Training a Super Model Look-Alike: Featuring Edit Distance, N-Gram Occurrence, and One Reference Translation.* In: Proceedings of the Workshop on Machine Translation Evaluation: Towards Systemizing MT Evaluation. 2003.
3. Snover M., Dorr B., Schwartz R., Micciulla L., Makhoul J.: *A Study of Translation Edit Rate with Targeted Human Annotation*. In: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas. Cambridge. 2006.
4. Satanjeev B.: *METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments.* In: Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization. 2005.
5. Culy C., Riehemann S. Z.: *The limits of N-gram translation evaluation metrics*. Machine Translation Summit IX. 2003.

Languages of Mathematics
Random Walking in the Mathematics of Languages

Petr Sojka

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

Abstract. An essay about mathematics being a sublanguage of other natural languages: how it may be represented, stored, searched and handled in several projects of (European) Digital Mathematics Libraries as DML-CZ or EuDML.
A framework for solving problem of computing of similar papers in a digital library is proposed, allowing several types of similarity type definitions: *plagiarity* counting on common word *n*-grams, *topicality* counting on common topics, or *conarrativity* counting on the same narrative. The vector of the most similar documents for a given similarity type is suggested to be computed using the algorithm by Page for web page ranking, often explained as 'random walking'.

Science is based on trust and integrity. – Venkatraman Ramakrishnan
Nobel laureate 2009

1 Introduction

The language of mathematics can be viewed as a sublanguage of other natural languages. The recent initiatives Towards a Digital Mathematics Library [1,2,3] aim at virtual multilingual digital library with the papers published as peer-reviewed verified archive knowledge in the area of mathematics. The area is well-defined by review databases Mathematical Reviews and Zentralblatt with almost 3,000,000 (metadata and reviews) items of mathematical scientific literature. The integration even on the level of full texts has started, and brings questions like:

– how to represent mathematical language, formulae?
– how to index it, search it?
– how to deal with semantics of mathematics?
– how to classify mathematics, which ontologies to use?
– how to deal with mix of 'informal' texts and formal proofs and specifications?

For big software firms like Google (Google Scholar) and ABBYY (FineReader) mathematics is very small niche with very big problems to face. There are, fortunately, several smaller digital mathematics library initiatives like NUMDAM[1] or DML-CZ[2] where new best practices are created and tested, in addition to the development of tools for solving at least some of the problems of handling mathematics. For the final solution of problems like mathematical OCR or semantic representation and searching mathematics handling there are still funds missing, though.

In this paper we shortly sum up current level of understanding of these issues based on the experience of five years of working on the DML-CZ project. We discuss math representation issues in Section 2. We follow with topic of math search and digital libraires in Sections 3 and 4. Finally, in Section 5, we define several kinds of 'similarity' usable not only for mathematical papers and suggest novel framework to compute these general versions of 'similarity' using iterative algorithm used sofar for ranking web pages [4].

Where possible, the systems will share a common application or database, or perhaps a more common data structure that will allow one system to import / export data with another system without sharing their applications or platforms. – Report by the 511 Interoperability Task Force, April 4, 2005

2 Domain of Mathematics

For communication of mathematics in a digital library, several formats are used: the relations and laws are either expressed verbally in plain language, or formulas and formalisms are used.

On the authoring side, the most widespread and preferred format is the plain TEX's notation or it's markup extensions defined in AMSLATEX. TEX or its successors as pdf(e)TEX are said to be used for the production of more than 90 % of the world's scientific printed journals.

For communication between bots, programs and applications, MathML standard by W3C is supported for mathematics exchange. One can cut and paste formula from Mathematica and paste it in Maple to derive it, and import the result into web page rendered by Firefox.

We can classify the levels mathematics is handled now:

1.0 lexical – words, *strings* of characters or TEX's \$ \$ notation.
2.0 syntactical – phrases, *parsed* formulas (represented as trees in MathML).
3.0 semantical – *meaning* of parsed phrases (cloud tags/ontologies/OpenMath).

The problem is that the author's message (it's incarnation in the paper's *content* and *form*) does not survive (no standard representation of math) when communicated (via the paper or over the web) to the readers.

Although semantical representations of mathematical formulas in MathML version 3 [3] or in OpenMath's Content Dictionaries [4] are well defined, they are not

[1] `http://numdam.org` [2] `http://dml.cz` [3] `http://www.w3.org/TR/MathML3/`
[4] `http://openmath.org`

used by authors, probably because there is no strong incentive and benefits for authors. On the opposite, semantical markup gives additional burden to authors to disambiguate their thoughts, when they hurry for publication (Publish or Perish). The cost of semantic-rich markup is not usually willing to be absorbed by publishers – they claim that the price tag is too high. There are estimates that the growth of production costs from standard paper/PDF-only LaTeX to PDF production to LaTeX to validated XML+MathML to LaTeX to PDF is tenfold (from \$6 to \$60 per page, even if it is outsourced to India or other cheap labour country). Others oppose that it is a must anyway and that by developing authoring tools that take as much of logical markup from author as possible into the source file publisher may leverage the costs to minimum. In the case publisher would not have rich semantically marked XML+MathML+SVG files to build it's services on, it would not be able to compete on the publishing market. Current ability iof some publishers to generate Epub or DAISY formats from their rich XML based representation shows that it actually pays back very quickly. New architectures and services start to appear, based on the rich XML+MathML markup [5].

Quite different requirements have theorem proving systems and computer algebra systems. They use usually their own internal representation of mathematics, with MathML (or LaTeX) as the interface languages.

As simple as possible, but not simpler. – Albert Einstein

3 Search

Neither format mentioned in the previous section is widely accepted and used, though. When one tries to search for citations of Kováčik and Rákosník's paper [6] by Google Scholar,[5] one finds more than a dozen of different citation clusters of it, depending on the OCR errors in this paper author's names and in the 'representation' of math formulas in the paper's title. It may be seen as a clear evidence of current mess of different ways of mathematics representation and treatment. There are attempts to sort out this mess, ambitions of e.g. *Math WebSearch*[6] are much higher.

The widely used *Google Search* only pays attention to the ranking when delivering (math) search results – there is no sign of math representation or disambiguation. *SearchPoint*[7], on the other hand allows walking in the meaning spaces: in the clusters of related pages with different meanings of terms in the question posed.

Mathematical search has both many specifics [7] and many common problems of information retrieval:

- Mathematical notation is context-dependent, e.g. binomial coefficients has different form in different languages and language contexts: $\binom{n}{k}$, ${}_nC^k$, C_k^n, C_n^k all denote the same semantically equivalent notion.
- Identical presentations can stand for multiple distinct mathematical objects, e.g. $\int f(x)\,dx$ for several anti-derivative operators (Riemann, Lebesgue,...).

[5] http://scholar.google.cz/scholar?q=Kovacik+Rakosnik

[6] http://search.mathweb.org/index.xhtml [7] http://searchpoint.ijs.si/

– Certain variations of notations are widely considered irrelevant, e.g. $\int f(x)\,dx$ and $\int f(y)\,dy$.

There are several math search systems and platforms available:

– *MathWebSearch*[8], by I. Şucan, M. Kohlhase (Bremen, GE);
– *MathDex*, by R. Miner et al. (Design Science, US) or *DLMF search*, A. Youssef (Washington, US);
– *EgoMath/Egothor*, J. Mišutka, L. Galamboš (Prague, CZ).

Other notable related work is:

– Mathematical formulae recognition from PDF, J. Baker, A. Sexton, V. Sorge, Birmingham, UK.
– Infty system, M. Suzuki, Kyushu, JP.
– ActiveMath web-based math-learning environment, P. Libbrecht, DKFI, Saarbrücken, GE.
– SWiM: A Semantic Wiki for Mathematical Knowledge Management, KWARC, Bremen, GE.

Math search system has to solve many technical aspects of search. In EgoMath system, these are e.g.

– normalization;
– linearization (search engine may work on strings/words);
– partial evaluation (e.g. distributivity);
– generalization (introduction of variables in the index) or
– ordering (for commutative operators).

Complexity of these issues are probably causing that there is not a widely used web search engine handling math yet.

Automating the creation of useful digital libraries – that is, digital libraries affording searchable text and reusable output – is a complicated process, whether the original library is paper-based or already available in electronic form. – Simske and Lin [8]

4 Math Digital Libraries

There is the vision of the world-wide digital mathematics library [9].

We may classify levels of digital libraries of mathematics:

1.0 classical library + scanned bitmaps.
2.0 interconnected, crosslinked and validated repository of peer reviewed documents, possibly fully (not only metadata) indexed on the syntactic level.

[8] `http://www.mathweb.org/wiki/MathWebSearch`

3.0 dynamically personalized, formalized knowledge in rich semantic representation with logical inference and deduction.

Most DMLs today strive to attach rich metadata to the scanned page bitmaps (level one). The ideal 3.0 world remains as a vision for the next decades. There are attempts towards level 2.0 (DML-CZ, NUMDAM, Euclid[9]). Reference lists are considered as paper metadata and made available and linkable by current leading systems (as CrossRef[10]).

More and more applications can be build using the [richly tagged] paper full texts. One that is admired by users of digital library is application that provides links to similar papers ('see also' types of suggestions).

When the music changes, so does the dance. – African proverb

5 Math Paper Similarities

Showing similar papers functionality starts to be offered by several digital libraries and publisher. But how to find similar papers among other milions? How to evaluate the possible candidate lists? Which type of similarity is preferred?

We have tried to think about these kind of questions and did some experiments with the data of DML-CZ and NUMDAM. We have used bag of words vector models for paper representation, and computed similarities by three methods: *TFIDF* term weighting, *Latent Semantic Analysis* (LSI) and by *Random projections* [10]. They are available for author's evaluation on the DML-CZ web pages. We were stuck with evaluation, as almost no author was willing to go through computed lists of similar papers and to compare the results given by different methods. Top ordering comparisons done by experts was evaluated as too costly and unfeasible within the budget and time constraints. The only information available we can base the evaluation on are available metadata as MSC numbers, and article full texts. But another solution came to our mind: *random walking*.

Let us remind method of Larry Page to compute ranking of web pages [4]. Let $G = \langle N, L\rangle$ be a graph of interlinked documents and let $W_0[i,j] = 1$ iff there is link from node n_i to n_j. Let we define forward neighbours of a document as $F(i) = \{n_j | W_0[i,j] = 1\}$. Let we now row-normalize adjacency matrix of G: $W[i,j] = \frac{1}{|F(i)|}$ if $W_0[i,j] = 1$ and $W[i,j] = 0$ otherwise.

Page's algorithm takes row-normalized adjacency matrix $\boldsymbol{W}$ and vector $\boldsymbol{e}$ (internal source of score of n_i, constant across iterations) and iteratively computes

$$\boldsymbol{a}^{(k)} = \alpha\boldsymbol{a}^{(k-1)}\boldsymbol{W} + (1-\alpha)\boldsymbol{e}\,.$$

Resulting vector is $\boldsymbol{a} = \langle a_1, a_2, \ldots, a_{|N|}\rangle$, where a_i represents the 'score' (pagerank) of node n_i. For more information we refer to the original paper or to the recent application of it in the area of Natural Language Engineering [11].

[9] `http://projecteuclid.org` [10] `http://crossref.org`

Let now take one document of interest n_k, for which we want to compute the most similar ones. We think of forward neighbours set $F(i)$ as a *support of similarity* to the document n_k of interest, based on the 'local knowledge' of document n_i.

Vector $\boldsymbol{e}$ can be used for smoothing (all values set to $\frac{1}{|N|}$), or as a source of explicit knowledge. It may be plausible to set non zero values only to all documents sharing same Mathematical Subject Classification (MSC)[11] codes as the document of interest. After the (convergence) computation, vector $\boldsymbol{a}$ contains similarity-ranking of document of interest (and DL may expose links to the ten documents having highest similarity scores a_i).

This framework allows solution of different tasks: different F and $\boldsymbol{e}$ can be used to compute different kinds of similarity – *simtypes*. We think of

topicality: this simtype should find thematically closest papers. F may be based on some vector space document model (LSA), $\boldsymbol{e}$ may reflect common MSC.

plagiarity: F should be based on the number and length of common word or word synsets n-grams.

narrativity: narrative qualities are often neglected when computing document similarities. New ways of representing narrative qualities as Markov chain start to appear as in the recent paper by Hoencamp et al. [12]. F should be sent for documents with similar or same Markov chain.

or their weighted combinations.

Computation will be time-consuming though: convergence for every task (simtype) and *every document (node)* has to be computed. It is yet to be shown how it will work in practice and whether these 'vis maior' simtypes will be praised by [Eu]DML users.

6 Conclusion

In this paper, we have identified some specifics of mathematical documents and suggested solution to the similarities problem – how to find documents close to the given one using different definitions of similarity metric.

Acknowledgments. This work has been partially supported by the Academy of Sciences of Czech Republic under the project 1ET200190513 and by the Ministry of Education of CR within the Centre of basic research LC536 and National Research Programme 2C06009.

References

1. Sojka, P., ed.: Towards a Digital Mathematics Library. In Sojka, P., ed.: Proceedings of DML 2008, Birmingham, UK, Masaryk University (2008) `http://www.fi.muni.cz/~sojka/dml-2008-program.xhtml`.

[11] `http://www.ams.org/msc`

2. Sojka, P., ed.: Towards a Digital Mathematics Library. In Sojka, P., ed.: Proceedings of DML 2009, Grand Bend, Ontario, CA, Masaryk University (2009) `http://www.fi.muni.cz/~sojka/dml-2009-program.html`.
3. Sojka, P.: Digitization Workflow in the Czech Digital Mathematics Library. Math-for-Industry Lecture Note Series **22** (2009) 272–280.
4. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: Seventh International World-Wide Web Conference (WWW 1998), Brisbane, Australia (1998).
5. Grigore, M., Wolska, M., Kohlhase, M.: Towards context-based disambiguation of mathematical expressions. Math-for-Industry Lecture Note Series **22** (2009) 262–271.
6. Kováčik, O., Rákosník, J.: On spaces $L_{p(x)}$ and $W_{k,p(x)}$. Czech Mathematical Journal **41** (1991) 592–618 `http://dml.cz/handle/10338.dmlcz/102493`.
7. Kohlhase, M., Sucan, I.: A Search Engine for Mathematical Formulae. In Calmet, J., Ida, T., Wang, D., eds.: AISC. Volume 4120 of Lecture Notes in Computer Science., Springer (2006) 241–253.
8. Simske, S.J., Lin, X.: Creating Digital Libraries: Content Generation and Re-Mastering. In: Proceedings of First International Workshop on Document Image Analysis for Libraries (DIAL 2004). (2004) pp. 33. `http://doi.ieeecomputersociety.org/10.1109/DIAL.2004.1263235`.
9. Jackson, A.: The Digital Mathematics Library. Notices Am. Math. Soc. **50**(4) (2003) 918–923.
10. Řehůřek, R., Sojka, P.: Automated Classification and Categorization of Mathematical Knowledge. In Autexier, S., Campbell, J., Rubio, J., Sorge, V., Suzuki, M., Wiedijk, F., eds.: Intelligent Computer Mathematics—Proceedings of 7th International Conference on Mathematical Knowledge Management MKM 2008. Volume 5144 of Lecture Notes in Computer Science LNCS/LNAI., Berlin, Heidelberg, Springer-Verlag (2008) 543–557.
11. Esuli, A., Sebastiani, F.: PageRanking WordNet Synsets: An Application to Opinion Mining. In: ACL, The Association for Computer Linguistics (2007) `http://aclweb.org/anthology-new/P/P07/P07-1054.pdf`.
12. Hoenkamp, E., Bruza, P., Song, D., Huang, Q.: An Effective Approach to Verbose Queries Using a Limited Dependencies Language Model. In Azzopardi, L., Kazai, G., Robertson, S.E., Rüger, S.M., Shokouhi, M., Song, D., Yilmaz, E., eds.: ICTIR. Volume 5766 of Lecture Notes in Computer Science., Springer (2009) 116–127 `http://dx.doi.org/10.1007/978-3-642-04417-5_11`.

Author Index

Book orders should be addressed to:

Pavel Mareček c/o FI MU
Botanická 68a
CZ-602 00 Brno
Phone: ++420 549 498 735
Email: `marecek@kup.to`

RASLAN 2009
Third Workshop on Recent Advances in Slavonic Natural Language Processing
Third Workshop on Recent Advances in Slavonic Natural Language Processing, RASLAN 2009
Karlova Studánka, Czech Republic, December 4–6, 2009
Proceedings

P. Sojka, A. Horák (Eds.)

Published by Masaryk University, Brno in 2009

Cover design: Petr Sojka

Typesetting: Petr Sojka

Data conversion: Adam Rambousek

Printing: `http://librix.eu`

First edition, 2009

Serial number INF-3/09-02/58

ISBN 978-80-210-5048-8