From Treebanking to Machine Translation

Zdeněk Žabokrtský

Habilitation Thesis

Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University in Prague
2009

This document has been typeset by the author using LaTeX2e with Geert-Jan M. Kruijff's bookufal class.

Abstract

The presented work is composed of two parts. In the first part we discuss one of the possible approaches to using the annotation scheme of the Prague Dependency Treebank for the task of Machine Translation (MT), and demonstrate it in detail within our highly modular transfer-based MT system called TectoMT.

The second part of the work consists of a sample of our publications, representing our research work from 2000 to 2009. Each article is accompanied with short comments on its context from a present day perspective. The articles are classified into four thematic groups: Annotating Prague Dependency Treebank, Parsing and Transformations of Syntactic Trees, Verb Valency, and Machine Translation.

The two parts naturally connect at numerous points, since most of the topics tackled in the second part—be it sentence analysis or synthesis, coreference resolution, etc.—have their obvious places in the mosaic of the translation process and are now in some way implemented in the TectoMT system described in the first part.

Contents

Part I

Machine Translation via Tectogrammatics

Chapter 1

Introduction

1.1 Motivation

In the following chapters we attempt to show how the annotation scheme of the Prague Dependency Treebank—both in the sense of "tangible" annotated data, software tools and annotation guidelines, and in the abstract sense of structured (layered), dependency-oriented "thinking about language"—can be used for Machine Translation (MT). We demonstrate it in our MT software system called TectoMT.

When we[1] started developing the pilot version of TectoMT in autumn 2005, our motivation for building the system was twofold.

First, we believe that the abstraction power offered by the tectogrammatical layer of language representation (as introduced by Petr Sgall in the 1960's and implemented within the Prague Dependency Treebank project in the last decade) can contribute to the state-of-the-art in Machine Translation. Not only that the system based on "tecto" does not loose its linguistic interpretability in any phase and thus it should allow for simple debugging and monotonic improvements, but compared to the popular n-gram translation models, there are also advantages from the statistical viewpoint. Namely, abstracting from the repertoires of language means (such as inflection, agglutination, word order, functional words, and intonation), which are used to varying extent in different languages for expressing non-lexical meanings, should make the training data contained in available parallel corpora much less sparse (data sparseness is a notorious problem in MT), and thus more machine-learnable.

Second, even if the first assumption could be wrong, we are sure it would be helpful for our team at the institute to be able to integrate existing NLP tools (be they ours or external) into a common software framework. Then we could ultimately get rid of the endless format conversions and frustrating ah-hoc tweaking of other people's source codes whenever one wants to perform any single operation on any single piece of linguistic data.

1.2 Related Work

MT is a broad research field nowadays: every year there are several conferences, workshops and tutorials dedicated to it (or even to its subfields), such as the ACL Workshop

[1] First person singular is avoided throughout this text. First person plural is used to refer to the present author.

on Statistical Machine Translation, the Workshop on Example-Based Machine Translation, or the European Machine Translation Conference. It goes beyond the scope of this work even to mention all the contemporary approaches to MT, but several elaborate surveys of current approaches to MT are already available to the reader elsewhere, e.g. in [Lopez, 2007].

A distinction is usually made between two MT paradigms: rule-based MT and statistical MT (SMT).[2] The rule-based MT systems are dependent on the availability of linguistic knowledge (such as grammar rules and dictionaries), whereas statistical MT systems require human-translated parallel text, from which they extract the translation knowledge automatically. Possible representatives of the first group are systems APAČ ([Kirschner, 1987]), RUSLAN ([Oliva, 1989]), and ETAP-3 ([Boguslavsky et al., 2004]),

Nowadays, probably the most popular representatives of the second group are phrase-based systems (in which the term 'phrase' stands simply for a sequence of words, not necessarily corresponding to phrases in constituent syntax), e.g. [Hoang et al., 2007], derived from the IBM models ([Brown et al., 1993]).

Of course, the two paradigms can be combined and hybrid systems can be created.[3] Linguistically relevant knowledge can be used in SMT systems: for example, factored translation [Koehn and Hoang, 2007] attempts to separate the translation of lemmas from the translation of morphological categories, with the following motivation:

> The current state-of-the-art approach to statistical machine translation, so-called phrase-based models, is limited to the mapping of small text chunks without any explicit use of linguistic information, may it be morphological, syntactic, or semantic. [...] Rich morphology often poses a challenge to statistical machine translation, since a multitude of word forms derived from the same lemma fragment the data and lead to sparse data problems.

SMT with a special type of lemmatization is also used in [Cuřín, 2006]. Conversely, there are also systems with 'rule-based' (linguistically interpretable) cores, which take advantage of the existence of statistical NLP tools such as taggers of parsers; see e.g. [Thurmair, 2004] for a discussion of this. Our MT system, which we present in the following chapters, combines linguistic knowledge and statistical techniques too.

Our MT system can be classified as a transfer-based system: first, it performs an analysis of input sentences to a certain level of abstraction, then it translates the abstract representation, and finally it performs sentence synthesis on the target-language side. Transfer-based systems often use syntactic trees as the transfer representation. Various sentence representations can be used as the transfer layer: e.g. (shallow) dependency trees are used in [Quirk et al., 2005], and constituency trees as e.g. in [Zhang et al., 2007]. Our system utilizes tectogrammatical trees as the transfer representation, which are remarkably similar to the normalized syntactic structures used for

[2] Example-based MT is occasionally considered as a third paradigm. However, it is difficult to find a clear boundary between Example-based MT and statistical MT.

[3] An overview of the possible combinations can be found at http://www.mt-archive.info/MTMarathon-2008-Eisele-ppt.pdf

translation in ETAP-3,[4] or to the logical forms used in [Menezes and Richardson, 2001]. All three representations capture a sentence as deep-syntactic dependency trees with nodes labeled with (lemmas of) autosemantic words and edges labeled with dependency relations.[5]

In our MT system, we use PDT-style tectogrammatical trees (t-trees for short). This option was discussed e.g. in [Hajič, 2002] and probably it was meant to be one of the applications of tectogrammatics much earlier. Experiments in a similar direction were published e.g. in [Čmejrek et al., 2003], [Fox, 2005], and [Bojar and Hajič, 2008].

1.3 Structure of the Thesis

The presented work is composed of two parts. After this introduction, Chapter 2 discusses how tectogrammatics fits to the task of MT. Chapter 3 introduces the notion of formemes aimed at facilitating translation of syntactic structures. Chapter 4 technically describes our software framework for developing NLP applications called TectoMT. Chapter 5 discusses how this framework can be used in English-Czech translation. Chapter 6 concludes the first part of this work.

The second part is a collection of our selected publications which have been published since 2000 in peer-reviewed conference proceedings or in the Prague Bulletin of Mathematical Linguistics. Most of the publications selected for this collection are joint works with other researchers, which is typical in computational linguistics.[6] Of course, the collection contains only articles in which the contribution of the present author was essential.

The articles in the second part are thematically divided into four groups: Annotating Prague Dependency Treebank, Parsing and Transformations of Syntactic Trees, Verb Valency, and Machine Translation.

The two parts are implicitly interlinked at numerous points, since most of the topics tackled in the second part played their role in the construction of the translation system described in the first part. To make the connection explicit, each paper included in the second part is referred to at least once in the first part. In addition, in front of each paper there is a brief preface, which looks at the paper from a broader perspective and sketches its relation to TectoMT.

[4] A preliminary comparison of tectogrammatical trees with trees used in Meaning-Text Theory (by which ETAP-3 is inspired) is sketched in [Žabokrtský, 2005].

[5] Another reincarnation of a similar idea—sentences represented as dependency trees with autosemantic words as nodes and "hidden" functional words—appeared recently in [Filippova and Strube, 2008]. However, the work was focused on text summarizing/compression, not on MT.

[6] For example, there are, on average, 3.0 authors per article in the Computational Linguistics journal of 2009 (volume 35, numbers 1-3), and 2.8 authors per paper in the proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics (one of the most prominent conferences in the field). One of the reasons is presumably the highly interdisciplinary nature of the field.

Chapter 2

Tectogrammatics in Machine Translation

2.1 Layers of Language Description in the Prague Dependency Treebank

As we work intensively with numerous constructs adopted from the annotation scheme (background linguistic theory, annotation conventions, file formats, software tools, etc.) of the Prague Dependency Treebank 2.0 (PDT for short, [Hajič et al., 2006]) in this work, we briefly summarize its main features first.

The PDT annotation scheme is based on Functional Generative Description (FGD) developed by Petr Sgall and his collaborators in Prague since the 1960's, [Sgall, 1967] and [Sgall et al., 1986]. One of the important features inherited by PDT from FGD is the stratification approach, which means that language description is decomposed into a sequence of descriptions – *strata* (called also levels or layers of description). There are three layers of annotation used in PDT: (1) morphological layer (m-layer for short), (2) analytical layer (a-layer), and (3) tectogrammatical layer (t-layer).[1]

At the morphological layer (detailed documentation in [Zeman et al., 2005]), each sentence is tokenized and morphological tags and lemmas are added to each token (word or punctuation mark).

At the analytical layer ([Hajič et al., 1999]), each sentence is represented as a surface-syntax dependency tree, in which each token from the original sentence is represented by one a-node. Each a-node is labeled by the analytical function, which captures the type of the node's dependency with respect to the governing node. Besides genuine dependencies (analytical function values such as Atr, Sb, and Adv), the analytical function also captures numerous rather technical issues (values such as AuxK for the sentence final full-stop, AuxV for an auxiliary verb in a complex verb form).

At the tectogrammatical layer ([Mikulová et al., 2005]), which is the most abstract and complex of the three, each sentence is represented as a deep-syntactic dependency tree, in which only autosemantic words (and coordination/apposition expressions) have nodes of their own. The nodes are labeled with tectogrammatical lemmas (ideally, pointers to a dictionary), and also with the functors, reflecting the dependency relations with respect to the governing nodes. According to applied valency theory (introduced in [Panevová, 1980]), functors distinguish actants (such as ACT for actor, PAT for patient) and free modifiers (various types of temporal, spatial, directional and other modifiers).

[1] Later in this text, we occasionally use the m-, a-, and t- prefixes for distinguishing which layer a given unit belongs to (a-tree, t-node, t-lemma, etc.).

Besides t-lemmas and functors, which constitute the core of the tectogrammatical tree structures, there are also numerous other attributes attached to t-nodes, corresponding to the individual "modules" of the t-layer description:

- There is a special attribute distinguishing conjuncts from shared modifiers in coordination and apposition constructions.
- For each verb node, there is a link to the used valency frame in the PDT-associated valency dictionary.
- There are attributes capturing information structure/communication dynamism.
- There are attributes called grammatemes, representing semantically indispensable, morphologically expressed meanings (such as number with nouns, tense with verbs, degree of comparison with adjectives).
- Miscellanea – there are attributes distinguishing roots of direct speeches, quotations, personal names, etc.

Besides linguistically relevant information stored on the individual layers, the layers' units are also equipped with links with connecting the given layer with the "lower" layer, as shown in Figure 1 in Section 7.2.

2.2 Terminological Note: Tectogrammatics or "Tectogrammatics"?

To avoid any terminological confusion, we should specify in which sense we use the term "tectogrammatics" (tectogrammatical layer of language representation), since there are several substantially different possible readings:

1. The term tectogrammatics was introduced in [Curry, 1963] in contrast to the term phenogrammatics. Sentence and noun phrase types are distinguished, a functional type hierarchy over them is considered, with functions from N to S, functions from functions from N to S to phrases of type S, etc. Tectogrammatical structure is built by combining such functions, while phenogrammatics looks at the result of evaluating tectogrammatical expressions.
2. The term tectogrammatics was used as a name for the highest level of language abstraction in Functional Generative Description in the 1960's, [Sgall, 1967]. The following levels were proposed: phonetic, morphonological, morphological, surface-syntactic, tectogrammatical.
3. Development of "Praguian" tectogrammatics continued in the following decades: new formalizations can be found in [Machová, 1977] or [Petkevič, 1987].
4. In the 1990's, tectogrammatics was chosen as the theoretical background for deep-syntactic sentence analysis in the Prague Dependency Treebank project. The initial version of the annotation guidelines (for annotating Czech sentences) were specified in [Panevová et al., 2001].

5. During the PDT annotation process, a lot of experience with applying tectogrammatics on real texts was gathered, which led to further modifications of the annotation rules. A final (and much larger) version of the PDT guidelines was published in [Mikulová et al., 2005] when the treebank was released.

6. The evolution of tectogrammatics still continues, for example in the project of annotating (English) Wall Street texts within the project Prague Czech-English Dependency Treebank, [Cinková et al., 2006].

In the following sections and chapters, we use the term "tectogrammatics" roughly in the sense of PDT 2.0 (reading 5). For MT purposes we perform additional minor changes, such as adding new attributes, different treatment of verb negation (in order to make it analogous to the treatment of negation of other word classes and to simplify the trees), and different interpretation of the linear ordering of tree nodes. The changes are always motivated by pragmatism, based on the empirical observations of the translation process.

We are aware that some of the changes might be in conflict with the theoretical presumptions of FGD, for example, not using t-node ordering for representing communication dynamism. However, despite such potentially controversial modifications, we decided to use the term tectogrammatics throughout this text and to refer to it even in the name of our translation system since

- we adhere to most of the core principles of tectogrammatics (each sentence is represented as a rooted tree with nodes corresponding to instances of autosemantic lexical units, edges corresponding to dependency relations among them, and other semantically indispensable meaning components captured as nodes' attributes) and adopt most of its implementation details specified in PDT 2.0 (e.g. naming node attributes and their values),

- as we have shown, due to continuous progress there is hardly any "the tectogrammatics" anyway, so using this term also in the context of TectoMT causes, in our opinion, less harm than trying to introduce our own new term (semitectogrammatics and MT-modified tectogrammatics were among the candidates) which would make the existence of those minor variances explicit.

2.3 Pros and Cons of Tectogrammatics in Machine Translation

2.3.1 Advantages

In our opinion, the main advantages of tectogrammatics from an MT viewpoint are the following:

- Tectogrammatics—even if it is not completely language independent—largely abstracts from language-specific repertories of means for expressing non-lexical meanings, such as inflection, agglutination, word order, or functional words. For example, the tense attribute attached to tectogrammatical nodes which represents heads of Czech finite verb clauses, captures the future tense by the same value regardless of whether the future tense was expressed by a prefix (*pojedu – I will go*), by inflection (*přijdu – I will come*), or by an auxiliary verb (*budu učit – I will teach*). This increases the similarity of sentence representations between typologically different languages, even if the lexical occupation remains, of course, different.

- Tectogrammatics "throws out" such information which is only imposed by grammar rules and thus is not semantically indispensable. For example, Czech adjectives in attributive positions express morphologically (by endings) their case, number, and gender categories, the values of which come from the governing nouns. So once we know that an adjective is in an attributive position, representing these categories becomes redundant. That is why adjectival tectogrammatical nodes do not store the values of the three categories at all.

- Tectogrammatics offers a natural factorization of the transfer step: lexical and non-lexical meaning components are "mixed" in a highly non-trivial way in the surface sentence shape, while they become (almost) orthogonal in its tectogrammatical representation ([Ševčíková-Razímová and Žabokrtský, 2006]). For example, the lexical value (stored in the t_lemma attribute) of a noun is clearly separated from its grammatical number (stored in the gram/number attribute). In a light of the two items above, it is clear that this is not the same as simply making a morphological analysis.

- We expect that local *tree* contexts in t-trees (i.e., children and especially the parent of a given t-node) carry more information (esp. for lexical choice) than local *linear* contexts in the original sentences.

We believe that these four features of tectogrammatics, i.e. (1) highlighting the similar structural core of different languages, (2) orthogonality/easy transfer factorization, (3) decreased redundancy, and (4) availability of dependency context (besides the linear context), could eventually help us to construct probabilistic translation models which are more efficient than phrase-based models in facing the notorious MT data sparsity problem.

2.3.2 Disadvantages

Despite the promising features of tectogrammatics from the MT viewpoint, there are also practical drawbacks in tecto-based MT (again, when compared to the state-of-the-art phrase-based models) which must be considered:

- Tectogrammatical data are highly structured and thus they require more complex memory representation and file formats, which limits the processing speed.

- Another disadvantage is caused by the fact that there several broadly used techniques for linear data (Hidden Markov Models, etc.), but similar tree-processing techniques (such as Hidden Markov Tree Models, [Diligenti et al., 2003]) are much less widely known.

- There are several open theoretical question in tectogrammatics. For example, it is not clear whether (and in what form) other linguistically relevant information could be added into t-trees (as pointed out in [Novák, 2008]), e.g. information about named entity hierarchy or definiteness in languages with articles.

- In our opinion, the most significant current obstacle in developing tecto-based MT is of a psychological nature: the developers are required to have at least a minimal insight into tectogrammatics (and the other PDT layers and relations among them), which—given the size of annotation guidelines and unavailability of short and clear introductory materials—has a strongly discouraging effect on the potential newcomers. In this aspect, relative simplicity and "flatness" is a great advantage of the phrase-based MT systems, and supports their much faster community growth.

- Another reason that limits the size of the community of developers of MT system based on dependency formalisms such as tectogrammatics is that the "dependency-oriented world" is smaller due to several historical reasons (as discussed e.g. in [Bolshakov and Gelbukh, 2000]). However, thanks to popular community events such as CoNLL-X Shared Task (competition in multilingual dependency parsing),[2] the dependency-oriented world seems to be growing.

[2] http://nextens.uvt.nl/˜conll/

Chapter 3

Formemes

3.1 Motivation for Introducing Formemes

Before giving our motivation for introducing the notion of formeme, we should first briefly explain this notion. Formeme can be seen as a property of a t-node which specifies in which morphosyntactic form this t-node was (in the case of analysis) or will be (in the case of synthesis) expressed in the surface sentence shape. The set of formeme values compatible with a given t-node is limited by the t-node's semantic part of speech: semantic nouns cannot be directly shaped into the form of subordinating clause, semantic verbs cannot be shaped into a possessive form, etc. Obviously, the set of formemes is highly language dependent, as languages differ not only in the repertory of morphosyntactic strategies they use, but also in the sets of values of the individual morphological categories (e.g. different case systems) and in the sets of available functional words (such as prepositions).

Here are some examples of formemes which we use for English:

- n:subj – semantic noun in subject position,
- n:for+X – semantic noun with the preposition *for*,
- n:X+ago – semantic noun with the postposition *ago*,
- n:poss – possessive form of a semantic noun,
- v:because+fin – semantic verb as a subordinating finite clause introduced by *because*,
- v:without+ger – semantic verb as a gerund after *without*,
- adj:attr – semantic adjective in attributive position,
- adj:compl – semantic adjective in complement position.

Our initial motivation for the introduction of formemes was as follows: during experiments with synthesis of Czech sentences from their t-trees (see Section10.1) we noticed that it might be advantageous to clearly differentiate between (a) deciding what surface form will be used for which t-node, and (b) performing the shaping changes (such as inflecting the t-lemmas, adding functional words and punctuation, reordering, etc.). The most informative attribute for the specification of the surface form of a given t-node is undoubtedly the t-node's functor, but many other local t-tree properties come into play, which makes the sentence synthesis directly from t-trees highly non-trivial. However, if we separate making decisions about the surface shape (i.e., specifying t-nodes' formemes) from performing the shaping changes, not only the modularity of the system increases, but the former part of the process becomes solvable by standard Machine Learning techniques, while the implementation of the latter part becomes solvable without probabilistic decision-making.

Another strong motivation for working with formemes in t-trees came later. As we have already mentioned, tectogrammatics helps us to factorize the translation, e.g. by separating information about the lemma of an adjective from information about its degree of comparison (the two can then be translated relatively independently). The transfer via tectogrammatics can be straightforwardly decomposed into three factors:[1,2]

1. translating lexical information captured in the t_lemma attribute,

2. translating morphologically expressed meaning components captured by the grammateme attributes, and

3. translating dependency relations, captured especially by the functor attribute.

We believe that as soon as we work with formemes in our t-trees, the task of the third factor (translating the sentence 'syntactization') can be implemented more directly by translating only the formemes. The underlying intuition is the following: Instead of assigning the functor TSIN to the expression 'since Monday' on the source-language side, keeping the functor during the transfer, and using this functor for assigning the morphosyntactic form on the target side (prepositional group with preposition *od* and genitive case), we could directly translate the English n:since+X formeme to the Czech n:od+2 formeme.[3] Moreover:

- If the transfer goes via functors, we need a system for assigning functors on the source-language side, a system for translating functors, and a system which decides what surface forms should be used on the target-language side give the functor labels. There is a trivial and probably satisfactory solution of the middle step (leave the same values), but the other two tasks are highly non-trivial and statistical/machine learning tools have to be applied (see e.g. [Žabokrtský et al., 2002]).

[1] Theoretically, a fourth transfer factor corresponding to information structure (IS) should be considered too. However, as far as we consider English-to-Czech translation direction, our experience with thousands of sentences confirms that errors caused by ignoring the IS factor are absolutely insignificant (both in number and subjective importance) compared to errors caused by other factors, especially by the lexical one. This holds not only for TectoMT, but also for our observations of other MT systems' outputs for this language pair.

[2] Of course, the three factors cannot be treated as completely independent. For example, translating 'come' as 'přijít' in the lexical factor might require changing the tense grammateme from present to future (there is no way to express present tense with perfective verbs in Czech).

[3] It should be mentioned that the set of functors used in PDT is heterogenous: there are classes of functors with very different functions. We plan to abstract away only from functors which label dependency relations (actants and free modifiers), whereas functors for coordination and apposition constructions will remain indispensable even in the formeme-based approach.

- If we work with formemes, it is the other way round: the formemes on the source side can be assigned deterministically (given the t-tree with links to the a-tree), then formeme-to-formeme translation follows, and then the synthesis of the target-language sentence is deterministic, given the t-tree with formeme labels. Now the first and last steps are deterministic and only the middle one is difficult. In this way we reduce undesirable chaining of statistical systems. The formeme-to-formeme translation model can be trained from aligned parallel t-trees, and all the features which would be otherwise used for functor assignment and translation can be used in formeme translation too.

To conclude: using formemes instead of functors should allow us to construct more compact models of sentence syntactization translation, while the main feature of tectogrammatics from an MT viewpoint—orthogonality offering a straightforward translation factorization—is still retained.

3.2 Related work

In literature, one can find attempts at an explicit description of morphological (and also syntactic)[4] requirements on surface forms of sentence elements especially in the relation with valency dictionaries. See [Žabokrtský, 2005] for a survey of numerous approaches, the first of them probably being [Helbig and Schenkel, 1969].

Of course, our own view on how surface forms should be formally captured was strongly influenced by our experience with the VALLEX lexicon ([Lopatková et al., 2008], also Section 9.1). But it should be noted that the set of formemes which we use in TectoMT is not identical with what is used in VALLEX. For example, since VALLEX contains only verbs, none of the slots contained in the frames in the lexicon can have a form of an attribute or of a relative clause.

The term ‘formeme’ (*formém* in Czech) was probably first used when FGD was introduced in [Sgall, 1967]. The following types of complex units of the morphological level called formemes were distinguished (p. 74): (a) lexical formemes, (b) case formemes (combinations of prepositions and cases, including zero preposition), (c) conjunction formemes (combination of a subordinating conjunction and verb mood), and (d) other grammatical formemes. Examples of formemes such as *o+L* (preposition *o* (about) and the locative case) and *když+VF* (subordinating conjunction *když* (when) and *verbum finitum*) were given (pp. 168-169).

The notion of formeme as defined in the original FGD obviously overlaps with the notion of formeme in this chapter. However, there are two important differences: (1) we do not treat formemes as units of the morphological level, but attach them as attributes to tectogrammatical nodes, and (2) our notion of formeme does not comprise ‘lexical formemes’.

[4] In our opinion, the fact that an expression has the form of a subordinating clause introduced with a certain conjunction, cannot be adequately expressed using the morphological level, but the surface syntax is needed too.

We decided to use the term 'formeme' instead of surface/morphemic/morphosyntactic form simply for pragmatic reasons: first, it is shorter, and second, together with the terms lexeme and grammateme it constitutes an easy-to-remember triad representing the three main factors of translation in TectoMT (as explained in section 3.1). To our knowledge, the term 'formeme' does not occur in the contemporary linguistic literature, so licensing it for our specific purpose will hopefully not cause too much harm.

3.3 Theoretical status of formemes

On one hand, adding the formeme attribute to tectogrammatical nodes allows for a relatively straightforward modeling of syntactization translation (compared to that based strictly on tectogrammatical functors). But on the other hand, it also means

1. "smuggling" elements of surface syntax into tectogrammatical (deep-syntactic) trees, which blurs the theoretical border between the layers of the original FGD.
2. increasing the redundancy of sentence representation (if formemes are added to full-fledged t-trees), because the information contained in formemes partially overlaps with the information captured by functors,
3. making the enriched t-trees more language specific, since the set of formeme values is more language specific than the set of functor values.

Our conclusion is the following: formeme attributes can be stored with t-nodes, which is technically easy and which could be very helpful for syntactization translation. However, from a strictly theoretical viewpoint they cannot be seen as a component of the tectogrammatical layer of language description in the sense of FGD, as it would not be compatible with some of the FGD's core ideas (clear layer separation, orthogonality, high degree of language independence). But neither can formemes be attached to a-layer nodes, because having both prescriptions for surface forms (e.g. a formeme for a prepositional group) and the surface units themselves (the prescribed preposition a-node) on the same layer would be redundant. Therefore, rather than belonging to one of these layers, formemes model a transition between them. But since in the PDT scheme there are only representations of layers and no separate representations of the transitions between them, we believe that the best (even if theoretically not fully adequate) way to store formemes in the form of t-node attributes.[5]

3.4 Formeme Values

Before having designed the set of formeme values which we currently use, we kept in mind the following desiderata:

- the values should be easily human-readable,

[5] Links between t-nodes and a-nodes are represented as pointers (a/lex.rf and a/aux.rf) stored as attributes of t-nodes in PDT too, even if they do not constitute a component of tectogrammatics.

- the values should be automatically parsable,

- if a formeme expresses that a certain functional word should be used on the surface (which is not always the case, as some formemes can imply only a certain type of agreement or certain requirements on linear position), then the functional word should be directly extractable from the formeme value, without the need of a decoding dictionary,

- the preceding rule must not hold in the case of synonymy on the lower layers: for example, Czech prepositional groups with short and vocalized forms of the same preposition (*k* and *ke*, *s* and *se*, etc.) are captured by the same formeme and not as two formemes, since preposition vocalization belongs to phonology rather than to morphosyntax ([Petkevič and Skoumalová, 1995]).

- different sets of formemes are applicable for t-nodes with a different semantic part of speech; it should be directly clear from the formeme values which semantic parts of speech they are compatible with,

- sets of formemes are undoubtedly language specific, however, we will attempt to use the same values for highly analogous cases; for example, there will be the same value adj:attr saying that an adjective is in the attributive position in Czech or in English, even if in Czech it is manifested by agreement which is not the case in English. Similarly, there will be the same value for heads of relative clauses both in Czech and English.

It is obvious that the set of formeme values is inherently structured: the same preposition appears in the English formemes n:without+X and in v:without+ger, the same case appears in the Czech formemes n:pro+4 and n:na+4, etc. However, we decided to represent a formeme value technically as an 'atomic' string attribute instead of a structure, since it significantly facilitates any manipulation with formemes.[6]

Now, we will provide examples for the individual parts of speech. Only examples of formemes applicable for Czech or English are given; completely different types of formemes might appear in typologically distant languages (such as suffixes in Hungarian or tones in Vietnamese).
Examples of formemes compatible with semantic verbs:

- v:fin – head of the finite clause (in a matrix clause, parenthetical clause, or direct speech, or a subordinated clause without any subordinating conjunction or relative pronoun), both in Czech and English

- v:that+fin, v:že+fin – subordinated clause introduced with the given subordinating conjunction, both in Czech and English

[6] A similar approach was used in the set of Czech positional morphological tags – the tags are represented as seemingly atomic strings, even if the set of all possible tags is in fact highly structured and the structure cannot be seen without string-wise decomposition of the tag values.

- v:rc – relative clause, both in Czech and English

- v:ger – gerund, (frequent) only in English

- v:without+ger – preposition with gerund, only in English

- v:attr – active or passive adjectival form (*fried fish*, *smiling guy*)

Examples of formemes compatible with semantic nouns:

- n:1, n:3, n:7. . . – noun in nominative (dative, instrumental) case (in Czech),

- n:subj, n:obj – noun in subject/object position (in English),

- n:attr – noun in attributive position (both in Czech and English, e.g. *pan kolega*, *world championship*, *Josef Novák*),

- n:poss – Saxon genitive in English or possessive adjective derived from noun in Czech (*Peter's*, *Petrův*) in the case of morphological nouns, or possessive forms of pronouns (*jeho*, *his*),

- n:s+7 – prepositional group with the given preposition *s* and the noun in genitive case (in Czech),

- n:with+X – prepositional group with the given preposition (in English),

- n:X+ago – postpositional group with the given postposition (in English),

Examples of formemes compatible with semantic adjectives:

- adj:attr – adjective in attributive position,

- adj:compl – adjective in complement position or after copula (*Stal se bohatým – He became rich*),

- adj:za+x – adjective in a nounless prepositional group (*Pokládal ho za bohatého – He considered him rich*).

Examples of formemes compatible with semantic adverbs:

- adv: – the adverb alone,

- adv:from+x – adverb with a preposition (*from when*).

3.5 Formeme Translation

One of the main motivations for introducing the notion of formemes was to facilitate translation of sentence syntactic structure. Of course, it would be possible to try creating a set of hand-crafted formeme-to-formeme translation rules. However, we decided to keep the formeme translation strictly data-driven and to extract such formeme dictionary from parallel data.

The translation mapping from English formemes to Czech formemes was obtained as follows. We analyzed 10,000 sentence pairs from the parallel text distributed during the Shared Task of Workshop in Statistical Machine Translation[7] up to the t-layer. We used Jan Hajič's tagger ([Hajič, 2004]) shipped with PDT 2.0 ([Hajič et al., 2006]) and the parser [McDonald et al., 2005] for Czech, and a rule-based conversion from the Czech a-layer to t-layer. The t-nodes were then labeled with formeme values. The procedure for analyzing the English sentences was more or less the same as that described in Sections 5.1.1–5.1.4. After finishing the analysis on both sides, we aligned t-nodes in the corresponding t-trees using the alignment procedure developed in [Mareček, 2008], inspired by [Menezes and Richardson, 2001]. Then we extracted the probabilistic formeme translation dictionary from the aligned t-node pairs. Fragments from the dictionary are shown in Table 3.1.

3.6 Open questions

The presented set of formemes should be seen as tentative and will probably undergo some changes in future, as there are still several issues that have not been satisfactorily solved. For example, it is not clear to what extent verb diathesis should influence the formeme value: should we distinguish in Czech the basic active verb form from the reflexive passivization (e.g. *vařit* vs. *vařit se*) by a formeme? Currently we do not. The same question holds for distinguishing passive and active deverbal attributes (e.g. *killing man* vs. *killed man*). Adding information about verb diathesis/voice (see Section 9.2) into the formeme attribute could be advantageous in some situations because of the fact that English passive forms which are often translated into Czech as reflexive passive forms could be modeled more directly. But on the other hand, the orthogonality of our system would suffer and the data sparsity problem would increase (the number of verb formemes would get multiplied by the number of diatheses).

[7] http://www.statmt.org/wmt08/

F_{en}	F_{cz}	$P(F_{cz}\|F_{en})$
adj:attr	adj:attr	0.9514
adj:attr	n:2	0.0138
n:subj	n:1	0.6483
n:subj	adj:compl	0.1017
n:subj	n:4	0.0786
n:subj	n:7	0.0293
n:obj	n:4	0.4231
n:obj	n:1	0.1828
n:obj	n:2	0.1377
v:fin	v:fin	0.9110
v:fin	v:rc	0.0232
v:fin	v:že+fin	0.0177
n:of+X	n:2	0.7719
n:of+X	adj:attr	0.0477
n:of+X	n:z+2	0.0402
n:in+X	n:v+6	0.5185
n:in+X	n:2	0.0878
n:in+X	adv:	0.0491
n:in+X	n:do+2	0.0414
n:poss	adj:attr	0.4056
n:poss	n:2	0.3798
n:poss	n:poss	0.1148
v:to+inf	v:inf	0.4817
v:to+inf	v:aby+fin	0.0950
v:to+inf	n:k+3	0.0702
v:to+inf	v:že+fin	0.0621
n:for+X	n:pro+4	0.2234
n:for+X	n:2	0.1669
n:for+X	n:4	0.0788
n:for+X	n:za+4	0.0775
n:on+X	n:na+6	0.2632
n:on+X	n:na+4	0.2180
n:on+X	n:2	0.0695
n:on+X	n:o+6	0.0602
n:from+X	n:z+2	0.4238
n:from+X	n:od+2	0.1951
n:from+X	n:2	0.0945
v:if+fin	v:pokud+fin	0.3067
v:if+fin	v:li+fin	0.2393
v:if+fin	v:kdyby+fin	0.1718
v:if+fin	v:jestliže+fin	0.1104
v:in+ger	n:při+6	0.3538
v:in+ger	v:inf	0.1077
v:in+ger	n:v+6	0.0923
v:while+fin	v:zatímco+fin	0.5263
v:while+fin	v:přestože+fin	0.1404
v:without+ger	v:aniž+fin	0.7500
v:without+ger	n:bez+2	0.1875
n:because_of+X	n:kvůli+3	0.4615
n:because_of+X	n:díky+3	0.3077

Table 3.1: Fragments from English-Czech probabilistic formeme translation dictionary. For each selected English formeme, several most probable Czech counterparts are listed, as well as conditional probability of the Czech formemes F_{cz} given the English formemes F_{en}.

Chapter 4

TectoMT Software Framework

TectoMT is a software framework for implementing NLP applications, focused especially on the task of Machine Translation (but by far not limited to it). The main motivation for building such a system was to allow for an easier integration of various existing NLP components (such as taggers, parsers, named entity recognizers, tools for anaphora resolution, sentence generators) and also to develop new ones in a common framework, so that larger systems and real-world applications can be built out of them in a simpler way than ever before.

We started to develop the framework at the Institute of Formal and Applied linguistics in autumn 2005. The architecture of the framework, the core technical components such as the application interface (API) to Perl representation of linguistic structures and various modules for processing linguistic data, have been implemented by the present author, but numerous other utilities have been created by roughly ten other contributing programmers, not to mention the work of authors of previously existing publicly available NLP tools integrated into TectoMT, many of which will be referred to in Chapter 5.

4.1 Main Design Decisions

During the development of TectoMT we have faced many design questions. The most important design decisions are the following:

- Modularity is emphasized in TectoMT. Any non-trivial NLP task should be decomposed into a sequence of subsequent steps, implemented in modules called *blocks*. The sequences of blocks (strictly linear, without branches) are called *scenarios*.
- Each block should have a well-documented, meaningful, and—if possible—also linguistically interpretable functionality, so that it can be easily substituted with an alternative solution (another block), which attempts to solve the same subtask using a different method/approach. Since granularity of the task decomposition is not given in advance, one block can have the same functionality as an alternative solution composed of several blocks (e.g., some taggers perform also lemmatization, whereas other taggers have to be followed by separate lemmatizers). As a rule of thumb, the size of a block should not exceed several hundred lines of code (of course, counting only the lines of the block itself and not the included modules).
- Each block is a Perl module (more specifically, a Perl class with an inherited interface). However, this does not mean that the solution of the task itself has to

be implemented in Perl too: the module itself can be only a wrapper for a binary application or a Java application, or a client of a web service running on a remote machine, etc.

- TectoMT is implemented in *Linux.* Full portability of the whole TectoMT to other operating systems is not realistic in the near future. But again, this does not exclude the possibility of releasing platform independent applications made of selected components. So, naturally, platform independent solutions should be sought after whenever possible.

- Processing of any type of linguistic data in TectoMT can be viewed as a path through the Vauquois diagram (with the vertical axis corresponding to the level/layer of language abstractions and the horizontal axis possibly corresponding to different languages, [Vauquois, 1973]). It should be always clear with which layers a given block works. By default, TectoMT mirrors the system of layers as developed in the PDT (morphological layer, analytical layer for surface dependency syntax, tectogrammatical layer for deep syntax), but other layers might be added too. By default, sentence representation at any level is supposed to form a tree (even if it is a flat tree on the morphological level and even if co-reference links might be seen as non-tree edges on the tectogrammatical layer).

- TectoMT is *neutral with respect to the methodology* employed in the individual blocks: fully stochastic, hybrid, or fully symbolic (rule-based) approaches can be used. The only preference is as follows: the solution which reaches the best evaluation result for the given subtask (according to some measurable criteria) is the best.

- Any block in TectoMT should be capable of *massive data processing.* It makes no sense to develop a block which needs on average more than a few hundred milliseconds per processed sentence (rule of thumb: the complete translation block sequence should not need more than a couple of seconds per sentence). Also, memory requirements of any block should not exceed reasonable limits, so that individual developers can run the blocks using their “home computers”.

- *TectoMT is composed of two parts.* The first part (the development part), which contains especially the processing blocks and other in-house tools and Perl libraries, is stored in an SVN repository so that it can be developed in parallel by more developers (and also outside the UFAL Linux network). The second part (the shared part), which contains downloaded libraries, downloaded software tools, independently existing linguistic data resources, generated data, etc., is shared without versioning because (a) it is supposed to be changed (more or less) only additively, (b) it is huge, as it contains large data resources, and (c) it should be automatically reconstructable (simply by redownloading, regeneration or reinstallation of its parts) if needed.

- TectoMT processing of linguistic data is usually composed of three steps: (1) convert the data (e.g. a plain text to be translated) into the tmt data format (PML-based format developed for TectoMT purposes), (2) apply the sequence of processing blocks, using the TectoMT object-oriented interface to the data, (3) convert the resulting structures to the desired output format (e.g., HTML containing the resulting translation).

- The main difference between the tmt data format and the PML applications used in PDT 2.0 is the following: in tmt, all representations of a textual document at the individual layers of language description are stored in a single file. As the number of linguistic layers in TectoMT might be multiplied by the number of processed languages (two or more in the case of parallel corpora) and by the direction of their processing (source vs. target during translation), manipulation with a growing number of files corresponding to a single textual document would become too cumbersome.

4.2 Linguistic Structures as Data Structures in TectoMT

4.2.1 Documents, Bundles, Trees, Nodes, Attributes

In TectoMT, linguistic representations of running texts are organized in the following hierarchy:

- One physical file corresponds to one document.

- A document consists of a sequence of bundles, mirroring a sequence of natural language sentences (typically, but not necessarily, originating from the same text). Attributes (attribute-value pairs) can be attached to a document as a whole.

- A bundle corresponds to one sentence in its various forms/representations (esp. its representations on various levels of language description, but also possibly including its counterpart sentence from a parallel corpus, or its automatically created translation, and their linguistic representations, be they created by analysis / transfer / synthesis). Attributes can be attached to a bundle as a whole.

- All sentence representations are tree-shaped structures – the term bundle stands for 'a bundle of trees'.

- In each bundle, its trees are "named" by the names of layers, such as SEnglishM (source-side English morphological representation, see the next section). In other words, there is, at most, one tree for a given layer in each bundle.

- Trees are formed by nodes and edges. Attributes can be attached only to nodes. Edges' attributes must be equivalently stored as the lower node's attributes. Trees' attributes must be stored as attributes of the root node.

- Attributes can bear atomic values or can be further structured (lists, structures etc.), as allowed by PML.

For those who are acquainted with the structures used in PDT 2.0, the most important difference lies in bundles: the level added between documents and trees, which comprises all layers of representation of a given sentence. As one document is stored as one physical file, all layers of language representations can be stored in one file in TectoMT (unlike in PDT 2.0).

4.2.2 'Layers' of Linguistic Structures

The notion of 'layer' has a combinatorial nature in TectoMT. It corresponds not only to the layer of language description as used e.g. in the Prague Dependency Treebank, but it is also specific for a given language (e.g., possible values of morphological tags are typically different for different languages) and even for how the data on the given layer were created (whether by analysis from the lower layer or by synthesis/transfer).

Thus, the set of TectoMT layers is a Cartesian product $\{S, T\} \times \{English, Czech\} \times \{W, M, P, A, T\}$, in which:

- values $\{S, T\}$ represent whether the data was created by analysis or transfer/synthesis (mnemonics: S and T correspond to (S)ource and (T)arget in MT perspective),

- values $\{English, Czech\}$ represent the language in question,

- values $\{W, M, P, A, T...\}$ represent the layer of description in terms of PDT 2.0 (W – word layer, M – morphological layer, A – analytical layer, T – tectogrammatical layer) or extensions (P – phrase-structure layer).

TectoMT layers are denoted by stringifying the three coordinates: for example, analytical representation of an English sentence acquired by sentence analysis is denoted as SEnlishA. This naming convention is used in many places in TectoMT: for naming trees in a bundle (and corresponding xml elements), for naming blocks, for node identifier generating, etc.

Unlike layers in PDT 2.0, the set of TectoMT layers should not be understood as totally ordered. Of course, there is a strong intuition basis for the abstraction axis of languages description (SEnglishT requires more abstraction than SEnglishM), but this intuition might not be sufficient in some cases (SEnglishP and SEnglishA represent roughly the same level of abstraction).

4.2.3 TectoMT API to linguistic structures

The linguistic structures in TectoMT are represented using the following object-oriented interface/types:

- document – `TectoMT::Document`

- bundle – `TectoMT::Bundle`
- node – `TectoMT::Node`
- document's, bundle's, and node's attributes – Perl scalars in case the PML schema prescribes an atomic type, or an appropriate class from `Fslib` corresponding to the type specified in the PML schema.

Classes TectoMT::Document,Bundle,Node have their own documentation, here we list only the basic methods for navigating through a TectoMT document (Perl variables such as `$document` are used only for illustration purposes, but there are no predefined variables like this in TectoMT). "Contained" objects encapsulated in "container" objects can be accessed as follows:

- `my @bundles = $document->get_bundles` – an array of bundles contained in the document
- `my $root_node = $bundle->get_tree($layer_name);` – the root node of the tree of the given type in the given bundle

There are also methods for accessing the container objects from the contained objects:

- `my $document = $bundle->get_document;` – the document in which the given bundle is contained
- `my $bundle = $node->get_bundle;` – the bundle in which the given node is contained
- `my $document = $node->get_document;` – composition of the two above

There are several methods for traversing tree topology, such as

- `my @children = $node->get_children;` – array of the node's children
- `my @descendants = $node->get_descendants;` – array of the node's children and their children and the children of their children ...
- `my $parent = $node->get_parent;` – parent node of the given node, or undef for root
- `my $root_node = $node->get_root;` – the root node of the tree into which the node belongs

Attributes of documents, bundles or nodes can be accessed by attribute getters and setters:

- `$document->get_attr($attr_name); $document->set_attr($attr_name, $attr_value);`
- `$bundle->get_attr($attr_name); $bundle->set_attr($attr_name, $attr_value);`

- `$node->get_attr($attr_name); $node->set_attr($attr_name, $attr_value);`

$attr_name is always a string (following the Fslib conventions in the case of structured attributes, e.g. using a slash in structured attributes, e.g. `gram/gender`).

New classes, with functionality specific only for some layers, can be derived from TectoMT::Node. For example, methods for accessing effective children/parents should be defined for nodes of dependency trees. Thus, there are, for example, classes named `TectoMT::Node::SEnglishA` or `TectoMT::Node::SCzechA` offering methods `get_eff_parents` and `get_eff_children`, which are inherited from a general analytical "abstract class" `TectoMT::Node::A` (which itself is derived from `TectoMT::Node`). Please note that the names of the 'terminal' classes are the same as the layer names. If there is no specific class defined for a layer, `TectoMT::Node` is used as a default for nodes on this layer.

All these classes are stored in `devel/libs/core`. Obviously, they are crucial for the functioning of most other components of TectoMT, so their functionality should be carefully checked after any changes.

4.2.4 Fslib as underlying representation

Technically, the full data structures are not stored in `TectoMT::{Document,Bundle,Node}` representation, but there is an underlying representation based on Petr Pajas's Fslib library[1] (tree-processing library distributed with the tree editor TrEd). Practically the only data stored in TectoMT objects (besides some indexing) are references to Fslib objects. The combination of a new object-oriented API (TectoMT) with the previously existing library (Fslib) used for the underlying memory representation was chosen because of the following reasons:

- In Fslib, it would not be possible to make the objects fully encapsulated, to introduce node-class hierarchy, and it would be very difficult to redesign the existing Fslib API (classes, functions, methods, data structures), as there is an excessive amount of existing code dependent on Fslib. So developing a new API seemed to be necessary.

- On the other hand, there are two important advantages of using the Fslib representation. First, we can use Prague Markup Language as the main file format, since serialization into PML (and reading PML) is fully implemented in Fslib. Second, since we use a Fslib-compatible file format, we can use also the tree editor TrEd for visualizing the structures and btred/ntred for comfortable batch processing of our data files.

Outside of the core libraries, there is almost no need to access the underlying Fslib representation – the data should be accessed exclusively via the TectoMT interface (unless some very special Fslib functionality is needed). However, the underlying Fslib representation can be accessed from the TectoMT instances as follows:

[1] http://ufal.mff.cuni.cz/ pajas/tred/Fslib.html

- `$document->get_tied_fsfile()` returns the underlying FSFile instance,
- `$bundle->get_tied_fsroot()` returns the underlying FSNode instance,
- `$node->get_tied_fsnode()` returns the underlying FSNode instance.

4.2.5 TMT File Format

The main file format used in TectoMT is TMT (.tmt suffix). TMT format is an application of PML. Thus, TMT files are PML instances of a PML schema. The schema is stored in `$TMT_ROOT/pml/tmt_schema.xml`. This schema merges and changes (more or less additively) the PML schemata from PDT 2.0.

The PML schema directly renders the logical structure of data: there can be one document in one tmt-file, the document has its attributes and contains a sequence of bundles, each bundle has its attributes and contains a set of trees (named by layer names), each tree consists of nodes, which again contain attributes.

Files in the TMT format are readable by the naked eye, but this is in fact useful only when writing and debugging format convertors from TMT to other formats or back. Otherwise, it is much more convenient to view the data in TrEd.

In TectoMT, one should never write components that directly access the TMT files (of course, with the single exception of convertors from other formats to TMT or back). Instead, the data should be accessed by the components exclusively via the above mentioned object-oriented Perl API.

4.3 Processing units in TectoMT

In TectoMT, there is the following hierarchy of processing units (i.e., software components that process data):

- The basic processing units are *blocks*. They serve for some very limited, well defined, and often linguistically interpretable tasks (e.g., tokenization, tagging, parsing). Blocks are not parametrizable. Technically, blocks are Perl classes inherited from `TectoMT::Block`.
- To solve a more complex task, selected blocks can be chained into a *block sequence*, also called a *scenario*. Technically, scenarios are instances of `TectoMT::Scenario` class, but in some situations (e.g. on the command line) it is sufficient to specify the scenario simply by listing block names separated by spaces.
- The highest unit is called application. Applications correspond to end-to-end tasks, be they real end-user applications (such as machine translation), or 'only' NLP-related experiments. Technically, applications are often implemented as Makefiles, which only glue together the components existing in TectoMT.

Technically, blocks are Perl classes derived from `TectoMT::Block`, with the following conventional structure:

1. block (package) name on the first line,
2. uses of pragmas and libraries,
3. possibly some initialization (e.g. loading external data),
4. declaration of the `process_document` method,
5. short POD documentation,
6. author’s copyright notice.

Example of a simple block, which causes that negation particles in English will be considered to be parts of verb forms during the transition from the SEnglishA layer to the SEnglishT layer:

```
package SEnglishA_to_SEnglishT::Mark_negator_as_aux;
use 5.008;
use strict;
use warnings;
use Report;
use base qw(TectoMT::Block);
use TectoMT::Document;
use TectoMT::Bundle;
use TectoMT::Node;

sub process_document {
  my ($self,$document) = @_;

  foreach my $bundle ($document->get_bundles()) {
    my $a_root = $bundle->get_tree(’SEnglishA’);

    foreach my $a_node ($a_root->get_descendants) {
      my ($eff_parent) = $a_node->get_eff_parents;
      if ($a_node->get_attr(’m/lemma’)=~/^(not|n\’t)$/
          and $eff_parent->get_attr(’m/tag’)=~/^V/ ) {
        $a_node->set_attr(’is_aux_to_parent’,1);
      }
    }
  }
}

1;
=over
=item SEnglishA_to_SEnglishT::Mark_negator_as_aux
’not’ is marked as aux_to_parent (which is used in the translation scenarios,
but not in preparing data for annotators)
=back
=cut

# Copyright 2008 Zdenek Zabokrtsky
```

Blocks are stored in subdirectories of the `libs/blocks/` directory. Most blocks are distributed among the directories according to their position along the virtual path through the Vauquois triangle. More specifically, they are part of a transition from layer L1 to layer L2. Such blocks are stored in the L1_toL2 directory, e.g. in SEnglishA_to_SEnglishT. But there are also blocks for other purposes, e.g. evaluation blocks (`libs/blocks/Eval/`) or data extraction blocks (`libs/blocks/Print/`).

Chapter 5

English-Czech Translation Implemented in TectoMT

The structure of this section directly reflects the sequence of blocks currently used for English-Czech translation in TectoMT. The translation process as a path along the well-know Vauquois "triangle" is sketched in Figure 5.1.

Two anomalies can be found in the diagram. First, there is an extra horizontal transition on the source language side, namely the transition from the English phrase-structure representation to the English analytical (surface-dependency) representation. This transition was included in the described version of our MT system because we had no English dependency parser available at the beginning of the experiment (however, we have it now, so the phrase-structure detour can be omitted in the more recent translation scenarios).

The second anomaly can be seen in the fact that the morphological layer seems to be missing on the target-language side. In fact, the two representations are merged and we build them more or less simultaneously: technically, the constraints on morphological categories are attached directly to a-nodes. The reason is that topological operations on the a-layer (such as adding new a-nodes or reordering them) are naturally interleaved with operations belonging rather to the m-layer (such as determining the values of morphological categories), and nothing would be gained if we forced ourselves to separate them strictly.

5.1 Translation Process Step by Step

Figure 5.2 illustrates the translation process by a sequence of tree representations for a sample sentence. The representations on each layer are presented in their final form (i.e., after finishing the transition to that layer).

5.1.1 From SEnglishW to SEnglishM

B1: The source English text is segmented into sentences. A new empty bundle is created for each sentence. A regular expression (covering the most frequent abbreviations) for finding sentence boundaries is used in this block. However, it will be necessary to use a more elaborate solution in the future, especially when translating HTML documents, in which the sentence boundaries should reflect also the formatting markup (e.g. paragraphs, titles and other block elements).

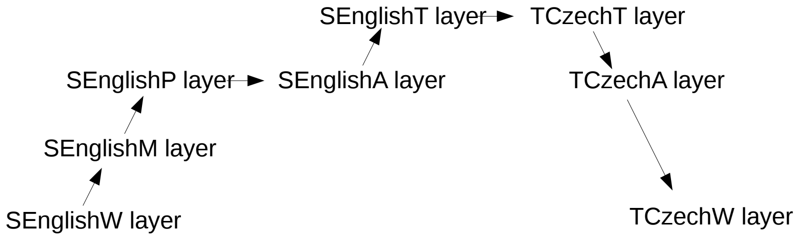


Figure 5.1: MT pyramid in terms of TectoMT layers.

B2: The sentences are split into sequences of tokens, roughly according to the Penn Treebank (PTB for short; [Marcus et al., 1994]) conventions, see the flat SEnglishM tree in Figure 5.2 The PTB-style tokenization is chosen because of compatibility with NLP tools trained on the PTB data. Robert MacIntyre's tokenization sed script[1] was modified for this purpose.

B3: The tokens are tagged with PTB-style POS tags using the TnT tagger ([Brants, 2000]); see the symbols such as `JJ` for the adjective *old* or `PRP$` for the possessive pronoun *her* in Figure 5.2 (a). Besides the TnT tagger, there are several alternative solutions available in TectoMT for tagging English sentences now: (a) Aaron Coburn's tagger `Lingua::EN::Tagger` Perl module,[2] (b) Adwait Ratnaparkhi's MxPost tagger,[3] and (c) Morce tagger arranged for English, [Spoustová et al., 2007].[4]

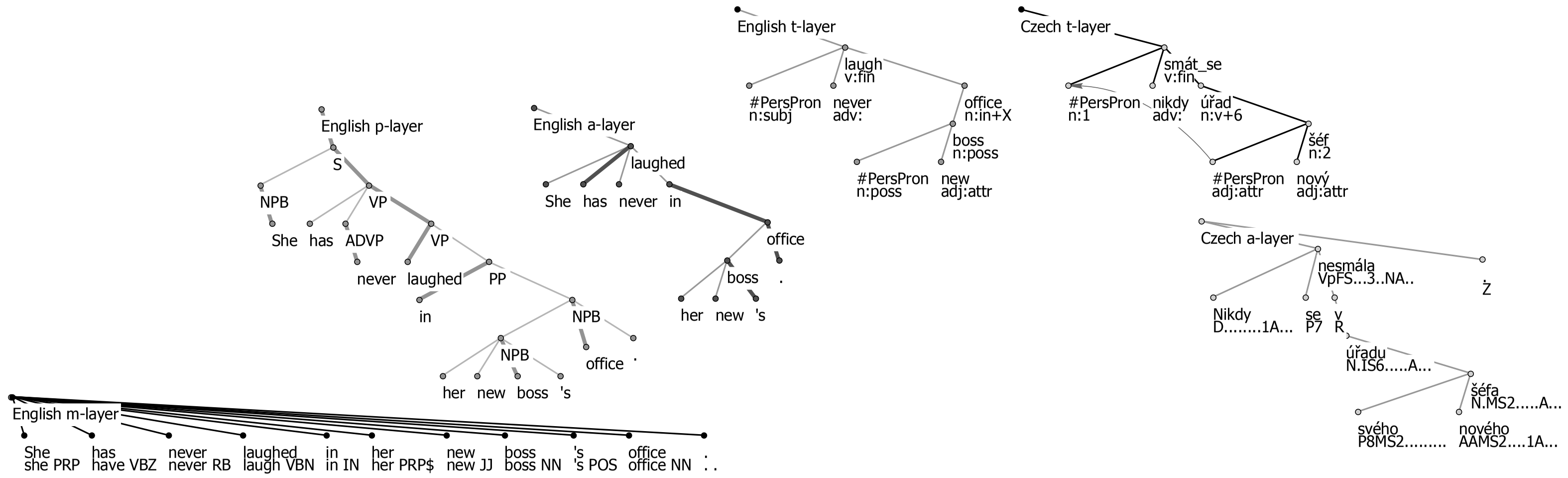
B4: Some tagging errors systematically made by the TnT tagger are fixed using a rule-

[1] http://www.cis.upenn.edu/ treebank/tokenizer.sed

[2] http://search.cpan.org/dist/Lingua-EN-Tagger/Tagger.pm

[3] http://www.inf.ed.ac.uk/resources/nlp/local_doc/MXPOST.html

[4] http://ufal.mff.cuni.cz/compost/



(a) (b) (c) (d) (e) (f)

Figure 5.2: Intermediate representations of the sentence *She has never laughed in her new boss's office.*) during its translation in TectoMT. The resulting Czech sentence is (*Nikdy se nesmála v úřadu svého nového šéfa.*).

based corrector.

B5: The tokens are lemmatized using `morpha` ([Minnen et al., 2000]) in combination with numerous hand-written rules and several lists of exceptions.[5] See e.g. the lemma *laugh* under the word form *laughed* in Figure 5.2 (a).

5.1.2 From SEnglishM to SEnglishP

B6: A PTB-style phrase-structure tree is built for each analyzed sentence using a parser [Collins, 1999] with Ken William's Perl interface `Lingua::CollinsParser`.[6] See the phrase-structure tree resulting from the sample sentence in Figure 5.2 (b).

5.1.3 From SEnglishP to SEnglishA

B7: In each phrase, the head node is marked. A hand-crafted ordered set of heuristic rules is used. In Figure 5.2 (b), the head child is distinguished in each phrase by the thick edge leading to it from its parent. For example, *boss* is the head of the noun phrase *her new boss's*.

B8: The phrase-structure trees are converted to (initial versions of) a-trees. Once the heads of all phrases are marked, a recursive procedure for transforming the phrase-structure subtrees into dependency subtrees can be used.[7] A very similar transformation is described in [Žabokrtský and Smrž, 2003] and [Žabokrtský and Kučerová, 2002] (Sections 8.3 and 8.2 in this work).

B9: Heuristic rules are applied for fixing apposition constructions.

B10: Another heuristic rules are applied for reattaching incorrectly positioned nodes. This postprocessing is necessary, because (as it was recently shown also in [Smrž et al., 2008]) the above mentioned algorithm for collapsing the phrase-tree head edges into dependency-tree nodes is not sufficient for all syntactic phenomena.

B11: The way in which multi-token prepositions (such as *because of*) and subordinating conjunctions (such as *provided that*) are treated is unified. We treat them in a way analogous to the guidelines for the Czech analytical layer of PDT: a canonically selected token of the "multi-token functional word" becomes the head and the other one(s) is (are) attached below it, being marked with a special analytical function value AuxP.

B12: Analytical functions are assigned where possible (especially those which are necessary for a correct treatment of coordination/apposition constructions).

[5] Another English lemmatizer which is faster and more accurate than `morpha` (and does not require additional postprocessing) has been recently implemented for TectoMT purposes by Martin Popel, see [Popel, 2009].

[6] http://search.cpan.org/~kwilliams/Lingua-CollinsParser-0.05/lib/Lingua/CollinsParser.pm

[7] Exploring the relationship between constituency and dependency sentence representation is not a new issue—the first studies go back to the 1960's ([Gaifman, 1965]).

5.1.4 From SEnglishA to SEnglishT

B13: Auxiliary a-nodes are marked (such as prepositions, subordinating conjunctions, auxiliary verbs, selected types of particles, etc.). In Figure 5.2 (c), the auxiliary tokens are connected to the autosemantic ones by thick arrows. Note that in some case the functional word is the a-tree parent of the autosemantic one (e.g. the preposition *in* in the figure), whereas in other cases it is the autosemantic word's child (auxiliary *has* or the Saxon genitive token in the figure).

B14: Token *not* is marked as an auxiliary node (but only if it is connected to a verb form).[8]

B15: Initial t-trees are built. Each a-node cluster formed by an autosemantic node and possibly several associated auxiliary nodes (i.e., nodes connected by the "thick" edges) is collapsed into a single t-node. T-tree dependency edges are derived from a-tree edges connecting the a-node clusters, as illustrated in Figure 5.2 (d): one t-node is created from the complex verb form *has laughed*, another t-node is created from the prepositional group *in office* (incorrectly connected also with the final full-stop), and there is an edge between the two t-nodes, originally corresponding to the edge between *laughed* and *in* a-nodes.

B16: T-nodes that are members of coordination (conjuncts) are distinguished from shared modifiers. It is necessary, as they all are attached below the coordination conjunction t-node, according to the PDT guidelines. For example, given the expression *fresh bananas and oranges*, t-nodes corresponding *fresh*, *bananas*, and *oranges* will all be attached below the coordination t-node *and*, but obviously the position of the shared modifier *fresh* must be somehow distinguished from the two conjunct positions.

B17: T-lemmas are modified in several specific cases. E.g., all kinds of personal pronouns are represented by the artificial t-lemma #PersPron (see the left child of the main predicate in Figure 5.2 (d)), which is equipped with grammatemes representing person, gender, and number categories.

B18: Functors are assigned that are necessary for a proper treatment of coordination and apposition constructions (e.g. CONJ for conjunction, DISJ for disjunction, ADVS for adversative) and some others (see our study on functor assignment in Section 7.1).

B19: Shared auxiliary words are distributed to all conjuncts in coordination constructions. For example, given the sentence *She is waiting for John and Mark*, t-nodes representing *John* and *Mark* should both refer also to the a-node representing the preposition *for*.

B20: T-nodes that are roots of t-subtrees corresponding to finite verb clauses are marked.

[8] Our treatment of verb negations differs from the approach in PDT 2.0, in which verb negation is represented as a separated t-node with a rhematizer function, whereas negation of nouns, adjectives and adverbs is represented using a special grammateme. For the purpose of MT, we find it more practical to represent negation of the four basic parts of speech by a grammateme.

In the sample sentence, only the *laugh* t-node is marked.

B21: Passive verb forms are marked.

B22: T-nodes which are roots of t-subtrees corresponding to relative clauses are marked.

B23: Coreference links between relative pronouns (or other relative pronominal word) and their nominal antecedents are identified. This will be important later after the transfer because of the required morphological gender and number agreement on the target language side.

B24: T-nodes that are the roots of t-subtrees corresponding to direct speeches are marked.

B25: T-nodes that are the roots of t-subtrees corresponding to parenthesized expressions are marked.

B26: The nodetype attribute – rough classification of t-nodes (see Section 7.2) – is filled.

B27: The sempos attribute (fine-grained classification of t-nodes) is filled.

B28: The grammateme attributes (semantically indispensable morphological categories, such as number for nouns, tense for verbs) are filled.

B29: The formeme (as introduced in Chapter 3) is determined of each t-node.

B30: Personal names are marked, and male and female first names are distinguished if possible.

5.1.5 From SEnglishT to TCzechT

B31: The target-side t-trees are initiated, simply by cloning the source-side t-trees (i.e., creating the t-tree by making a copying the a-tree topology).

B32: In each t-node, its formeme is translated (the formeme translation has been described in Section 3.5). Translated formemes are visible in 5.2 (e).

B33: T-lemma in each t-node is translated as its most probable target-language counterpart (which is compliant with the already chosen formeme), according to a probabilistic dictionary. The dictionary was created by merging the translation dictionary from PCEDT ([Cuřín et al., 2004]) and a translation dictionary extracted from part of the parallel corpus CzEng (see Section 10.2) aligned at word-level by Giza++ ([Och and Ney, 2003]).

B34: Manual rules are applied for fixing the formeme and lexeme choices, which are otherwise systematically wrong because of the simplifications in the previous steps.

B35: Fill the gender grammateme in t-nodes corresponding to denotative nouns, which becomes important in one of the later steps aimed at resolving grammatical agreement.

The gender value can in most cases be uniquely determined using only the tectogrammatical lemma attribute.[9]

B36: The aspect grammateme is filled in t-nodes corresponding to verbs. Information about aspect (perfective/imperfective) is necessary for making decisions about forming complex future tense in Czech (auxiliary *být* is used only for imperfectives).

B37: Rule-based correction of translated date/time expressions is applied (several templates such as *1970's*, *July 1*, etc.).

B38: Grammateme values in places where the English-Czech grammateme correspondence is not trivial, are fixed (e.g., if an English gerund expression is translated using Czech subordinating clause and thus the tense grammateme has to be filled).

B39: Verb forms are negated where some arguments of the verbs bear negative meaning, because of double negation in Czech. Note that there must be a negated verb form (*nesmála – did not laugh*) in the translation of the sample sentence because of the presence of a negative adverb among its children (*nikdy – never*).

B40: Verb t-nodes in active voice that have transitive t-lemma and no accusative object, are turned to reflexives (this is only a very rough heuristics, however it is worth doing, as its accuracy is above 50%).

B41: The t-nodes with genitive formeme or prepositional-group formeme, whose counterpart English t-nodes are located in front of the governing node, are moved to postmodification position. For example, *Prague map* goes to *mapa Prahy*.

B42: The dependency orientation between numeric expressions and counted nouns is reversed if the value of the numeric expression is greater than four and the noun without the numeral would be expressed in nominative or accusative case. For example: *Viděl jsem dvě děti – I saw* two_{acc} $kids_{acc}$, but *Viděl jsem pět dětí – I saw* $five_{acc}$ $kids_{gen}$.

B43: Coreference links from personal pronouns to their antecedents are identified, if the latter are in a subject position. This is important later for reflexivization: the presence of the coreference link in Figure 5.2 (e) causes the possessive reflexive pronoun *svého* (his_{refl}) to be chosen later in the SCzechA tree, and not the possessive pronoun *jeho* (*his*), which would be in this context incorrect. One of the possible approaches to resolution of pronominal anaphora is described in Section 7.3.

5.1.6 From TCzechT to TCzechA

B44: Initial a-trees is created by cloning t-trees (again, the tree topology is simply copied).

B45: The surface morphological categories are filled (gender, number, case, negation,

[9] This fact indicates that the presence of the gender attribute with denotative nouns in the PDT 2.0 is redundant. The same holds for the aspect attribute with verbs.

etc.) with values derived from the values of grammatemes, formemes, semantic parts of speech etc. In each a-node, the values of the categories are concatenated into a string (shown in 5.2 (f) as a node label) which later functions as regular expression filter for choosing the appropriate morphological tag in each a-node.

B46: The values of gender and number of relative pronouns are propagated from their antecedents (along the coreference links).

B47: The values of gender, number and person are propagated according to the subject-predicate agreement (i.e., subjects with finite verbs). In our example, feminine gender and singular number are propagated from the personal pronoun to the verb *smát se*.

B48: Agreement of adjectivals in attributive positions is resolved (copying gender, number, and case from their governing nouns). In our example, masculine gender, singular number, and genitive case are propagated to the two child nodes of the word *šéf*.

B49: Complement agreement is resolved (copying gender/number from subject to adjectival complement).

B50: Pro-drop is applied – deletion of personal pronouns in subject positions. In our example, the a-node corresponding to the subject of the verb *smát se* disappears from the a-tree.

B51: Preposition a-nodes are added (if implied by the t-node's formeme). The a-node bearing the preposition *v* is added above the noun *šéf*.

B52: A-nodes for subordinating conjunction are added (if implied by the t-node's formeme).

B53: A-nodes corresponding to reflexive particles are added for reflexive tantum verbs. A-node *se* now appears below the main verb.

B54: An a-node representing the auxiliary verb *být* (to be) is added in the case of compound passive verb forms, as would be needed e.g. in the expression *byl spatřen* (*(he) was seen*).

B55: A-nodes representing modal verbs are added, accordingly to the deontic modality grammateme, as would be needed e.g. in the expression *může to udělat* (*(he/she) can do it*).

B56: The auxiliary verb *být* is added in imperfective future-tense complex verb forms, as would be needed e.g. in the expression *budu zpívat* (*I will sing*).

B57: Verb forms such as *by/bys/bychom* expressing conditional verb modality are added, according to the value of grammateme verbmod, as would be needed e.g. in the expression *přišel by* (*he would come*).

B58: Auxiliary verb forms such as *jsem/jste* are added into past-tense complex verb forms whose subject is first or second person, as would be needed e.g. in the expression *spal* *jsem* (*I slept*).

B59: A-trees are partitioned into finite clauses: a-nodes belonging to the same clause are coindexed using a new technical attribute. In our example there is only one clause.

B60: In each clause, a-nodes which represent clitics are moved to the so-called second position in the clause (according to Wackernagel's law).[10]

B61: A-nodes corresponding to sentence-final punctuation mark are added.

B62: A-nodes corresponding to commas on boundaries between governing and subordinated clauses are added.

B63: A-nodes corresponding to commas in front of the conjunction *ale* and also commas in multiple coordinations are added.

B64: Pairs of parenthesis a-nodes are added if they appeared in the source language sentence.

B65: Morphological lemmas are chosen in a-nodes corresponding to personal pronouns (this can be done using a simple table).

B66: Resulting word forms are generated (derived from lemmas and tags) using the Czech word form generator described in [Hajič, 2004][11]. If more than one word form is allowed by a combination of the lemma with the regular expression filter on the morphological tags, then the form which is the most frequent in the Czech National Corpus is selected.

B67: Prepositions *k*, *s*, *v*, and *z* are vocalized accordingly to the prefix of the following word. We implemented a relatively straightforward solution based on a list of prefixes extracted from the Czech National Corpus. Other approaches based on hand-crafted rules adapted from [Petkevič and Skoumalová, 1995] or on automatically acquired decision trees are mentioned and evaluated in [Ptáček, 2008].

B68: The first word in each sentence is capitalized as well as in each direct speech.

5.1.7 From TCzechA to TCzechW

B69: The resulting sentences are created by flattening the a-trees. Heuristic rules for proper spacing around punctuation marks are used.

B70: The resulting text is created simply by concatenating the resulting sentences with spaces in between.

[10] At this point we plan to add another block which will sort the clitics if more than one appear. In Czech, auxiliary forms of the verb *být* (*to be*) such as *jsem*, *budu* or *bych* go first, then short forms of reflexive pronouns (or reflexive particles) follow, then short forms of pronouns in dative, and finally short forms of pronouns in accusative.

[11] Perl interface to this generator has been implemented by Jan Ptáček.

5.2 Employed Resources of Linguistic Data

In the following list we give a summary of the resources of linguistic data whose existence was—directly or indirectly (e.g. in the form of probabilistic models of previously existing NLP components trained from the data)—important for the above described version of our translation system.

- It was necessary to use Penn Treebank [Marcus et al., 1994] to train English taggers and parsers.
- British National Corpus[12] was used for improving English lemmatization.
- Czech National Corpus [cnk, 2005] was used for creating frequency lists of Czech word forms and lemmas, and also for extracting prefixes causing vocalization of prepositions.
- Prague Dependency Treebank 2.0 [Hajič et al., 2006] was used for training Czech taggers and parsers.
- Parallel sentences from the Shared Task of Workshop in Statistical Machine Translation were used for extracting formeme translation dictionary.
- Czech-English parallel corpus CzEng (Section 10.2) was used for improving English-Czech translation dictionary.
- Parallel Czech and English sentences manually aligned on the word layer collected in [Mareček, 2008] were used for training the perceptron-based t-tree aligner.
- Valency lexicon of Czech verbs VALLEX [Lopatková et al., 2008] was used for gathering lists of verbs with specific properties (such as verbs having actants in genitive case or in infinitive form).
- Probabilistic dictionary developed in [Cuřín, 2006] was used as one of the sources of English-Czech translation entries.

[12] http://www.natcorp.ox.ac.uk

Chapter 6

Conclusions and Final Remarks

We presented a new Machine Translation system employing the layered annotation scenario of the Prague Dependency Treebank. The system makes use of numerous existing resources of linguistic data as well as of existing NLP tools, but many new software components had to be implemented, too. At present, the system fully functions. Its translation quality was evaluated within the Shared Task of the Workshop in Statistical Machine Translation, see [Callison-Burch et al., 2008] and [Callison-Burch et al., 2009]. It does not outperform the state-of-the-art systems; however, there is still space for improvements, especially we plan to focus on the transfer phase using information from the target-side language model. The first promising experiments in this direction are described in [Žabokrtský and Popel, 2009] (Section 10.3).

Besides implementing the MT system itself, the second goal of developing TectoMT was to facilitate integration of various NLP components, share them in various applications, and to support cooperation in general. In our opinion, this goal has been fully achieved: a number of NLP components are already integrated in it, such as four taggers for English, two constituency parsers for English, two dependency parser for English, three taggers for Czech, two dependency parsers for Czech (one of them described in Section 8.1), a named entity recognizer for English, and two named entity recognizers for Czech. New components are still being added, as there are more than ten programmers contributing to the TectoMT repository at present. Besides developing the English-Czech translation scenario described in this work, TectoMT was also used for several other MT-related experiments, such as:

- MT based on Synchronous Tree Substitution Grammars and factored translation [Bojar and Hajič, 2008],
- aligning tectogrammatical representations of parallel Czech and English sentences, [Mareček et al., 2008],
- building a large, automatically annotated parallel English-Czech treebank CzEng 0.9 [Bojar and Žabokrtský, 2009],
- compiling a probabilistic English-Czech translation dictionary [Rouš, 2009],
- evaluating metrics for measuring translation quality [Kos and Bojar, 2009],

TectoMT was also used for several other purposes not directly related to MT. For example, TectoMT was used for

- complex pre-annotation of English tectogrammatical trees within the Prague Czech English Dependency Treebank project [Hajič et al., 2009b],
- tagging the Czech data set for the CoNLL Shared Task [Hajič et al., 2009a],
- gaining syntax-based features for prosody prediction [Romportl, 2008],
- experiments on information retrieval [Kravalová, 2009],
- experiments on named entity recognition [Kravalová and Žabokrtský, 2009],
- conversion between different deep-syntactic representations of Russian sentences [Mareček and Kljueva, 2009].

Part II

Selected Publications

Chapter 7

Annotating Prague Dependency Treebank

7.1 Automatic Functor Assignment in the Prague Dependency Treebank.

Full reference:

Zdeněk Žabokrtský: Automatic Functor Assignment in the Prague Dependency Treebank, In TSD2000, Proceedings of Text, Speech and Dialogue (eds. P. Sojka, I. Kopeček, K. Pala). Springer-Verlag Berlin Heidelberg. pp. 45–50. 2000.

Comments:

This paper presented our attempt at automatizing part of the transition from analytical to tectogrammatical trees within the PDT project, namely, assigning functors to autosemantic words. The aim was to save part of the experts' work and make the annotation process faster. As described in [Žabokrtský et al., 2002], the performance of this tool was later improved by putting more emphasis on using Machine Learning and by employing additional sources of linguistic data. The tool was incorporated into the tree editor Tred and used by the annotators for preprocessing tectogrammatical structures from 2001 to 2004. Later we also developed several modifications of the tool, for example, to assign analytical functions in Czech and Arabic analytical trees; in the latter case, the assigner has been used by the annotators of the Prague Arabic Dependency Treebank [Hajič et al., 2004].

Nowadays, our tool for assigning functors is outperformed by the system described in [Klimeš, 2006], the Czech and English version of which are integrated in the TectoMT framework.

Automatic Functor Assignment in the Prague Dependency Treebank ⋆

Zdeněk Žabokrtský

Czech Technical University, Department of Computer Science
121 35 Praha 2, Karlovo nám. 13, Czech Republic
zabokrtz@cs.felk.cvut.cz

Abstract. This paper presents work in progress, the goal of which is to develop a module for automatic transition from analytic tree structures to tectogrammatical tree structures within the Prague Dependency Treebank project. Several rule-based and dictionary-based methods were combined in order to be able to make maximal use of both information extractable from the training set and a priori knowledge. The implementation of this approach was verified on a testing set, and a detailed evaluation of the results achieved so far is presented.

1 Introduction

The process of syntactic tagging in the Prague Dependency Treebank (PDT) is divided into two steps: The first step results in *analytic tree structures* (ATS), in which every word form and punctuation mark is explicitly represented as a node of rooted tree, with no additional nodes added (except for the root of the tree of every sentence). The second step results in *tectogrammatical tree structures* (TGTS), which approximate the underlying sentence representations according to [4]. In contrast to the ATSs, only autosemantic words have nodes of their own in TGTSs, informations about functional words (prepositions, subordinating conjunctions etc.) are contained in the tags attached to the autosemantics nodes. Figure 1 depicts an example of a TGTS.

Besides slight changes in the topology of the input ATS (for instance, pruning of synsemantic nodes), the transition from ATSs to TGTSs involves the assignment of the tectogramatical function (*functor*) to every node in the tree. There are roughly 60 functors divided into two subroups (cf. [4]): (i) *actants* (ACTor, PATient, ADDRessee, EFFect, ORIGin) and (ii) *free modifiers*: TWHEN (time-when), LOCaction, MEANS, EXTent, BENeficiary, ATTribute ...).

Presently, the topological conversion and the assignment of a few functors (e.g., ACT, PAR, PRED) are solved automatically by the procedure of Böhmová et al. [1]. However, most of the functors have to be assigned manually. The

⋆ I would like to thank my advisor Ivana Kruijff-Korbayová for her permanent support. I am also indebted to Alevtina Bémová for consultations about functor assignment, to Petr Pajas for the help with the data preprocessing and to Julia Hockenmaier for her useful comments.

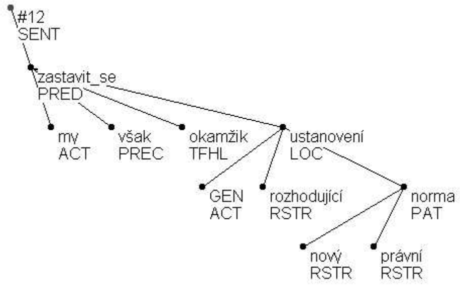


Fig. 1. TGTS of the sentence *Zastavme se však na okamžik u rozhodujících ustanovení nové právní normy.* (Let's however stop for a moment at the most important paragraphs of the new legal norm.)

amount of labor involved in the manual annotation obviously slows down the growth of the PDT on the tectogrammatical level. Decreasing the amount of manual annotation has been the motivation for developing the more complex *automatic functor assignment system* (AFA) presented in this paper. Let us describe the starting position.

- No general unambiguous rules for functor assignment are known, human annotators use mostly only their language experience and intuition. We cannot reach 100% correctness of AFA since even the results of individual annotators sometimes differ.
- The annotators usually use the whole sentence context for their decision. It has not been measured how often it is really unavoidable to take the full context into account or how large the context must be.
- Preliminary measurements revealed that the distribution of functors is very non-uniform. The 15 most frequent functors cover roughly 90% of nodes. Conversely, there are hardly any examples for the least frequent functors.
- It would be very time consuming to test the performance AFA on randomly selected ATSs and find errors manually. Fortunately we can use the ATSs for which manually created TGTSs are already avaliable, annotate them automatically and compare the results against the manually annotated TGTSs.
- The available TGTSs contain imperfect data. Some errors are inherited from ATSs, and functor assignments are in some cases ambiguous (nodes with more than one functor) or incomplete (some nodes have no functor yet).

2 Materials

Training and Testing Sets When I started working on AFA, 18 TGTS files were available, each containing up to 50 sentences from newspaper articles. This was a sufficient amount of data for mining knowledge, which can improve the AFA's performance. But in order to reliably measure AFA's correctness, it is necessary to have a separate data set not used for knowledge mining. Therefore I randomly selected 15 files for the training set and 3 files for the testing set. After removing incomplete and ambiguously assigned nodes, the training set contained 6049 annotated nodes, and the testing set 1089 annotated nodes.

Data Preprocessing Neither the maximum degree of a node (i.e., the number of outgoing edges) nor the depth of a tree are limited in TGTSs. The trees thus can be very complex, and working with the whole tree context of the individual nodes would make AFA unnecessarily complicated. For the sake of the experiments described here, I assumed that reasonable correctness can be achieved using only information about the node to be annotated and about its governing node (i.e., about the edge in the tree). So the first step of the preprocessing was the *transformation from the tree structure into the list of edges.*

Each node in PDT can have tens of attributes, majority of them beeing useless for AFA. Hence, a selection of the relevant attributes is performed next (*feature selection*). I chose the following set: word form, lemma, full morphological tag and analytical function of both the governing and dependent node, preposition or conjunction which binds the governing and the dependent node, and the functor of the dependent node.

In order to make the subsequent processing easier, 3 additional simple attributes (the parts of speech of both nodes, the morphological case of the dep. node) were extracted from these 10 attributes (*feature extraction*). Finally, each accented character has been substituted with the corresponding ASCII character followed by "_". Having a vector of 13 symbolic attributes, the task of AFA can be now formulated as the *classification of the symbolic vectors into 60 classes.*

3 Implementation

The AFA system has been designed as a collection of small programs written mostly in Perl. Each method of functor assignment forms a separate program (script), the data to be assigned goes through a sequence of these scripts in a pipeline fashion. Each method can assign only those nodes which have not been assigned by any of the previous scripts yet. This approach enables flexible tuning of the AFA characteristics (precision, recall) simply by reordering or removing the individual methods. This advantage would be lost in the case of one compact complicated program.

Rule-based Methods (RBM) The RBMs consist of simple hand written decision trees. They use no external data and therefore are independent of the quality of the training set. They do not bring any new information into the PDT, only transform the information contained in an ATS. Currently I have 7 methods with reasonable precision:

1. `verbs_active`: if the governing node is a verb in active form then
   - if the analytical function (afun) is subject, then the node is assigned the functor ACT (→ ACT)

   - if afun is object and case is dative then → ADDR
   - if afun is object and case is accusative then → PAT
2. `verbs_passive`: if the governing node is a verb in passive form:
   - if afun is subject then → PAT
   - if afun is object and case is dative then → ADDR
   - if afun is object and case is instrumental then → ACT
3. `adjectives`: if the node corresponds to an adjective
   - if it is a possessive adjective then → RSTR
   - else → RSTR
4. `pronounposs`: if the node is a possessive pronoun then → APP
5. `numerals`: if the node is a numeral then → RSTR
6. `pnom`: if afun is PNOM then → PAT
7. `pred`: if afun is PRED then → PRED

Dictionary-based Methods (DBM) It is not feasible to resolve all the remaining unassigned functors using only simple RBMs like those above, since we could not profit from the growing volume and diversity of the training set.

I have so far developed four methods using different types of dictionaries:

- `adverbs`: The couples *adverb–functor* were automatically extracted from the training set, and added to the list of adverbs from [2]; from the combined list, the *unambiguous* (accompanied always with the same functor) adverbs were extracted. Such a dictionary can be used to assign functors to adverbs. Examples from the dictionary: *výlučně* (exclusively) RHEM, *výrazně* (extensively) EXT
- `subconj`: A dictionary of unambiguous *subordinative conjunctions* was constructed in the same way as the dictionary of adverbs. If a verb is related to its governing node by one of these conjunctions, the functor can be easily assigned.
  Examples from the dictionary: *i když* (even when) CNCS,
  jelikož (because) CAUS, *jen co* (as soon as) TWHEN, *jestli* (if) COND
- `prepnoun`: All the *preposition–noun* pairs (a preposition followed by a noun) were extracted from the training set. The unambiguous couples which occured at least twice were inserted into the dictionary.
  Examples from the dictionary: *v roce* (in year) TWHEN, *pro podnikatele* (for businessman) BEN, *od doby* (from time) TSIN, *z odvětví* (from branch) DIR1, *v zemích* (in countries) LOC
- `similarity`: The dictionary is formed by the entire training set. The functor of the most similar vector found in the training set is used for assignment. The (in)equality of individual attributes has different impact (weight) on the similarity function, e.g., the part of speech is more important than the lemma. The weights were determined experimentally. Example: for *zálohy na daně* (pre-payments of taxes), where the dependent node *daně* (taxes) is to be assigned a functor, the most similar record found is *návrh na stanovení* (proposal of determination), so the functor PAT of the dependent node is used.

4 Results

Testing Set The testing set was not used in the mining of knowledge (dictionaries), therefore we can apply both rule-base and dictionary method on it. For each method, six quantitative characteristics have been determined (Table 1):

- *Cover* = the number of all nodes assigned by the given method
- *Relative cover* = cover divided by number of all functors to be assigned (1089 in the training set). This number also reflects the frequency of several phenomenona (e.g., possessive pronouns).
- *Errors* = the number of incorrectly assigned functors
- *Hits* = the number of correctly assigned functors
- *Recall* = the percentage of correct functor assignments by the given method among all functors to be assigned (hit/1089·100%)
- *Precision* = the percentage of of correct functor assignments by the given method among all functors assigned by this method (hits/cover·100%)

Method	Cover	Rel. cover	Hits	Recall	Errors	Precision
pred	104	9.5 %	104	9.6 %	0	100 %
verbs_active	199	18.3 %	184	16.9 %	15	92.5 %
verbs_passive	7	0.6 %	6	0.6 %	1	85.7 %
pnom	34	3.1 %	32	2.9 %	2	94.1 %
adjectives	177	16.3 %	170	15.6 %	7	96.0 %
numerals	21	1.9 %	15	1.4 %	6	71.4 %
pronounpos	16	1.5 %	13	1.2 %	3	81.3 %
subconj	3	0.3 %	2	0.2 %	1	66.7 %
adverbs	34	3.1 %	30	2.8 %	4	88.2 %
prepnoun	9	0.8 %	9	0.8 %	0	100 %
similarity	485	44.5 %	287	26.4 %	198	59.2 %
Total	Σ=1089	Σ=100 %	Σ=852	Σ=78.2 %	Σ=237	78.2 %

Table 1. Results of AFA on the testing set

The methods in Table 1 are sorted in the same order as they were executed. This order permutation reaches the highest precision. The `similarity` method is handicapped by the fact that all easily solvable cases are assigned by its predecessors. If we use `similarity` alone, we get Recall=Precision=73%.

I believe that the results of this first implementation are satisfactory, I had expected lower overall precision. However, I cannot compare it to anything else, since there is no other AFA implementation with comparable recall within the PDT project.

Rule-based Methods on Training Set In order to verify the precision of RBMs, we can apply them on the training set as well (see Table 2). Note that the size of the training set is 6049 nodes.

Method	Cover	Rel. cover	Hits	Recall	Errors	Precision
pred	574	9.5 %	554	9.2 %	20	96.5 %
verbs_active	973	16.1 %	907	15.0 %	66	93.2 %
verbs_passive	34	0.6 %	27	0.4 %	7	79.4 %
pnom	164	2.7 %	152	2.5 %	12	92.7 %
adjectives	1063	17.6 %	976	16.1 %	87	91.8 %
numerals	92	1.5 %	66	1.1 %	26	71.7 %
pronounpos	64	1.1 %	61	1.0 %	3	95.3 %
Total	Σ=2964	Σ=49.0 %	Σ=2743	Σ=45.3 %	Σ=221	92.5 %

Table 2. Results of RBMs on the training set

Precision versus Recall We have to decide whether we prefer to *minimize the number or errors* (maximizing precision using only the methods with the best precision) or *maximize the number of correctly assigned nodes* (maximizing recall using all the methods with admissible precision). The optimal compromise should be influenced by the *misclassification cost* which can be estimated as an amount of annotators' work for finding and correcting incorrectly assigned functors.

5 Conclusion and Future Work

I implemented several methods for automatic functor assignment, tested them and evaluated their characteristics. Methods based on rules had higher precision than dictionary-based methods. The possibility of combining individual approaches opened the question whether we prefer to assign, e.g., 49 % of functors with 92 % precision or to assign everything with 78 % precision.

All the available TGTSs are from newspaper articles. The distance between the training set and the testing set is thus rather small. If AFA were to be applied to other than newspaper articles, it is likely that precision would be slightly lower.

As more manually annotated TGTSs become available, we can expect improvements of the dictionary-based methods. Moreover, it will hopefully be possible to discover some more rules for functor assignment using the machine learning system C4.5; we have so far obtained promising preliminary results.

References

1. Böhmová, A., Panevová, J., Sgall, P.: *Syntactic Tagging: Procedure for the Transition from the Analytic to the Tectogrammatical Tree Structures.* Text, Speech and Dialogue, Springer (1999)
2. Hajičová, E., Panevová, J., Sgall, P.: *Manuál pro tektogramatické značkování.* ÚFAL MFF UK (1999)
3. Panevová, J.: *Formy a funkce ve stavbě české věty.* Academia (1980)
4. Petr Sgall, Eva Hajičová, and Jarmila Panevová: *The Meaning of the Sentence in its Semantic and Pragmatic Aspects.* Reidel, Dordrecht, The Netherlands, 1986.

7.2 Annotation of Grammatemes in the Prague Dependency Treebank 2.0

Full reference:

Razímová Magda, Žabokrtský Zdeněk: Annotation of Grammatemes in the Prague Dependency Treebank 2.0, in Proceedings of the LREC Workshop on Annotation Science, ELRA, Genova, Italy, ISBN 2-9517408-2-4, pp. 12-19, 2006

Comments:

Grammatemes constitute an indispensable component of tectogrammatical sentence description—without them it would not be possible to translate correctly e.g. number of nouns or tense of verbs during the tectogrammatical transfer. The notion of grammatemes was roughly described in [Sgall et al., 1986], and then elaborated in more detail (including the set of values) in the initial version of tectogrammatical annotation guidelines, [Panevová et al., 2001]. We added a hierarchy of types of tectogrammatical nodes (published in [Razímová and Žabokrtský, 2005]) which allows formally ensuring the presence or absence of individual grammatemes with a given node, and enriching the set of grammatemes with new ones dedicated to pronominal and numerical expressions, as described in [Ševčíková-Razímová and Žabokrtský, 2006]. These modifications were also adopted by the new version of annotation guidelines [Mikulová et al., 2005] published with PDT 2.0.

Given tectogrammatical trees with manually corrected topology and functors, and also reliable annotation on analytical and morphological layers, it seemed to be possible for most of the grammateme attributes to be filled automatically with very high precision. Therefore we implemented a rule-based system for assigning the grammatemes, and used it for annotating the tectogrammatical data of PDT 2.0 (only a very small amount of manual annotation work was needed). This tool for automatic grammateme assignment was later incorporated into TectoMT and its English version was created too, which is now used in the English-Czech translation implemented in TectoMT.

Annotation of Grammatemes in the Prague Dependency Treebank 2.0

Magda Razímová, Zdeněk Žabokrtský

Institute of Formal and Applied Linguistics, Charles University, Prague
Malostranské náměstí 25, Prague 1, 118 00, Czech Republic
{razimova,zabokrtsky}@ufal.mff.cuni.cz

Abstract

In this paper we report our work on the system of grammatemes (mostly semantically-oriented counterparts of morphological categories such as number, degree of comparison, or tense), the concept of which was introduced in Functional Generative Description, and has been recently further elaborated in the layered annotation scenario of the Prague Dependency Treebank 2.0. We present also a hierarchical typology of tectogrammatical nodes, which is used as a formal means for ensuring presence or absence of respective grammatemes.

1. Introduction

Human language, as an extremely complex system, has to be described in a modular way. Many linguistic theories attempt to reach the modularity by decomposing language description into a set of layers, usually linearly ordered along an abstraction axis (from text/sound to semantics/pragmatics). One of the common features of such approaches is that word forms occurring in the original surface expression are substituted (for the sake of higher abstraction) with their lemmas at the higher layer(s). Obviously, the inflectional information contained in the word forms is not present in the lemmas. Some information is 'lost' deliberately and without any harm, since it is only imposed by government (such as case for nouns) or agreement (congruent categories such as person for verbs or gender for adjectives). However, the other part of the inflectional information (such as number for nouns, degree for adjectives or tense for verbs) is semantically indispensable and must be represented by some means, otherwise the sentence representation becomes deficient (naturally, the representations of sentence pairs such as '*Peter met his youngest brother*' and '*Peter meets his young brothers*' must not be identical at any level of abstraction). At the tectogrammatical layer of Functional Generative Description (FGD, (Sgall, 1967), (Sgall et al., 1986)), which we use as the theoretical basis of our work, these means are called grammatemes.[1]

The theoretical framework of FGD has been implemented in the Prague Dependency Treebank 2.0 project (PDT 2.0, (Hajičová et al., 2001)), which aims at a complex annotation of large amount of Czech newspaper texts. Although grammatemes are present in the FGD for decades, in the context of PDT they were paid for a long time a considerably less attention, compared e.g. to valency, topic-focus articulation, or coreference. However, in our opinion grammatemes will play a crucial role in NLP applications of FGD and PDT (e.g., machine translation is impossible without realizing the differences in the above pair of example sentences). That is why we decided to further elaborate the system of grammatemes and to implement it in the PDT 2.0 data. This paper outlines some of the results of more than two years of the work on this topic.

The paper is structured as follows: after introducing the basic properties of the PDT 2.0 with focus on the tectogrammatical layer in Section 2., we will describe the classification of t-layer nodes in Section 3., enumerate and exemplify the individual grammatemes and their values in Section 4. After outlining the basic facts about the (mostly automatic) annotation procedure in Section 5. we will add some final remarks in Section 6.

2. Sentence Representation in the Prague Dependency Treebank 2.0

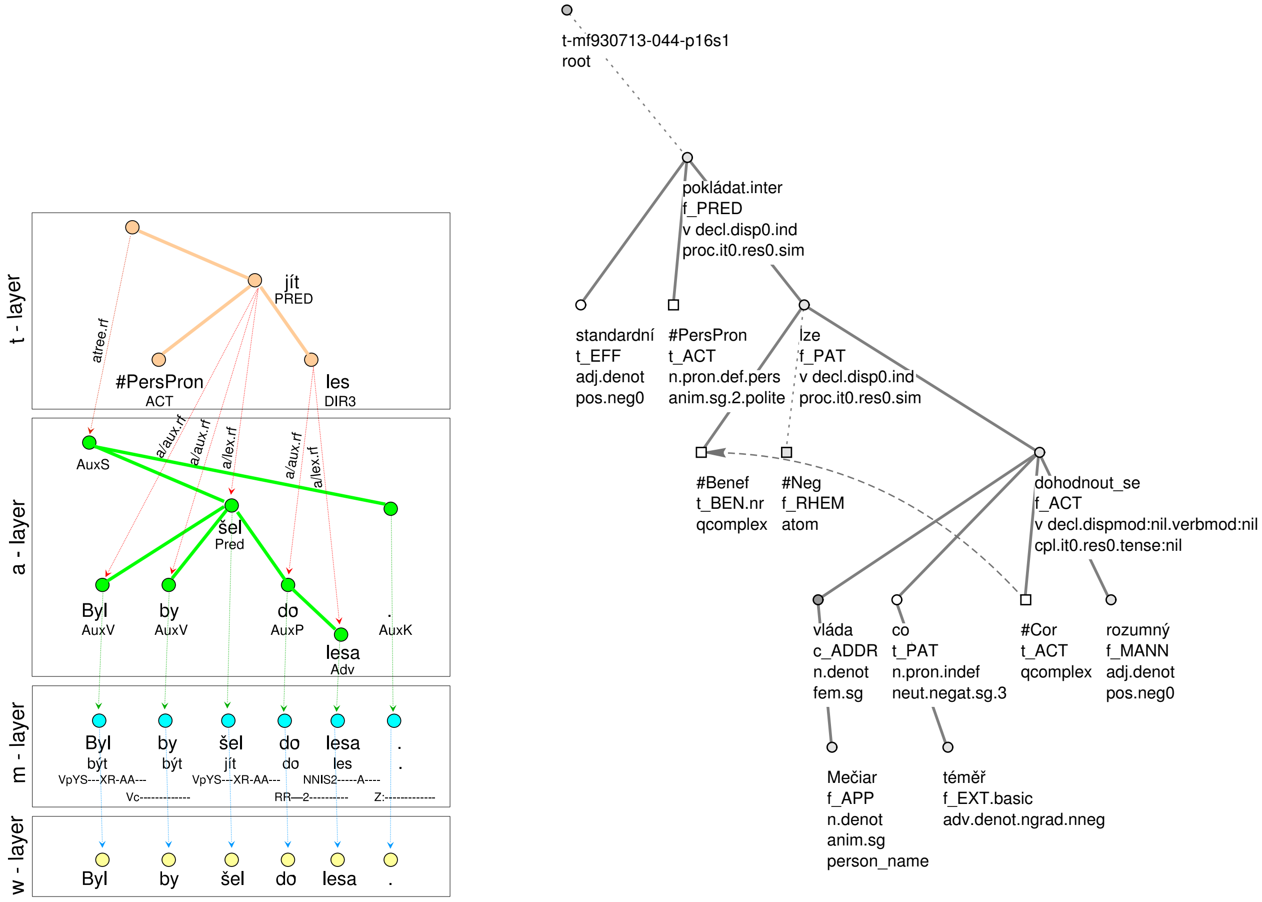
In the Prague Dependency Treebank annotation scenario, three layers of annotation are added to Czech sentences (see Figure 1 (a)):[2]

- morphological layer (m-layer), on which each token is lemmatized and POS-tagged,
- analytical layer (a-layer), on which a sentence is represented as a rooted ordered tree with labeled nodes and edges, corresponding to the surface-syntactic relations; one a-layer node corresponds to exactly one m-layer token,
- tectogrammatical layer (t-layer), which will be briefly described later in this section.

The full version of the PDT 2.0 data consists of 7,129 manually annotated textual documents, containing altogether 116,065 sentences with 1,960,657 tokens (word forms and punctuation marks). All these documents are annotated at the m-layer. 75 % of the m-layer data are annotated at the a-layer (5,338 documents, 87,980 sentences, 1,504,847 tokens). 59 % of the a-layer data are annotated also at the t-layer (i.e. 44 % of the m-layer data; 3,168 documents,

[1] Just for curiosity: almost the same term 'grammemes' is used for the same notion in the Meaning-Text Theory (Mel'čuk, 1988), although to a large extent the two approaches were created independently.

[2] Technically, there is also one more layer below these three layers which is called w-layer (word layer); on this layer the original raw-text is only segmented into documents, paragraphs and tokens and all these units are enriched with identifiers.



(a) (b)

Figure 1: (a) PDT 2.0 annotation layers (and the layer interlinking) illustrated (in a simplified fashion) on the sentence *Byl by šel do lesa.* ([He] would have gone into forest.), (b) tectogrammatical representation of the sentence: *Pokládáte za standardní, když se s Mečiarovou vládou nelze téměř na ničem rozumně dohodnout?* (Do you find it standard if almost nothing can be reasonably agreed on with Mečiar's government?)

49,442 sentences, 833,357 tokens).[3] The annotation at the t-layer started in 2000 and was divided into four areas:

a. building the dependency tree structure of the sentence including labeling of dependency relations and valency annotation,

b. topic / focus annotation,

c. annotation of coreference (i.e. relations between nodes referring to the same entity),

d. annotation of grammatemes and related attributes, the description of which is the main objective of this paper.

After the annotation of data had finished in 2004, an extensive cross-layer checking took over a year. The CD-ROM including the final annotation of PDT 2.0-data, a detailed documentation as well as software tools is to be publicly released by Linguistic Data Consortium in 2006.[4]

At the t-layer, the sentence is represented as a dependency tree structure built of nodes and edges (see Figure 1 (b)). Tectogrammatical nodes (t-nodes) represent auto-semantic words (including pronouns and numerals) while functional words such as prepositions have no node in the tree (with some exception of technical nature: e.g. coordinating conjunctions used for representation of coordination constructions are present in the tree structure). Each t-node is a complex data structure – it can be viewed as a set of attribute-value pairs, or even as a typed feature structure as used in unification grammars such as HPSG (Pollard and Sag, 1994).

For the purpose of our contribution, the most important attributes are the attribute t-lemma (tectogrammatical lemma), attribute functor, grammatemes and the classifying attributes nodetype and sempos. The annotation of attributes t-lemma and functor belongs to the area marked above as (a); these attributes will be introduced in the next paragraphs. Grammatemes and the attributes nodetype and sempos – all of them coming under the area (d) – will be characterized from the standpoint of annotation in Section 3. (The annotation of attributes belonging to the areas

[3] The previous version of the treebank, PDT 1.0, was smaller and contained only m-layer and a-layer annotation (Hajič et al., 2001).

[4] See `http://ufal.mff.cuni.cz/pdt2.0/`

(b) and (c) goes beyond the scope of this paper.)

The attribute t-lemma contains the lexical value of the t-node, or an 'artificial' lemma. The lexical value of the t-node is mostly a sequence of graphemes corresponding to the 'normalized' form of the represented word (i.e. infinitive for verbs or nominative form for nouns). In some cases, the t-lemma corresponds to the basic word from which the represented word was derived, e.g. in Figure 1 (b), the possessive adjective *Mečiarova* (*Mečiar's*) is represented by the t-lemma *Mečiar*, or the adverb *rozumně* (*reasonably*) is represented by the adjectival t-lemma *rozumný* (*reasonable*). The artificial t-lemma appears at t-nodes that have no counterpart in the surface sentence structure (e.g. the t-lemma #Gen at a verbal complementation not occurring in the surface structure because of its semantic generality), or it corresponds to personal pronouns, no matter whether expressed on the surface or not (e.g. the t-lemma #PersPron at the t-node in Figure 1 (b)). The dependency relation between the t-node in question and its parent t-node is stored in the attribute functor, e.g. functor EFF at the t-node with t-lemma *standardní* (*standard*), which plays the role of an effect of the predicate in the sentence displayed in Figure 1 (b).

3. Two-level Typing of Tectogrammatical Nodes

While the attributes t-lemma and functor are attached to each t-node of the tectogrammatical tree, grammatemes are relevant only for some of them. The reason for this difference consists in the fact that only some words represented by t-nodes bear morphological meanings.

3.1. Types of Tectogrammatical Nodes

To differentiate t-nodes that bear morphological meanings from those without such meanings, a classification of t-nodes was necessary. Based on the information captured by the above mentioned attributes t-lemma and functor, eight types of t-nodes were distinguished. The appurtenance of the t-node to one of the types is stored in the attribute nodetype.[5]

- **Complex nodes** (nodetype='complex') as the most important node type should be named in the first place: since they represent nouns, adjectives, verbs, adverbs and also pronouns and numerals (i.e. words expressing morphological meanings), they are the only ones with which grammatemes are to be assigned.

The other seven types of t-nodes and the corresponding values of the attribute nodetype are as follows:

- **The root of the tectogrammatical tree** (nodetype='root') is a technical t-node the child t-node of which is the governing t-node of the sentence structure.

- **Atomic nodes** (nodetype='atom') are t-nodes with functors RHEM, MOD etc. – they represent rhematizers, modal modifications etc.

- **Roots of coordination and apposition constructions** (nodetype='coap') contain the t-lemma of the coordinating conjunction or an artificial t-lemma of a punctuation symbol (e.g. #Comma).

- **Parts of foreign phrases** (nodetype='fphr') are components of phrases that do not follow rules of Czech grammar (labeled by a special functor FPHR in the tree).

- **Dependent parts of phrasemes** (nodetype='dphr') represent words that constitute a single lexical unit with their parent t-node (labeled by a special functor DPHR in the tree); the meaning of this unit does not follow from the meanings of its component parts.

- **Roots of foreign and identification phrases** (nodetype='list') are nodes with special artificial t-lemmas (#Forn and #Idph), which play the role of a parent of a foreign phrase (i.e. of nodes with nodetype='fphr' – see above) or the role of a parent of a phrase having a function of a proper name.

- So called **quasi-complex nodes** (nodetype= 'qcomplex') stand mostly for obligatory verbal complementations that are not present in the surface sentence structure (i.e. they have the same functors as complex nodes but, unlike them, quasi-complex t-nodes have artificial t-lemmas, e.g. #Gen).

3.2. Semantic Parts of Speech

Not all morphological meanings (chosen as tectogrammatically pertinent) are relevant for all complex t-nodes (cf., for example, the category of tense at nouns or the degree of comparison at verbs). As we did not want to introduce any 'negative' value to identify the non-presence of the given morphological meaning at a t-node (i.e., if all grammatemes would be annotated at each complex t-node, the negative value would be filled in at the irrelevant ones), the attribute sempos for sorting the t-nodes according to morphological meanings they bear had to be introduced into the attribute system.

The groups into which the complex t-nodes were further divided are called semantic parts of speech. According to basic onomasiological categories of substance, quality, event and circumstance (Dokulil, 1962), four semantic parts of speech were distinguished: semantic nouns, semantic adjectives, semantic verbs and semantic adverbs. These groups are not identical with the 'traditional' parts of speech: while ten traditional parts of speech are discerned in Czech and the appurtenance of the word to one of them is captured by a morphological tag (i.e. by an attribute of m-layer in the PDT 2.0), the 'only' four semantic parts of speech are categories of the t-layer and are captured by the attribute sempos (values n, adj, v and adv). The relations between semantic and traditional parts of speech are demonstrated in Figure 2. We would like to illustrate them on the example of semantic adjectives in more detail.

The following groups traditionally belonging to different parts of speech count among the semantic adjectives: (i) traditional adjectives, (ii) deadjectival adverbs, (iii) adjectival pronouns, and (iv) adjectival numerals.

[5] Some of the nodetype values are present in Figure 1 (b). If none of the nodetype values is indicated with the t-node, the nodetype is 'complex'.

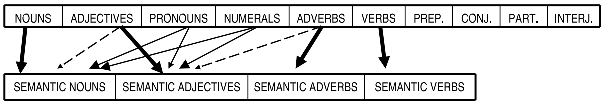


Figure 2: Relations of traditional parts of speech to their semantic counterparts. Arrows in bold denote a prototypical relation, thin arrows indicate the distribution of pronouns and numerals into semantic parts of speech and dotted arrows stand for the classification according to derivational relations.

(i) Traditional adjectives, e.g. *standardní* (*standard*) in Figure 1 (b), are mostly regarded as semantic adjectives (with the already mentioned exception of possessive adjectives converted to nouns).
(ii) At the t-layer, deadjectival adverbs, e.g. *rozumně* (*reasonably*) in Figure 1 (b), are represented by the t-lemma of the corresponding adjective, here by the t-lemma *rozumný* (*reasonable*). In this way, a derivational relation is followed: the word is represented by its basic word. Other types of derivational relations analyzed in PDT 2.0 will be introduced in the next sections.
(iii) and (iv) Since there are no groups such as 'semantic pronouns' or 'semantic numerals' at the t-layer, these words were distributed into semantic nouns and adjectives according to their function they fill in the sentence. While pronouns and numerals filling typical positions of nouns (such as agent or patient) belong to semantic nouns, pronouns and numerals playing an adjectival role are classified as semantic adjectives. For examples of nominal usage of the pronoun *který* (*which*) and of the numeral *sto* (*hundred*) see sentences (1), and (2) respectively:

(1) *Kurz, který*.n *jsem si vybral, je špatný.*
The course that I have chosen is bad.

(2) *Už vedl sto*.n *kurzů.*
He has already taught one hundred courses.

For examples of adjectival usage of the pronoun *který* (*which*) and of the numeral *tři* (*three*) see sentences (3), and (4) respectively:

(3) *Který*.adj *kurz si mám vybrat?*
Which course should I choose?

(4) *Vyučuje tři*.adj *kurzy.*
He teaches three courses.

The subgroups of semantic adjectives presented above are viewed as constituting the inner structure of this class. Also the classes of semantic nouns and semantic adverbs were sub-classified in a similar way. (Semantic verbs cannot be subdivided by the same principles as the other semantic parts of speech.)[6] The appurtenance of a t-node to a concrete subgroup of semantic parts of speech is captured as a detailed value of the attribute sempos (e.g. adj.denot or adj.quant.def in Figure 3).

The t-node hierarchy including the detailed subclassification of semantic adjectives is displayed in Figure 3.

4. Grammatemes and Their Values

There are 15 grammatemes at the t-layer of PDT 2.0. Grammatemes number, gender, person and politeness were assigned to t-nodes belonging to the subclasses of semantic nouns. The grammatemes degcmp, negation, numertype and indeftype were annotated with semantic nouns as well as with semantic adjectives, the latter two of them also with semantic adverbs. The other seven grammatemes belong to semantic verbs: tense, aspect, verbmod, deontmod, dispmod, resultative, and iterativeness.
All the grammatemes will be explained and exemplified in the following subsections one by one. A separate subsection is devoted to a more detailed discussion about pronominal words.

4.1. Number

The grammateme **number** is the tectogrammatical counterpart of the morphological category of number – the grammateme values, sg (for singular) and pl (for plural), mostly correspond to the values of this morphological category, e.g. the noun *vláda*.sg (*government*) in Figure 1 (b) is in singular while *vlády*.pl (*governments*) would be plural. However, as the grammateme captures the 'semantic' number, its value differs from that of the morphological category in some cases: e.g. while the morphological number of pluralia tantum is always 'plural' (e.g. the Czech word *dveře*, *door*), the tectogrammatical singular in a sentence like (5) is discerned from the tectogrammatical plural in the sentence (6) – at these nouns, the decision by an annotator was necessary; if such a decision were not possible on the basis of context (e.g. in the sentence (7)), a special value nr ('not recognized') was assigned.

(5) *Neotevírej tyto dveře*.sg
Do not open this door.

(6) *Šel dlouhou chodbou*
He walked through a long corridor
a minul několikery dveře.pl
and passed several doors.

(7) *Otevřel dveře*.nr
He opened the door/doors.

4.2. Gender

In PDT 2.0, values of the grammateme **gender** correspond to the morphological gender: anim (for masculine animate), inan (for masculine inanimate), fem (for feminine), and neut (for neuter).

4.3. Person and Politeness

The grammatemes **person** and **politeness** have been assigned to one subclass of semantic nouns that contains personal pronouns. These words are represented by the artificial t-lemma #PersPron at the t-layer (e.g. in the Figure 1 (b), where the t-node with the t-lemma #PersPron represents the actor that is not present in the surface sentence structure). The values of the former grammateme (1, 2, 3) distinguish among the 1st, 2nd and 3rd person pronouns;

[6] The sub-classification of semantic verbs is one of our future aims; properties of verbal systems in other languages (as studied e.g. in (Bybee, 1985)) will be considered.

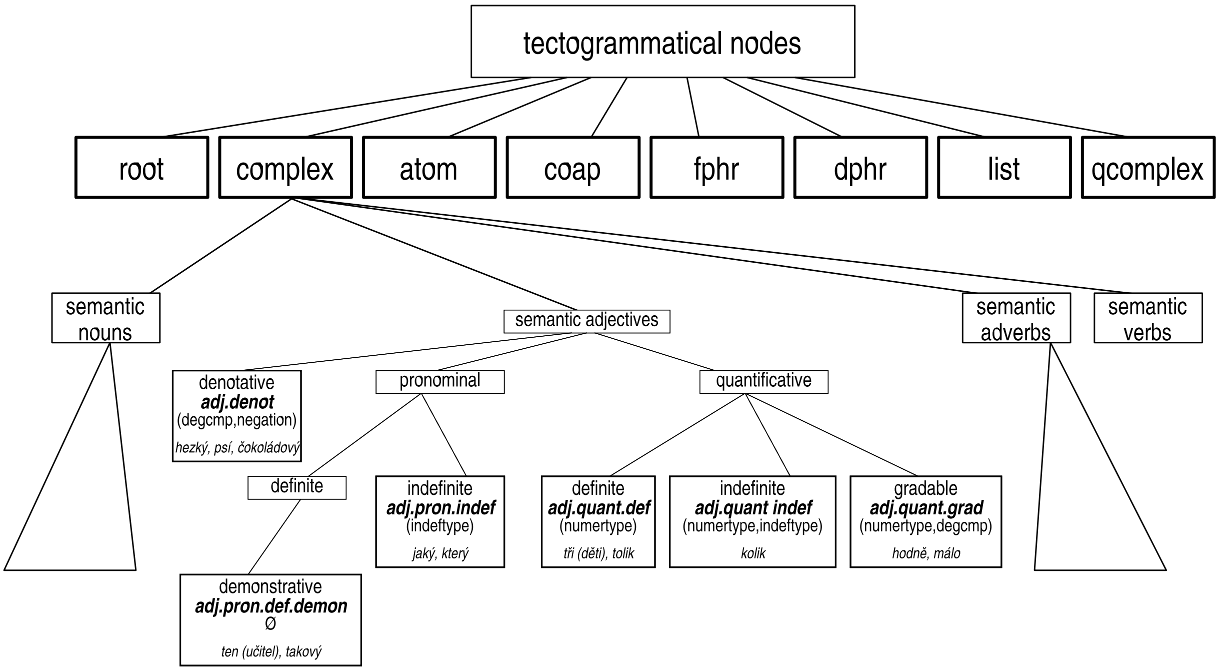


Figure 3: Hierarchy of t-nodes. The first branching renders the nodetype distinctions. Then, only complex t-nodes are further subdivided into four semantic parts of speech. Semantic nouns, semantic adjectives and semantic adverbs are further subclassified. Due to space limitations, only the subclassification of semantic adjectives is displayed in detail. In the leaf t-nodes of this subclassification, the values of attribute sempos is given on the second line and the list of grammatemes associated with the given class follows on the third line in the boxes.

the values of the latter one (basic, polite) discern the common from the polite usage of 2nd person pronouns. The surface pronoun is derived from the combination of t-lemma and values of grammatemes number, gender, person and politeness. E.g., the pronoun *vy* (*you*) in the sentence (8) is derived from the tectogrammatical representation #PersPron+**pl**+anim+2+**basic** in contrast to the same pronoun in the sentence (9) that is derived from the representation #PersPron+**sg**+anim+2+**polite**.

(8) *Vy jste vybrali dobrý kurz.*
‘You have chosen a good course’
(- said to a group of persons)

(9) *Vy jste vybral dobrý kurz.*
‘You have chosen a good course’
(- said politely to a single person)

4.4. Degree of Comparison

The grammateme **degcmp** corresponds to the morphological category of degree of comparison. Besides the values pos (for positive), comp (comparative) and sup (superlative), a special value acomp for comparative forms of adjectives/adverbs without a comparative meaning (so called ‘absolute comparative’, also ‘elative’) was established. The common usage of comparative forms such as *Jan je starší*.comp *než ona* (*Jan is elder than her*) was distinguished from the absolute usage e.g. in *starší*.acomp *muž* (*an elder man*) by the manual annotation.

4.5. Types of Numeral and Pronominal Expressions

Neither the grammateme **numertype** nor **indeftype** have a counterpart in the traditional set of morphological categories. They capture information on derivational relations among numerals, and pronominal words respectively, analyzed at the t-layer: derived words are represented by the t-lemma of its basic word and the feature that would be lost by such a representation is captured by values of these grammatemes. As all types of numerals are seen as derivations from the corresponding basic numeral and thus represented by its t-lemma, the grammateme numertype captures the type of the numeral in question. The surface numeral is then derived from the t-lemma and the value of this grammateme, e.g. the ordinal numeral *třetí* (*the third*) is derived form the following tectogrammatical representation: t-lemma *tři* (*three*) + numertype=‘ord’ (for ordinal). Besides the value ord, the value set of this grammateme involves four other values: basic for basic numerals (*tři kurzy–three courses*), frac for fractional numerals (*třetina kurzu–the third of the course*), kind for numerals concerning the number of kinds/sorts (*trojí víno–three sorts of wine*), and set for numerals with meaning of the number of sets (*troje klíče–three sets of keys*).

In a similar vein, indefinite, negative, interrogative, and relative pronouns are represented by the t-lemma corresponding to the relative pronoun – the specific semantic feature is stored in the grammateme indeftype. Surface pronouns are derived from the lemma and the value of this grammateme: e.g. the indefinite pronoun *někdo* (*somebody*) and the negative pronoun *nikdo* (*nobody*) are derived from the following tectogrammatical representations: t-lemma *kdo* + indeftype='indef', and t-lemma *kdo* + indeftype='negat' respectively.[7] Such representation of derivational relations makes it possible to represent all these words by a very small set of t-lemmas. The question of applying similar principles to pronominal words in other languages will be mentioned in Subsection 4.11.

4.6. Negation

Also the grammateme **negation** captures a lexical information needed for derivation of surface forms: it enables to represent both, the positive and the negative forms of adjectives, adverbs and (temporarily, only a group of) nouns by a single t-node with the same t-lemma – e.g. the adjective *standardní* (*standard*) in Figure 1 (b) as well as its negative form *nestandardní* (*non-standard*) are represented by the t-node with t-lemma *standardní* and the absence/presence of negation is captured by the value of the grammateme: the value neg0 was assigned to the t-node representing the positive form, the value neg1 to the t-node corresponding to the negative form.[8]

4.7. Tense

The grammateme **tense** corresponds to the morphological category of tense. The values sim (simultaneous with the moment of speech/with other event), ant (anterior to the moment of speech/to other event), and post (posterior to the moment of speech/to other event)[9] have been assigned automatically.

4.8. Aspect

The grammateme **aspect** is the tectogrammatical counterpart of the category of aspect. As there are verbs in Czech that can express both, imperfective and perfective aspects by the same forms (so called bi-aspectual verbs), manual annotation was necessary to make a decision with these verbs.

4.9. Verbal Modalities

There are three grammatemes concerning modality. The grammateme **verbmod** captures if the represented verbal form expresses the indicative (value ind), the imperative (imp), or the conditional mood (cdn). Since modal verbs do not have a t-node of their own at the t-layer (for explanation see (Panevová et al., 1971)), the deontic modality expressed by these verbs is stored in the grammateme **deontmod**, e.g. the predicate of the sentence *Už může odejít* (*He can already leave*) is represented by a t-node with t-lemma *odejít* (*to leave*) and the modality is stored as the value poss (for possibilitive) in the grammateme deontmod. The last of the modality grammatemes, the grammateme **dispmod**, concerns the so-called dispositional modality. This type of modality is represented by a special syntactic construction involving a 'reflexive-passive' verb construction, a dative form of a noun/personal pronoun playing the role of agent, and a modal adverb, e.g. the sentence (10):

(10) *Studentům se ta kniha čte dobře.*
Lit. *To students the book reads well.*
It is easy for the students to read the book.

4.10. Resultative and Iterativeness

While the grammateme **resultative** (values res1, res0) reflects the fact whether the event is/is not presented as a resultant state, the last verbal grammateme **iterativeness** indicates whether the event is/is not viewed as a repeated (multiplied) action (values it1, it0).

4.11. Pronominal Words at the T-layer

In this chapter, we would like to provide a deeper view into the principles of representation of pronominal words at the t-layer of PDT 2.0, and then to outline how this representation can be applied to such words in English or German.

As already mentioned above, pronouns are represented by a minimal set of t-lemmas at the t-layer. Personal pronouns by a single (artificial) t-lemma #PersPron; grammatemes assigned to the t-nodes of personal pronouns were presented in the previous chapter. Indefinite, negative, in-

T-lemma:	*kdo*	*co*	*kter´y*	*jak´y*
indefype:				
relat	kdo	co	který, jenž	jaký
indef1	někdo	něco	některý	nějaký
indef2	kdosi kdos	cosi cos	nějakýsi	jakýsi
indef3	kdokoli kdokoliv	cokoli cokoliv	kterýkoli kterýkoliv	jakýkoli jakýkoliv
indef4	ledakdo leckdo	ledaco lecco	leckterý ledakterý	lecjaký ledajaký
indef5	kdekdo	kdeco	kdekterý	kdejaký
indef6	kdovíkdo málokdo	kdovíco máloco	kdovíkterý málokterý	kdovíjaký všelijaký
inter	kdo kdopak	co copak	který kterýpak	jaký jakýpak
negat	nikdo	nic	žádný	nijaký
total1	všechen	všechno vše	-	-
total2	-	-	každý	-

Table 1: The indeftype grammateme has actually eleven values (1st column in the table). It makes it possible to represent all semantic variants of pronouns *kdo* (*somebody*), *co* (*something*), *který* (*that*) and *jaký* (*what*) (in the 2nd, 3rd, 4th and 5th column) by only four t-lemmas at the t-layer.

[7] A similar treatment of indefinite and negative pronouns as of two subtypes of the same entity can be found in (Helbig, 2001).

[8] Unlike this representation, negative verbal forms (verbal negation is expressed also by the prefix *ne-* in Czech) are represented by a sub-tree consisting of a t-node with a verbal t-lemma the child of which is a t-node with the artificial t-lemma #Neg; cf. the representation of the negated verb *nelze* ((it) *can not be*) by two t-nodes, with the t-lemmas *lze* ((it) *can be*) and #Neg, in Figure 1 (b). The explanation can be found in (Hajičová, 1975).

[9] As the class of semantic verbs has not been sub-classified yet and all verbal grammatemes were annotated with each verbal t-node, a special value nil was inserted into the value system for cases when the represented word does not express a feature captured by the grammateme (cf. the value of grammateme tense at a t-node representing an infinitive form).

terrogative and relative pronouns are all represented by a t-lemma corresponding to the relative pronoun. In this way, only four lemmas – i.e. *kdo* (*somebody*), *co* (*something*), *který* (*which*) and *jaký* (*what*) – are sufficient to represent all Czech pronouns of named types at the t-layer. The pronouns with corresponding values of the grammateme indeftype are displayed in Table 1.

Since the semantic features stored in the grammateme indeftype are expressed also by other words of pronominal character in Czech, e.g. by pronominal adverbs *nikde* (*nowhere*) or *nějak* (*somehow*), or by an indefinite numeral *několik* (*a few*), we can use this grammateme also for the tectogrammatical representation of these words.[10]

As the groups of pronominal words are unproductive classes with (at least to a certain extent) transparent derivational relations not only in Czech, but also in other languages, we believe that similar regularities to those captured in Czech by the indeftype grammateme can be found also elsewhere. However, as it is obvious from the preliminary sketch of several English and German pronouns classified in Table 2,[11] the application of our scheme to other languages will not be straightforward and various subtle differences have to be taken into account. For instance, there is only one negative form *nikdo* corresponding to the t-lemma *kdo* in Czech, therefore the present system provides no means for distinguishing German negative pronouns *niemand* and *niergendjemand*. A new question arises also in the case of English *anybody* when used in negative clauses, which has no counterpart in Czech or German.

5. Implementation

The procedure for assigning grammatemes (and nodetype and sempos) to nodes of tectogrammatical trees was implemented in ntred[12] environment for processing the PDT data. Besides almost 2000 lines of Perl code, we formulated a number of rules for grammateme assignment written in a text file using a special economic notation (roughly 2000 lines again), and numerous lexical resources (e.g. special-purpose list of verbs or adverbs). As we intensively used all information available also at the two 'lower' levels of the PDT (morphological and analytical), most of the annotation could have been done automatically with a highly satisfactory precision.

It should be emphasized that the inter-layer links played a key role in the procedure. As it is clear from Figure 1 (a), it would not be possible to set e.g. the value of the number grammateme of the (already lemmatized) t-node *les* (forest) without having the access to the morphological tag of the corresponding m-layer unit in the given sentence, or to find out that the verb *jít* (to go) is in conditional mood (verbmod=cdn) without knowing that the corresponding a-layer complex verb form subgraph contains the node *by*.

	English	English	German	German
T-lemma	*who*	*what*	*wer*	*was*
indeftype:				
relat	who	what	wer	was
indef1	somebody	something	jemand	etwas
indef2	-	-	irgendjemand	irgendetwas
indef3	whoever	whatever	-	-
inter	who	what	wer	was
negat	nobody	nothing	niemand	nichts
total1	all	everything	alle	alles
total2	each	each	jeder	jedes

Table 2: Selected English and German pronouns preliminarily classified according to the indeftype grammateme.

Due to the fact that a lot of effort had been spent on checking and correcting of the inter-layer pointers in PDT 2.0, finally we needed only around 5 man-months of human annotation for solving just the very specific issues (as mentioned at single grammatemes in the previous section).

Now we would like to show a fragment of the above mentioned rules. For a given t-node: if the lemma of the corresponding m-node is *který* (which), the t-node itself is not in the attributive syntactic position and participates in grammatical coreference (i.e., it forms a relative construction), then sempos=n.pron.indef, indeftype=relat, and the values of the grammatemes gender and number are inherited from the coreference antecedent. This rule would be applied on the sentence (1).

To further demonstrate that grammatemes are not just dummy copies of what was already present in the morphological tag of the node, we give two examples:

- Deleted pronouns in subject positions (which must be restored at the t-layer) might inherit their gender and/or number from the agreement with the governing verb (possibly complex verbal form), or from an adjective (if the governor was copula), or from its antecedent (in the sense of textual coreference).

- Future verbal tense in Czech can be realized using simple inflection (perfectives), or auxiliary verb (imperfectives), or prefixing (lexically limited).

The procedure was repeatedly tested on the PDT data, which was extremely important for debugging and further improvements of the procedure. Final version of the procedure was applied to all the available tectogrammatical data (as for its size, recall the second paragraph in Section 2.). This data, enriched with node classification and grammateme annotation, will be included in PDT 2.0 distribution.

Due to the highly structured nature of the task, it is difficult to present the results of the annotation procedure from the quantitative viewpoint. However, at least the distribution of the values of nodetype and sempos are shown in Tables 3 and 4.

[10] The indeftype grammateme is applied to indefinite numerals together with the above-mentioned grammateme numertype – thus only a single t-lemma *kolik* (*how many*) represent words of different nature: e.g. *několik´at´y* (*not the first*), *kolikr´at* (*how many times*) etc.

[11] We chose English and German, because, first, the two languages are the most familiar to the present authors, and second, certain experiments concerning their t-layer have already been performed, see e.g. (Cinková, 2004) or (Kučerová and Žabokrtský, 2002).

[12] `http://ufal.mff.cuni.cz/~pajas`

complex	550947
root	49442
qcomplex	46015
coap	35747
atom	34035
fphr	4549
list	2512
dphr	1282

Table 3: Values of nodetype sorted according to the number of occurences in the PDT 2.0 t-layer data.

n.denot	236926
adj.denot	100877
v	88037
n.pron.def.pers	32903
adj.quant.def	19441
n.denot.neg	18831
n.pron.indef	11343
adv.denot.ngrad.nneg	8947
n.quant.def	7994
adj.pron.def.demon	5746
n.pron.def.demon	4759
adj.pron.indef	3383
adv.pron.indef	3107
adv.pron.def	2928
adj.quant.grad	1865
adv.denot.grad.neg	1315
adv.denot.grad.nneg	1139
adv.denot.ngrad.neg	751
adj.quant.indef	655

Table 4: Detailed values of sempos sorted according to the number of occurences in the PDT 2.0 t-layer data.

6. Conclusion

We believe that two important novel goals have been achieved in the present enterprise:

- We proposed a formal classification of tectogrammatical nodes and described its consequences on the system of grammatemes, and thus the tectogrammatical tree structures become formalizable e.g. by typed feature structures.
- We implemented an automatic and highly-complex procedure for capturing the node classification, the system of grammatemes and derivations, and verified it on large-scale data, namely on the whole tectogrammatical data of PDT 2.0. Thus the results of our work will be soon publicly available.

In the paper we do not compare our achievements with related work, since we are simply not aware of a comparably structured annotation on comparably large data in any other publicly available treebank. For instance, to our knowledge no other treebank attemps at reducing the (semantically redundant) morphological attributes imposed only by agreement, or at specifying verbal tense for a complex verb form as for a whole, or at representing a noun (or a personal pronoun) and the corresponding possessive adjective (or possessive pronoun, respectively) in a unified fashion. However, from the theoretical viewpoint the presented model bears some resemblances with the system of grammemes in the deep-syntactic level of the already mentioned Meaning-Text Theory (Mel'čuk, 1988).

In the near future, we plan to separate the grammatemes that bear the derivational information (such as numertype) from the grammatemes having their direct counterpart in traditional morphological categories. The long-term aim is to describe further types of derivation: we should concentrate on productive types of derivation (diminutive formation, formation of feminine counterparts of agentive nouns etc.). The set of 'derivational' grammatemes will be extended in this way. The next issue is the problem of subclassification of semantic verbs. The challenging topic is also the study of grammatemes in other languages.

Acknowledgements

The research reported in this paper was supported by the projects 1ET101120503, GA-UK 352/2005 and GD201/05/H014. We would also like to thank professors Jarmila Panevová and Eva Hajičová for numerous comments on the draft of the paper.

7. References

Joan L. Bybee. 1985. *Morphology: A study of the relation between meaning and form.* Benjamins, Philadelphia.

Silvie Cinková. 2004. Manuál pro tektogramatickou anotaci angličtiny. Technical report, ÚFAL/CKL MFF UK.

Miloš Dokulil. 1962. *Tvoření slov v češtině I.* Academia, Prague.

Jan Hajič, Eva Hajičová, Petr Pajas, Jarmila Panevová, Petr Sgall, and Barbora Vidová Hladká. 2001. Prague Dependency Treebank 1.0.

Eva Hajičová, Jan Hajič, Barbora Vidová-Hladká, Martin Holub, Petr Pajas, Veronika Kolářová-Řezníčková, and Petr Sgall. 2001. The Current Status of the Prague Dependency Treebank. In *Proceedings of the 5th International Conference on Text, Speech and Dialogue*, pages 11–20, Berlin, Heidelberg, New York. Springer-Verlag.

Eva Hajičová. 1975. *Negace a presupozice ve významové stavbě věty.* Academia, Prague.

Hermann Helbig. 2001. *Die semantische Struktur natürlicher Sprache.* Springer-Verlag, Berlin, Heidelberg, New York.

Ivona Kučerová and Zdeněk Žabokrtský. 2002. Transforming Penn Treebank Phrase Trees into (Praguian) Tectogrammatical Dependency Trees. *Prague Bulletin of Mathematical Linguistics*, (78):77–94.

Igor A. Mel'čuk. 1988. *Dependency Syntax: Theory and Practice.* State University of New York Press.

Jarmila Panevová, Eva Benešová, and Petr Sgall. 1971. *Čas a modalita v češtině.* Univerzita Karlova, Prague.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar.* The University of Chicago Press, Chicago.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects.* D. Reidel Publishing Company, Dordrecht.

Petr Sgall. 1967. *Generativní popis jazyka a česká deklinace.* Academia, Prague.

7.3 Anaphora in Czech: Large Data and Experiments with Automatic Anaphora Resolution

Full reference:

Kučová Lucie, Žabokrtský Zdeněk: Anaphora in Czech: Large Data and Experiments with Automatic Anaphora, in Lecture Notes in Computer Science, Vol. 3658, Proceedings of the 8th International Conference, TSD 2005, Copyright Springer, Západočeská univerzita v Plzni, Berlin / Heidelberg, ISBN 3-540-28789-2, ISSN 0302-9743, pp. 93-98, 2005

Comments:

From 2002 to 2005 we participated in adding coreference relations to tectogrammatical trees in PDT. Technically, the added relations were represented (and also visualized) as "pointers" from one t-node (anaphor) to another t-node (antecedent), which proved better tractable than the coreference representations suggested in [Petkevič, 1987] and in [Panevová et al., 2001].

Annotation instructions were completed gradually during the annotation process. They were first published as [Kučová et al., 2003], and later incorporated into the PDT 2.0 annotation guidelines [Mikulová et al., 2005].

In order to make the annotation faster, certain classes of coreference links were pre-annotated automatically (described in [Kučová et al., 2003] too) from the early stages of annotation process. More recent experiments with automatic coreference resolution based on the PDT scheme can be found in [Kučová and Žabokrtský, 2005], [Němčík, 2006], [Linh and Žabokrtský, 2007].

Another recognizer of several types of coreference links (especially grammatical coreference in the case of relative clauses and reflexive pronouns, which seem to be the most important ones from the MT viewpoint) is now implemented in TectoMT and used during the translation process, as already mentioned in Section 5.1.4.

Anaphora in Czech: Large Data and Experiments with Automatic Anaphora Resolution

Lucie Kučová and Zdeněk Žabokrtský *

Institute of Formal and Applied Linguistics, Charles University (MFF),
Malostranské nám. 25, CZ-11800 Prague, Czech Republic
{kucova,zabokrtsky}@ufal.mff.cuni.cz
http://ufal.mff.cuni.cz

Abstract. The aim of this paper is two-fold. First, we want to present a part of the annotation scheme of the Prague Dependency Treebank 2.0 related to the annotation of coreference on the tectogrammatical layer of sentence representation (more than 45,000 textual and grammatical coreference links in almost 50,000 manually annotated Czech sentences). Second, we report a new pronoun resolution system developed and tested using the treebank data, the success rate of which is 60.4 %.

1 Introduction

Coreference (or co-reference) is usually understood as a symmetric and transitive relation between two expressions in the discourse which refer to the same entity. It is a means for maintaining language economy and discourse cohesion ([1]). Since the expressions are linearly ordered in the time of the discourse, the first expression is often called *antecedent.* Then the second expression (*anaphor*) is seen as 'referring back' to the antecedent. Such a relation is often called *anaphora.*[1] The process of determining the antecedent of an anaphor is called *anaphora resolution* (AR).

Needless to say that AR is a well-motivated NLP task, playing an important role e.g. in machine translation. However, although the problem of AR has attracted the attention of many researches all over the world since 1970s and many approaches have been developed (see [2]), there are only a few works dealing with this subject for Czech, especially in the field of large (corpus) data.

The present paper summarizes the results of studying the phenomenon of coreference in Czech within the context of the Prague Dependency Treebank 2.0 (PDT 2.0).[2] PDT 2.0 is a collection of linguistically annotated data and documentation and is based on the theoretical framework of Functional Generative Description (FGD). The annotation scheme of the PDT 2.0 consists of

* The research reported on in this paper has been supported by the grant of the Charles University in Prague 207-10/203329 and by the project 1ET101120503.

[1] Unfortunately, these terms tend to be used inconsistently in literature.

[2] PDT 2.0 is to be released soon by the Linguistic Data Consortium.

three layers: morphological, analytical and tectogrammatical. Within this system, coreference is captured at the tectogrammatical layer of annotation.

2 Theoretical Background

In FGD, the distinction between grammatical and textual coreference is drawn ([6]). One of the differences is that (individual subtypes of) grammatical coreference can occur only if certain local configurational requirements are fulfilled in the dependency tree (such as: if there is a relative pronoun node in a relative clause and the verbal head of the clause is governed by a nominal node, then the pronoun node and nominal node are coreferential), whereas textual coreference between two nodes does not imply any syntactic relation between the nodes in question or any other constraint on the shape of the dependency tree. Thus textual coreference easily crosses sentence boundaries.

Grammatical Coreference. In the PDT 2.0, grammatical coreference is annotated in the following situations (see a sample tree in Fig. 1):[3] (i) relative pronouns in relative clauses, (ii) reflexive and reciprocity pronouns (usually coreferential with the subject of the clause), (iii) control (in the sense of [7]) – both for verbs and nouns of control.

Textual Coreference. For the time being, we concentrate on the case of textual coreference in which a demonstrative or an anaphoric pronoun (also in its zero form) are used.[4] The following types of textual coreference links are special (see a sample tree in Fig. 2):[5]

- a link to a particular node if this node represents an antecedent of the anaphor or a link to the governing node of a subtree if the antecedent is represented by this node plus (some of) its dependents:[6] *Myslíte, že rozhodnutí NATO, zda se [**ono**] rozšíří, či nikoli, bude záviset na postoji Ruska?* (Do you think that the decision of NATO whether [**it**] will be enlarged or not will depend on the attitude of Russia?)
- a specifically marked link (segm) denoting that the referent is a whole segment of text, including also cases, when the antecedent is understood by inferencing from a broader co-text: *Potentáti v bance koupí za 10, prodají si za 15.(. . .) Odhaduji, že do 2 let budou schopni splatit bance dluh a třetím*

[3] We only list the types of coreference in this paper; detailed linguistic description will be available in the documentation of the PDT 2.0.

[4] With the demonstrative pronoun, we consider only its use as a noun, not as an adjective; we do not include pronouns of the first and second persons.

[5] Besides the listed coreference types, there is one more situation where coreference occurs but is difficult to be identified and no mark is stored into the attributes for coreference representation. It is the case of nodes with tectogrammatical lemma #Unsp (unspecified); see [9]. Example: *Zmizení tohoto 700 kg těžkého přístroje hygienikům ohlásili (**Unsp**) 30. června letošního roku.* (Lit.: The disappearance of the medical instrument weighing 700 kg to hygienists[**they**] announced on June 30th this year.)

[6] This is also the way how a link to a clause or a sentence is being captured.

rokem už budou dělat na sebe. A na práci najmou jen schopné lidi. Kdo ***to*** *pochopí, má náskok.* (The big shots buy in a bank for 10 and sell for 15. (. . .) I guess that within two years they will be able to pay back the debt to the bank and in the third year they will work for themselves. And they will hire only capable people, it will be in their best interest. Those who understand **this**, will have an advantage.)

– a specifically marked link (exoph) denoting that the referent is "out" of the co-text, it is known only from the situation: *Následuje dramatická pauza a pak již vchází* ***On*** *nebo* ***Ona****.* (Lit. (there) follows dramatic pause and then already enters **He** or **She**.)

3 Annotated Data

Data Representation. When designing the data representation on coreference links, we took into account the fact that each tectogrammatical node is equipped with an identifier which is unique in the whole PDT. Thus the coreference link can be easily captured by storing the identifier of the antecedent node (or a sequence of identifiers, if there are more antecedents for the same anaphor) into a distinguished attribute of the anaphor node. We find this 'pointer' solution more transparent (and – from the programmer's point of view – much easier to cope with) than the solutions proposed in [3] or [4].

At present, there are three node attributes used for representing coreference: (i) coref_gram.rf – identifier or a list of identifiers of the antecedent(s) related via grammatical coreference; (ii) coref_text.rf – identifier or a list of identifiers of the antecedent(s) related via textual coreference; (iii) coref_special – values segm (segment) and exoph (exophora) standing for special types of textual coreference.

We used the tree editor TrEd developed by Petr Pajas as the main annotation interface.[7] More details concerning the annotation environment can be found in [8]. In this editor (as well as in Figures 1 and 2 in this paper), a coreference link is visualized as a non-tree arc pointing from the anaphor to its antecedent.

Quantitative Properties. PDT 2.0 contains 3,168 newspaper texts annotated at the tectogrammatical level. Altogether, they consist of 49,442 sentences with 833,357 tokens (summing word forms and punctuation marks). Coreference has been annotated manually (disjunctively[8]) in all this data. After finishing the manual annotation and post-annotation checks and corrections, there are 23,266 links of grammatical coreference (dominating relative pronouns as the anaphor – 32 %) and 22,365[9] links of textual coreference (dominating personal and possessive pronouns as the anaphor – 83 %), plus 505 occurrences of segm and 120 occurrences of exoph).

[7] `http://ufal.mff.cuni.cz/~pajas`

[8] Independent parallel annotation of the same sentences were performed only in the starting phase of the annotation, only as long as the annotation scheme stabilized and reasonable inter-annotator agreement was reached (see [8])

[9] Similarity of the numbers of textual and grammatical coreference links is only a more or less random coincidence. If we would have annotated also e.g. bridging anaphora, the numbers would be much more different.

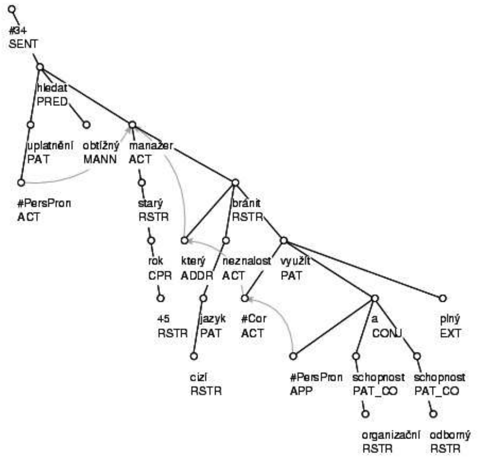


Fig. 1. Simplified PDT sample with various subtypes of grammatical coreference. The structure is simplified, only tectogrammatical lemmas, functors, and coreference links are depicted. The original sentence is '*Obtížněji hledají své uplatnění manažeři starší 45 let, kterým neznalost cizích jazyků brání plně využít své organizační a odborné schopnosti.*' (Lit.: More difficultly search their self-fulfillment manages older than 45 years, to which unknowledge of foreign languages hamper to use their organization and specialized abilities).

4 Experiments and Evaluation of Automatic Anaphora Resolution

In [8] it was shown that it is easy to get close to 90 % precision when considering only grammatical coreference.[10] Obviously, textual coreference is more difficult to resolve (there are almost no reliable clues as in the case of grammatical coreference). So far, we attempted to resolve only the textual coreference links 'starting' in nodes with tectogrammatical lemma #PersPron. This lemma stands for personal (and personal possessive) pronouns, be they expressed on the surface (i.e., present in the original sentence) or restored during the annotation of the tectogrammatical tree structure.

We use the following procedure (numbers in parentheses were measured on the training part of the PDT 2.0):[11] For each detected anaphor (lemma #PersPron):

[10] This is not surprising, since in the case of grammatical coreference most of the information can be derived from the topology and basic attributes of the tree (supposing that we have access also to the annotation of morphological and analytical level of the sentence). However, it opens the question of redundancy (at least for certain types of grammatical coreference).

[11] The procedure is based mostly on our experience with the data. However, it undoubtedly bears many similarities with other approaches ([2]).

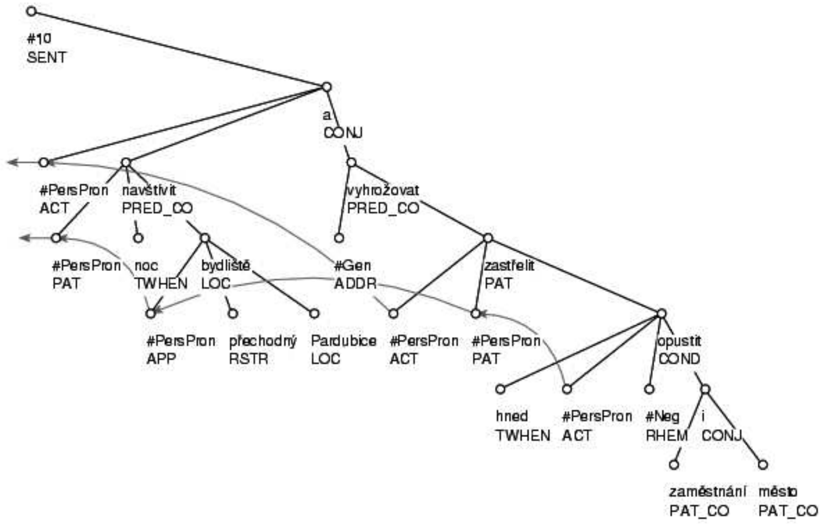


Fig. 2. Simplified PDT sample containing two textual coreference chains. The original sentence is '*Navštívil ji v noci v jejím přechodném bydlišti v Pardubicích a vyhrožoval, že ji zastřelí, pokud hned neopustí zaměstnání i město.*' (Lit.: [He] visited her in night in her temporary dwelling in Pardubice and threatened [her] that [he] will shoot her if [she] instantly does not leave her job and city.).

– First, an initial set of antecedent candidates is created: we used all nodes from the previous sentence and current sentence (roughly 3.2 % of correct answers disappear from the set of candidates in this step).
– Second, the set of candidates is gradually reduced using various filters: (1) candidates from the current sentence not preceding the anaphor are removed (next 6.2 % lost), (2) candidates which are not semantic nouns (nouns, pronouns and numeral with nominal nature, possessive pronouns, etc.), or at least conjunctions coordinating two or more semantic nouns, are removed (5.6 % lost), (3) candidates in subject position which are in the same clause as the anaphor are removed, since the anaphor would be probably expressed by a reflexive pronoun (0.7 % lost) (4) all candidates disagreeing with the anaphor in gender or number are removed (3.7 % lost), (5) candidates which are parent or grandparent of the anaphor (in the tree structure) are removed (0.6 % lost), (6) if both the node and its parent are in the set of candidates, then the child node is removed (1.6 % lost), (7) if there is a candidate with the same functor with anaphor, then all candidates having different functor are removed (3.4 % lost), (8) if there is a candidate in a subject position, then all candidates in different than subject positions are removed (2.4 % lost),
– Third, the candidate is chosen from the remaining set which is (linearly) the closest to the given anaphor (12.5 % lost).

When measuring the performance only on the evaluation-purpose part of the PDT 2.0 data (roughly 10 % of the whole), the final success rate (number of correctly resolved antecedents divided by the number of pronoun anaphors) is 60.4 %.[12]

The whole system consists of roughly 200 lines of Perl code and was implemented using ntred[13] environment for accessing the PDT data. The question of speed is almost irrelevant: since the system is quite straightforward and fully deterministic, ntred running on ten networked computers needs less than one minute to resolve all #PersPron node in PDT.

5 Final Remarks

We understand coreference as an integral part of a dependency-based annotation of underlying sentence structure which prepares solid grounds for further linguistic investigations. It proved to be useful in the implemented AR system, which profits from the existence of the tectogrammatical dependency tree (and also from the annotations on the two lower levels).

As for the results achieved by our AR system, to our knowledge there is no other system for Czech reaching comparable performance and verified on comparably large data.

References

1. Halliday M. A. K., Hasan, R.: Cohesion in English, Longman, London (1976)
2. Mitkov, R.: Anaphora resolution. Longman, London (2001)
3. Plátek, M., Sgall, J., Sgall, P.: A Dependency Base for a Linguistic Description. In: Sgall, P.(ed.): Contributions to Functional Syntax, Semantics and Language Comprehension. Academia, Prague (1984) 63-97
4. Hajičová, E., Panevová J., Sgall, P.: Coreference in Annotating a Large Corpus. In: Proceedings of LREC 2000, Vol. 1. Athens, Greece (2000) 497-500
5. Hajičová, E., Panevová, J., Sgall, P. Manuál pro tektogramatické značkování. Technical Report ÚFAL-TR-7 (1999)
6. Panevová, J.: Koreference gramatická nebo textová? In: Banys, W., Bednarczuk, L., Bogacki, K. (eds.): Etudes de linguistique romane et slave, Krakow (1991)
7. Panevová, J.: More Remarks on Control. In: Prague Linguistic Circle Papers, Benjamin Publ. House, Amsterdam – Philadelphia (1996) 101-120
8. Kučová L., Kolářová V., Pajas P., Žabokrtský Z., and Čulo O.: Anotování koreference v Pražském závislostním korpusu. Technical Report of the Center for Computational Linguistics, Charles University, Prague (2003)
9. Kučová L., Hajičová E. 2004), Coreferential Relations in the Prague Dependency Treebank. Presented at 5th Discourse Anaphora and Anaphor Resolution Colloquium, San Miguel, Azores (2004)
10. Barbu C., Mitkov, R.: Evaluation tool for rule-based anaphora resolution methods. In: Proceedings of ACL'01, Toulouse, France (2001) 34-41

[12] For instance, the results in pronoun resolution in English reported in [10] was also around 60 %.

[13] http://ufal.mff.cuni.cz/~pajas

Chapter 8

Parsing and Transformations of Syntactic Trees

8.1 Combining Czech Dependency Parsers

Full reference:

Holan Tomáš, Žabokrtský Zdeněk: Combining Czech Dependency Parsers, in Lecture Notes in Computer Science, No. 4188, Proceedings of the 9th International Conference, TSD 2006, Copyright Springer-Verlag Berlin Heidelberg, Masarykova univerzita, Berlin / Heidelberg, ISBN 3-540-39090-1, ISSN 0302-9743, pp. 95-102, 2006

Comments:

The paper represents two directions of our research in dependency parsing: first, we implemented our own rule-based parser, and second, we experimented with combining the existing parsers in order to achieve higher accuracy than that of any parser in isolation.

Our rule-based parser described in the paper does not outperform the state-of-the-art statistical parsers, but does significantly contribute in a parser combination – it seems that it is better to combine several highly diverse parsers than to combine only the high-accuracy ones. Besides the Czech version, we also created mutations of the parser for German, Romanian, Slovenian (used for pre-annotation of the Slovene Dependency Treebank, [Džeroski et al., 2006]), and Polish (used in experiments on correcting OCR of medical texts, [Piasecki, 2007]). The Czech version is now integrated in the TectoMT environment and used in light-weight applications (such as on-line analysis of Czech sentences in TrEd).

We also participated in other parsing experiments [Zeman and Žabokrtský, 2005] and [Novák and Žabokrtský, 2007]. In the latter work we have shown that it is possible to significantly reduce the size of the model used by the state-of-the-art MST parser [McDonald et al., 2005] by removing the least-weighted features and thus making the parsing process more effective (reducing the time and memory requirements) without sacrificing accuracy. The adapted parser models are now used in TectoMT both for parsing Czech and English.

Combining Czech Dependency Parsers*

Tomáš Holan and Zdeněk Žabokrtský

Faculty of Mathematics and Physics, Charles University
Malostranské nám. 25, CZ-11800 Prague, Czech Republic
{tomas.holan,zdenek.zabokrtsky}@mff.cuni.cz

Abstract. In this paper we describe in detail two dependency parsing techniques developed and evaluated using the Prague Dependency Treebank 2.0. Then we propose two approaches for combining various existing parsers in order to obtain better accuracy. The highest parsing accuracy reported in this paper is 85.84 %, which represents 1.86 % improvement compared to the best single state-of-the-art parser. To our knowledge, no better result achieved on the same data has been published yet.

1 Introduction

Within the domain of NLP, dependency parsing is nowadays a well-established discipline. One of the most popular benchmarks for evaluating parser quality is the set of analytical (surface-syntactic) trees provided in the Prague Dependency Treebank (PDT). In the present paper we use the beta (pre-release) version of PDT 2.0,[1] which contains 87,980 Czech sentences (1,504,847 words and punctuation marks in 5,338 Czech documents) manually annotated at least to the analytical layer (a-layer for short).

In order to make the results reported in this paper comparable to other works, we use the PDT 2.0 division of the a-layer data into training set, development-test set (d-test), and evaluation-test set (e-test). Since all the parsers (and parser combinations) presented in this paper produce full dependency parses (rooted trees), it is possible to evaluate parser quality simply by measuring its accuracy: the number of correctly attached nodes divided by the number of all nodes (not including the technical roots, as used in the PDT 2.0). More information about evaluation of dependency parsing can be found e.g. in [1].

Following the recommendation from the PDT 2.0 documentation for the developers of dependency parsers, in order to achieve more realistic results we use morphological tags assigned by an automatic tagger (instead of the human annotated tags) as parser input in all our experiments.

The rest of the paper is organized as follows: in Sections 2 and 3, we describe in detail two types of our new parsers. In Section 4, two different approaches to parser combination are discussed and evaluated. Concluding remarks are in Section 5.

* The research reported on in this paper has been carried out under the projects 1ET101120503, GAČR 207-13/201125, 1ET100300517, and LC 536.

[1] For a detailed information and references see http://ufal.mff.cuni.cz/pdt2.0/

2 Rule-based Dependency Parser

In this section we will describe a rule-based dependency parser created by one of the authors. Although the first version of the parser was implemented already in 2002 and its results have been used in several works (e.g. [2]), no more detailed description of the parser itself has been published yet.

The parser in question is not based on any grammar formalism (however, it has been partially inspired by several well-known formal frameworks, especially by unification grammars and restarting automata). Instead, the grammar is 'hardwired' directly in Perl code. The parser uses tred/btred/ntred[2] tree processing environment developed by Petr Pajas. The design decisions important for the parser are described in the following paragraphs.

One tree per sentence. The parser outputs exactly one dependency tree for any sentence, even if the sentence is ambiguous or incorrect. As illustrated in Figure 1 step 1, the parser starts with a flat tree – a sequence of nodes attached below the auxiliary root, each of them containing the respective word form, lemma, and morphological tag. Then the linguistically relevant oriented edges are gradually added by various techniques. The structure is connected and acyclic at any parsing phase.

No backtracking. We prefer greedy parsing (allowing subsequent corrections, however) to backtracking. If the parser makes a bad decision (e.g. due to insufficient local information) and it is detected only much later, then the parser can 'rehang' the already attached node (rehanging becomes necessary especially in the case of coordinations, see steps 3 and 6 in Figure 1). Thus there is no danger of exponential expansion which often burdens symbolic parsers.

Bottom-up parsing (reduction rules). When applying reduction rules, we use the idea of a 'sliding window' (a short array), which moves along the sequence of 'parentless' nodes (the artificial root's children) from right to left.[3] On each position, we try to apply simple hand-written grammar rules (each implemented as an independent Perl subroutine) on the window elements. For instance, the rule for reducing prepositional groups works as follows: if the first element in the window is an unsaturated preposition and the second one is a noun or a pronoun agreeing in morphological case, then the parser 'hangs' the second node below the first node, as shown in the code fragment below (compare steps 9 and 10 in Figure 1):

```
sub rule_prep_noun($) {
  my $win = shift;
  if (preposition($win->[0])
    and nominal($win->[1])
    and not $win->[0]->{p_saturated}){
      $win->[0]->{p_saturated}=1;
      return hang($win->[1],$win->[0]);
  } else {  return 0 }
}
```

```
sub rule_adj_noun($) {
  my $win = shift;
  if (adjectival($win->[0]) and noun($win->[1])
       and ($win->[0]->{p_ordinal} or
         (agr_case($win->[0],$win->[1]) and
          agr_number($win->[0],$win->[1]) and
          agr_gender($win->[0],$win->[1])))) {
    return hang($win->[0],$win->[1]);
    } else {  return 0 }
}
```

[2] http://ufal.mff.cuni.cz/~pajas/tred/index.html

[3] Our observations show that the direction choice is important, at least for Czech.

The rules are tried out according to their pre-specified ordering; only the first applicable rule is always chosen. Then the sliding window is shifted several positions to the right (outside the area influenced by the last reduction, or to the right-most position), and slides again on the shortened sequence (the node attached by the last applied rule is not the root's child any more). Presently, we have around 40 reduction rules and – measured by the number of edges – they constitute the most productive component of the parser.

Interface to the tagset. Morphological information stored in the morphological tags is obviously extremely important for syntactic analysis. However, the reduction rules never access the morphological tags directly, but exclusively via a predefined set of 'interface' routines, as it is apparent also in the above rule samples. This routines are not always straightforward, e.g. the subroutine `adjectival` recognizes not only adjectives, but also possessive pronouns, some of the negative, relative and interrogative pronouns, some numerals etc.

Auxiliary attributes. Besides the attributes already included in the node (word form, lemma, tag, as mentioned above), the parser introduces many new auxiliary node attributes. For instance, the attribute `p_saturated` used above specifies whether the given preposition or subordinating conjunction is already 'saturated' (with a noun or a clause, respectively), or special attributes for coordination. In these attributes, a coordination conjunction which coordinates e.g. two nouns pretends itself to be a noun too (we call it the effective part of speech), so that e.g. a shared attribute modifier can be attached directly below this conjunction.

External lexical lists. Some reduction rules are lexically specific. For this purpose, various simple lexicons (containing e.g. certain types of named entities or basic information about surface valency) have been automatically extracted either from the Czech National Corpus or from the training part of the PDT, and are used by the parser.

Clause segmentation. In any phase of the parsing process, the sequence of parentless nodes is divided into segments separated by punctuation marks or coordination conjunctions; the presence of a finite verb form is tested in every segment, which is extremely important for distinguishing interclausal and intraclausal coordination.[4]

Top-down parsing. The application of the reduction rules can be viewed as bottom-up parsing. However, in some situations it is advantageous to switch to the top-down direction, namely in the cases when we know that a certain sequence of nodes (which we are not able to further reduce by the reduction rules) is of certain syntactic type, e.g. a clause delimited on one side by a subordinating conjunctions, or a complex sentence in a direct speech delimited from both sides by quotes. It is important especially for the application of fallback rules.

Fallback rules. We are not able to describe all language phenomena by the reduction rules, and thus we have to use also heuristic fallback rules in some situations. For instance, if we are to parse something what is probably a single

[4] In our opinion, it is especially coordination (and similar phenomena of non-dependency nature) what makes parsing of natural languages so difficult.

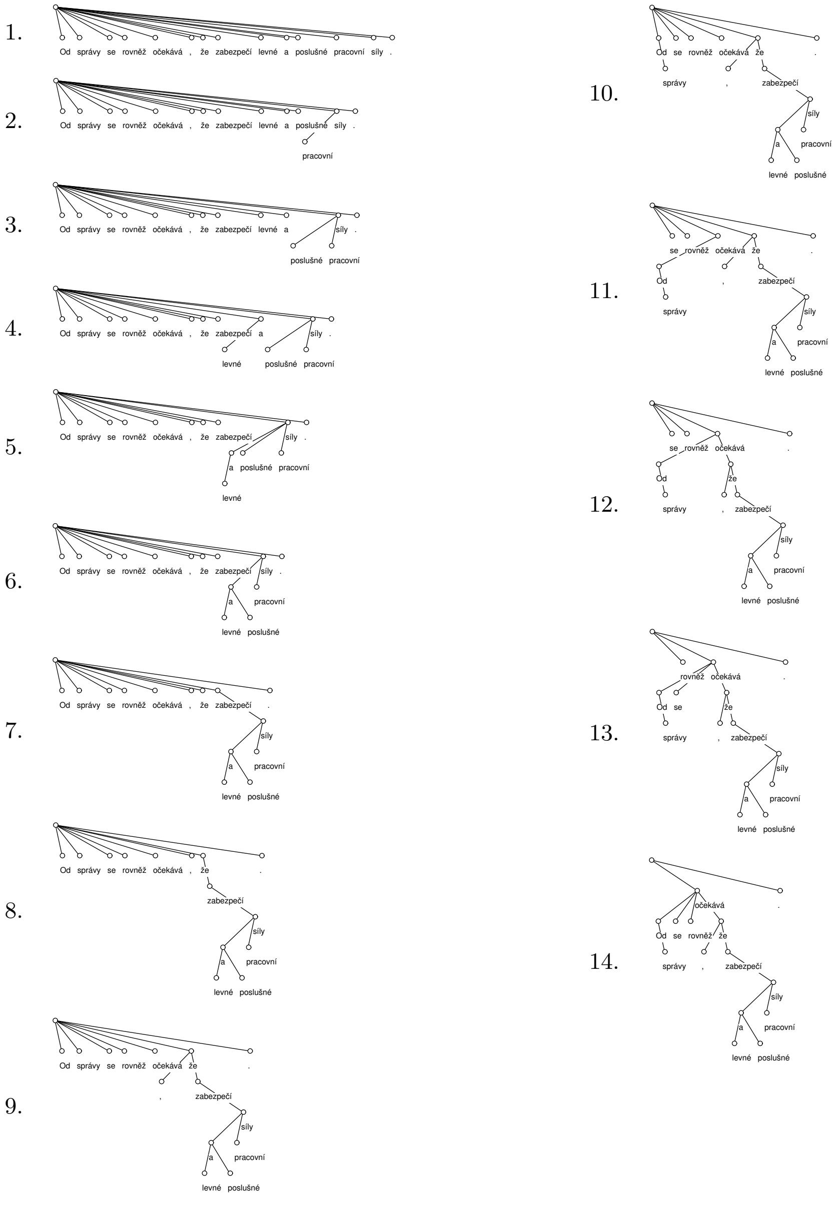


Fig. 1. Step-by-step processing of the sentence *'Od správy se rověž očekává, že zabezpečí levné a poslušné pracovní síly.'* (The administration is also supposed to ensure cheap and obedient manpower.) by the rule-based parser.

clause and no reduction rules are no longer applicable, then the finite verb is selected as the clause head and all the remaining parentless nodes are attached below it (steps 11-14 in Figure 1).

Similar attempts to parsing based on hand-coded rules are often claimed to be hard to develop and maintain because of the intricate interplay of various language phenomena. In our experience and contrary to this expectation, it is possible to reach a reasonable performance (see Table 1), speed and robustness within one or two weeks of development time (less than 2500 lines of Perl code). We have also verified that the idea of our parser can be easily applied on other languages – the preliminarily estimated accuracy of our Slovene, German, and Romanian rule-based dependency parsers is 65-70 % (however, the discussion about porting the parser to other languages goes beyond the scope of this paper).

As for the parsing speed, it can be evaluated as follows: if the parser is executed in the parallelized ntred environment employing 15 Linux servers, it takes around 90 seconds to parse all the PDT 2.0 a-layer development data (9270 sentences), which gives roughly 6.9 sentences per second per server.

3 Pushdown Dependency Parsers

The presented pushdown parser is similar to those described in [3] or [4]. During the training phase, the parser creates a set of premise-action rules, and applies it during the parsing phase. Let us suppose a stack represented as a sequence $n_1..n_j$, where n_1 is the top element; stack elements are ordered triplets $< form, lemma, tag >$. The parser uses four types of actions:

- read a token from the input, and push it into the stack,
- attach the top item n_1 of the stack below the artificial root (i.e., create a new edge between these two), and pop it from the stack,
- attach the top item n_1 below some other (non-top) item n_i, and pop the former from the stack,
- attach a non-top item n_i below the top item n_1, and remove the former from the stack.[5]

The forms of the rule premises are limited to several templates with various degree of specificity. The different templates condition different parts of the stack and of the unread input, and previously performed actions.

In the training phase, the parser determines the sequence of actions which leads to the correct tree for each sentence (in case of ambiguity we use a prespecified preference ordering of the actions). For each performed action, the counters for the respective premise-action pairs are increased.

During the parsing phase, in each situation the parser chooses the premise-action pair with the highest score; the score is calculated as a product of the value of the counter of the given pair and of the weight of the template used in

[5] Note that the possibility of creating edges from or to the items in the middle of the stack enables the parser to analyze also non-projective constructions.

the premise (see [5] for the discussion about template weights), divided by the exponentially growing penalty for the stack distance between the two nodes to be connected.

In the following section we use four versions of the pushdown parser: L2R – the basic pushdown parser (left to right), R2L – the parser processing the sentences in reverse order, L23 and R23 – the parsers using 3-letter suffices of the word forms instead of the morphological tags.

The parsers work very quickly; it takes about 10 seconds to parse 9270 sentences from PDT 2.0 d-test on PC with one AMD Athlon 2500+. Learning phase takes around 100 seconds.

4 Experiments with Parser Combinations

This section describes our experiments with combining eight parsers. They are referred to using the following abbreviations: McD (McDonnald's maximum spanning tree parser, [6]),[6] COL (Collins's parser adapted for PDT, [7]), ZZ (rule-based dependency parser described in Section 2), AN (Holan's parser ANALOG which has no training phase and in the parsing phase it searches for the local tree configuration most similar to the training data, [5]), L2R, R2L, L23 and R32 (pushdown parsers introduced in Section 3). For the accuracy of the individual parsers see Table 1.

We present two approaches to the combination of the parsers: (1) Simply Weighted Parsers, and (2) Weighted Evaluation Classes.

Simply Weighted Parsers (SWP). The simplest way to combine the parsers is to select each node's parent out of the set of all suggested parents by simple parser voting. But as the accuracy of the individual parsers significantly differ (as well as the correlation in parser pairs), it seems natural to give different parsers different weights, and to select the eventual parent according to the weighted sum of votes. However, this approach based on local decisions does not guarantee cycle-free and connected resulting structure. To guarantee its 'treeness', we decided to build the final structure by the Maximum Spanning Tree algorithm (see [6] for references). Its input is a graph containing the union of all edges suggested by the parsers; each edge is weighted by the sum of weights of the parsers supporting the given edge. We limited the range of weights to small natural numbers; the best weight vector has been found using a a simple hill-climbing heuristic search.

We evaluated this approach using 10-fold cross evaluation applied on the PDT 2.0 a-layer d-test data. In each of the ten iterations, we found the set of weights which gave the best accuracy on 90 % of d-test sentences, and evaluated the accuracy of the resulting parser combination on the unseen 10 %. The average accuracy was 86.22 %, which gives 1.98 percent point improvement compared to McD. It should be noted that all iterations resulted in the same weight vector:

[6] We would like to thank Václav Novák for providing us with the results of McD on PDT 2.0.

Table 1. Percent accuracy of the individual parsers when applied (separately) on the PDT 2.0 d-test and e-test data.

	McD	COL	ZZ	AN	R2L	L2R	R23	L23
d-test	84.24	81.55	76.06	71.45	73.98	71.38	61.06	54.88
e-test	83.98	80.91	75.93	71.08	73.85	71.32	61.65	53.28

(10, 10, 9, 2, 3, 2, 1, 1) for the same parser ordering as in Table 1. Figure 2 shows that the improvement with respect to McD is significant and relatively stable.

When the weights were 'trained' on the whole d-test data and the parser combination was evaluated on the e-test data, the resulting accuracy was 85.84 % (1.86 % improvement compared to McD), which is the best e-test result reported in this paper.[7]

Weighted Equivalence Classes (WEC). The second approach is based on the idea of partitioning the set of parsers into equivalence classes. At any node, the pairwise agreement among the parsers can be understood as an equivalence relation and thus implies partitioning on the set of parsers. Given 8 parsers, there are theoretically 4133 possible partitionings (in fact, there are only 3,719 of them present in the d-test data), and thus it is computationally tractable.

In the training phase, each class in each partitioning obtains a weight which represents the conditional probability that the class corresponds to the correct result, conditioned by the given partitioning. Technically, the weight is estimated as the number of nodes where the given class corresponds to the correct answer divided by the number of nodes where the given partitioning appeared.

In the evaluation phase, at any node the agreement of results of the individual parsers implies the partitioning. Each of the edges suggested by the parsers then corresponds to one equivalence class in this partitioning, and thus the edge obtains the weight of the class. Similarly to the former approach to parser combination, the Maximum Spanning Tree algorithm is applied on the resulting graph in order to obtain a tree structure.

Again, we performed 10-fold cross validation using the d-test data. The resulting average accuracy is 85.41 %, which is 1.17 percentage point improvement compared to McD. If the whole d-test is used for weight extraction and the resulting parser is evaluated on the whole e-test, the accuracy is 85.14 %.

The interesting property of this approach to parser combination is that if we use the same set of data both for the training and evaluation phase, the resulting accuracy is the upper bound for of all similar parser combinations based only on the information about local agreement/disagreement among the parsers. If this experiment is performed on the whole d-test data, the obtained upper bound is 87.15 %.

[7] Of course, in all our experiments we respect the rule that the e-test data should not be touched until the developed parsers (or parser combinations) are 'frozen'.

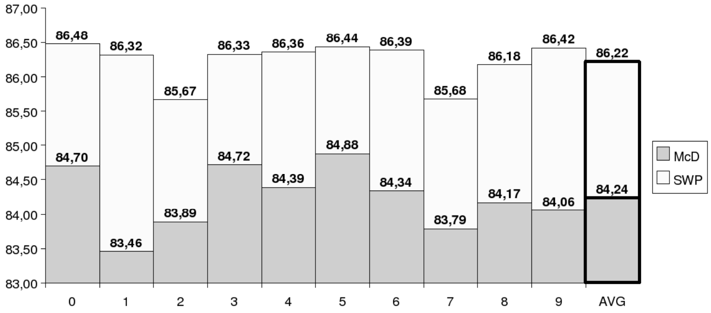


Fig. 2. Accuracy of the SWP parser combination compared to the best single McD parser in 10-fold evaluation on the d-test data.

5 Conclusion

In our opinion, the contribution of this paper is threefold. First, the paper introduces two (types of) Czech dependency parsers, the detailed description of which has not been published yet. Second, we present two different approaches to combining the results of different dependency parsers; when choosing the dependency edges suggested by the individual parsers, we use the Maximum Spanning Tree algorithm to assure that the output structures are still trees. Third, using the PDT 2.0 data, we show that both parser combinations outperform the best existing single parser. The best reported result 85.84 % corresponds to 11.6 % relative error reduction, compared to 83.98 % of the single McDonald's parser.

References

1. Zeman, D.: Parsing with a Statistical Dependency. PhD thesis, Charles University, MFF (2004)
2. Zeman, D., Žabokrtský, Z.: Improving Parsing Accuracy by Combining Diverse Dependency Parsers. In: Proceedings of the 9th International Workshop on Parsing Technologies, Vancouver, B.C., Canada (2005)
3. Holan, T.: Tvorba závislostního syntaktického analyzátoru. In: Sborník semináře MIS 2004. Matfyzpress, Prague, Czech Republic (2004)
4. Nivre, J., Nilsson, J.: Pseudo-Projective Dependency Parsing. In: Proceedings of ACL'05, Ann Arbor, Michigan (2005)
5. Holan, T.: Genetické učení závislostních analyzátorů. In: Sborník semináře ITAT 2005. UPJŠ, Košice (2005)
6. McDonald, R., Pereira, F., Ribarov, K., Hajič, J.: Non-Projective Dependency Parsing using Spanning Tree Algorithms. In: Proceedings of HTL/EMNLP'05, Vancouver, BC, Canada (2005)
7. Hajič, J., Collins, M., Ramshaw, L., Tillmann, C.: A Statistical Parser for Czech. In: Proceedings ACL'99, Maryland, USA (1999)

8.2 Transforming Penn Treebank Phrase Trees into (Praguian) Tectogrammatical Dependency Trees

Full reference:

Kučerová Ivona, Žabokrtský Zdeněk: Transforming Penn Treebank Phrase Trees into (Praguian) Tectogrammatical Dependency Trees, in Prague Bulletin of Mathematical Linguistics, No. 78, Copyright MFF UK, Univerzita Karlova, ISSN 0032-6585, pp. 77–94, 2002

Comments:

The following paper describes the procedure which was used for the automatic generation of tectogrammatical trees in the Prague Czech-English Dependency Treebank, [Cuřín et al., 2004]. The procedure was later reimplemented within English-Czech translation in TectoMT (as shown in Chapter 5). Besides MT, the procedure was also used for preparing data for annotators of English t-trees ([Šindlerová et al., 2007]).

Transforming Penn Treebank Phrase Trees into (Praguian) Tectogrammatical Dependency Trees

Zdeněk Žabokrtský and Ivona Kučerová

Abstract

The aim of this article is to document a work in progress on experiments with transforming a part of the (English) Penn Treebank into tectogrammatical tree structures, similar to those which are defined in the annotation scheme of the (Czech) Prague Dependency Treebank[1]. After a brief outline of the main properties of the sentence representations used in both projects, the transformation from one representation to the other is described in detail. The cornerstones of the transformation are (i) a recursive procedure for translating the topology of a phrase tree into the Praguian tectogrammatical dependency tree topology, (ii) the procedure for functor ("thematic role") assignment, and (iii) the procedure for grammateme assignment.

By applying the transformation to the Wall Street Journal part of the Penn Treebank, the tectogrammatical tree structures for roughly 48,000 English sentences have been automatically created. Roughly 1000 trees have been manually corrected. An evaluation of the differences between the data before and after manual corrections is presented. It also allows for a general estimate of the quality of the automatically created trees.

One of the differences between tectogrammatical and phrase trees is the fact that the original sentences can be trivially reconstructed only from the latter ones. That is why we include here also a few remarks on how to generate sentences from tectogrammatical trees.

1 Introduction

1.1 Penn Treebank, Wall Street Journal

The Penn Treebank project (PTB, [6])[2] consists of about 1,500,000 tokens from English newspaper texts. The treebank bracketing style is based on constituent syntax. Not only syntactic elements, but also several types of structural reconstructions (traces) are realized on the surface. Samples of the PennTreebank bracketing and a set of frequent labels are presented in the Appendix.

The largest subpart of the PTB texts is taken from the Wall Street Journal. PTB project selected 2,499 stories from a three-year Wall Street Journal (WSJ) collection of 98,732 stories for syntactic annotation (1 million words, about 40,000 sentences). The transformation tools described below have been proceeded only on WSJ subpart of the treebank.

1.2 Prague Dependency Treebank and Tectogrammatical Tree Structures

The Prague Dependency Treebank (see [5] for references) is a research project running at the Center for Computational Linguistics[3] and the Institute of Formal and Applied Linguistics[4], Charles University, Prague. It aims at creating a complex annotation of a part of the Czech National Corpus[5]. The sentences are assigned their underlying representations in three steps

[1] The research described herein has been supported by the Ministry of Education of the Czech Republic, project LN00A063, and by the NSF Award #IIS-0121285.

[2] LDC catalog no.: LDC99T42, version 3: `http://www.ldc.upenn.edu/Catalog/LDC99T42.html`

[3] `http://ckl.mff.cuni.cz`

[4] `http://ufal.mff.cuni.cz`

[5] `http://ucnk.ff.cuni.cz`

of annotation: morphological, analytical, and tectogrammatical. The data on the first two levels, which were annotated in a semiautomatic way, consist of more than a million tokens[6]. Presently, the semiautomatic annotation on the third level has been finished for roughly 20,000 sentences.

The annotation of a sentence on the tectogrammatical level results in a *tectogrammatical tree structure* (TGTS). A TGTS is a dependency tree, whose main properties are the following: *(i)* only autosemantic (lexical, meaningful) words have a node of their own; *(ii)* the correlates of function words (i.e. synsemantic, auxiliary words) are attached as labels to the autosemantic words to which they belong (i.e. auxiliary verbs and subordinating conjunctions to the verbs, prepositions to nouns, etc.); coordinating conjunctions remain as nodes of their own; *(iii)* each node is labeled with a *functor* (arguments or theta roles, and adjuncts); *(iv)* the nodes contain a backward link to the node(s) on the second (analytical) level from which they were created, in order to retrieve the information contained there for various purposes (lemma, morphological tag, analytical function, form, surface word order etc.).

A functor represents the role of the node within the sentence, for example Actor, Patient, Addressee, Effect, Origin, various types of spatial and temporal circumstantials, Means, Manner, Condition, etc. There are roughly 60 functors. Functors provide detailed information on the relation between a node and its governing node. See Appendix B for the list of the most frequent functors.

2 Transformation Procedure

2.1 Outline

The transformation of the Penn Treebank phrase trees to the tectogrammatical trees consists of the following steps:

1. **Marking Heads** - the head is chosen in each phrase (using a program written by Jason Eisner ([2]));

2. **Lemmatisation** - a lemma is attached to each word in the sentence (using a program written by Martin Čmejrek (see Chapter 2 in [3]);

3. **Structural Transformations** - the topology of the tectogrammatical tree is derived from the topology of the PTB tree, and each node is labeled with the information from the PTB tree. In this step, the concept of head of a PTB subtree plays a key role;

4. **Functor Assignment** - a functor is assigned to each node of the tectogrammatical tree;

5. **Grammateme Assignment** - morphological (e.g. Tense, Degree of Comparison) and syntactic grammatemes (e.g. TWHEN_AFT(er)) are assigned to each node of the tectogrammatical tree. The assignment of the morphological attributes is based on PennTreebank tags and reflects basic morphological properties of the language. The syntactic grammatemes capture more specific information about deep syntactic structure. At the moment, there are no automatic tools for the assignment of the latter ones.

The transformation tool described in this document covers the last three steps. The tool was written in Perl and consists of roughly 1000 lines of code. The resulting tectogrammatical trees (a sample of which is available on the Internet[7]) are stored in fs-format and can be viewed using the tree editor Tred[8] ([4]).

[6] LDC catalog no.: LDC2001T10, version 1.0; `http://www.ldc.upenn.edu/Catalog/LDC2001T10.html`

[7] http://ckl.mff.cuni.cz/zabokrtsky/wsj2tgts/

[8] http://ckl.mff.cuni.cz/pajas/tred/

2.2 Structural Transformation

The structural transformation can be divided into two steps. First, an "initial dependency tree" (see Fig. 1) is created, in which each word and punctuation mark has its own node[9]. Second, the nodes which are not autosemantic (punctuation marks, prepositions, determiners, subordinating conjunctions, certain particles, auxiliary verbs, modal verbs) are marked as "deleted" (instead of physical deletion, they are just marked as hidden). Selected information from the deleted nodes is copied into the governing autosemantic nodes. Traces are also processed in the second step.

The topology of the initial tree is derived from the topology of the phrase tree by a recursive procedure, which has the following input arguments: phrase tree T_{phr}, initial tree T_{dep}, one particular node s_{phr} from T_{phr} – root of the phrase subtree to be processed, and node p_{dep} from T_{dep} – future parent of the tectogrammatical subtree resulting from s_{Phr} subtree. The recursion looks as follows:

1. if s_{phr} is a terminal node, then create a single tectogrammatical node n_{dep} in T_{dep} and attach it below p_{dep}; return n_{dep},

2. else (it is a nonterminal): choose the head node h_{phr} among the children of s_{phr}, run this recursive procedure with h_{phr} as the phrase subtree root argument, and it returns node r_{dep} (root of the recursively created dep. subtree); run the recursive procedure for each remaining s_{phr}'s child $n_{phr,i}$, get the subtree root $o_{dep,i}$ and attach it below r_{dep}; return r_{dep}.

Obviously, the concept of the head[10] of a phrase subtree plays the key role for the structural transformation. The notion of head used in our approach slightly differs from that of Jason Eisner's head assigning script. Therefore we occasionally use different rules for head selection (for example in case of apposition, prepositional phrases etc.).

Treating Traces. The PennTreebank annotation scheme reflects not only surface realization of a sentence, but it also contains several types of traces. Some of them can be used for generating TGTS nodes that are not realized on the surface.

- A-movement traces: marked as numbered asterisks without other letter specification (e.g. *-1); they can be transformed into several types of coreferential nodes (eg. Cor.ACT) and to nodes of so called general participants (eg. Gen.ACT); the procedure is based on the theoretical assumption that full information about predicate-argument structure is present at the tectogrammatical level; examples can be seen in the Appendix C.2.

- A'-movement traces: they are marked as numbered asterisks with letter "T" specification (e.g. *T*-1); they are used only when assigning functors to wh-word within relative clauses; in the future they could be used also for generating topic-focus articulation as one of the important pieces of information need for capturing this complex phenomenon; an example can be seen in Appendix C.1.

Other types of traces are not captured by the transformation procedure yet.

2.3 Functor Assignment

Various properties of both the phrase tree and the tectogrammatical tree are used for the functor assignment, for example:

[9] However, the initial dependency tree differs from the analytic tree as defined in the annotation scheme of the PDT. For example, the head of a prepositional phrase is not the preposition.

[10] The implementation of the head choosing part of the transformation was partly inspired by a code written by Christian Korthals for similar purposes.

- part-of-speech tags can be used in certain cases; for instance, if a word was tagged as PRP$ (possessive pronoun), then the functor APP (appurtenance) is assigned (PRP$ → APP for short; see the Appendix for the full tag sets), JJ → RSTR, JJR → CPR, etc.
- function tags: BNF → BEN, DTV → ADDR, LGS → ACT, etc.
- lemma: "not" → RHEM, "only" → RHEM, "both" → RSTR, "very" → EXT, etc.

2.4 Grammateme Assignment

Various properties of both the phrase tree and the tectogrammatical tree are used for the grammateme assignment, for example:

- in the case of nouns, number can be derived from the POS-tag (NN and NNP singular, NNS and NNPS plural)
- in the case of certain pronouns, number and gender can be derived from their lemma ("she" FEM SG, etc.)[11]
- the degree of comparison for adjectives and adverbs can be derived from their POS-tag (e.g. JJS → SUP) or from deleted function words (e.g. *more interesting* → COMP)
- tense is derived either from the POS-tag (e.g. VBZ → present) or from the combination of (deleted) auxiliary verbs
- certain grammatemes obtain automatically only their default value (e.g. IT0 for iterativeness).

3 Node Attributes

When the tectogrammatical trees are being created, each node is equipped with many attributes. Some of them are defined within the tectogrammatical level of language description (trlemma, functor, grammatemes). On the other hand, many of them have only different technical functions and do not belong to the (theoretical) tectogrammatical representation as such.

3.1 Technical Attributes

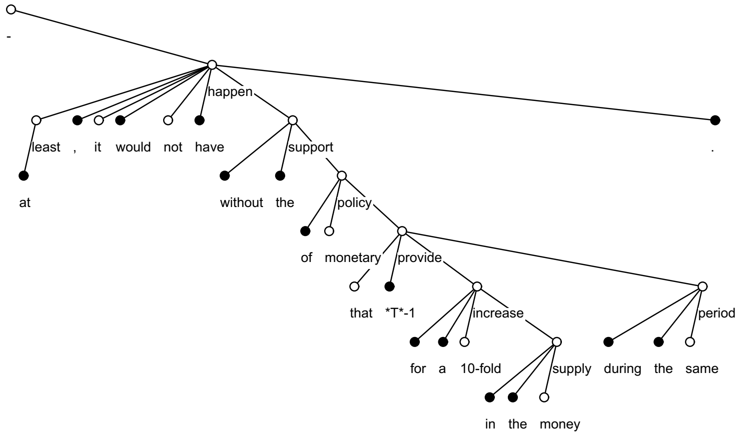
1. FORM - original word form;
2. FW (function word) - word form of a (hidden) preposition or subordinating conjunction attached below the given node;
3. X_PHRASE_SEQUENCE - sequence of labels of non-terminal nodes (of the phrase tree), which "collapsed" into one node of the tectogrammatical tree; the labels are separated by ";"; e.g.: `NN;NP~;PP-TMP;`
4. X_MODALVERB - if a hidden node with a modal verb is governed by the given node, then the word form of the modal verb is copied into this attribute of the autosemantic verb; this grammateme is used for DEONTMOD grammateme assignment;
5. X_AUXVERB_FORMS - the same thing, but with auxiliary verb forms; this attribute is used for verbal grammatemes assignment;

[11] Note that the "surface lemma" of a pronoun might differ from its tectogrammatical lemma, e.g.: "myself" → "I", "she" → "he", etc.

(a) **input data** – the original PTB bracketing enriched with head markers and lemmas:

```
WSJ_1795.MRG:21::(S (ADVP (@IN @at at) (JJS @least least)) (, @, ,) (NP~-SBJ (@PRP @it it))
(@VP (MD @would would) (RB @not not)(@VP~(VB @have have) (@VP~ (@VBN @happened happen) (PP (@IN
@without without) (NP~ (@NP (DT @the the) (@NN @support support)) (PP (@IN @of of) (NP~ (@NP
(JJ @monetary monetary) (@NN @policy policy)) (SBAR (@WHNP-1 (@WDT  @that that)) (S~ (NP~-SBJ
(@-NONE- @*T*-1 *T*-1)) (@VP (@VBD @provided provide) (PP-CLR (@IN @for for) (NP~ (@NP (DT @a a)
(JJ @10-fold 10-fold) (@NN @increase increase)) (PP-LOC (@IN @in in) (NP~ (DT @the the) (NN
@money money) (@NN @supply supply))))) (PP-TMP (@IN @during during) (NP~ (DT @the the) (JJ @same
same) (@NN @period period)))))))))))) (. @. .))
```

(b) **the initial dependency tree**: each word or punctuation mark has its own node; the (technical) root node is added. Nodes to be deleted (prepositions, determiners, modal and auxiliary verbs, punctuation marks, A'-movement trace) are depicted as black circles.



(c) **the resulting tectogrammatical tree**: nodes which are not autosemantic are deleted; values of functors and grammatemes are assigned (only selected verbal grammatemes are visible in the figure).

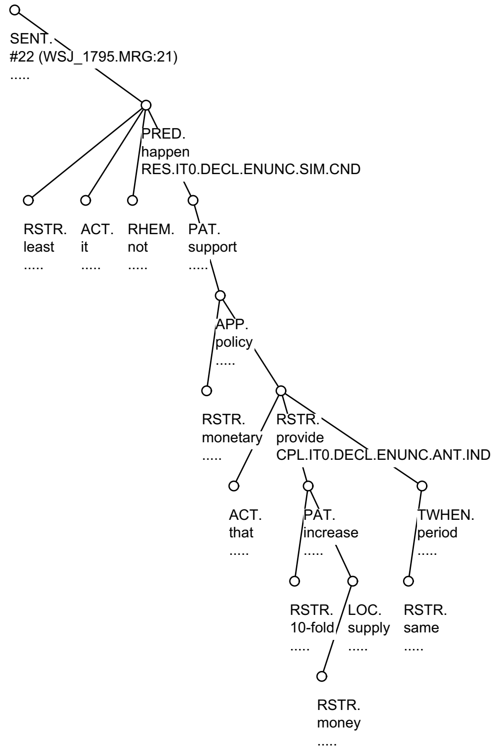


Figure 1: Process of creation of the tectogrammatical tree structure of the sentence "*At least, it would not have happened without the support of monetary policy that provided for a 10-fold increase in the money supply during the same period.*"

6. X_AUXVERB_LEMMAS - the same thing, but with auxiliary verb lemmas; this attribute is used for verbal grammatemes assignment;
7. X_DETERMINER - word form of a hidden child node with a determiner (note: "this", "these" etc. are marked as determiners in the PTB, but we treat them as adjectives);
8. X_WSJ_ID - word identifier (format: sentence_id/word_number), e.g.: `WSJ_1795.MRG:21/13`;
9. X_TRANSLATION - translation of the lemma into Czech, which should ease the manual annotation in the case of complicated sentences (for Czech annotators, obviously).

3.2 Genuine Tectogrammatical Attributes

Note: Some non-essential distinctions between Czech and English can be found even in functor assignment, but much more serious distinctions are expected in the assignment of grammatemes, because the sets of values of grammatemes are much more language dependent. The theoretical distinctions between the tectogrammatical level for English and the tectogrammatical level for Czech have not been properly studied yet, therefore we had to use the tectogrammatical tag set as developed for Czech. This fact might be the source of certain representational inadequacies.

3.2.1 Morphological grammatemes assigned by the automatic procedure

- for verbs and deverbal forms (e.g. gerunds)
  - ASPECT
    - **PROC** - processual, i.e. analogical to the Czech imperfective form; *economist who use.*PROC *the total employment figures*
    - **CPL** - complex, i.e. analogical to perfective form; *the trade gap is expected to widen.*CPL *st.*
    - **RES** - resultative; *experts are thought to have risen.*RES *strongly in August*
  - ITERATIVENESS
    - **IT0** - *economists said.*IT0*: Exports are...*
    - **IT1** - assigned only manually
- for finite verbs only
  - SENTMOD (mode of a sentence)
    - **ENUNC** - indicative mode of the clause (applicable also for relative clauses)
    - **EXCL** - exclamatory; assigned only manually
    - **DESID** - optative; assigned only manually
    - **IMPER** - imperative; assigned only manually
  - VERBMOD (mode of a finite verb)
    - **IND** - indicative mode of the verb; *economist who use.*IND *the total employment figures*
    - **CND** - conditional form of the verb: *they could arrive.*CND *only on Monday*
    - **IMPER** - imperative; assigned only manually
  - DEONTMOD (modality of a verb)
    - **DECL** - non-modal form; *he came.*DECL *on Monday*
    - **DEB** - debitive; *he must come.*DEB

        - **HRT** - hortative; *he should come.*HRT
        - **VOL** - volitive; *he wants to come.*VOL
        - **POSS** - possibilitive; *he can come.*POSS, *he would be able to come.*POSS
        - **PERM** - permissive; *he may come.*PERM
        - **FAC** - facultative; assigned only manually
    - Tense
        - **SIM** - simultaneous; *he wants to come.*SIM; *he has not done.*SIM *it yet*
        - **ANT** - anterior; *he wanted to come.*ANT; *he had not done.*ANT *it before entering the university*
        - **POST** - posterior; *he will come.*POST *on Monday*; *he will have done.*POST *it by Monday*
- for nouns and pronouns
    - Number
        - **SG** - *he met one girl.*SG
        - **PL** - *he met girls.*PL
- only for pronouns
    - Gender
        - **ANIM** - *he*; *his*
        - **FEM** - *she*; *her*
        - **NEUT** - *it*; *its*
- for adjectives and adverbs
    - Degcmp (Degree of Comparison)
        - **POS** - positive; *small,well*
        - **COMP** - comparative; *smaller*; *more interesting*
        - **SUP** - superlative; *smallest*; *the most interesting*

3.2.2 Values of the structural grammateme memberof

- memberof
    - **CO** - at all conjoined items (CONJ) except common dependents; *all.*NIL *boys.*CO *and girls.*CO *came to the party*
    - **AP** - at items of an apposition; *John Benjamin.*AP*, 45.*AP*, was assigned ...*

4 Manual Annotation

In order to gain a "gold standard" (high-quality data set), roughly 1,000 sentences have been manually corrected after the automatic procedure has been run on them.

These data are assigned morphological grammatemes (the full set of values) and syntactic grammatemes, and the nodes within the trees are reordered according to topic-focus articulation.

4.1 Differences from the Automatic Procedure

Differences in assignment of morphological grammatemes can be illustrated on several examples of verbal attributes:

- attribute Iterativeness can be set to IT1 (*he used to play tennis every day*);
- the interpretation of the tense is preferred to morphological realization (*he said he would come.*POST *on Monday*);
- attribute SENTMOD can have other values (*Come!.*IMPER; *Do you want to come?.*INTER) that cannot be assigned automatically according to the punctuation marks because of the presence of finite verbs within relative clauses (*he asked.*ENUNC *whether we could come.*INTER - interrogative mode is based on the lexical semantics of the finite verb).

The assignment of syntactic grammatemes is done only manually. The grammatemes specify the semantic interpretation mainly of temporal and locative functors. This interpretation is closely connected with the form of preposition, but instead of the original form of a preposition it contains a more general value. This set partly bears Czech forms of prepositions.

The assignment of syntactic grammatemes is related to some kind of "neutral" speech situation; this means for example that the sets for different locative functors are the same (*he was at school* vs. *he run to the school* with the same value of the syntactic grammateme).

The list of the most frequent values of syntactic grammateme GRAM:

- basic values (applicable for all functors)
  - **NIL** - unmarked realization
  - **APPX** - approximate value; *it costs about $ 100.*APPX
- values specific for locative and partly temporal functors
  - **v** - *he was in the garden.*LOC_v; *he was at the school.*LOC_v; *he runs to the cinema.*DIR3_v
  - **mezi.1** - *among*, *amid*
  - **mezi.2** - *between*
  - **na** - *knock at the door.*DIR3_na
  - **za** - *behind*
  - **vedle** - *beside*
  - **před** - *in_front_of*
- values only for temporal functors TWHEN and THO
  - **BEF** - *he arrived after the holiday.*BEF
  - **AFT** - *he was there a week ago.*AFT
- other values (used only with relevant functors)
  - functor=BEN (benefactor)
    - **NIL** - *for somebody*
    - **AGST** - *against somebody*

file	# sentences	# words and punct. marks	# tgts nodes	# incorrectly attached nodes	# incorrectly assigned functors
wsj_1789	48	1201	835	46 (5.5%)	143 (17.1%)
wsj_1790	45	1037	762	39 (5.1%)	122 (16.0%)
wsj_1795	60	1551	996	60 (6.0%)	191 (19.1%)
wsj_2100	53	1429	1002	94 (9.4%)	161 (16.0%)
wsj_2104	39	1298	889	49 (5.5%)	201 (22.6%)
total	245	6516	4484	288 (6.4%)	818 (18.2%)

Table 1: Evaluation of the quality of automatically created tectogrammatical trees (differences in deep word order are not counted here).

- functor=ACMP (accompaniment)
  - **NIL** - *with somebody*
  - **WOUT** - *without somebody*
- functor=CPR (comparison)
  - **NIL** - *the economy has become open as the other industrialized nations.*NIL
  - **DFR** - *he is more clever than me.*DFR
- functor=EXT
  - **MORE** - *he is too.*MORE *young to be her brother*
  - **LESS** - *she is almost.*LESS *thirty*
- functor=REG
  - **NIL** - *an excursus of little relevance to its central point.*NIL
  - **WOUT** - *no matter why he couldn't come.*WOUT *they...*

4.2 Evaluation

The evaluation of the automatic procedure is based on a comparison of the automatically generated and then manually corrected sentences from 5 files. The results are summarized in Table 1. Due to the low variation of the error rates, the presented transformation tool seems to be sufficiently robust.

5 Open questions

There are several unsolved topics in the automatic transformation of context-free trees to TGTS. The following three of them seem to us to be the most important:

- **assignment of functors and grammatemes**: finding rules for a better assignment of functors is needed for an automatic assignment of syntactic grammatemes;

- **morphological grammatemes for English**: creating a set of morphological grammatemes specific for English is necessary; solution should not be independent from the revision of the set of functors; for example *one of the most important topics* would be assigned in Czech the functor DIR1 because of the specific Czech surface realization (literally: *one from* ...): the question is whether we should use this functor also for the English variant or whether we should use a different, more general (or more specific?) functor, e.g. for selection from a semantic container or group;

- **topic-focus articulation**: the transformation tool described here doesn't even attempt at solving this problem because of its complexity; possible hints are definite and indefinite articles, information of verbal aspect and A'-movement traces in the original Penn-Treebank data.

6 Remarks on Text Generation

When generating the text from the tectogrammatical trees ([3]), the following difficult problems will have to be faced: how to (i) reconstruct function words, (ii) find an appropriate word order, and (iii) find an appropriate word form for each node.

6.1 Reconstructing function words

- **prepositions** - this is the most difficult problem; the majority of them could be reconstructed using the functor (if there is a dominating surface realization of the functor). However, some functors have a great variation in surface realization, none of them being significantly dominant; it is mainly the case of local and temporal circumstantials, for example in "*The cat slept on/below/behind/near the table*" the functor of "the table" is always LOC. The subtle differences should have been captured via grammatemes, but these are not assigned by the automatic procedure. In other words, when generating from the automatically generated trees, sometimes it is not possible to reconstruct the correct (pre)position of the sleeping cat (without looking into the FW attributes, which is a little bit of cheating).

- **auxiliary verbs** - the auxiliary verbs in complex verb forms can be derived from the combination of the values of the grammatemes ASPECT, SENTMOD, VERBMOD, TENSE (difficult, but feasible); note that negation is not a grammateme, but a child node of the verb node.

- **modal verbs** - grammateme DEONTMOD can be used (DEB → "must", HRT → "should", etc.).

- **subordinating conjunctions** - this should be quite straightforward: a table which maps the functor of the head of the subordinating clause to the appropriate conjunction (COND → "if" etc.) could be hopefully constructed.

- **determiners** - the "official" tectogrammatical level does not give a tool for representing the determiners (it was developed for Czech, which does not have them). It is obvious that in an English sentence the determiners cannot be first deleted and then reconstructed with certainty without knowing the numerous conventions, the world, and—what is the worst—the sentence context. However, after an appropriate study, at least the topic-focus annotation and the deep word order could be used for inserting determiners.

Besides function words, also the punctuation marks have to be reconstructed.

6.2 Finding appropriate word forms

Word forms[12] are not present on the tectogrammatical level and must be derived from the lemma and the values of respective grammatemes. This is trivial in some cases (e.g., generating

[12] Whenever we speak about finding word forms here, we mean in fact finding the appropriate POS tags, from which (and the lemma) the word forms can be easily created using any morphological dictionary of English.

plural for nouns is influenced only by the grammateme NUMBER of the same node), but it is non-trivial in others:

- **subject-verb agreement** - the correct verb form must agree in person and number with the subject (the subject might be coordinated).
- **complex verb forms** - several grammatemes (TENSE, ASPECT, SENTMOD, VERBMOD, DEONTMOD, NUMBER, PERSON), and the existence of the negation child node must be considered when searching for the correct word forms of a given autosemantic verb and possibly auxiliary verb(s).

7 Conclusion

We have shown that the phrase trees from the Penn Treebank can be automatically transformed into the tectogrammatical trees with a reasonably high reliability. The quality evaluation (based on the comparison with manually annotated trees) can be summarized as follows: there are about 6% of wrongly aimed dependecies (wrongly attached nodes), and about 18% of wrongly assigned functors.

Acknowledgment

Our thanks go to Jan Hajič, Eva Hajičová, Jiří Havelka, and Petr Sgall for many useful comments on this text.

References

[1] Bies, Ann, Mark Fergusona, Karen Katz, and Robert MacIntyre. Bracketing Guidelines for Treebank II Style. Penn Treebank Project, University of Pennsylvania (1995)

[2] Eisner, Jason: Smoothing a Probabilistic Lexicon Via Syntactic Transformations. University of Pennsylvania (2001)

[3] Hajič Jan et al.: Generation in the context of MT. Final Report. Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD. In prep. (2002)

[4] Hajič, Jan, Petr Pajas, Barbora Hladká: The Prague Dependency Treebank: Annotation Structure and Support, IRCS Workshop on Linguistic Databases, Philadelphia, PA (2001)

[5] Hajičová, Eva, Jan Hajič, Barbora Hladká, Martine Holub, Petr Pajas, Veronika Řezníčková, and Petr Sgall: The Current Status of the Prague Dependency Treebank, proceedings of Text, Speech and Dialogue, Springer-Verlag (2001)

[6] Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz: Building a Large Annotated Corpus of English: The Penn Treebank, Computational Linguistics (1994)

[7] Sgall, Petr, Eva Hajičová, and Jarmila Panevová: The Meaning of the Sentence and Its Semantic and Pragmatic Aspects. Academia/Reidel Publishing Company, Prague, Czech Republic/Dordrecht, Netherlands (1986)

A Notation used in the Penn Treebank

The following summary was extracted from [1].

A.1 Part-of-Speech Tags

CC coordinating conjunction (*and*)

CD cardinal number (*1, third*)

DT determiner (*the*)

EX existential there (*there is*)

FW foreign word (*d'hoevre*)

IN preposition/subordinating conjunction (*in, of, like*)

JJ adjective (*green*)

JJR adjective, comparative (*greener*)

JJS adjective, superlative (*greenest*)

LS list marker (*1)*)

MD modal (*could, will*)

NN noun, singular or mass (*table*)

NNS noun plural (*tables*)

NNP proper noun, singular (*John*)

NNPS proper noun, plural (*Vikings*)

PDT predeterminer (*¡i¿both¡/i¿ the boys*)

POS possessive ending (*friend's*)

PRP personal pronoun (*I, he, it*)

PRP$ possessive pronoun (*my, his*)

RB adverb (*however, usually, naturally, here, good*)

RBR adverb, comparative (*better*)

RBS adverb, superlative (*best*)

RP particle (*give up*)

TO to (*to go, to him*)

UH interjection (*uhhuhhuhh*)

VB verb, base form (*take*)

VBD verb, past tense (*took*)

VBG verb, gerund/present participle (*taking*)

VBN verb, past participle (*taken*)

VBP verb, sing. present, non-3d (*take*)

VBZ verb, 3rd person sing. present (*takes*)

WDT wh-determiner (*which*)

WP wh-pronoun (*who, what*)

WP$ possessive wh-pronoun (*whose*)

WRB wh-abverb (*where, when*)

A.2 Phrase Labels

S simple declarative clause

SBAR clause introduced by a subord. conjunction

SBARQ direct question introduced by a wh-word

SINV inverted declarative sentence

SQ inverted yes/no question

ADJP adjective phrase

ADVP adverb phrase

CONJP conjunction phrase

FRAG fragment

INTJ interjection

LST list marker

NAC not a constituent

NX something like N-bar level

PP prepositional phrase

PRN parenthetical

PRT particle

QP quantifier phrase

RRC reduced relative clause

UCP unlike coordinated phrase

VP verb phrase

WHADJP wh-adjective phrase

WHADVP wh-adverb phrase

WHNP wh-noun phrase

WHPP wh-prepositional phrase

X unknown

A.3 Function tags

-ADV adverbial

-NOM nominal

-DTV dative

-LGS logical subject

-PRD predicate

-PUT loc. complement of put

-SBJ surface subject

-TPC topicalized

-VOC vocative

-BNF benefactive

-DIR direction

-EXT extent

-LOC locative

-MNR manner

-PRP purpose or reason

-TMP temporal

-CLR closely related

-CLF cleft

-HLN headline

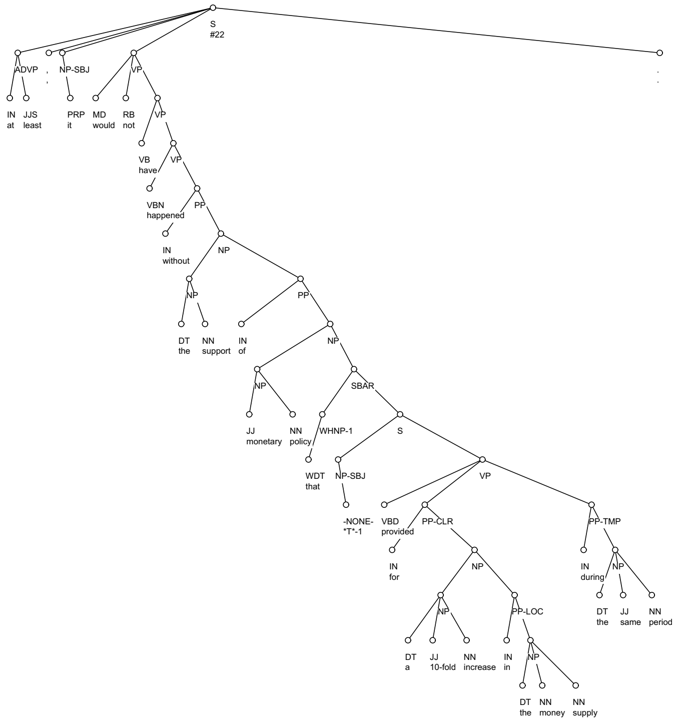
-TTL title

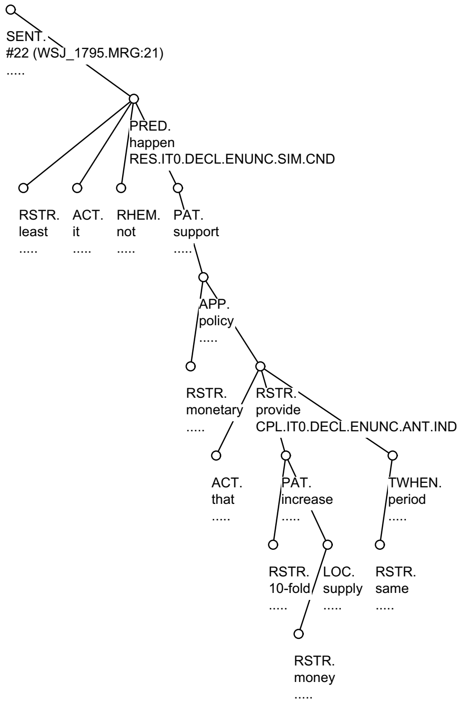
B Alphabetically Ordered List of 40 Functors Most Frequent in the Prague Dependency Treebank

ACMP (accompaniment): mothers with *children*
ACT (actor): *Peter* read a letter.
ADDR (addressee): Peter gave *Mary* a book.
ADVS (adversative): He came there, *but* didn't stay long.
AIM (aim): He came there to *look* for Jane.
APP (appurtenance, i.e., possession in a broader sense): *John's* desk
APPS (apposition): Charles the Fourth, (i.e.) *the Emperor*
ATT (attitude): They were here *willingly.*
BEN (benefactive): She made this for her *children.*
CAUS (cause): She did so since they *wanted* it.
COMPL (complement): They painted the wall *blue.*
COND (condition):If they *come* here, we'll be glad.
CONJ (conjunction): Jim *and* Jack
CPR (comparison): *taller* than Jack
CRIT (criterion): According to *Jim*, it was raining there.
DENOM (denomination): *Chapter 5* (e.g. as a title)
DIFF (difference): taller by two *inches*
DIR1 (direction-from): He went from the *forest* to the village.
DIR2 (direction-through): He went through the *forest* to the village
DIR3 (direction-to): He went from the forest to the *village.*
DISJ (disjunction): here *or* there
DPHR (dependent part of a phraseme): in *no* way, *grammar* school
EFF (effect): We made him the *secretary.*
EXT (extent): *highly* efficient
FPHR (foreign phrase): *dolcissimo*, as they say
ID (entity): the river *Thames*
LOC (locative): in *Italy*
MANN (manner): They did it *quickly.*
MAT (material): a bottle of *milk*
MEANS (means): He wrote it by *hand.*
MOD (mod): He *certainly* has done it.
PAR (parentheses): He has, as we *know*, done it yesterday.
PAT (patient): I saw *him.*
PHR (phraseme): in no *way*, grammar *school*
PREC (preceding, particle referring to context): *therefore*, *how ever*
PRED (predicate): I *saw* him.
REG (regard): with regard to *George*
RHEM (rhematizer, focus sensitive particle): *only*, *even*, *also*
RSTR (restrictive adjunct): a *rich* family
THL (temporal-how-long): We were there for three *weeks.*
THO (temporal-how-often) We were there very *often.*
TWHEN (temporal-when): We were there at *noon.*

C Samples of WSJ phrase trees and their (automatically created) tectogrammatical counterparts

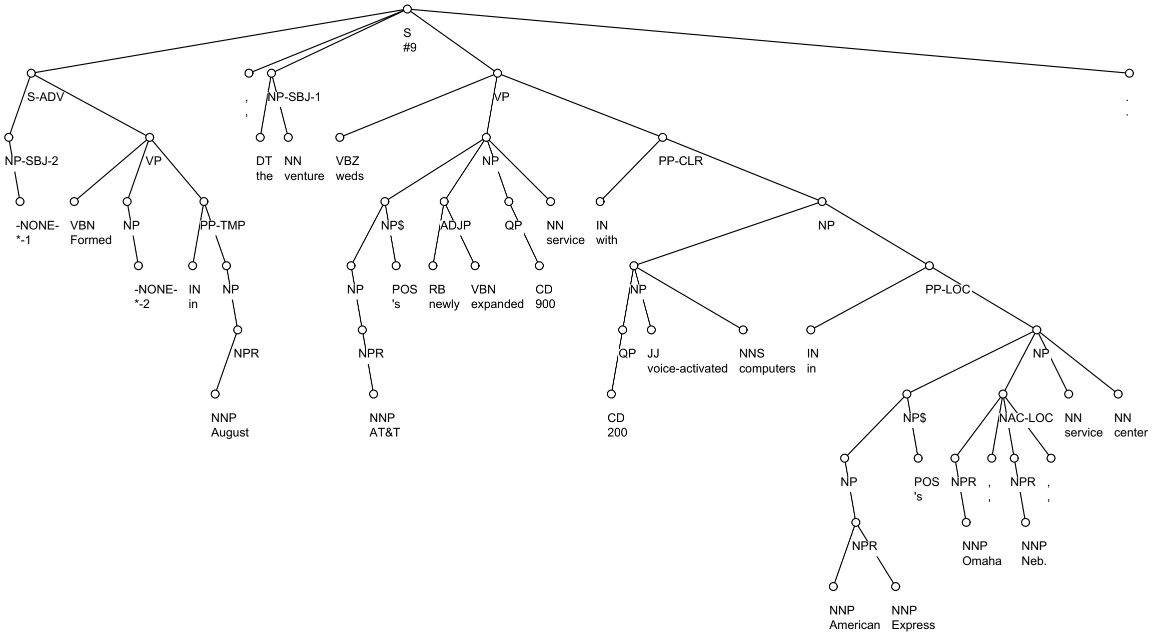
C.1 Sample sentence: *"At least, it would not have happened without the support of monetary policy that provided for a 10-fold increase in the money supply during the same period."*

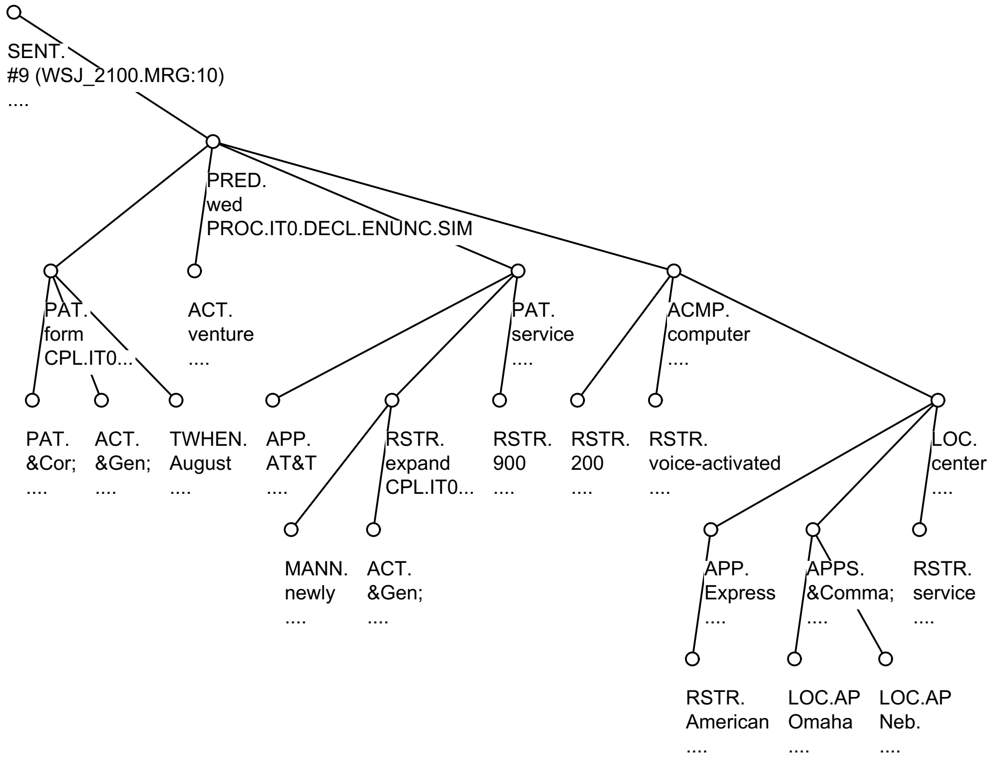




C.2 Sample sentence: *“Formed in August, the venture weds AT&T’s newly expanded 900 service with 200 voice-activated computers in American Express’s Omaha, Neb., service center.”*

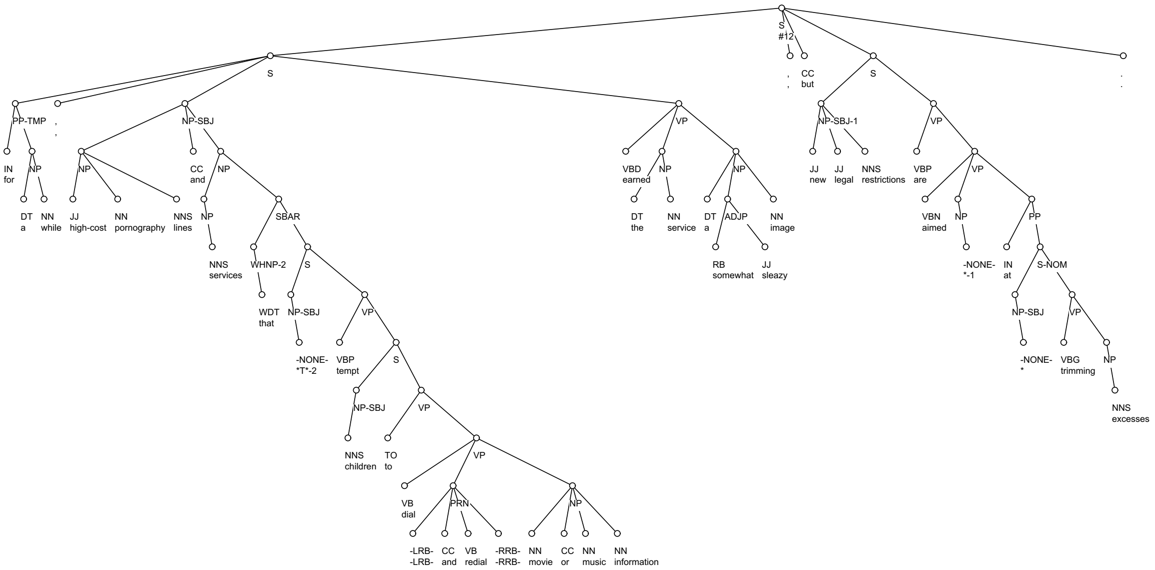
Note the A-movement traces and the apposition (the grammateme MEMBEROF is filled).

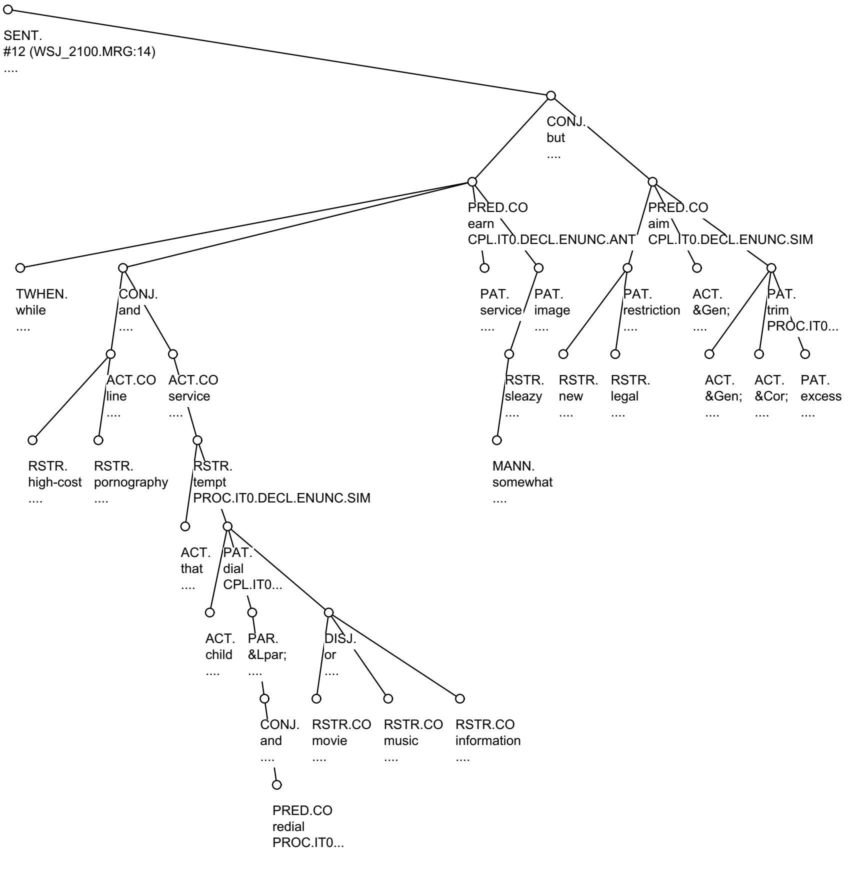




C.3 Sample sentence: *"For a while, high-cost pornography lines and services that tempt children to dial (and redial) movie or music information earned the service a somewhat sleazy image, but new legal restrictions are aimed at trimming excesses."*

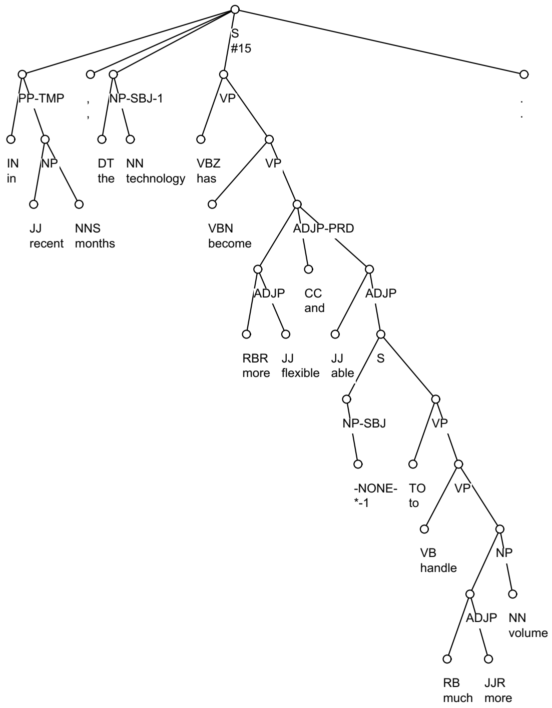
Note how the A-movement traces were processed (for passive voice). The figure also contains the coordination (again, the grammateme MEMBEROF is assigned) and the parenthesis.

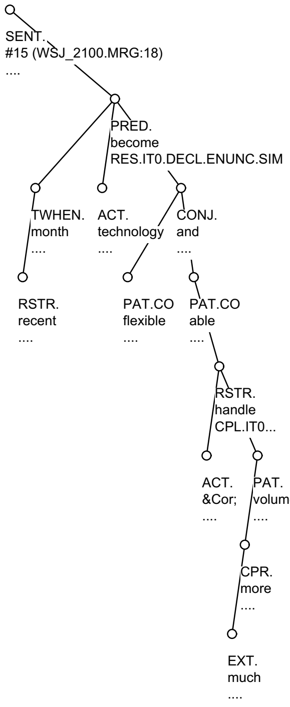




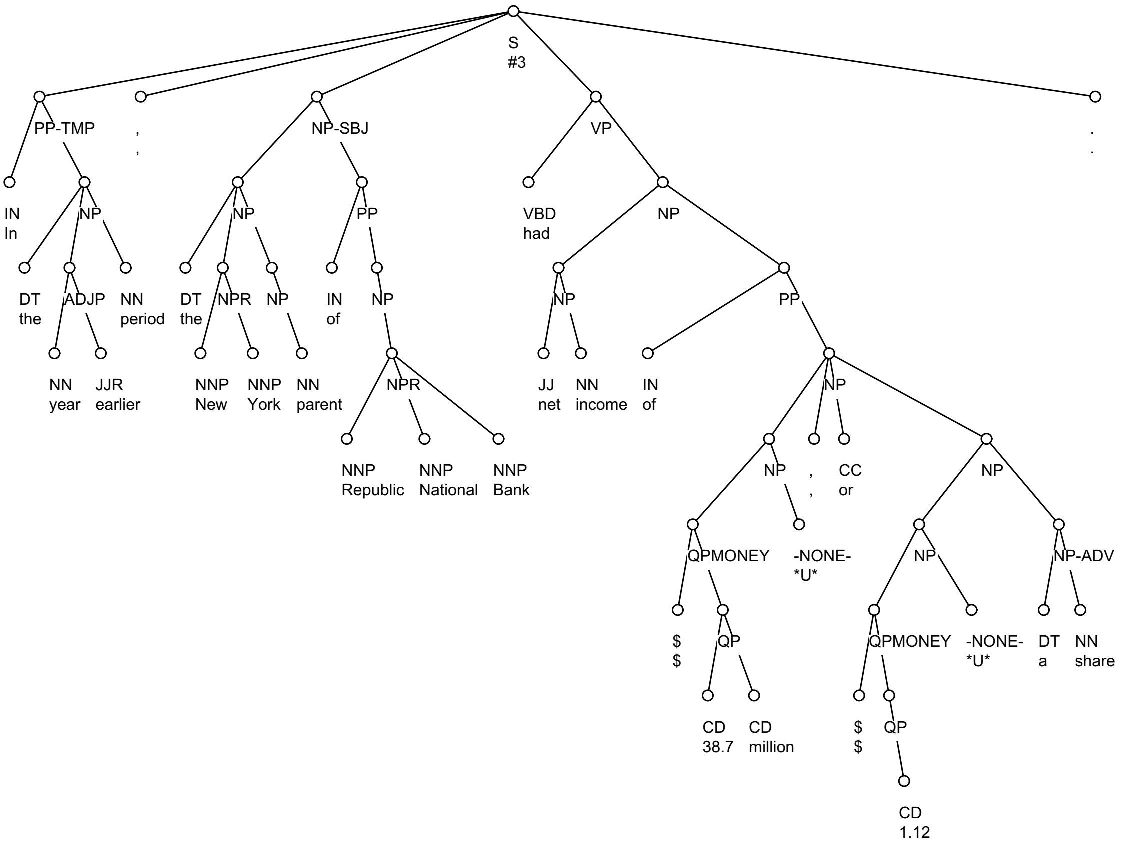
C.4 Sample sentence: *“In recent months, the technology has become more flexible and able to handle much more volume.”*

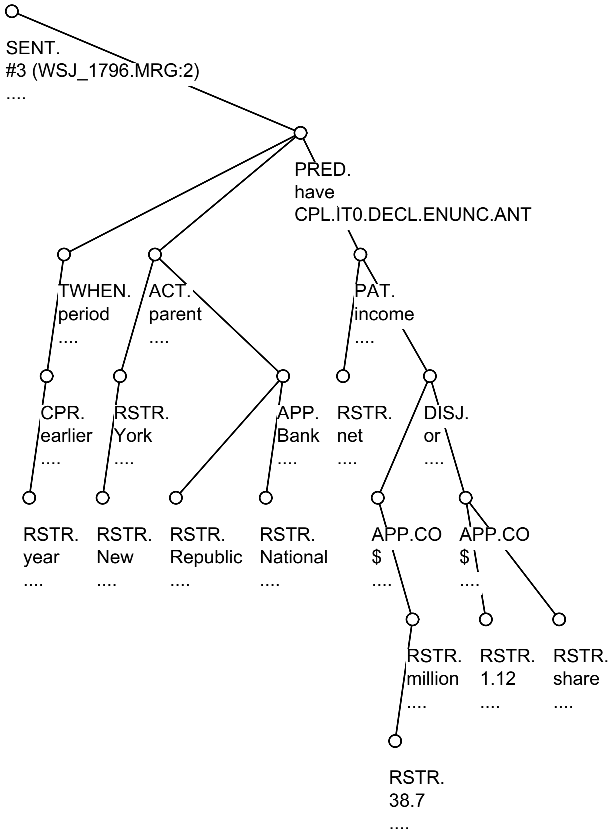
Note the comparative form of adjective *flexible*. The grammateme DEGCMP of the corresponding node is set to COMP.





C.5 Sample sentence: *"In the year earlier period, the New York parent of Republic National Bank had net income of $ 38.7 million, or $ 1.12 a share."*





8.3 Arabic Syntactic Trees: from Constituency to Dependency

Full reference:

Žabokrtský Zdeněk, Smrž Otakar: Arabic Syntactic Trees: from Constituency to Dependency, in EACL 2003 Conference Companion, EACL 2003 Conference Companion, Copyright Association for Computational Linguistics, Budapest, Hungary, ISBN 1-932432-01-9, pp. 183–186, April 2003

Comments:

This paper describes our experiments on transforming Arabic phrase-structure trees from the Penn Arabic Treebank [Maamouri et al., 2003] to dependency trees as defined in the Prague Arabic Dependency Treebank [Hajič et al., 2004]. The motivation was to share the created resources despite of the fact that the underlying formalisms differ.

Recently, the conversion of Arabic phrase-structure trees to dependency trees has been newly implemented by Otakar Smrž, as described in [Smrž et al., 2008]. The new procedure, which employs not only head selection heuristics as the original solution, but also elaborated rules focused on individual syntactic phenomena, is now used for enlarging the forthcoming Prague Arabic Dependency Treebank 2.0 with dependency trees automatically converted from the Penn Arabic Treebank.

Arabic Syntactic Trees: from Constituency to Dependency

Zdeněk Žabokrtský and **Otakar Smrž**
Center for Computational Linguistics
Faculty of Mathematics and Physics
Charles University in Prague
{zabokrtsky,smrz}@ckl.mff.cuni.cz

Abstract

This research note reports on the work in progress which regards automatic transformation of phrase-structure syntactic trees of Arabic into dependency-driven analytical ones. Guidelines for these descriptions have been developed at the Linguistic Data Consortium, University of Pennsylvania, and at the Faculty of Mathematics and Physics and the Faculty of Arts, Charles University in Prague, respectively.

The transformation consists of (i) a recursive function translating the topology of a phrase tree into a corresponding dependency tree, and (ii) a procedure assigning analytical functions to the nodes of the dependency tree.

Apart from an outline of the annotation schemes and a deeper insight into these procedures, model application of the transformation is given herein.

1 Introduction

Exploring the relationship between constituency and dependency sentence representations is not a new issue—the first studies go back to the 60's (Gaifman (1965); for more references, see e.g. Schneider (1998)). Still, some theoretical findings had not been applicable until the first dependency treebanks with well-defined annotation schemes came into existence just in the very last years (Hajič et al., 2001).

The need to convert Arabic treebank data of different descriptions arises from a co-operation between the Linguistic Data Consortium (LDC), University of Pennsylvania, and three concerned institutions of Charles University in Prague, namely the Center for Computational Linguistics, the Institute of Formal and Applied Linguistics, and the Institute of Comparative Linguistics.

The two parties intend to share the resources they create. Prior to this exchange, 10,000 words from the LDC Arabic Newswire A Corpus were manually annotated in both syntactic styles as a step to ensure that the annotations are re-usable and their concepts mutually compatible. Here we attempt the constituency–dependency direction of the transfer.

1.1 Phrase-structure trees

The input data come from the LDC team (Maamouri et al., 2003). The annotation scheme is based on constituent-syntax bracketing style used at the University of Pennsylvania (Maamouri and Cieri, 2002). The trees include nodes for surface text tokens as well as non-terminal nodes following from the descriptive grammar. Not only syntactic elements, but also several kinds of structural reconstructions (traces) are captured here.

1.2 Analytical trees

Under the analytical tree structure we understand a representation of the surface sentence in form of a dependency tree. The node set consists of all the tokens determined after morphological analysis of the text, and the sentence root node. The description recovers the relation between a governor and a node dependent on it. The nature of the government is expressed by the analytical functions of the nodes being linked.

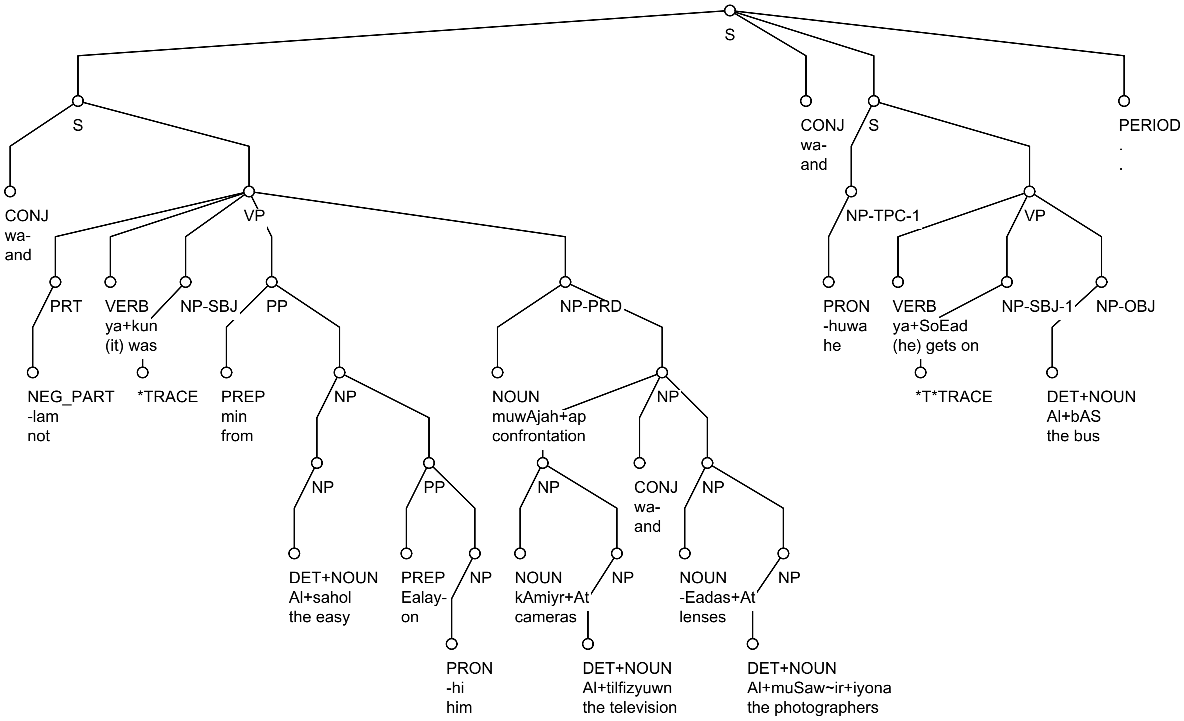


Figure 1: The model sentence in the phrase-structure syntactic description. The nodes are labeled either with part-of-speech (POS) tags, or with the names of non-terminals.

1.3 Model sentence

Let us give a model sentence which in its phonetic transcript and translation reads

> *Wa lam yakun mina ’s-sahli ʿalay hi muwāǧahatu kāmīrāti ’t-tilfizyūni wa ʿadasāti ’l-muṣawwirīna wa huwa yaṣʿadu ’l-bāṣa.*
>
> It was not easy for him to face the television cameras and the lenses of photographers as he was getting on the bus.

Its respective representations in Figures 1 and 2 use glossed tokens which are further split into morphemes and transliterated in Tim Buckwalter’s notation of graphemes of the Arabic script.

There are three phenomena to focus on in the trees. Firstly, occurrence of the empty trace `(*TRACE) NP-SBJ` or the `(*T*TRACE) NP-SBJ-1` one with its contents moved to `NP-TPC-1`. Secondly, subtree interpretation may be sensitive to other than the top-level nodes, like when the coordination `S CONJ S` produces the subordinate complement clause `Pred (Atv)` due to the idiomatic context of the pronoun. Finally to note are complex rearrangements of special constructs, as is the case of `NP-SBJ PP NP-PRD` versus `AuxP AuxP Sb` nodes and their subtrees. More discussion follows.

1.4 Outline of the transformation

The two tree types in question differ in the topology as well as in the attributes of the nodes. Thus, the problem is decomposed into two parts:

i) creation of the dependency tree topology, i.e. contraction of the phrase-structure tree based mostly on the concept of phrase heads and on resolution of traces,

ii) assignment of labels describing the analytical function of the node within the target tree.

2 Structural Transformation

2.1 The core algorithm

The principle of the conversion of phrase structures into dependency structures is described clearly in Xia and Palmer (2001) as (a) mark the

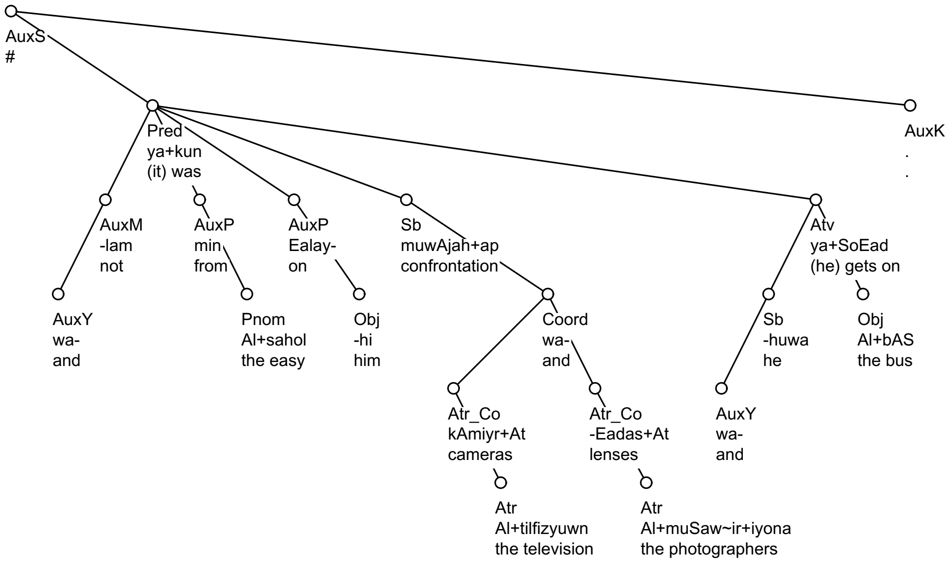


Figure 2: The model sentence in the dependency analytical description, showing the nodes and their functions in the hierarchy.

head child of each node in a phrase structure, using the head percolation table, and (b) in the dependency structure, make the head of each non-head child depend on the head of the head-child.

In our implementation, the topology of the analytical tree is derived from the topology of the phrase tree by a recursive function, which has the following input arguments: original phrase tree T_{phr}, dependency tree T_{dep} being created, one particular node s_{phr} from T_{phr} (the root of the phrase subtree to be processed), and node p_{dep} from T_{dep} (the future parent of the subtree being processed). The function returns the root of the created analytical subtree. The recursion works like this:

1. If s_{phr} is a terminal node, then create a single analytical node n_{dep} in T_{dep} and attach it below p_{dep}; return n_{dep};

2. Otherwise (s_{phr} is a nonterminal), choose the head node h_{phr} among the children of s_{phr}, recursively call the function with h_{phr} as the phrase subtree root argument, and store its return value r_{dep} (root of the recursively created dependency subtree); recursively call the function for each remaining s_{phr}'s child $n_{phr,i}$, and attach the returned subtree root $o_{dep,i}$ below r_{dep}; return r_{dep}.

2.2 Appointing heads

Rules for the selection of phrase heads follow from the analytical annotation guidelines. Predicates are considered the uppermost nodes of a clause, prepositions govern the rest of a prepositional phrase, auxiliary words are annotated as leaves etc. Non-verbal predication, so frequent in Arabic syntax, is also formalized into the terms of dependency, cf. Smrž et al. (2002).

With the algorithm taking decisions about the head child before scanning the subtrees of the level, the already mentioned clause *huwa yaṣʿadu ’l-bāṣa* qualifies improperly as a sister to the predicate *yakun* of the main clause. In fact, we are dealing with the so called state or complement clause. Therefore, corrective shuffling in this respect is inevitable.

2.3 Tree post-processing

Completion of the dependency tree also involves pruning of subtrees which are co-indexed with some trace, and attaching them in place of the referring trace node. Typically, this is the case for clauses having an explicit subject before the predicate. In the model sentence, *yaṣʿadu* retains its role as a predicate of the clause, no matter what function it receives from its governor.

3 Analytical Function Assignment

The analytical function can be deduced well from the POS of the node and the sequence of labels of all its ancestors in the phrase tree, and from the POS or the lexical attributes of its parent in the dependency tree. That is why this step succeeds the structural changes.

Problems may appear though if the declared constituents are not consistent enough, relative to the analytical concept. While `NP-SBJ`, `PP` and `NP-PRD` would normally imply `Sb`, `AuxP` and `Pnom`, these get in principal conflict in the type of nominal predicates like *mina 's-sahli* followed by an optional object and a rhematic subject. The Figures provide the best insight into the differences.

4 Evaluation and Conclusion

Preliminary evaluation gives 60 % accuracy of the generated tree topology, and roughly the same rate for analytical function assignment. The measure is the percentage of correct values of parents/functions among all values. The work is in progress, however. According to our experience with similar task for Czech, English (Žabokrtský and Kučerová, 2002) and German, we expect the performance to improve up to 90 % and 85 % as more phenomena are treated.

The experience made during this task shall be useful for the development of a rule-based dependency partial analysis, which shall pre-process data for manual analytical annotation.

Acknowledgements

Development and fine-tuning of the transformation procedures would not have been possible without the TrEd tree editor by Petr Pajas of the Charles University in Prague. The Figures were produced with it, too.

The phonetic transcription of Arabic within this paper was typeset using the ArabTEX package for TEX and LATEX by Prof. Dr. Klaus Lagally of the University of Stuttgart.

The research described herein has been supported by the Ministry of Education of the Czech Republic, projects LN00A063 and MSM113200006.

References

Haim Gaifman. 1965. Dependency Systems and Phrase-Structure Systems. *Information and Control*, pages 304–337.

Jan Hajič, Eva Hajičová, Petr Pajas, Jarmila Panevová, Petr Sgall, and Barbora Vidová-Hladká. 2001. Prague Dependency Treebank 1.0 (Final Production Label). CDROM CAT: LDC2001T10, ISBN 1-58563-212-0.

Mohamed Maamouri and Christopher Cieri. 2002. Resources for Natural Language Processing at the Linguistic Data Consortium. In *Proceedings of the International Symposium on Processing of Arabic*, pages 125–146, Tunisia, April 18th–20th. Facultè des Lettres, University of Manouba.

Mohamed Maamouri, Ann Bies, Hubert Jin, and Tim Buckwalter. 2003. Arabic Treebank: Part 1 v 2.0. LDC catalog number LDC2003T06, ISBN 1-58563-261-9.

Gerold Schneider. 1998. A Linguistic Comparison Constituency, Dependency, and Link Grammar. Master's thesis, University of Zurich.

Otakar Smrž, Jan Šnaidauf, and Petr Zemánek. 2002. Prague Dependency Treebank for Arabic: Multi-Level Annotation of Arabic Corpus. In *Proceedings of the International Symposium on Processing of Arabic*, pages 147–155, Tunisia, April 18th–20th. Facultè des Lettres, University of Manouba.

Fei Xia and Martha Palmer. 2001. Converting Dependency Structures to Phrase Structures. In *Proceedings of the Human Language Technology Conference (HLT-2001)*, San Diego, CA, March 18–21.

Zdeněk Žabokrtský and Ivona Kučerová. 2002. Transforming Penn Treebank Phrase Trees into (Praguian) Tectogrammatical Dependency Trees. *Prague Bulletin of Mathematical Linguistics*, (78):77–94.

Chapter 9

Verb Valency

9.1 Valency Frames of Czech Verbs in VALLEX 1.0

Full reference:

Žabokrtský Zdeněk, Lopatková Markéta: Valency Frames of Czech Verbs in VALLEX 1.0, in HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation, HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation, Copyright Association for Computational Linguistics, Boston, pp. 70–77, May 2004
Comments:

This paper described the first version of VALLEX, the valency lexicon of Czech verbs, which was later enlarged and recently issued as a book [Lopatková et al., 2008]. There are two contact points between our research on valency and MT. First, the need for describing the limitations of surface forms on individual valency slots led to the introduction of the formeme notion. Second, the valency lexicon is used in TectoMT as a source of verb lists with specific properties (such as verbs allowing genitive, infinitive, or *že*-clause forms in their valency frames, verbs which are reflexiva tantum). We also plan to use it a source of aspectual pairs (perfective and corresponding imperfective verbs), so that the lexical transfer becomes separated from the choice of grammatical aspect (and probabilistic models specialized on the aspect choice could be created).

Valency Frames of Czech Verbs in VALLEX 1.0

Zdeněk Žabokrtský
Center for Computational Linguistics,
Charles University,
Malostranské nám. 25,
CZ-11800 Prague, Czech Republic
zabokrtsky@ckl.mff.cuni.cz

Markéta Lopatková
Center for Computational Linguistics,
Charles University,
Malostranské nám. 25,
CZ-11800 Prague, Czech Republic
lopatkova@ckl.mff.cuni.cz

Abstract

The Valency Lexicon of Czech Verbs, Version 1.0 (VALLEX 1.0) is a collection of linguistically annotated data and documentation, resulting from an attempt at formal description of valency frames of Czech verbs. VALLEX 1.0 is closely related to Prague Dependency Treebank. In this paper, the context in which VALLEX came into existence is briefly outlined, and also three similar projects for English verbs are mentioned. The core of the paper is the description of the logical structure of the VALLEX data. Finally, we suggest a few directions of the future research.

1 Introduction

The Prague Dependency Treebank[1] (PDT) meets the wide-spread aspirations of building corpora with rich annotation schemes. The annotation on the underlying (tectogrammatical) level of language description ((Hajičová et al., 2000)) – serving among other things for training stochastic processes – allows to acquire a considerable amount of data for rule-based approaches in computational linguistics (and, of course, for ’traditional’ linguistics). And valency belongs undoubtedly to the core of all rule-based methods.

PDT is based on Functional Generative Description of Czech (FGD), being developed by Petr Sgall and his collaborators since the 1960s ((Sgall et al., 1986)). Within FGD, the theory of valency has been studied since the 1970s (see esp. (Panevová, 1992)). Its modification is used as the theoretical background in VALLEX 1.0 (see (Lopatková, 2003) for a detailed description of the framework).

Valency requirements are considered for autosemantic words – verbs, nouns, adjectives, and adverbs. Now, its principles are applied to a huge amount of data – that means a great opportunity to verify the functional criteria set up and the necessity to expand the ‘center’, ‘core’ of the language being described.

Within the massive manual annotation in PDT, the problem of consistency of assigning the valency structure increased. This was the first impulse leading to the decision of creating a valency lexicon. However, the potential usability of the valency lexicon is certainly not limited to the context of PDT – several possible applications have been illustrated in ((Straňáková-Lopatková and Žabokrtský, 2002)).

The Valency Lexicon of Czech Verbs, Version 1.0 (VALLEX 1.0) is a collection of linguistically annotated data and documentation, resulting from this attempt at formal description of valency frames of Czech verbs. VALLEX 1.0 contains roughly 1400 verbs (counting only perfective and imperfective verbs, but not their iterative counterparts).[2] They were selected as follows: (1) We started with about 1000 most frequent Czech verbs, according to their number of occurrences in a part of the Czech National Corpus[3] (only ‘*být*’ (to be) and some modal verbs were excluded from this set, because of their non-trivial status on the tectogrammatical level of FGD). (2) Then we added their perfective or imperfective aspectual counterparts, if they were missing; in other words, the set of verbs in VALLEX 1.0 is closed under the relation of ‘aspectual pair’.

The preparation of the first version of VALLEX has taken more than two years. Although it is still a work in progress requiring further linguistic research, the first

[1] http://ufal.mff.cuni.cz/pdt

[2] Besides VALLEX, a larger valency lexicon (called PDT-VALLEX, (Hajič et al., 2003)) has been created during the annotation of PDT. PDT-VALLEX contains more verbs (5200 verbs), but only frames occuring in PDT, whereas in VALLEX the verbs are analyzed in the whole complexity, in all their meanings. Moreover, richer information is assigned to particular valency frames in VALLEX.

[3] http://ucnk.ff.cuni.cz

version has been already publically released. The whole VALLEX 1.0 can be downloaded from the Internet after filling the on-line registration form at the following address: http://ckl.mff.cuni.cz/zabokrtsky/vallex/1.0/

From the very beginning, VALLEX 1.0 was designed with an emphasis on both human and machine readability. Therefore both linguists and developers of applications within the Natural Language Processing domain can use and critically evaluate its content. In order to satisfy different needs of these different potential users, VALLEX 1.0 contains the data in the following three formats:

- **Browsable version**. HTML version of the data allows for an easy and fast navigation through the lexicon. Verbs and frames are organized in several ways, following various criteria.
- **Printable version**. For those who prefer to have a paper version in hand. For a sample from the printable version, see the Appendix.
- **XML version**. Programmers can run sophisticated queries (e.g. based on XPATH query language) on this machine-tractable data, or use it in their applications. Structure of the XML file is defined using a DTD file (Document Type Definition), which naturally mirrors logical structure of the data (described in Sec. 3).

2 Similar Projects for English Verbs[4]

2.1 FrameNet

FrameNet ((Fillmore, 2002)) groups lexical units (pairings of words and senses) into sets according to whether they permit parallel semantic descriptions. The verbs belonging to a particular set share the same collection of frame-relevant semantic roles. The 'general-purpose' semantic roles (as Agent, Patient, Theme, Instrument, Goal, and so on) are replaced by more specific 'frame-specific' role names (e.g. Speaker, Addressee, Message and Topic for 'speaking verbs').

2.2 Levin Verb Classes

Levin semantic classes ((Levin, 1993)) are constructed from verbs which undergo a certain number of alternations (where an alternation means a change in the realization of the argument structure of a verb, as e.g. 'conative alternation' Edith cuts the bread – Edith cuts at the bread). These alternations are specific to English. For Czech, e.g. particular types of diatheses can be considered as useful alternations.

Both FrameNet and Levin classification are focused (at least for the time being) only on selected meanings of verbs.

2.3 PropBank

In the PropBank corpus ((Kingsbury and Palmer, 2002)) sentences are annotated with predicate-argument structure. The human annotators use the lexicon containing verbs and their 'frames' – lists of their possible complementations. The lexicon is called 'Frame Files'. Frame Files are mapped to individual members of Levin classes.

There is only a minimal specification of the connections between the argument types and semantic roles – in principle, a one-argument verb has arg0 in its frame, a two-argument verb has arg0 and arg1, etc. Frame Files store all the meanings of the verbs, with their description and examples.

3 Logical Structure of the VALLEX Data

3.1 Word Entries

On the topmost level, VALLEX 1.0 is divided into word entries (the HTML 'graphical' layout of a word entry is depicted on Fig. 1). Each word entry relates to one or more headword lemmas[5] (Sec. 3.2). The word entry consists of a sequence of frame entries (Sec. 3.5) relevant for the lemma(s) in question (where each frame entry usually corresponds to one of the lemma's meanings). Information about the aspect (Sec. 3.16) of the lemma(s) is assigned to each word entry as a whole.

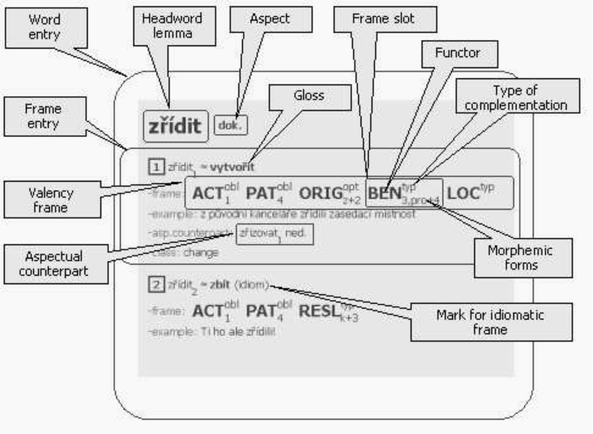


Figure 1: HTML layout of a word entry.

Most of the word entries correspond to lemmas in a simple one-to-one manner, but the following two nontrivial situations (and even combinations of them) appear as well in VALLEX 1.0:

[4] For comparison of PropBank, Lexical Conceptual Database, and PDT, see (Hajičová and Kučerová, 2002).

[5] Remark on terminology: The terms used here either belong to the broadly accepted linguistic terminology, or come from the Functional Generative Description (FGD), which we have used as the background theory, or are defined somewhere else in this text.

- lemma variants (Sec. 3.3)
- homonyms (Sec. 3.4)

The content of a word entry roughly corresponds to the traditional term of lexeme.

3.2 Lemmas

Under the term of lemma (of a verb) we understand the infinitive form of the respective verb, in case of homonym (Sec. 3.4) followed by a Roman number in superscript (which is to be considered as an inseparable part of the lemma in VALLEX 1.0!).

Reflexive particles *se* or *si* are parts of the infinitive only if the verb is reflexive tantum, primary (e.g. *bát se*) as well as derived (e.g. *zabít se*, *šířit se*, *vrátit se*).

3.3 Lemma Variants

Lemma variants are groups of two (or more) lemmas that are interchangable in any context without any change of the meaning (e.g. *dovědět se/dozvědět se*). The only difference usually is just a small alternation in the morphological stem, which might be accompanied by a subtle stylistic shift (e.g. *myslet/myslit*, the latter one being bookish). Moreover, although the infinitive forms of the variants differ in spelling, some of their conjugated forms are often identical (*mysli* (imper.sg.) both for *myslet* and *myslit*).

The term 'lemma variants' should not be confused with the term 'synonymy'.

3.4 Homonyms

There are pairs of word entries in VALLEX 1.0, the lemmas of which have the same spelling, but considerably differ in their meanings (there is no obvious semantic relation between them). They also might differ as to their etymology (e.g. *nakupovat*^{I} - to buy vs. *nakupovat*^{II} - to heap), aspect (Sec. 3.16) (e.g. *stačit*^{I} pf. - to be enough vs. *stačit*^{II} impf. - to catch up with), or conjugated forms (*žilo* (past.sg.fem) for *žít*^{I} - to live vs. *žalo*(past.sg.fem) *žít*^{II} - to mow). Such lemmas (homonyms)[6] are distinguished by Roman numbering in superscript. These numbers should be understood as an inseparable part of lemma in VALLEX 1.0.

3.5 Frame Entries

Each word entry consists of a non-empty sequence of frame entries, typically corresponding to the individual meanings (senses) of the headword lemma(s) (from this point of view, VALLEX 1.0 can be classified as a Sense Enumerated Lexicon).

The frame entries are numbered within each word entry; in the VALLEX 1.0 notation, the frame numbers are attached to the lemmas as subscripts.

The ordering of frames is not completely random, but it is not perfectly systematic either. So far it is based only on the following weak intuition: primary and/or the most frequent meanings should go first, whereas rare and/or idiomatic meanings should go last. (We do not guarantee that the ordering of meanings in this version of VALLEX 1.0 exactly matches their frequency of the occurrences in contemporary language.)

Each frame entry[7] contains a description of the valency frame itself (Sec. 3.6) and of the frame attributes (Sec. 3.13).

3.6 Valency Frames

In VALLEX 1.0, a valency frame is modeled as a sequence of frame slots. Each frame slot corresponds to one (either required or specifically permitted) complementation[8] of the given verb.

The following attributes are assigned to each slot:

- functor (Sec. 3.7)
- list of possible morphemic forms (realizations) (Sec. 3.8)
- type of complementation (Sec. 3.11)

Some slots tend to systematically occur together. In order to capture this type of regularity, we introduced the mechanism of slot expansion (Sec. 3.12) (full valency frame will be obtained after performing these expansions).

3.7 Functors

In VALLEX 1.0, functors (labels of 'deep roles'; similar to theta-roles) are used for expressing types of relations between verbs and their complementations. According to FGD, functors are divided into inner participants (*actants*) and free modifications (this division roughly corresponds to the argument/adjunct dichotomy). In VALLEX 1.0, we also distinguish an additional group of quasi-valency complementations.

Functors which occur in VALLEX 1.0 are listed in the following tables (for Czech sample sentences see (Lopatková et al., 2002), page 43):

Inner participants:

- ACT (actor): *<u>Peter</u> read a letter.*
- ADDR (addressee): *Peter gave <u>Mary</u> a book.*

[6] Note on terminology: we have adopted the term 'homonyms' from Czech linguistic literature, where it traditionally stands for what was stated above (words identical in the spelling but considerably different in the meaning); in English literature the term 'homographs' is sometimes used to express the same notion.

[7] Note on terminology: The content of 'frame entry' roughly corresponds to the term of lexical unit ('lexie' in Czech terminology).

[8] Note on terminology: in this text, the term 'complementation' (dependent item) is used in its broad sense, not related to the traditional argument/adjunct (complement/modifier) dichotomy (or, if you want, covering both ends of the dichotomy).

- PAT (patient): *I saw him.*
- EFF (effect): *We made her the secretary.*
- ORIG (origin): *She made a cake from apples.*

Quasi-valency complementations:

- DIFF (difference): *The number has swollen by 200.*
- OBST(obstacle): *The boy stumbled over a stumb.*
- INTT (intent): *He came there to look for Jane.*

Free modifications:

- ACMP (accompaniement): *Mother came with her children.*
- AIM (aim): *John came to a bakery for a piece of bread.*
- BEN (benefactive): *She made this for her children.*
- CAUS (cause): *She did so since they wanted it.*
- COMPL (complement): *They painted the wall blue.*
- DIR1 (direction-from): *He went from the forest to the village.*
- DIR2 (direction-through): *He went through the forest to the village.*
- DIR3 (direction-to): *He went from the forest to the village.*
- DPHR (dependent part of a phraseme): *Peter talked horse again.*
- EXT (extent): *The temperatures reached an all time high.*
- HER (heritage): *He named the new villa after his wife.*
- LOC (locative): *He was born in Italy.*
- MANN (manner): *They did it quickly.*
- MEANS (means): *He wrote it by hand.*
- NORM (norm): *Peter has to do it exactly according to directions.*
- RCMP (recompense): *She bought a new shirt for 25 $.*
- REG (regard): *With regard to George she asked his teacher for advice.*
- RESL (result): *Mother protects her children from any danger.*
- SUBS (substitution): *He went to the theatre instead of his ill sister.*
- TFHL (temporal-for-how-long): *They interrupted their studies for a year.*
- TFRWH (temporal-from-when): *His bad reminiscences came from this period.*
- THL (temporal-how-long): *We were there for three weeks.*
- TOWH (temporal-to when): *He put it over to next Tuesday.*
- TSIN (temporal-since-when): *I have not heard about him since that time.*
- TWHEN (temporal-when): *His son was born last year.*

Note 1: Besides the functors listed in the tables above, also value DIR occurs in the VALLEX 1.0 data. It is used only as a special symbol for slot expansion (Sec. 3.12).

Note 2: The set of functors as introduced in FGD is richer than that shown above, moreover, it is still being elaborated within the Prague Dependency Treebank. We do not use its full (current) set in VALLEX 1.0 due to several reasons. Some functors do not occur with a verb at all (e.g. APP - appuertenace, '*my.*APP *dog*'), some other functors can occur there, but represent other than dependency relation (e.g. coordination, '*Jim or.*CONJ *Jack*'). And still others can occur with verbs as well, but their behaviour is absolutely independent of the head verb, thus they have nothing to do with valency frames (e.g. ATT - attitude, '*He did it willingly.*ATT').

3.8 Morphemic Forms

In a sentence, each frame slot can be expressed by a limited set of morphemic means, which we call forms. In VALLEX 1.0, the set of possible forms is defined either explicitly (Sec. 3.9), or implicitly (Sec. 3.10). In the former case, the forms are enumerated in a list attached to the given slot. In the latter case, no such list is specified, because the set of possible forms is implied by the functor of the respective slot (in other words, all forms possibly expressing the given functor may appear).

3.9 Explicitly Declared Forms

The list of forms attached to a frame slot may contain values of the following types:

- **Pure (prepositionless) case.** There are seven morphological cases in Czech. In the VALLEX 1.0 notation, we use their traditional numbering: 1 - nominative, 2 - genitive, 3 - dative, 4 - accusative, 5 - vocative, 6 - locative, and 7 - instrumental.
- **Prepositional case.** Lemma of the preposition (i.e., preposition without vocalization) and the number of the required morphological case are specified (e.g., *z+2, na+4, o+6*. . .). The prepositions occurring in VALLEX 1.0 are the following: *bez, do, jako, k, kolem, kvůli, mezi, místo, na, nad, na úkor, o, od, ohledně, okolo, oproti, po, pod, podle, pro, proti, před, přes, při, s, u, v, ve prospěch, vůči, v zájmu,*

z, za. ('*jako*' is traditionally considered as a conjunction, but it is included in this list, as it requires a particular morphological case in some valency frames).

- **Subordinating conjunction.** Lemma of the conjunction is specified. The following subordinating conjunctions occur in VALLEX 1.0: *aby, ať, až, jak, zda,*[9] *že*.
- **Infinitive construction.** The abbreviation 'inf' stands for infinitive verbal complementation. 'inf' can appear together with a preposition (e.g. '*než+inf*'), but it happens very rarely in Czech.
- **Construction with adjectives.** Abbreviation 'adj-digit' stands for an adjective complementation in the given case, e.g. adj-1 (*Cítím se slabý* - I feel weak).
- **Constructions with '*být*'** . Infinitive of verb '*být*' (to be) may combine with some of the types above, e.g. *být+adj-1* (e.g. *zdá se to být dostatečné* - it seems to be sufficient).
- **Part of phraseme.** If the set of the possible lexical values of the given complementation is very small (often one-element), we list these values directly (e.g. '*napospas*' for phraseme '*ponechat napospas*' - to expose).

3.10 Implicitly Declared Forms

If no forms are listed explicitly for a frame slot, then the list of possible forms implicitly results from the functor of the slot according to the following (yet incomplete) lists:

- LOC: adverb, na+6, v+6, u+2, před+7, za+7, nad+7, pod+7, okolo+2, kolem+2, při+6, vedle+2, mezi+7, mimo+4, naproti+3, podél+2 . . .
- MANN: adverb, 7, na+4, . . .
- DIR3: adverb, na+4, v+4, do+2, před+4, za+4, nad+4, pod+4, vedle+2, mezi+4, po+4, okolo+2, kolem+2, k+3, mimo+4, naproti+3 . . .
- DIR1: adverb, z+2, od+2, zpod+2, zpoza+2, zpřed+2 . . .
- DIR2: adverb, 7, přes+4, podél+2, mezi+7, . . .
- TWHEN: adverb, 2, 4, 7, před+7, za+4, po+6, při+6, za+2, o+6, k+3, mezi+7, v+4, na+4, na+6, kolem+2, okolo+2, . . .
- THL: adverb, 4, 7, po+4, za+4, . . .
- EXT: adverb, 4, na+4, kolem+2, okolo+2, . . .
- REG: adverb, 7, na+6, v+6, k+3, při+6, ohledně+2, nad+7, na+4, s+7, u+2, . . .
- TFRWH: z+2, od+2, . . .
- AIM: k+3, na+4, do+2, pro+4, proti+3, aby, ať, že, . . .
- TOWH: na+4 . . .
- TSIN: od+2 . . .
- TFHL: na+4, pro+4, . . .
- NORM: podle+2, v duchu+2, po+6, . . .
- MEANS: 7, v+6,na+6,po+6, z+2, že, s+7, na+4, za+4, pod+7, do+2, . . .
- CAUS: 7, za+4, z+2, kvůli+2, pro+4, k+3, na+4, že, . . .

3.11 Types of Complementations

Within the FGD framework, valency frames (in a narrow sense) consist only of inner participants (both obligatory[10] and optional, 'obl' and 'opt' for short) and obligatory free modifications; the dialogue test was introduced by Panevová as a criterium for obligatoriness. In VALLEX 1.0, valency frames are enriched with quasi-valency complementations. Moreover, a few non-obligatory free modifications occur in valency frames too, since they are typically ('typ') related to some verbs (or even to whole classes of them) and not to others. (The other free modifications can occur with the given verb too, but are not contained in the valency frame, as it was mentioned above (Sec. 3.7))

The attribute 'type' is attached to each frame slot and can have one of the following values: 'obl' or 'opt' for inner participants and quasi-valency complementations, and 'obl' or 'typ' for free modifications.

3.12 Slot Expansion

Some slots tend systematically to occur together. For instance, verbs of motion can be often modified with direction-to and/or direction-through and/or direction-from modifier. We decided to capture this type of regularity by introducing the abbreviation flag for a slot. If this flag is set (in the VALLEX 1.0 notation it is marked with an upward arrow), the full valency frame will be obtained after slot expansion.

If one of the frame slots is marked with the upward arrow (in the XML data, attribute 'abbrev' is set to 1), then the full valency frame will be obtained after substituting this slot with a sequence of slots as follows:

- $\uparrow\mathrm{DIR}^{typ} \rightarrow \mathrm{DIR1}^{typ}\ \mathrm{DIR2}^{typ}\ \mathrm{DIR3}^{typ}$

[9] Note: form '*zda*' is in fact an abbreviation for couple of conjunctions '*zda*' and '*jestli*'.

[10] It should be emphasized that in this context the term obligatoriness is related to the presence of the given complementation in the deep (tectogrammatical) structure, and not to its (surface) deletability in a sentence (moreover, the relation between deep obligatoriness and surface deletability is not at all straightforward in Czech).

- $\uparrow$DIR1^{obl} → DIR1^{obl} DIR2^{typ} DIR3^{typ}
- $\uparrow$DIR2^{obl} → DIR1^{typ} DIR2^{obl} DIR3^{typ}
- $\uparrow$DIR3^{obl} → DIR1^{typ} DIR2^{typ} DIR3^{obl}
- $\uparrow$TSIN^{obl} → TSIN^{obl} THL^{typ} TTIL^{typ}
- $\uparrow$THL^{typ} → TSIN^{typ} THL^{typ} TTIL^{typ}

3.13 Frame Attributes

In VALLEX 1.0, frame attributes (more exactly, attribute-value pairs) are either obligatory or optional. The former ones have to be filled in every frame. The latter ones might be empty, either because they are not applicable (e.g. some verbs have no aspectual counterparts), or because the annotation was not finished (e.g. attribute class (Sec. 3.15) is filled only in roughly one third of frames). Obligatory frame attributes:

- gloss – verb or paraphrase roughly synonymous with the given frame/meaning; this attribute is not supposed to serve as a source of synonyms or even of genuine lexicographic definition – it should be used just as a clue for fast orientation within the word entry!
- example – sentence(s) or sentence fragment(s) containing the given verb used with the given valency frame.

Optional frame attributes:

- control (Sec. 3.14)
- class (Sec. 3.15)
- aspectual counterparts (Sec. 3.16)
- idiom flag (Sec. 3.17)

3.14 Control

The term 'control' relates in this context to a certain type of predicates (verbs of control)[11] and two coreferential expressions, a 'controller' and a 'controllee'. In VALLEX 1.0, control is captured in the data only in the situation where a verb has an infinitive modifier (regardless of its functor). Then the controllee is an element that would be a 'subject' of the infinitive (which is structurally excluded on the surface), and controller is the co-indexed expression. In VALLEX 1.0, the type of control is stored in the frame attribute 'control' as follows:

- if there is a coreferential relation between the (unexpressed) subject ('controllee') of the infinitive verb and one of the frame slots of the head verb, then the attribute is filled with the functor of this slot ('controller');
- otherwise (i.e., if there is no such co-reference) value 'ex.' is used.

Examples:

- *pokusit se* (to try) - control: ACT
- *slyšet* (to hear), e.g. 'slyšet někoho přicházet' (to hear somebody come) - control: PAT
- *jít*, in the sense 'jde to udělat' (it is possible to do it) - control: ex

3.15 Class

Some frames are assigned semantic classes like 'motion', 'exchange', 'communication', 'perception', etc. However, we admit that this classification is tentative and should be understood merely as an intuitive grouping of frames, rather than a properly defined ontology.

The motivation for introducing such semantic classification in VALLEX 1.0 was the fact that it simplifies systematic checking of consistency and allows for making more general observations about the data.

3.16 Aspect, Aspectual Counterparts

Perfective verbs (in VALLEX 1.0 marked as 'pf.' for short) and imperfective verbs (marked as 'impf.') are distinguished between in Czech; this characteristic is called aspect. In VALLEX 1.0, the value of aspect is attached to each word entry as a whole (i.e., it is the same for all its frames and it is shared by the lemma variants, if any).

Some verbs (i.e. *informovat* - to inform, *charakterizovat* - to characterize) can be used in different contexts either as perfective or as imperfective (obouvidová slovesa, 'biasp.' for short).

Within imperfective verbs, there is a subclass of of iterative verbs (iter.). Czech iterative verbs are derived more or less in a regular way by affixes such as *-va-* or *-iva-*, and express extended and repetitive actions (e.g. *čítávat, chodívat*). In VALLEX 1.0, iterative verbs containing double affix *-va-* (e.g. *chodívávat*) are completely disregarded, whereas the remaining iterative verbs occur as aspectual counterparts in frame entries of the corresponding non-iterative verbs (but have no own word entries, still).

A verb in its particular meaning can have aspectual counterpart(s) - a verb the meaning of which is almost the same except for the difference in aspect (that is why the counterparts constitute a single lexical unit on the tectogrammatical level of FGD; however, each of them has its own word entry in VALLEX 1.0, because they have different morphemic forms). The aspectual counterpart(s) need not be the same for all the meanings of the given verb, e.g., *odpovědět* is a counterpart of *odpovídat* - to answer, but not of *odpovídat* - to correspond. Therefore the aspectual counterparts (if any) are listed in frame attribute 'asp. counterparts' in VALLEX 1.0. Moreover, for

[11] Note on terminology: in English literature the terms 'equi verbs' and 'raising verbs' are used in a similar context.

perfective or imperfective counterparts, not only the lemmas are specified within the list, but (more specifically) also the frame numbers of the counterpart frames (which is of course not the case for the iterative counterparts, for they have no word entries of their own as stated above).

One frame might have more than one counterpart because of two reasons. Either there are two counterparts with the same aspect (impf. *působit* and impf. *způsobovat* for pf. *způsobit*), or there are two counterparts with different aspects (impf. *scházet*, pf. *sejít*, iter. *scházívat*).

3.17 Idiomatic frames

When building VALLEX 1.0, we focused mainly on primary or usual meanings of verbs. We also noted many frames corresponding to peripheral usages of verbs, however their coverage in VALLEX 1.0 is not exhaustive. We call such frames idiomatic and mark them with label 'idiom'. An idiomatic frame is tentatively characterized either by a substantial shift in meaning (with respect to the primary sense), or by a small and strictly limited set of possible lexical values in one of its complementations, or by occurence of another types of irregularity or anomaly.

4 Future Work

We plan to extend VALLEX in both quantitative and qualitative aspects. At this moment, word entries for 500 new verbs are being created, and further batches of verbs will follow in near future (selected with respect to their frequency, again). As for the theoretical issues, we intend to focus on capturing the structure on the set of frames/senses (e.g. the relations between primary and metaphorical usages of a verb), on improving the semantic classification of frames, and on exploring the influence of word-formative process on valency frames (for example, regularities in the relations between valency frames of a basic verb and of a verb derived from it by prefixing, are expected).

Acknowledgements

VALLEX 1.0 has been created under the financial support of the projects MSMT LN00A063 and GACR 405/04/0243.

We would like to thank for an extensive linguistic and also technical advice to our colleagues from CKL and UFAL, especially to professor Jarmila Panevová.

References

Charles Fillmore. 2002. Framenet and the linking between semantic and syntactic relations. In *Proceedings of COLING 2002*, pages xxviii–xxxvi.

Jan Hajič, Jarmila Panevová, Zdeňka Urešová, Alevtina Bémová, Veronika Kolářová, and Petr Pajas. 2003. PDT-VALLEX: Creating a Large-coverage Valency Lexicon for Treebank Annotation. In *Proceedings of The Second Workshop on Treebanks and Linguistic Theories*, volume 9 of *Mathematical Modeling in Physics, Engineering and Cognitive Sciences*, pages 57–68. Vaxjo University Press, November 14–15, 2003.

Eva Hajičová and Ivona Kučerová. 2002. Argument/Valency Structure in PropBank, LCS Database and Prague Dependency Treebank: A Comparative Pilot Study. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, pages 846–851. ELRA.

Eva Hajičová, Jarmila Panevová, and Petr Sgall, 2000. *A Manual for Tectogrammatical Tagging of the Prague Dependency Treebank*.

Paul Kingsbury and Martha Palmer. 2002. From Treebank to PropBank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, Las Palmas, Spain.

Beth C. Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press, Chicago, IL.

Markéta Lopatková, Zdeněk Žabokrtský, Karolina Skwarska, and Václava Benešová. 2002. Tektogramaticky anotovaný valenční slovník českých sloves. Technical Report TR-2002-15.

Markéta Lopatková. 2003. Valency in the Prague Dependency Treebank: Building the Valency Lexicon. *Prague Bulletin of Mathematical Linguistics*, (79–80).

Jarmila Panevová. 1992. Valency frames and the meaning of the sentence. In Ph. L. Luelsdorff, editor, *The Prague School of Structural and Functional Linguistics*, pages 223–243, Amsterdam-Philadelphia. John Benjamins.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. D. Reidel Publishing Company, Dordrecht.

Hana Skoumalová. 2002. Verb frames extracted from dictionaries. *The Prague Bulletin of Mathematical Linguistics 77*.

Markéta Straňáková-Lopatková and Zdeněk Žabokrtský. 2002. Valency Dictionary of Czech Verbs: Complex Tectogrammatical Annotation. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, volume 3, pages 949–956. ELRA.

Naďa Svozilová, Hana Prouzová, and Anna Jirsová. 1997. *Slovesa pro praxi*. Academia, Praha.

APPENDIX – Sample from the printed version of VALLEX 1.0

$\boxed{2}$ hrát si$_2$ ≈ **předstírat** (idiom)
–frame: ACT_{1}^{obl} PAT_{na+4}^{obl}
–example: *Petr si hraje na machra*
–asp.counterparts: hrávat si iter.

hrozit impf.

$\boxed{1}$ hrozit$_1$ ≈ **vyhrožovat**
–frame: ACT_{1}^{obl} $ADDR_{3}^{obl}$ $PAT_{7,\text{že}}^{obl}$
–example: *hrozil nám udáním / že nás udá*
–asp.counterparts: hrozívat iter.
–class: communication

$\boxed{2}$ hrozit$_2$ ≈ **vyhrožovat gestem**
–frame: ACT_{1}^{obl} PAT_{3}^{opt} $MEANS_{7}^{typ}$
–example: *hrozil nám rukou*
–asp.counterparts: hrozívat iter.

$\boxed{3}$ hrozit$_3$ ≈ **blížit se**
–frame: ACT_{3}^{obl} $PAT_{1,\text{že}}^{obl}$ LOC^{typ}
–example: *hrozil mu neúspěch; v Mongolsku hrozí hladomor*
–asp.counterparts: hrozívat iter.

hrozit se impf.

$\boxed{1}$ hrozit se$_1$ ≈ **obávat se; děsit se**
–frame: ACT_{1}^{obl} $PAT_{2,aby,\text{že}}^{obl}$
–example: *hrozil se neúspěchu*
–asp.counterparts: hrozívat se iter.

hýbat impf.

$\boxed{1}$ hýbat$_1$ ≈ **pohybovat; měnit polohu něčeho**
–frame: ACT_{1}^{obl} $PAT_{7,s+7}^{obl}$
–example: *hýbat klikou / rukou / s nábytkem*
–asp.counterparts: hýbnout$_1$ pf.

$\boxed{2}$ hýbat$_2$ ≈ **vzbuzovat zájem / rozruch** (idiom)
–frame: ACT_{1}^{obl} PAT_{7}^{obl}
–example: *nové myšlenky hýbou světem*

hýbat se impf.

$\boxed{1}$ hýbat se$_1$ ≈ **pohybovat se; měnit polohu**
–frame: ACT_{1}^{obl} LOC^{typ} $\uparrow DIR^{typ}$
–example: *Nehýbejte se!; větev se hýbá ve větru*
–class: motion

hýbnout pf.

$\boxed{1}$ hýbnout$_1$ ≈ **pohnout; změnit polohu něčeho**
–frame: ACT_{1}^{obl} $PAT_{7,s+7}^{obl}$
–example: *hýbnout hlavou / se skříní*
–asp.counterparts: hýbat$_1$ impf.

CH

charakterizovat biasp.

$\boxed{1}$ charakterizovat$_1$ ≈ **popsat, popisovat; vystihnout, vystihovat**
–frame: ACT_{1}^{obl} PAT_{4}^{obl} $MEANS_{7}^{typ}$ $COMPL_{jako+4}^{typ}$
–example: *problém charakterizoval těmito slovy; ta vlastnost ho dost charakterizuje; charakterizoval přítele jako dobráka*
–class: communication

chodit impf.

$\boxed{1}$ chodit$_1$ ≈ **pohybovat se pomocí nohou; přemísťovat se (s nějakým záměrem)**
–frame: ACT_{1}^{obl} $INTT_{na+4,inf}^{opt}$ $MANN^{typ}$ $\uparrow DIR^{typ}$
–example: *chodit domů pěšky; chodit od hospody k hospodě; chodit rychle; dítě už chodí; chodí stejně (ale jako Jirka.CPR); chodit na borůvky / na nákup / nakupovat; chodit k lékaři na kontroly*
–asp.counterparts: chodívat iter.
–class: motion
–control: ACT

$\boxed{2}$ chodit$_2$ ≈ **absolvovat chůzí**
–frame: ACT_{1}^{obl} PAT_{4}^{obl}
–example: *chodit pochod*
–asp.counterparts: chodívat iter.
–class: motion

$\boxed{3}$ chodit$_3$ ≈ **být doručován** (idiom)
–frame: ACT_{1}^{obl} $BEN_{3,pro+4}^{typ}$
–example: *pošta chodí i v neděli;'chodí špatné zprávy z Rwandy*
–asp.counterparts: chodívat iter.

$\boxed{4}$ chodit$_4$ ≈ **fungovat** (idiom)
–frame: ACT_{1}^{obl} $MANN^{typ}$
–example: *chodit bez chyby o stroji; ten stroj už chodí*
–asp.counterparts: chodívat iter.

$\boxed{5}$ chodit$_5$ ≈ **ujídat** (idiom)
–frame: ACT_{1}^{obl} PAT_{na+4}^{obl} $\uparrow DIR^{typ}$
–example: *chodit na hrušky / na cukroví do komory*
–asp.counterparts: chodívat iter.

$\boxed{6}$ chodit$_6$ ≈ **být upraven** (idiom)
–frame: ACT_{1}^{obl} $PAT_{adj\text{-}1}^{obl}$
–example: *chodit otrhaný; chodí na bál přestrojená*
–asp.counterparts: chodívat iter.

$\boxed{7}$ chodit$_7$ ≈ **mít partnera** (idiom)
–frame: ACT_{1}^{obl} PAT_{s+7}^{obl}
–example: *chodit s někým*
–asp.counterparts: chodívat iter.
–class: social interaction

$\boxed{8}$ chodit$_8$ ≈ **být oblečen** (idiom)
–frame: ACT_{1}^{obl} $COMPL_{jako+1,\ za+4}^{typ}$
–example: *chodí jako maškara / za maškaru o masopustu*
–asp.counterparts: chodívat iter.

9.2 Valency Lexicon of Czech Verbs: Alternation-Based Model

Full reference:

Lopatková Markéta, Žabokrtský Zdeněk, Skwarska Karolina: Valency Lexicon of Czech Verbs: Alternation-Based Model, in Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006) , ELRA, Genova, Italy, ISBN 2-9517408-2-4, pp. 1728-1733, 2006

Comments:

This paper presents the Alternation-based Model introduced in [Žabokrtský, 2005]. The primary aim of the model was to reduce redundancy of the valency lexicon VALLEX by capturing several types of regularities present in the lexicon. From the MT viewpoint, the model could help us in facing the notorious data sparsity problem – less instances of valency frames (of a given verb or a set of verbs) would be necessary to train the translation system if we could profit from knowledge about the relations within the set of frames. However, this idea has not been verified experimentally so far.

Valency Lexicon of Czech Verbs: Alternation-Based Model

Markéta Lopatková, Zdeněk Žabokrtský, Karolina Skwarska

Institute of Formal and Applied Linguistics, Charles University, Prague
Malostranské náměstí 25, Prague 1, 118 00, Czech Republic
{lopatkova,zabokrtsky}@ufal.mff.cuni.cz

Abstract

The main objective of this paper is to introduce an alternation-based model of valency lexicon of Czech verbs VALLEX. Alternations describe regular changes in valency structure of verbs – they are seen as transformations taking one lexical unit and return a modified lexical unit as a result. We characterize and exemplify 'syntactically-based' and 'semantically-based' alternations and their effects on verb argument structure. The alternation-based model allows to distinguish a minimal form of lexicon, which provides compact characterization of valency structure of Czech verbs, and an expanded form of lexicon useful for some applications.

Introduction

The verb is traditionally considered to be the center of the sentence, and the description of syntactic and syntactic-semantic behavior of verbs is a substantial task for linguists. Theoretical aspects of valency are challenging. Moreover, valency information stored in a lexicon (as valency properties are diverse and cannot be described by general rules) belongs to the core information for any rule-based task of NLP (from lemmatization and morphological analysis through syntactic analysis to such complex tasks as e.g. machine translation).

There are tens of different theoretical approaches, tens of language resources and hundreds of publications related to the study of verbal valency in various natural languages. It goes far beyond the scope of this paper to give an exhaustive survey of all these efforts – Žabokrtský (2005) gives a survey and short characteristics of the most prominent projects (i.e. (Fillmore, 2002), (Babko-Malaya et al., 2004), (Erk et al., 2003) and (Mel'čuk and Zholkovsky, 1984)).

The present paper is structured as follows: in the first section the valency lexicon VALLEX is introduced. Section 2. deals with the concept of alternations – we present alternations as transformations that describe regular changes in the valency structure of verbs (and reduce lexicon redundancy). We characterize basic rules for their representation and exemplify basic types of alternations. Section 3. gives a brief sketch of minimal and expanded form of the lexicon.

1. Valency lexicon VALLEX

The valency lexicon VALLEX is a collection of linguistically annotated data and documentation, resulting from an attempt at a formal description of valency frames of roughly 4300 most frequent Czech verbs. It is closely related to Prague Dependency Treebank (PDT), see (Hajič, 2005).[1] VALLEX provides information on the valency structure of verbs in their particular meanings / senses, possible morphological forms of their complementations and additional syntactic information, accompanied with glosses and examples (briefly described below; the theoretical background of Functional Generative Description of Czech is presented in (Sgall et al., 1986) and (Panevová, 1994), its application on VALLEX is specified in (Lopatková, 2003)). All verb entries in VALLEX are created manually; manual annotation and accent put on consistency of annotation are highly time consuming and limit the speed of quantitative growth, but allow for reaching desired quality.

VALLEX version 1.0 was publicly released in autumn 2003. The second version of the lexicon, VALLEX 2.0, which adopted the alternation-based model will be available this autumn (2006) at http://ufal.mff.cuni.cz/vallex/.

1.1. Structure of VALLEX

VALLEX can be seen as having two components, a data component and a grammar component.

Formally, the **data component** consists of word entries corresponding to verb lexemes. Lexeme is an abstract twofold data structure which associates lexical form(s) and lexical unit(s) (see Fig. 1).

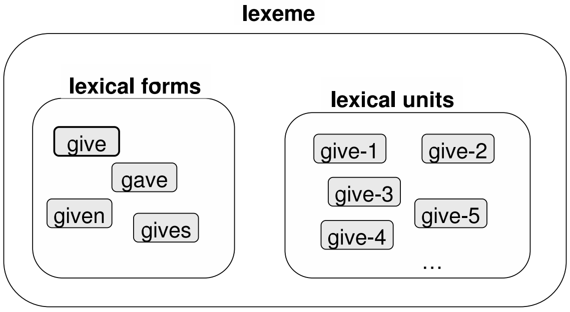


Figure 1: Lexeme, lexical form, and lexical unit.

Lexical forms are all possible manifestations of a lexeme in an utterance, as e.g. perfective, imperfective and iterative verb lemmas, all their morphological manifestations, reflexive and irreflexive forms etc. In the lexicon, all lexical

[1] However, VALLEX is not to be confused with a bit larger valency lexicon PDT-VALLEX created during the annotation of PDT, see (Hajič et al., 2003). PDT-VALLEX has originated as a set of valency frames instantiated in PDT, whereas in the more complex and more elaborated VALLEX verbs are analyzed in all their complexity.

forms of a lexeme are represented by perfective, imperfective and iterative infinitive forms (if they exist), the so called **(headword) lemma(s)**.
Concerning **lexical units (LUs)**, the concept introduced in (Cruse, 1986) has been adopted: LUs are "form-meaning complexes with (relatively) stable and discrete semantic properties". Particular lexical unit is specified by particular meaning / sense, loosely speaking,'given word in the given sense'.[2] Each lexical unit is characterized by a **gloss** (i.e. a verb or a paraphrase roughly synonymous with the given meaning / sense) and by **example(s)** (i.e. sentence fragment(s) containing the given verb used with the given valency frame). The core valency information is encoded in the **valency frame** consisting of a set of **valency members / slots**. Each of these valency members corresponds to an individual – either required or specifically permitted – complementation of the given verb (assigned with its possible morphological forms and a flag for obligatorness). In addition to this obligatory information, also optional attributes may appear in each LU: a flag for idiom, information on control, affiliation to a syntactic-semantic class and a list of alternations that can be applied to this LU (accompanied by examples as illustrated below), see Fig. 2.

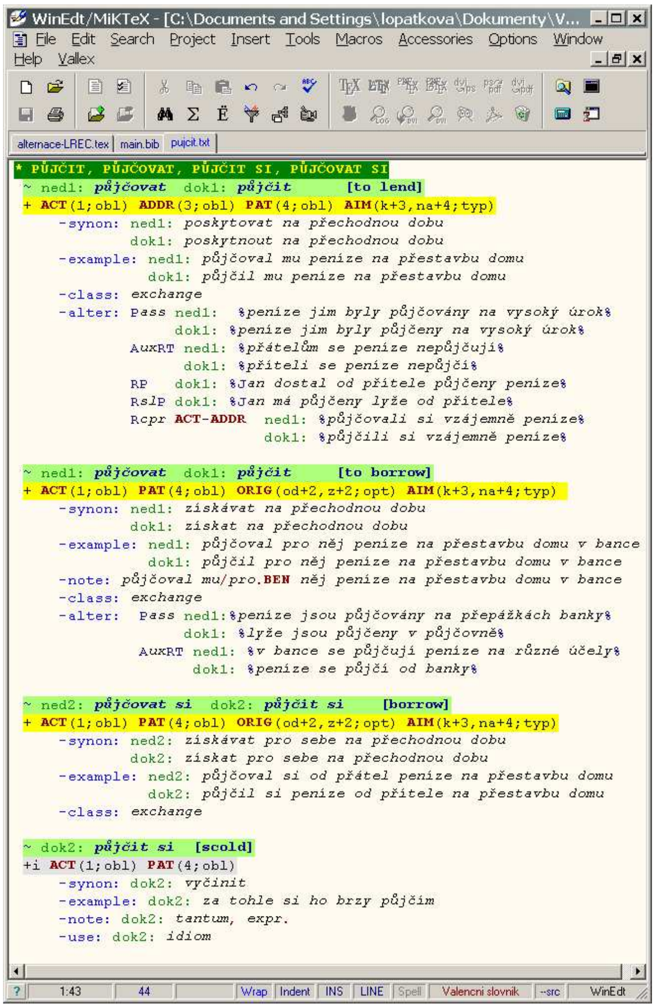
The **grammar component** consists of a set of transformations that can be applied to particular LUs (as specified in the data component) to obtain derived LUs and thus an expanded form of the lexicon. These transformations explicitly cover possible alternation constructions for individual verb forms (they are described in more details in Section 2.2.).

1.2. Basic quantitative characteristics of VALLEX

VALLEX 2.0 contains almost 2100 lexemes. Valency frames of around 6350 LUs are stored in the lexicon. From the other point of view, it describes roughly 4300 verbs (counting perfective forms (ca 1950 verbs), imperfective forms (2250 verbs) as well as biaspectual forms (96 verbs); in addition to these numbers, VALLEX contain also 335 iterative verbs).

2. Alternations

When studying the valency of Czech verbs, it proves to be fruitful to exploit the concept of Levin's alternations (Levin, 1993) and to adapt it for Czech. Levin's alternations describe different changes in argument structure of lexical units. Though our main goal is rather different from that of Levin (Levin builds semantically coherent classes from verbs which undergo particular sets of alternations), the concept of alternations enables us to systematically describe regular changes in argument structure of verbs. Levin recognizes around 45 alternations for English (some of them with more variants). Similar behavior of verbs can be detected in Czech in spite of the typological character of this inflective language. Several of these alternations are described in Czech linguistic works, e.g. in (Daneš, 1985), (Mlu, 1987), (Panevová, 1999), but no Czech lexicon has reflected this model yet.



```
* PŮJČIT, PŮJČOVAT, PŮJČIT SI, PŮJČOVAT SI
 ~ ned1: půjčovat  dok1: půjčit       [to lend]
 + ACT(1;obl) ADDR(3;obl) PAT(4;obl) AIM(k+3,na+4;typ)
     -synon: ned1: poskytovat na přechodnou dobu
             dok1: poskytnout na přechodnou dobu
     -example: ned1: půjčoval mu penize na přestavbu domu
               dok1: půjčil mu penize na přestavbu domu
     -class: exchange
     -alter: Pass ned1:  %penize jim byly půjčovány na vysoký úrok%
                  dok1: %penize jim byly půjčeny na vysoký úrok%
             AuxRT ned1: %přátelům se penize nepůjčuji%
                    dok1: %příteli se penize nepůjčí%
             RP   dok1: %Jan dostal od přítele půjčeny penize%
             RslP dok1: %Jan má půjčeny lyže od přítele%
             Rcpr ACT-ADDR  ned1: %půjčovali si vzájemně penize%
                            dok1: %půjčili si vzájemně penize%

 ~ ned1: půjčovat  dok1: půjčit      [to borrow]
 + ACT(1;obl) PAT(4;obl) ORIG(od+2,z+2;opt) AIM(k+3,na+4;typ)
     -synon: ned1: získávat na přechodnou dobu
             dok1: získat na přechodnou dobu
     -example: ned1: půjčoval pro něj penize na přestavbu domu v bance
               dok1: půjčil pro něj penize na přestavbu domu v bance
     -note: půjčoval mu/pro.BEN něj penize na přestavbu domu v bance
     -class: exchange
     -alter:  Pass ned1:%penize jsou půjčovány na přepážkách banky%
                   dok1: %lyže jsou půjčeny v půjčovně%
              AuxRT ned1: %v bance se půjčuji penize na různé účely%
                     dok1: %penize se půjčí od banky%

 ~ ned2: půjčovat si  dok2: půjčit si     [borrow]
 + ACT(1;obl) PAT(4;obl) ORIG(od+2,z+2;opt) AIM(k+3,na+4;typ)
     -synon: ned2: získávat pro sebe na přechodnou dobu
             dok2: získat pro sebe na přechodnou dobu
     -example: ned2: půjčoval si od přátel penize na přestavbu domu
               dok2: půjčil si penize od přítele na přestavbu domu
     -class: exchange

 ~ dok2: půjčit si  [scold]
 +i ACT(1;obl) PAT(4;obl)
     -synon: dok2: vyčinit
     -example: dok2: za tohle si ho brzy půjčim
     -note: dok2: tantum, expr.
     -use: dok2: idiom
```

Figure 2: VALLEX lexeme for the lemma *půjčit/půjčovat/půjčit si/půjčovat si* [to lend / to borrow]. The highlighting mode in WinEdt text editor, the annotation tool for VALLEX.

The problem is that many verbs can be used in different contexts in the same or only slightly different meanings, which can be accompanied by small changes in their syntactic properties. When describing valency really explicitly, such changes imply introduction of new LUs, which is rather unintuitive and causes problems in building a lexicon (it is a substantial source of inconsistency during annotation, it causes redundancy in the lexicon). As an illustration:

(1) *Martin.ACT nastříkal barvu.PAT na zed'.DIR3*
Martin sprayed paint on the wall.

(2) *Martin.ACT nastříkal zed'.PAT barvou.*MEANS
Martin sprayed the wall with paint.

Clearly, different frames (containing different functors, i.e. labels of 'deep roles')[3] are instantiated in both pairs. Thus we have to have two LUs for these two utterances of verb

[2] This concept of LU corresponds to the Filipec's 'monosemic lexeme' as specified in (Filipec, 1994).

[3] Here the labels ACT and PAT stand for inner participants Actor/Bearer and Patient, respectively, the labels DIR3 and MEANS stand for free modifications Direction-where and Means.

despite the similarity of their meanings. The point here is that instead of having two unrelated LUs in the lexicon, it is more economical (less redundant) to store only one of them (considered as a basic LU) accompanied with information about particular alternation(s) that is/are applicable on this LU (and a derived LU can be generated 'on demand').

2.1. Threefold effect of alternations

In our approach, alternations are seen as transformations that take one LU as an argument and return another LU as a result. The effect of alternations is manifested by (at least one of) the following ways:

- change in **(complex) verb form**,
- change in **valency frame**, i.e.
  - changes in list of valency members,
  - changes in obligatorness of particular members,
  - changes in the sets of possible morphological forms of particular complementations,
- change in **lexical meaning** (with a possible change in the syntactic-semantic class).

Each alternation should be applicable on a whole group of LUs and its manifestation must be completely regular – all the changes (in form, in valency frame as well as in meaning) must be predictable from the input LU and the type of alternation.

2.2. Alternations as transformations

According to the alternation-based model, LUs are grouped into **LU clusters**, as is sketched in Fig. 3. Each cluster contains a **basic LU**, which has to be physically stored in the lexicon, and possibly a number of **derived LUs**, which are present only virtually in the lexicon – these derived LUs are obtained as results of transformations (for alternations applicable on the basic LU).

As the effects of alternations are completely regular, each alternation can be described in the grammar component of the lexicon as **set(s) of transformation rules** that can be applied on a basic LU. These transformations cover all changes in a LU relevant for a particular alternation.

Let us stress here that some alternations can be composed. Thus the LU cluster (see Fig. 3) can be seen as an oriented graph with one distinguished node (basic LU), from which there is an oriented path to all remaining nodes.

Concerning the choice of the basic LU, linguists do not offer in general any simple and explicit solution. Practically, this choice depends on the list of alternations introduced in the lexicon, so it is arbitrary to some extent (only the formal criterion that all other LUs are reachable from the chosen one must be fulfilled). Therefore certain conventions were adopted, some of them more obvious (as e.g. active construction is considered as the basic structure and particular passive constructions as the derived ones), other more arbitrary (as e.g. choice of basic LU for 'cause co-occurrence' alternation, see examples (5)-(6)).

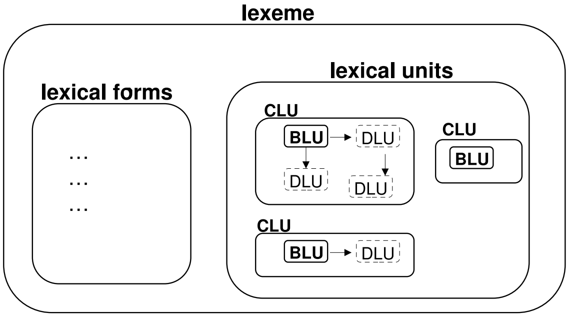


Figure 3: Basic and derived LUs (BLUs and DLUs) forming clusters of LUs (CLU).

Since some alternations can be combined the transformation rules must specify also changes in the list of alternations applicable to the output LU (see below, examples (3)-(4) and (5)-(6)).

The concept of transformations is described in detail on the 'recipient passive' alternation and 'cause co-occurrence' alternation in the following sections.

2.2.1. 'Recipient passive' alternation

The 'recipient passive' alternation can be exemplified on the sentences (3)-(4).

(3) *Pojišťovna.ACT zaplatila výrobcům.ADDR ztráty.PAT*
[insurance_company$_{Nom}$-covered-(to)producers$_{Dat}$-losses$_{Acc}$]
The insurance company covered losses to the producers.

(4) *Výrobci.ADDR dostali od pojišťovny.ACT zaplaceny ztráty.PAT*
[producers$_{Nom}$-got-from-insurance_company$_{Gen}$-covered-losses$_{Acc}$]
The producers have got covered their losses from the insurance company.

The active construction of a meaningful verb (here the verb *zaplatit* [to cover / to pay]) is considered as the basic LU, and thus it is contained in the VALLEX lexicon, see LU in Fig. 4. The set of applicable alternations (together with the examples) is listed in the atribute 'alter'.

It is specified in the grammar component, that the 'recipient passive' construction (marked RP in VALLEX) consists of the finite form of the verb *dostat* [to get] plus passive participle of the meaningful verb. The passive participle has either the form for neuter gender, or it agrees with the noun in accusative case (we draw on the description proposed in (Daneš, 1985) and (Mlu, 1987)).

Clearly, the 'recipient passive' construction has the same valency frame (i.e. the same set of valency complementations) as the active construction. However, the possible morphological forms are different – in active sentence, ACTor is in Nominative and ADDRessee in Dative case; in recipient passive, ACTor is either in Instrumental, or it is realized as a prepositional group *od* [from]+Genitive and ADDRessee is in Nominative (PATient is in Accusative case in both sentences).

```
ZAPLATIT
  ∼ pf: zaplatit [to cover / to pay]
    + ACT(1;obl) ADDR(3;opt) PAT(4;obl)
      -gloss: uhradit [to cover / to pay]
      -example: zaplatit mechanikovi opravu
                   [to pay the repair to a mechanic]
      -class: exchange
      -alter: Pass %oprava byla zaplacena v eurech%
                   [the repair was paid in euros]
              AuxRT %oprava se zaplatila v eurech%
                   [the repair was paid in euros]
              RP %opravu dostali zaplacenu v eurech%
                   [they have got the repair covered in euros]
              RslP %rodiče měli dovolenou zaplacenu %
                   [parents have the holidays paid]
              Rcpr    ACT-ADDR
                   %zaplatili si (navzájem) všechny pohledávky%
                   [they covered their claims each to other]
```

Figure 4: The basic LU for the particular sense of the verb *zaplatit* [to cover / to pay] in the annotation format.

In VALLEX, a transformation notation developed by Petr Pajas (originally used for consistency checking of valency frames in PDT) was adopted for describing different types of alternations. Informally, the set of rules for RP alternation looks as follows:

- change in verb form:
  ⇒ +*dostat* [to get], finite form
  active ⇒ passive participle
  (neuter gender | agreement with the noun in Accusative)
- changes in valency frame :
  not applicable (NA in the sequel)
- changes of possible morphological forms:
  ACT(1) ⇒ – ACT(1), +ACT(7), +ACT(od+2)
  ADDR(3) ⇒ – ADDR(3), +ADDR(1)[4]
- change of syntactic-semantic class:
  NA
- change in the list of applicable alternations:
  ⇒ – Pass
  ⇒ – AuxRT
  ⇒ – RP
  ⇒ – RslP
  ⇒ – Rcpr

As a result of this transformation rule (applied to the basic LU for the verb *zaplatit* [to cover / to pay]), the derived LU for the 'recipient passive' construction is obtained, see Fig. 5 (the example is copied from the relevant alter attribute of the basic LU).

2.2.2. 'Cause co-occurrence' alternation

The 'cause co-occurrence' alternation concerns a group of verbs that express putting things / substances into containers or putting them on surface (for Czech described in (Daneš, 1985), for English see (Levin, 1993), Section 2.3).

```
  ∼ pf: zaplatit [to cover / to pay]
    + ACT(7,od+2;obl) ADDR(1;opt) PAT(4;obl)
      -gloss: uhradit [to cover / to pay]
      -example: opravu dostali zaplacenu v eurech
                   [they have got the repair covered in euros]
      -class: exchange
```

Figure 5: The derived LU for the 'recipient passive' construction for the verb *zaplatit* [to cover / to pay].

(5) *Dělníci.ACT* naložili vagony.PAT uhlím.MEANS
The workers loaded the wagons with coal.

(6) *Dělníci.ACT naložili uhlí.PAT do vagonů.DIR3*
The workers loaded coal on the wagons.

Sentences (5)-(6) show two possible underlying syntactic structures that these verbs can create, see Table 1.

	agens / causator	container / surface	thing / substance
ex. (5)	ACT	PAT	MEANS
ex. (6)	ACT	DIR3	PAT

Table 1: Two possible underlying syntactic structures for the 'cause co-occurrence' alternation.

In VALLEX, the syntactic structure realized in the sentence (5) is considered as the primary one – thus the basic LU for the relevant sense of the verb *nakládat / naložit* [to load] is such as in Fig. 6 ('CCo' labels 'cause co-occurrence' alternation). All alternations applicable to this verb sense are presented here just to illustrate the possibility of alternations to compose.

```
NAKLÁDAT, NALOŽIT
  ∼ impf: nakládat pf: naložit [to load]
    + ACT(1;obl) PAT(4;obl) MEANS(;typ)
      -gloss: impf: plnit pf: naplnit [to load]
      -example: impf: nakládat vůz senem
                pf: naložit vůz senem
                [to load a wagon with hay]
      -class: providing
      -alter:
          Pass impf: %vozy byly nakládány dřevem po okraj%
                pf: %vozy byly naloženy dřevem po okraj%
                [wagons were loaded with timber to the brim]
          AuxRT impf: %vozy se nakládaly dřevem po okraj%
                pf: %vozy se naložily dřevem po okraj%
                [wagons were loaded with timber to the brim]
          RslP pf: %mít vůz naložený dřevem po okraj%
                [to have wagon loaded with timber to the brim]
          CCo impf: %nakládat seno na vůz%
                pf: %naložit seno na vůz%
                [to load hay on wagon]
```

Figure 6: The basic LU for the particular sense of the verb *nakládat / naložit* [to load].

[4] This is interpreted as: concerning ACT, remove Nominative case, add Instrumenal and prepositional group *od*+Genitive; concerning ADDR, remove Dative case and add Nominative.

The transformation rule in the grammar component of VALLEX specifies the way how to obtain a derived LU for particular alternations. Concerning CCo, the following changes are relevant:

- change in verb form:
  NA
- changes in valency frame (list of complementations as well as obligatorness of particular members):
  MEANS ⇒ – MEANS
  ⇒ +DIR3(;obl)
- changes of possible morphological forms:
  NA
- change of syntactic-semantic class:
  providing ⇒ location
- change in list of applicable alternations:
  ⇒ – CCo

The result of the CCo transformation rule applied to the appropriate basic LU for the verb *nakládat / naložit* [to load] is shown in Fig. 7.

NAKLÁDAT, NALOŽIT
∼ impf: *nakládat* pf: *naložit* [to load]
+ ACT(1;obl) PAT(4;obl) DIR3(;obl)
-gloss: impf: *plnit* pf: *naplnit* [to load]
-example: impf: *nakládat seno na vůz*
pf: *naložit seno na vůz*
[to load hay on wagon]
-class: *location*
-alter: Pass
AuxRT
RslP

Figure 7: The derived LU for the 'cause co-occurrence' alternation for the verb *nakládat / naložit* [to load].

As the lists of alternations applicable to derived LU's are gained from the transformation rules in the grammar component (not from the data component), there cannot be examples of their instantiations in derived LUs (we minimize this minus by ordering alternations, see Section 2.3.).

2.3. Typology of alternations

Basically, we distinguish two groups of alternations, tentatively characterized as 'syntactically-based' alternations and 'semantically-based' ones.

2.3.1. 'Syntactically-based' alternations

A group of 'syntactically-based' alternations primarily consists of different types of 'diathesis' in Czech. Further, reciprocal alternations are ranged with this type and also some additional (more sparse) constructions. These alternations are characterized by changes in the verb form.

We have exemplified some of these alternations in the previous section in Figures 4 and 6, where label Pass stands for passive voice, AuxRT for reflexive passive, RP and RslP for recipient and resultative passive with *dostat* [to get] and *mít* [to have], respectively, plus passive participle constructions. We take into account also, e.g., alternations IP-I and IP-II for constructions *dát / nechat* plus infinitive (as in *dává / nechává si vyprat špinavé košile* [he has/gets his dirty shirts washed]). Label Rcpr (see Fig. 4) is used for reciprocal constructions described for Czech in (Panevová, 1999).

The 'syntactically-based' alternations cover constructions described in details in Czech grammars, another 'diatheses' are regular enough to be covered by general rules (e.g. 'dispositional modality' or impersonal constructions), so it is redundant to store them in a lexicon (see esp. (Mlu, 1987) and (Daneš, 1985)).

2.3.2. 'Semantically-based' alternations

Let us give here at least several examples to illustrate 'semantically-based' alternations. Levin stated that alternations are language dependent, though several of English examples have their Czech counterparts, e.g. 'cause co-occurrence' alternation (see examples (1)-(2)) together with its variant 'lose co-occurrence' alternation match with Levin's 2.3 alternations. The following Table 2 shows some other examples of semantically-based alternations (examples marked with * are described in (Benešová, 2004)).

1.4	*vyjít kopec / vyjít na kopec** [to climb the mountain / to climb up the mountain]
2.4	*chlapec roste v muže / z chlapce roste muž* [a boy grows into a man / a man grows from a boy]
1.1	*Slunce vyzařuje teplo / teplo vyzařuje ze slunce* [the Sun radiates heat / heat radiates from the Sun]
2.1	*poslat dopis mamince / poslat peníze do Indie** [to send mamma a letter / to send money to India]
???	*soustředit se v centru města/ soustředit se do centra** [to mass in the city center / to mass into the city center]

Table 2: Examples of corresponding Czech and English alternations (numbers in first column stand for Levin's types of alternations).

Distinguishing two basic groups of alternations is not an enterprize for its own sake – these two groups exhibit different behavior:

- Alternations belonging to the same group typically cannot be composed (with the rare exception of Rcpr alternation where subject is not involved – this case must be treated separately).
- Typically, alternations from different groups can be mutually composed.
- Though in general, alternations from different groups can be composed in any order, we have not found a single example where the order of composition is relevant. That means that the result of composition is the same regardless the order.

These observations result in an important constraint – it allows us to prescribe the order in which alternations can be composed: if two alternations are to be applied to any LU, then the 'semantically-based' one is (by convention) considered as the first one, the 'syntactically-based' one follows.

This constraint has both theoretical and practical impact. It guarantees the tree structure of LU clusters (compare Fig. 3 in Section 2.). From the practical point of view it ensures that 'semantically-based' alternations are exemplified in the lexicon. Considering the exhaustive description of passive constructions in grammar books (and also description of other constructions which come under ‘syntactically-based’ alternations), it seems to be acceptable to have these types of alternations without examples in the expanded form of the lexicon.

3. Minimal and expanded form of the lexicon

The VALLEX lexicon (in its minimal form) contains only the basic LU with an associated list of applicable alternations. However, there are various tasks for which it could be useful to include the derived LUs to the lexicon (e.g. frame disambiguation, i.e. assigning LUs to verb occurrences in text). This requirement leads to distinguishing minimal and expanded form of valency lexicon VALLEX – the expanded one (containing all LUs covered either explicitly or implicitly in the lexicon) can be derived from the minimal one (containing only basic LUs) by a fully automatic procedure. The formal alternation-based model of VALLEX is described in details in (Žabokrtský, 2005), where also the main software components of the dictionary production system developed for VALLEX are outlined (including annotation format, www interface for searching the text format as well as XML data format).

Conclusions

Despite the variety of valency behavior of lexical units, in the valency lexicon of Czech verbs VALLEX the stress is laid on an adequate and consistent description of regular properties of verbs as lexical units. The alternation-based model gives a more powerful description of Czech verbs and shows regular changes in their argument structure. It makes it possible to decrease redundancy in the lexicon and to make the lexicon more consistent.

In future, we will especially focus on the ‘semantically-based’ alternations in Czech, the adequate description of which requires further linguistic research. We aim to empirically confirm the adequacy of tree-structure constraint on LU clusters. Depending on the progress in this field, we intend to involve newly specified alternations to the lexicon. We plan to extend VALLEX also in quantitative aspects.

The alternation-based model is a novelty in Czech computational lexicography. Though only a limited number of alternations has been practically implemented in VALLEX, its asset to adequate description of valency properties of verbs has been clearly proved.

Acknowledgement

The research reported in this paper has been supported by the grant of the Grant Agency of Czech Republic No. 405/04/0243.

4. References

Olga Babko-Malaya, Martha Palmer, Nianwen Xue, Aravind Joshi, and Seth Kulick. 2004. Proposition Bank II: Delving Deeper. In A. Meyers, editor, *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 17–23, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Václava Benešová. 2004. Delimitace lexií českých sloves z hlediska jejich syntaktických vlastností. Master’s thesis, Filozofická fakulta Univerzity Karlovy.

D. A. Cruse. 1986. *Lexical Semantics*. Cambridge University Press, Cambridge.

František Daneš. 1985. *Věta a text*. Academia, Praha.

Katrin Erk, Andrea Kowalski, Sebastian Pado, and Manfred Pinkal. 2003. Towards a Resource for Lexical Semantics: A Large German Corpus with Extensive Semantic Annotation. In *Proceedings of ACL-03*, Sapporo, Japan.

Josef Filipec. 1994. Lexicology and Lexicography: Development and State of the Research. In Philip A. Luelsdorff, editor, *The Prague School of Structural and Functional Linguistics*, pages 164–183. John Benjamins Publishing Company.

Charles J. Fillmore. 1968. The case for case. In E. Bach and R. Harms, editors, *Universals in Linguistic Theory*, pages 1–90. New York.

Charles J. Fillmore. 1977. The case for case reopened. In J.M. Sadock P. Cole, editor, *Syntax and Semantics 8*, pages 59–81.

Charles J. Fillmore. 2002. FrameNet and the Linking between Semantic and Syntactic Relations. In Shu-Cuan Tseng, editor, *Proceedings of COLING 2002*, pages xxviii–xxxvi. Howard International House.

Jan Hajič, Jarmila Panevová, Zdeňka Urešová, Alevtina Bémová, Veronika Kolářová, and Petr Pajas. 2003. PDT-VALLEX: Creating a Large-coverage Valency Lexicon for Treebank Annotation. In *Proceedings of The Second Workshop on Treebanks and Linguistic Theories*, volume 9 of *Mathematical Modeling in Physics, Engineering and Cognitive Sciences*, pages 57–68.

Jan Hajič. 2005. Complex Corpus Annotation: The Prague Dependency Treebank. In *Insight into Slovak and Czech Corpus Linguistics*, pages 54–73. Veda Bratislava, Slovakia.

Beth C. Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press, Chicago, IL.

Markéta Lopatková. 2003. Valency in Prague Dependency Treebank: Building Valency Lexicon. *The Prague Bulletin of Mathematical Linguistics 79-80*.

Igor A. Mel’čuk and Alexander K. Zholkovsky. 1984. *Explanatory Combinatorial Dictionary of Modern Russian*. Wiener Slawistischer Almanach, Vienna.

1987. *Mluvnice češtiny III*. Academia, Praha.

Jarmila Panevová. 1994. Valency Frames and the Meaning of the Sentence. In Philip A. Luelsdorff, editor, *The Prague School of Structural and Functional Linguistics*, pages 223–243. John Benjamins Publishing Company.

Jarmila Panevová. 1999. Česká reciproční zájmena a slovesná valence. *Slovo a slovesnost*, 4(60):269–275.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. D. Reidel Publishing Company, Dordrecht.

1964. *Slovník spisovného jazyka českého*. Praha.

Zdeněk Žabokrtský. 2005. *Valency Lexicon of Czech Verbs*. Ph.D. thesis, Charles University in Prague, Faculty of Mathematics and Physics.

Chapter 10

Machine Translation

10.1 Synthesis of Czech Sentences from Tectogrammatical Trees

Full reference:

Ptáček Jan, Žabokrtský Zdeněk: Synthesis of Czech Sentences from Tectogrammatical Trees, in Lecture Notes in Computer Science, No. 4188, Proceedings of the 9th International Conference, TSD 2006, Copyright Springer-Verlag Berlin Heidelberg, Masarykova univerzita, Berlin / Heidelberg, ISBN 3-540-39090-1, ISSN 0302-9743, pp. 221-228, 2006

Comments:

This article introduces a generator of Czech sentences from their tectogrammatical representations. Later, the present author created another Czech sentence generator, implemented in a more modular fashion—i.e., decomposed into a long sequence of processing blocks—in the TectoMT environment. This version, in which the notion of formemes was used for the first time, is currently part of the English-Czech translation as implemented in TectoMT. Recently, this generator has been adapted by Jan Ptáček for English in [Ptáček, 2008].

Synthesis of Czech Sentences from Tectogrammatical Trees*

Jan Ptáček, Zdeněk Žabokrtský

Institute of Formal and Applied Linguistics
Malostranské náměstí 25, 11800 Prague, Czech Republic
{ptacek,zabokrtsky}@ufal.mff.cuni.cz

Abstract. In this paper we deal with a new rule-based approach to the Natural Language Generation problem. The presented system synthesizes Czech sentences from Czech tectogrammatical trees supplied by the Prague Dependency Treebank 2.0 (PDT 2.0). Linguistically relevant phenomena including valency, diathesis, condensation, agreement, word order, punctuation and vocalization have been studied and implemented in Perl using software tools shipped with PDT 2.0. BLEU score metric is used for the evaluation of the generated sentences.

1 Introduction

Natural Language Generation (NLG) is a sub-domain of Computational Linguistics; its aim is studying and simulating the production of written (or spoken) discourse. Usually the discourse is generated from a more abstract, semantically oriented data structure. The most prominent application of NLG is probably transfer-based machine translation, which decomposes the translation process into three steps: (1) analysis of the source-language text to the semantic level, maximally unified for all languages, (2) transfer (arrangements of the remaining language specific components of the semantic representation towards the target language), (3) text synthesis on the target-language side (this approach is often visualized as the well-known machine translation pyramid, with hypothetical interlingua on the very top; NLG then corresponds to the right edge of the pyramid). The task of NLG is relevant also for dialog systems, systems for text summarizing, systems for generating technical documentation etc.

In this paper, the NLG task is formulated as follows: given a Czech tectogrammatical tree (as introduced in Functional Generative Description, [1], and recently elaborated in more detail within the PDT 2.0 project[1,2]), generate a Czech sentence the meaning of which corresponds to the content of the input tree. Not surprisingly, the presented research is motivated by the idea of transfer-based machine translation with the usage of tectogrammatics as the highest abstract representation.

[*] The research has been carried out under projects 1ET101120503 and 1ET201120505.

[1] `http://ufal.mff.cuni.cz/pdt2.0/`

[2] In the context of PDT 2.0, sentence synthesis can be viewed as a process inverse to treebank annotation.

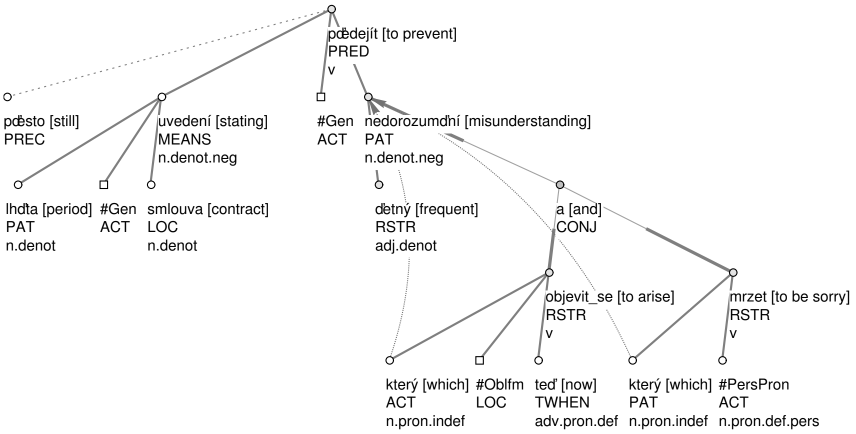


Fig. 1. Simplified t-tree fragment corresponding to the sentence *'Přesto uvedením lhůty ve smlouvě by se bylo předešlo četným nedorozuměním, která se nyní objevila a která nás mrzí.'* (But still, stating the period in the contract would prevent frequent misunderstandings which have now arisen and which we are sorry about.)

In the PDT 2.0 annotation scenario, three layers of annotation are added to Czech sentences: (1) *morphological layer* (m-layer), on which each token is lemmatized and POS-tagged, (2) *analytical layer* (a-layer), on which a sentence is represented as a rooted ordered tree with labeled nodes and edges corresponding to the surface-syntactic relations; one a-layer node corresponds to exactly one m-layer token, (3) *tectogrammatical layer* (t-layer), on which the sentence is represented as a deep-syntactic dependency tree structure (t-tree) built of nodes and edges (see Figure 1). T-layer nodes represent auto-semantic words (including pronouns and numerals) while functional words such as prepositions, subordinating conjunctions and auxiliary verbs have no nodes of their own in the tree. Each tectogrammatical node is a complex data structure – it can be viewed as a set of attribute-value pairs, or even as a typed feature structure. Word forms occurring in the original surface expression are substituted with their t-lemmas. Only semantically indispensable morphological categories (called grammatemes) are stored in the nodes (such as number for nouns, or degree of comparison for adjectives), but not the categories imposed by government (such as case for nouns) or agreement (congruent categories such as person for verbs or gender for adjectives). Each edge in the t-tree is labeled with a functor representing the deep-syntactic dependency relation. Coreference and topic-focus articulations are annotated in t-trees as well. See [2] for a detailed description of the t-layer.

The pre-release version of the PDT 2.0 data consists of 7,129 manually annotated textual documents, containing altogether 116,065 sentences with 1,960,657 tokens (word forms and punctuation marks). The t-layer annotation is available for 44 % of the whole data (3,168 documents, 49,442 sentences).

2 Task Decomposition

Unlike stochastic 'end-to-end' solutions, rule-based approach, which we adhere to in this paper, requires careful decomposition of the task (due to the very complex nature of the task, a monolithic implementation could hardly be maintainable). The decomposition was not trivial to find, because many linguistic phenomena are to be considered and some of them may interfere with others; the presented solution results from several months of experiments and a few re-implementations.

In our system, the input tectogrammatical tree is gradually changing – in each step, new node attributes and/or new nodes are added. Step by step, the structure becomes (in some aspects) more and more similar to a-layer tree. After the last step, the resulting sentence is obtained simply by concatenating word forms which are already filled in the individual nodes, the ordering of which is also already specified.

A simplified data-flow diagram corresponding to the generating procedure is displayed in Figure 2. All the main phases of the generating procedure will be outlined in the following subsections.

2.1 Formeme Selection, Diatheses, Derivations

In this phase, the input tree is traversed in the depth-first fashion, and so called *formeme* is specified for each node. Under this term we understand a set of constraints on how the given node can be expressed on the surface (i.e., what morphosyntactic form is used). Possible values are for instance simple case *gen* (genitive), prepositional case *pod+7* (preposition *pod* and instrumental), *v-inf* (infinitive verb),[3] *že+v-fin* (subordinating clause introduced with subordinating conjunction *že*), *attr* (syntactic adjective), etc.

Several types of information are used when deriving the value of the new *formeme* attribute. At first, the valency lexicon[4] is consulted: if the governing node of the current node has a valency frame, and the valency frame specifies constraints on the surface form for the functor of the current node, then these constraints imply the set of possible formemes. In case of verbs, it is also necessary to specify which diathesis should be used (active, passive, reflexive passive etc.; depending on the type of diathesis, the valency frame from the lexicon undergoes certain transformations). If the governing node does not have a valency frame, then the formeme default for the functor of the current node (and subfunctor, which specifies the type of the dependency relations in more detail) is

[3] It is important to distinguish between infinitive as a formeme and infinitive as a surface-morphological category. The latter one can occur e.g. in compound future tense, the formeme of which is not infinitive.

[4] There is the valency lexicon PDT-VALLEX ([3]) associated with PDT 2.0. On the t-layer of the annotated data, all semantic verbs and some semantic nouns and adjectives are equipped with a reference to a valency frame in PDT-VALLEX, which was used in the given sentence.

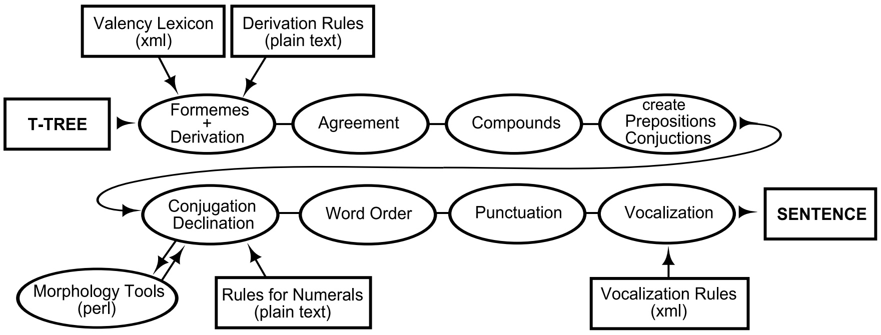


Fig. 2. Data-flow diagram representing the process of sentence synthesis.

used. For instance, the default formeme for the functor `ACMP` (accompaniment) and subfunctor `basic` is *s+7* (with), whereas for `ACMP.wout` it is *bez+2* (without).

It should be noted that the formeme constraints depend also on the possible word-forming derivations applicable on the current node. For instance, the functor APP (appurtenance) can be typically expressed by formemes *gen* and *attr*, but in some cases only the former one is possible (some Czech nouns do not form derived possessive adjectives).

2.2 Propagating Values of Congruent Categories

In Czech, which is a highly inflectional language, several types of dependencies are manifested by agreement of morphological categories (agreement in gender, number, and case between a noun and its adjectival attribute, agreement in number, gender, and person between a finite verb and its subject, agreement in number and gender between relative pronoun in a relative clause and the governor of the relative clause, etc.). As it was already mentioned, the original tectogrammatical tree contains those morphological categories which are semantically indispensable. After the formeme selection phase, value of case should be also known for all nouns. In this phase, oriented agreement arcs (corresponding to the individual types of agreement) are conceived between nodes within the tree, and the values of morphological categories are iteratively spread along these arcs until the unification process is completed.

2.3 Expanding Complex Verb Forms

Only now, when person, number, and gender of finite verbs is known, it is possible to expand complex verb forms where necessary. New nodes corresponding to reflexive particles (e.g. in the case of reflexiva tantum), to auxiliary verbs (e.g. in the case of complex future tense), or to modal verbs (if deontic modality of the verb is specified) are attached below the original autosemantic verb.

2.4 Adding Prepositions and Subordinating Conjunctions

In this phase, new nodes corresponding to prepositions and subordinating conjunctions are added into the tree. Their lemmas are already implied by the value of node formemes.

2.5 Determining Inflected Word Forms

After the agreement step, all information necessary for choosing the appropriate inflected form of the lemma of the given node should be available in the node. To perform the inflection, we employ morphological tools (generator and analyzer) developed by Hajič ([4]). The generator tool expects a lemma and a positional tag (as specified in [5]) on the input, and returns the inflected word form. Thus the task of this phase is effectively reduced to composing the positional morphological tag; the inflection itself is performed by the morphological generator.

2.6 Special Treatment of Definite Numerals

Definite numerals in Czech (and thus also in PDT 2.0 t-trees) show many irregularities (compared to the rest of the language system), that is why it seems advantageous to generate their forms separately. Generation of definite numerals is discussed in [6].

2.7 Reconstructing Word Order

Ordering of nodes in the annotated t-tree is used to express information structure of the sentences, and does not directly mirror the ordering in the surface shape of the sentence. The word order of the output sentence is reconstructed using simple syntactic rules (e.g. adjectival attribute goes in front of the governing noun), functors, and topic-focus articulation. Special treatment is required for clitics: they should be located in the 'second' position in the clause (Wackernagel position); if there are more clitics in the same clause, simple rules for specifying their relative ordering are used (for instance, the clitic *by* always precede short reflexive pronouns).

2.8 Adding Punctuation Marks

In this phase, missing punctuation marks are added to the tree, especially (i) the terminal punctuation (derived from the `sentmod` grammateme), (ii) punctuations delimiting boundaries of clauses, of parenthetical constructions, and of direct speeches, (iii) and punctuations in multiple coordinations (commas in expressions of the form *A, B, C and D*).

Besides adding punctuation marks, the first letter of the first token in the sentence is also capitalized in this phase.

2.9 Vocalizing Prepositions

Vocalization is a phonological phenomenon: the vowel *-e* or *-u* is attached to a preposition if the pronunciation of the prepositional group would be difficult without the vowel (e.g. *ve výklenku* instead of **v výklenku*). We have adopted vocalization rules precisely formulated in [7] (technically, we converted them into the form of an XML file, which is loaded by the vocalization module).

3 Implementation and Evaluation

The presented sentence generation system was implemented in ntred[5] environment for processing the PDT data. The system consists of approximately 9,000 lines of code distributed in 28 Perl modules. The sentence synthesis can also be launched in the GUI editor tred providing visual insight into the process.

As illustrated in Figure 2, we took advantage of several already existing resources, especially the valency lexicon PDT-VALLEX ([3]), derivation rules developed for grammateme assignment ([8]), and morphology analyzer and generator ([4]).

We propose a simple method for estimating the quality of a generated sentence: we compare it to the original sentence from which the tectogrammatical tree was created during the PDT 2.0 annotation. The original and generated sentences are compared using the BLUE score developed for machine translation ([9]) – indeed, the annotation-generation process is viewed here as machine translation from Czech to Czech. Obviously, in this case BLEU score does not evaluate directly the quality of the generation procedure, but is influenced also by the annotation procedure, as depicted in Figure 3.

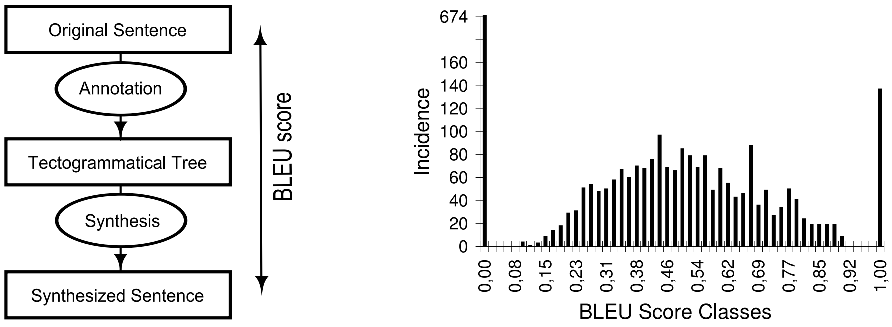


Fig. 3. Evaluation scheme and distribution of BLEU score in a development test sample counting 2761 sentences.

[5] http://ufal.mff.cuni.cz/~pajas

It is a well-known fact that BLEU score results have no direct common-sense interpretation. However, a slightly better insight can be gained if the BLEU score result of the developed system is compared to some baseline solution. We decided to use a sequence of t-lemmas (ordered in the same way as the corresponding t-layer nodes) as the baseline.

When evaluating the generation system on 2761 sentences from PDT 2.0 development-test data, the obtained BLEU score is **0.477**.[6] Distribution of the BLEU score values is given in Figure 3. Note that the baseline solution reaches only 0.033 on the same data.

To give the reader a more concrete idea of how the system really performs, we show several sample sentences here. The *O* lines contain the original PDT 2.0 sentence, the *B* lines present the baseline output, and finally, the *G* lines represent the automatically generated sentences.

(1) *O*: Dobře ví, o koho jde.
B: vědět dobrý jít kdo
G: Dobře ví, o koho jde.

(2) *O*: Trvalo to až do roku 1928, než se tento problém podařilo překonat.
B: trvat až rok 1928 podařit_se tento problém překonat
G: Trvalo až do roku 1928, že se podařilo tento problém překonat.

(3) *O*: Stejně tak si je i adresát výtky podle ostrosti a výšky tónu okamžitě jist nejen tím, že jde o něj, ale i tím, co skandál vyvolalo.
B: stejně tak být i adresát výtka ostrost a výška tón okamžitý jistý nejen jít ale i skandál vyvolat co
G: Stejně tak je i adresát výtky podle ostrosti a podle výšky tónu okamžitě jistý, nejen že jde o něj, ale i co skandál vyvolalo.

(4) *O*: Pravda o tom, že žvýkání pro žvýkání bylo odjakživa činností veskrze lidskou – kam paměť lidského rodu sahá.
B: pravda žvýkání žvýkání být odjakživa činnost lidský veskrze paměť rod lidský sahat kde
G: Pravda, že žvýkání pro žvýkání bylo odjakživa veskrze lidská činnost (kam paměť lidského rodu sahá).

4 Final Remarks

The primary goal of the presented work – to create a system generating understandable Czech sentences out of their tectogrammatical representation – has been achieved. This conclusion is confirmed by high BLUE-score values. Now we are incorporating the developed sentence generator into a new English-Czech

[6] This result seems to be very optimistic; moreover, the value would be even higher if there were more alternative reference translations available.

transfer-based machine translation system; the preliminary results of the pilot implementation seem to be promising.

As for the comparison to the related works, we are aware of several experiments with generating Czech sentences, be they based on tectogrammatics (e.g. [10], [11], [12]) or not (e.g. [13]), but in our opinion no objective qualitative comparison of the resulting sentences is possible, since most of these systems are not functional now and moreover there are fundamental differences in the experiment settings.

References

1. Sgall, P.: Generativní popis jazyka a česká deklinace. Academia (1967)
2. Mikulová, M., Bémová, A., Hajič, J., Hajičová, E., Havelka, J., Kolářová, V., Lopatková, M., Pajas, P., Panevová, J., Razímová, M., Sgall, P., Štěpánek, J., Urešová, Z., Veselá, K., Žabokrtský, Z., Kučová, L.: Anotace na tektogramatické rovině Pražského závislostního korpusu. Anotátorská příručka. Technical Report TR-2005-28, ÚFAL MFF UK (2005)
3. Hajič, J., Panevová, J., Urešová, Z., Bémová, A., Kolářová-Řezníčková, V., Pajas, P.: PDT-VALLEX: Creating a Large-coverage Valency Lexicon for Treebank Annotation. In: Proceedings of The Second Workshop on Treebanks and Linguistic Theories, Vaxjo University Press (2003) 57–68
4. Hajič, J.: Disambiguation of Rich Inflection – Computational Morphology of Czech. Charles University – The Karolinum Press, Prague (2004)
5. Hana, J., Hanová, H., Hajič, J., Vidová-Hladká, B., Jeřábek, E.: Manual for Morphological Annotation. Technical Report TR-2002-14 (2002)
6. Ptáček, J.: Generování vět z tektogramatických stromů Pražského závislostního korpusu. Master's thesis, MFF, Charles University, Prague (2005)
7. Petkevič, V., ed.: Vocalization of Prepositions. In: Linguistic Problems of Czech. (1995) 147–157
8. Razímová, M., Žabokrtský, Z.: Morphological Meanings in the Prague Dependency Treebank 2.0. LNCS/Lecture Notes in Artificial Intelligence/Proceedings of Text, Speech and Dialogue (2005)
9. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a Method for Automatic Evaluation of Machine Translation. Technical report, IBM (2001)
10. Panevová, J.: Random generation of Czech sentences. In: Proceedings of the 9th conference on Computational linguistics, Czechoslovakia, Academia Praha (1982) 295–300
11. Panevová, J.: Transducing Components of Functional Generative Description 1. Technical Report IV, Matematicko-fyzikální fakulta UK, Charles University, Prague (1979) Series: Explizite Beschreibung der Sprache und automatische Textbearbeitung.
12. Hajič, J., Čmejrek, M., Dorr, B., Ding, Y., Eisner, J., Gildea, D., Koo, T., Parton, K., Penn, G., Radev, D., Rambow, O.: Natural Language Generation in the Context of Manchine Translation. Technical report, Johns Hopkins University, Baltimore, MD (2002)
13. Hana, J.: The AGILE System. Prague Bulletin of Mathematical Linguistics (78) (2001) 147–157

10.2 CzEng: Czech-English Parallel Corpus

Full reference:

Bojar Ondřej, Žabokrtský Zdeněk: CzEng: Czech-English Parallel Corpus, Release version 0.5, in Prague Bulletin of Mathematical Linguistics, Vol. 86, Univerzita Karlova, ISSN 0032-6585, pp. 59-62, 2006

Comments:

Machine Translation, as almost all other modern NLP applications, requires large amount of data resources, out of which parallel corpora are probably the most important. Several years ago, before we started prototyping our MT system, there was no large broad-domain parallel corpus for English and Czech (even though some narrow-domain corpora were already available). Therefore, we started to compile our own parallel corpus. The corpus is called CzEng. Currently we are participating in preparing the corpus for its next public release.

The existence of CzEng is crucial for TectoMT, as it served as the main resource for building the probabilistic dictionary used in English-Czech translation (and also in experiments with aligning Czech and English tectogrammatical trees, [Mareček et al., 2008]), but CzEng has also been used outside TectoMT, especially for training a phrase-base MT system as described in [Bojar, 2008], and was offered as training data to participants of "Shared Task: Machine Translation for European Languages" organized within the EACL 2009 4th Workshop on Statistical Machine Translation.[1]

[1] http://www.statmt.org/wmt09/translation-task.html

CzEng: Czech-English Parallel Corpus

Release version 0.5

Ondřej Bojar, Zdeněk Žabokrtský
{bojar,zabokrtsky}@ufal.mff.cuni.cz

Abstract

We introduce CzEng 0.5, a new Czech-English sentence-aligned parallel corpus consisting of around 20 million tokens in either language. The corpus is available on the Internet and can be used under the terms of license agreement for non-commercial educational and research purposes. Besides the description of the corpus, also preliminary results concerning statistical machine translation experiments based on CzEng 0.5 are presented.

1 Introduction

CzEng 0.5[1] is a Czech-English parallel corpus compiled at the Institute of Formal and Applied Linguistics, Charles University, Prague in 2005-2006. The corpus contains no manual annotation. It is limited only to texts which have been already available in an electronic form and which are not protected by authors' rights in the Czech Republic. The main purpose of the corpus is to support Czech-English and English-Czech machine translation research with the necessary data. CzEng 0.5 is available free of charge for educational and research purposes, however, the users should become acquainted with the license agreement.[2]

2 CzEng 0.5 Data

CzEng 0.5 consists of a large set of parallel textual documents mainly from the fields of European law, information technology, and fiction, all of them converted into a uniform XML-based file format and provided with automatic sentence alignment. The corpus contains altogether 7,743 document pairs. Full details on the corpus size are given in Table 1.

2.1 Data Sources

We have used texts from the following publicly available sources:

- Acquis Communautaire Parallel Corpus (Ralf et al., 2006),
- The European Constitution and KDE documentation from corpus OPUS (Tiedemann and Nygaard, 2004),
- Readers' Digest texts were partially made available already in (Čmejrek et al., 2004),
- Kačenka was previously released as (Rambousek et al., 1997); because of the authors' rights, CzEng 0.5 can include only its subset, namely the following books:
  - D. H. Lawrence: Sons and Lovers / Synové a milenci,
  - Ch. Dickens: The Pickwick Papers / Pickwickovci,
  - Ch. Dickens: Oliver Twist,
  - T. Hardy: Jude the Obscure / Neblahý Juda,

[1] http://ufal.mff.cuni.cz/czeng/

[2] http://ufal.mff.cuni.cz/czeng/license.html

  - T. Hardy: Tess of the d'Urbervilles / Tess z d'Urbervillu,
- Other E-books were obtained from various Internet sources; the English side comes mainly from Project Gutenberg.[3] CzEng 0.5 includes these books:
  - Jack London: The Star Rover / Tulák po hvězdách,
  - Franz Kafka: Trial / Proces,
  - E.A. Poe: The Narrative of Arthur Gordon Pym of Nantucket: Dobrodružství A.G.Pyma,
  - E.A. Poe: A Descent into the Maelstrom / Pád do Malströmu,
  - Jerome K. Jerome: Three Men in a Boat / Tři muži ve člunu.

	Document pairs	Sentences		Words+Punctuation	
		Czech	English	Czech	English
Acquis Communautaire	6,272	1,101,610	930,626	14,619,572	16,079,043
	81.0%	77.6%	71.8%	78.9%	76.6%
European Constitution	47	11,506	10,380	138,853	176,096
	0.6%	0.8%	0.8%	0.7%	0.8%
Samples from European Journal	8	5,777	4,993	104,560	133,136
	0.1%	0.4%	0.4%	0.6%	0.6%
Readers' Digest	927	121,203	128,305	1,794,827	2,234,047
	12.0%	8.5%	9.9%	9.7%	10.6%
Kačenka	5	62,696	69,951	1,034,642	1,188,029
	0.1%	4.4%	5.4%	5.6%	5.7%
E-Books	5	17,140	17,495	330,118	399,607
	0.1%	1.2%	1.4%	1.8%	1.9%
KDE	479	98,789	133,897	495,052	784,316
	6.2%	7.0%	10.3%	2.7%	3.7%
Total	7,743	1,418,721	1,295,647	18,517,624	20,994,274
	100.0%	100.0%	100.0%	100.0%	100.0%

Table 1: CzEng 0.5 sections and data sizes.

2.2 Preprocessing

Since the individual sources of parallel texts differ in many aspects, a lot of effort was required to integrate them into a common framework. Depending on the type of the input resource, (some of) the following steps have been applied on the Czech and English documents:

- conversion from PDF, Palm text (PDB DOC), SGML, HTML and other formats,
- encoding conversion (everything converted into UTF-8 character encoding), sometimes manual correction of mis-interpreted character codes,
- removing scanning errors, removing end-of-line hyphens,
- file renaming, directory restructuring,
- sentence segmentation,
- tokenization,
- removing long text segments having no counterpart in the corresponding document,
- adding sentence and token identifiers,
- conversion to a common XML format.

For the sake of simplicity, the tokenization and segmentation rules were reduced to a minimum. This decision leads to some unpleasant differences in tokenization and segmentation compared to the "common standard" of Penn-Treebank-like or Prague-Dependency-Treebank-like annotation.[4]

[3] `http://www.gutenberg.org/`

[4] A different character class (digit, letter, punctuation) always starts a new token. Adjacent punctuation characters are encoded as separate tokens. Consecutive periods (...) thus lead to a sequence of one-token sentences. Moreover, no abbreviations were searched for. This hurts especially with titles (Dr.) or abbreviated names (O. Bojar), because a period followed by an upper-case letter is treated as the sentence boundary. All such expressions are thus split into several sentences.

English-Czech	1-1	0-1	1-2	2-1	1-0	1-3	0-2	3-1	Other
Alignment pairs	924,543 71.6%	97,929 7.6%	70,879 5.5%	69,558 5.4%	64,490 5.0%	23,538 1.8%	8,526 0.7%	6,768 0.5%	24,943 1.9%

Table 2: Sentence alignment pairs according to number of sentences.

2.3 Sentence Alignment

All the documents were sentence-aligned using the `hunalign` tool[5] (Varga et al., 2005). All the settings were kept default and we did not use any dictionary to bootstrap from. Hunalign collected its own temporary dictionary to improve sentence-level alignments.

The number of alignment pairs according to the number of sentences on the English and Czech side is given in Table 2.

3 First Machine-Translation Results Using CzEng 0.5

To provide a baseline for MT quality, we report BLEU (Papineni et al., 2002) scores of a state-of-the-art phrase-based MT system Moses.[6]

For this experiment, we selected 1-1 aligned sentences up to 50 words from CzEng 0.5. From this subcorpus, a random selection of three independent test sets (3000 sentences each) was kept aside and the remaining 870k sentences were used for training. The training data contained 9.7M Czech and 11.4M English tokens (words and punctuation).

Table 3 reports baseline BLEU scores on 3000-sentence test set with 1 reference translation. The texts were only lowercased (including the reference translation) and no other special preprocessing was performed. No advanced features of Moses such as factored translation were utilized. We ran the experiment three times, always using one of the test sets to tune model parameters, another to evaluate the performance on unseen sentences and ignoring the third test set. For curiosity we also report BLEU scores when not translating at all, i.e. pretending that the source text is a translation in the target language. Only some punctuation, numbers or names thus score.

Our results cannot be compared to previously reported Czech-English machine translation experiments (Čmejrek, Cuřín, and Havelka, 2003; Bojar, Matusov, and Ney, 2006),[7] because those experiments used a different 4 or 5-reference test set consisting of 250 sentences only.

The relatively high scores we have achieved are caused by the nature of our data. Most of our training data come from Acquis Communautaire and contain European legislation texts. Although there should be no reoccuring documents in our collection, there is a significant portion of sentences that repeat verbatim in the texts. Naturally, such frequent sentences can get to the randomly chosen test sets. A check of the three test sets revealed that only 1823±13 sentence pairs did not occur in training data. In other words, more than a third of the sentences in each test set appears already in the training data.

4 Summary And Further Plans

We have presented CzEng 0.5, a collection of Czech-English parallel texts. The corpus of about 20 million tokens is automatically sentence aligned. CzEng 0.5 is available free of charge for educational and research purposes, the licence allows collecting statistical data and making short citations. To our

[5] `http://mokk.bme.hu/resources/hunalign`

[6] Moses has been developed during a summer workshop at Johns Hopkins University, as a drop-in replacement for Pharaoh (Koehn, 2004). See `http://www.clsp.jhu.edu/ws2006/groups/ossmt/` for more details.

[7] English→Czech translation has also been attempted at the JHU workshop, report forthcoming.

	To English	To Czech
Not translating at all	5.98±0.68	5.93±0.67
Baseline Moses translation	42.57±0.55	37.41±0.58

Table 3: BLEU scores of a baseline MT system trained and evaluated on CzEng 0.5 data. Test set of 3000 sentences, 1 reference translation.

knowledge, it is the biggest and the most diverse publicly available parallel corpus for the Czech-English pair.

In the future, we plan to further enlarge CzEng. Even now we are aware of various sources of parallel material available on the Internet and not included in CzEng; however, in most of these cases it seems impossible to make any use of such data without breaking the authors' rights.

Future versions of CzEng will contain (machine) annotation of the data on various levels up to deep syntactic layer. We also plan to designate subsections of CzEng as standard development and evaluation data sets for machine translation, paying proper attention to cleaning up of these sets. Our future plans also include experimenting with several machine translation systems.

5 Acknowledgement

The work on this project was partially supported by the grants GAČR 405/06/0589, GAUK 351/2005, Collegium Informaticum GAČR 201/05/H014, ME838 and MŠMT ČR LC536.

References

Bojar, Ondřej, Evgeny Matusov, and Hermann Ney. 2006. Czech-English Phrase-Based Machine Translation. In *FinTAL 2006*, volume LNAI 4139, pages 214–224, Turku, Finland, August. Springer.

Čmejrek, Martin, Jan Cuřín, and Jiří Havelka. 2003. Czech-English Dependency-based Machine Translation. In *EACL 2003 Proceedings of the Conference*, pages 83–90. Association for Computational Linguistics, April.

Čmejrek, Martin, Jan Cuřín, Jiří Havelka, Jan Hajič, and Vladislav Kuboň. 2004. Prague Czech-English Dependecy Treebank: Syntactically Annotated Resources for Machine Translation. In *Proceedings of LREC 2004*, Lisbon, May 26–28.

Koehn, Philipp. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In Robert E. Frederking and Kathryn Taylor, editors, *AMTA*, volume 3265 of *Lecture Notes in Computer Science*, pages 115–124. Springer.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *ACL 2002, Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania.

Ralf, Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, Dan Tufiş, and Dániel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, pages 2142–2147. ELRA.

Rambousek, Jiří, Jana Chamonikolasová, Daniel Mikšík, Dana Šlancarová, and Martin Kalivoda. 1997. KAČENKA (Korpus anglicko-český - elektronický nástroj Katedry anglistiky). http://www.phil.muni.cz/angl/kacenka/kachna.html.

Tiedemann, Jörg and Lars Nygaard. 2004. The OPUS corpus - parallel & free. In *Proceedings of Fourth International Conference on Language Resources and Evaluation, LREC 2004*, Lisbon, May 26–28.

Varga, Dániel, László Németh, Péter Halácsy, András Kornai, Viktor Trón, and Viktor Nagy. 2005. Parallel corpora for medium density languages. In *Proceedings of the Recent Advances in Natural Language Processing RANLP 2005*, pages 590–596, Borovets, Bulgaria.

10.3 Hidden Markov Tree Model in Dependency-based Machine Translation

Full reference:

Žabokrtský Zdeněk, Popel Martin: Hidden Markov Tree Model in Dependency-based Machine Translation, in Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, Association for Computational Linguistics, Suntec, Singapore, pp. 145-148, 2009

Comments:

The main disadvantage of the transfer procedure described in Section 5.1.1 is that it does not make use of the target language model (and thus it cannot utilize large monolingual data available for Czech). In theory, one could generate a number of target t-tree hypotheses in the transfer phase, synthesize target sentences for all of them, and rank the sentences using a standard n-gram technique. However, such an approach would lead to serious time complexity issues because the total number of hypotheses is huge (exponential). Our strategy described in the paper is different: instead of using standard target language *n-gram model*, we use target language *t-tree model*, and thus we can combine the translation model and the target language tree model in a single optimization step, without repeating the sentence synthesis. As we discuss in the paper, such an approach can be naturally modelled using Hiddent Markov Tree Models (for which a time-efficient modification of Viterbi algorithm exists) and a substantial performance improvement can be gained.

Hidden Markov Tree Model in Dependency-based Machine Translation*

Zdeněk Žabokrtský
Charles University in Prague
Institute of Formal and Applied Linguistics
zabokrtsky@ufal.mff.cuni.cz

Martin Popel
Charles University in Prague
Institute of Formal and Applied Linguistics
popel@matfyz.cz

Abstract

We would like to draw attention to Hidden Markov Tree Models (HMTM), which are to our knowledge still unexploited in the field of Computational Linguistics, in spite of highly successful Hidden Markov (Chain) Models. In dependency trees, the independence assumptions made by HMTM correspond to the intuition of linguistic dependency. Therefore we suggest to use HMTM and tree-modified Viterbi algorithm for tasks interpretable as labeling nodes of dependency trees. In particular, we show that the transfer phase in a Machine Translation system based on tectogrammatical dependency trees can be seen as a task suitable for HMTM. When using the HMTM approach for the English-Czech translation, we reach a moderate improvement over the baseline.

1 Introduction

Hidden Markov Tree Models (HMTM) were introduced in (Crouse et al., 1998) and used in applications such as image segmentation, signal classification, denoising, and image document categorization, see (Durand et al., 2004) for references.

Although Hidden Markov Models belong to the most successful techniques in Computational Linguistics (CL), the HMTM modification remains to the best of our knowledge unknown in the field.

The first novel claim made in this paper is that the independence assumptions made by Markov Tree Models can be useful for modeling syntactic trees. Especially, they fit dependency trees well, because these models assume conditional dependence (in the probabilistic sense) only along tree edges, which corresponds to intuition behind dependency relations (in the linguistic sense) in dependency trees. Moreover, analogously to applications of HMM on sequence labeling, HMTM can be used for labeling nodes of a dependency tree, interpreted as revealing the hidden states[1] in the tree nodes, given another (observable) labeling of the nodes of the same tree.

The second novel claim is that HMTMs are suitable for modeling the transfer phase in Machine Translation systems based on deep-syntactic dependency trees. Emission probabilities represent the translation model, whereas transition (edge) probabilities represent the target-language tree model. This decomposition can be seen as a tree-shaped analogy to the popular n-gram approaches to Statistical Machine Translation (e.g. (Koehn et al., 2003)), in which translation and language models are trainable separately too. Moreover, given the input dependency tree and HMTM parameters, there is a computationally efficient HMTM-modified Viterbi algorithm for finding the globally optimal target dependency tree.

It should be noted that when using HMTM, the source-language and target-language trees are required to be isomorphic. Obviously, this is an unrealistic assumption in real translation. However, we argue that tectogrammatical deep-syntactic dependency trees (as introduced in the Functional Generative Description framework, (Sgall, 1967)) are relatively close to this requirement, which makes the HMTM approach practically testable.

As for the related work, one can found a number of experiments with dependency-based MT in the literature, e.g., (Boguslavsky et al., 2004), (Menezes and Richardson, 2001), (Bojar, 2008). However, to our knowledge none of the published systems searches for the optimal target representa-

* The work on this project was supported by the grants MSM 0021620838, GAAV ČR 1ET101120503, and MŠMT ČR LC536. We thank Jan Hajič and three anonymous reviewers for many useful comments.

[1] HMTM looses the HMM's time and finite automaton interpretability, as the observations are not organized linearly. However, the terms "state" and "transition" are still used.

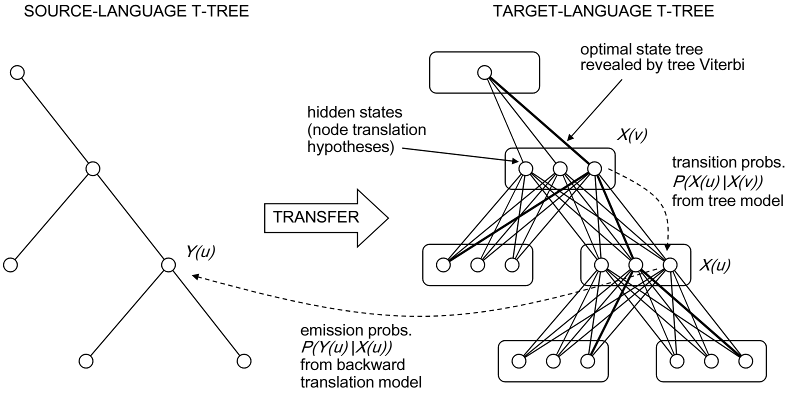


Figure 1: Tectogrammatical transfer as a task for HMTM.

tion in a way similar to HMTM.

2 Hidden Markov Tree Models

HMTM are described very briefly in this section. More detailed information can be found in (Durand et al., 2004) and in (Diligenti et al., 2003).

Suppose that $V = \{v_1, \ldots, v_{|V|}\}$ is the set of tree nodes, r is the root node and ρ is a function from $V \backslash r$ to V storing the parent node of each non-root node. Suppose two sequences of random variables, $\boldsymbol{X} = (X(v_1), \ldots, X(v_{|V|}))$ and $\boldsymbol{Y} = (Y(v_1), \ldots, Y(v_{|V|}))$, which label all nodes from V. Let $X(v)$ be understood as a hidden state of the node v, taking a value from a finite state space $S = \{s_1, \ldots, s_K\}$. Let $Y(v)$ be understood as a symbol observable on the node v, taking a value from an alphabet $K = \{k_1, \ldots, k_2\}$. Analogously to (first-order) HMMs, (first-order) HMTMs make two independence assumptions: (1) given $X(\rho(v))$, $X(v)$ is conditionally independent of any other nodes, and (2) given $X(v)$, $Y(v)$ is conditionally independent of any other nodes. Given these independence assumptions, the following factorization formula holds:[2]

$$P(\boldsymbol{Y}, \boldsymbol{X}) = P(Y(r)|X(r))P(X(r)) \cdot \prod_{v \in V \backslash r} P(Y(v)|X(v))P(X(v)|X(\rho(v))) \qquad (1)$$

We see that HMTM (analogously to HMM, again) is defined by the following parameters:

- $P(X(v)|X(\rho(v)))$ – transition probabilities between the hidden states of two tree-adjacent nodes,[3]
- $P(Y(v)|X(v))$ – emission probabilities.

Naturally the question appears how to restore the most probable hidden tree labeling given the observed tree labeling (and given the tree topology, of course). As shown in (Durand et al., 2004), a modification of the HMM Viterbi algorithm can be found for HMTM. Briefly, the algorithm starts at leaf nodes and continues upwards, storing in each node for each state and each its child the optimal downward pointer to the child's hidden state. When the root is reached, the optimal state tree is retrieved by downward recursion along the pointers from the optimal root state.

3 Tree Transfer as a Task for HMTM

HMTM Assumptions from the MT Viewpoint. We suggest to use HMTM in the conventional tree-based analysis-transfer-synthesis translation scheme: (1) First we analyze an input sentence to a certain level of abstraction on which the sentence representation is tree-shaped. (2) Then we use HMTM-modified Viterbi algorithm for creating the target-language tree from the source-language tree. Labels on the source-language nodes are treated as emitted (observable) symbols, while labels on the target-language nodes are understood as hidden states which are being searched for

[2] In this work we limit ourselves to *fully stationary* HMTMs. This means that the transition and emission probabilities are independent of v. This "node invariance" is an analogy to HMM's time invariance.

[3] The need for parametrizing also $P(X(r))$ (prior probabilites of hidden states in the root node) can be avoided by adding an artificial root whose state is fixed.

(Figure 1). (3) Finally, we synthesize the target-language sentence from the target-language tree.

In the HMTM transfer step, the HMTM emission probabilities can be interpreted as probabilities from the "backward" (source given target) node-to-node translation model. HMTM transition probabilities can be interpreted as probabilities from the target-language tree model. This is an important feature from the MT viewpoint, since the decomposition into *translation model* and *language model* proved to be extremely useful in statistical MT since (Brown et al., 1993). It allows to compensate the lack of parallel resources by the relative abundance of monolingual resources.

Another advantage of the HMTM approach is that it allows us to disregard the ordering of decisions made with the individual nodes (which would be otherwise nontrivial, as for a given node there might be constraints and preferences coming both from its parent and from its children). Like in HMM, it is the notion of hidden states that facilitates "summarizing" distributed information and finding the global optimum.

On the other hand, there are several limitations implied by HMTMs which we have to consider before applying it to MT: (1) There can be only one labeling function on the source-language nodes, and one labeling function on the target-language nodes. (2) The set of hidden states and the alphabet of emitted symbols must be finite. (3) The source-language tree and the target-language tree are required to be isomorphic. In other words, only node labeling can be changed in the transfer step.

The first two assumption are easy to fulfill, but the third assumption concerning the tree isomorphism is problematic. There is no known linguistic theory guaranteeing identically shaped tree representations of a sentence and its translation. However, we would like to show in the following that the tectogrammatical layer of language description is close enough to this ideal to make the HMTM approach practically applicable.

Why Tectogrammatical Trees? Tectogrammatical layer of language description was introduced within the Functional Generative Description framework, (Sgall, 1967) and has been further elaborated in the Prague Dependency Treebank project, (Hajič and others, 2006).

On the tectogrammatical layer, each sentence is represented as a tectogrammatical tree (t-tree for short; abbreviations t-node and t-layer are used in the further text too). The main features of t-trees (from the viewpoint of our experiments) are following. Each sentence is represented as a dependency tree, whose nodes correspond to autosemantic (meaningful) words and whose edges correspond to syntactic-semantic relations (dependencies). The nodes are labeled with the lemmas of the autosemantic words. Functional words (such as prepositions, auxiliary verbs, and subordinating conjunctions) do not have nodes of their own. Information conveyed by word inflection or functional words in the surface sentence shape is represented by specialized semantic attributes attached to t-nodes (such as number or tense).

T-trees are still language specific (e.g. because of lemmas), but they largely abstract from language-specific means of expressing non-lexical meanings (such as inflection, agglutination, functional words). Next reason for using t-trees as the transfer medium is that they allow for a natural transfer factorization. One can separate the transfer into three relatively independent channels:[4] (1) transfer of lexicalization (stored in t-node's lemma attribute), (2) transfer of syntactizations (stored in t-node's formeme attribute),[5] and (3) transfer of semantically indispensable grammatical categories[6] such as number with nouns and tense with verbs (stored in specialized t-node's attributes).

Another motivation for using t-trees is that we believe that local tree contexts in t-trees carry more information relevant for correct lexical choice, compared to linear contexts in the surface sentence shapes, mainly because of long-distance dependencies and coordination structures.

Observed Symbols, Hidden States, and HMTM Parameters. The most difficult part of the tectogrammatical transfer step lies in transfer-

[4] Full independence assumption about the three channels would be inadequate, but it can be at least used for smoothing the translation probabilities.

[5] Under the term syntactization (the second channel) we understand morphosyntactic form – how the given lemma is "shaped" on the surface. We use the t-node attribute *formeme* (which is not a genuine element of the semantically oriented t-layer, but rather only a technical means that facilitates modeling the transition between t-trees and surface sentence shapes) to capture syntactization of the given t-node, with values such as n:subj – semantic noun (s.n.) in subject position, n:for+X – s.n. with preposition *for*, n:poss – possessive form of s.n., v:because+fin – semantic verb as a subordinating finite clause introduced by *because*), adj:attr – semantic adjective in attributive position.

[6] Categories only imposed by grammatical constraints (e.g. grammatical number with verbs imposed by subject-verb agreement in Czech) are disregarded on the t-layer.

ring lexicalization and syntactization (attributes lemma and formeme), while the other attributes (node ordering, grammatical number, gender, tense, person, negation, degree of comparison etc.) can be transferred by much less complex methods. As there can be only one input labeling function, we treat the following ordered pair as the observed symbol: $Y(v) = (L^{src}(v), F^{src}(v))$ where $L^{src}(v)$ is the source-language lemma of the node v and $F^{src}(v)$ is its source-language formeme. Analogously, hidden state of node v is the ordered couple $X(v) = (L^{trg}(v), F^{trg}(v))$, where $L^{trg}(v)$ is the target-language lemma of the node v and $F^{trg}(v)$ is its target-language formeme. Parameters of such HMTM are then following:

$P(X(v)|X(\rho(v))) = P(L^{trg}(v), F^{trg}(v)|L^{trg}(\rho(v)), F^{trg}(\rho(v)))$
– probability of a node labeling given its parent labeling; it can be estimated from a parsed target-language monolingual corpus, and

$P(Y(v)|X(v)) = P(L^{src}(v), F^{src}(v)|L^{trg}(v), F^{trg}(v))$
– backward translation probability; it can be estimated from a parsed and aligned parallel corpus.

To summarize: the task of tectogrammatical transfer can be formulated as revealing the values of node labeling functions L^{trg} and F^{trg} given the tree topology and given the values of node labeling functions L^{src} and F^{src}. Given the HMTM parameters specified above, the task can be solved using HMTM-modified Viterbi algorithm by interpreting the first pair as the hidden state and the second pair as the observation.

4 Experiment

To check the real applicability of HMTM transfer, we performed the following preliminary MT experiment. First, we used the tectogrammar-based MT system described in (Žabokrtský et al., 2008) as a baseline.[7] Then we substituted its transfer phase by the HMTM variant, with parameters estimated from 800 million word Czech corpus and 60 million word parallel corpus. As shown in Table 1, the HMTM approach outperforms the baseline solution both in terms of BLEU and NIST metrics.

System	BLEU	NIST
baseline system	0.0898	4.5672
HMTM modification	0.1043	4.8445

Table 1: Evaluation of English-Czech translation.

5 Conclusion

HMTM is a new approach in the field of CL. In our opinion, it has a big potential for modeling syntactic trees. To show how it can be used, we applied HMTM in an experiment on English-Czech tree-based Machine Translation and reached an improvement over the solution without HMTM.

[7] For evaluation purposes we used 2700 sentences from the evaluation section of WMT 2009 Shared Translation Task. http://www.statmt.org/wmt09/

References

Igor Boguslavsky, Leonid Iomdin, and Victor Sizov. 2004. Multilinguality in ETAP-3: Reuse of Lexical Resources. In *Proceedings of Workshop Multilingual Linguistic Resources, COLING*, pages 7–14.

Ondřej Bojar. 2008. *Exploiting Linguistic Data in Machine Translation*. Ph.D. thesis, ÚFAL, MFF UK, Prague, Czech Republic.

Peter E. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*.

Matthew Crouse, Robert Nowak, and Richard Baraniuk. 1998. Wavelet-based statistical signal processing using hidden markov models. *IEEE Transactions on Signal Processing*, 46(4):886–902.

Michelangelo Diligenti, Paolo Frasconi, and Marco Gori. 2003. Hidden tree Markov models for document image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:2003.

Jean-Baptiste Durand, Paulo Goncalvès, and Yann Guédon. 2004. Computational methods for hidden Markov tree models - An application to wavelet trees. *IEEE Transactions on Signal Processing*, 52(9):2551–2560.

Jan Hajič et al. 2006. Prague Dependency Treebank 2.0. Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase based translation. In *Proceedings of the HLT/NAACL*.

Arul Menezes and Stephen D. Richardson. 2001. A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In *Proceedings of the workshop on Data-driven methods in machine translation*, volume 14, pages 1–8.

Petr Sgall. 1967. *Generativní popis jazyka a česká deklinace*. Academia, Prague, Czech Republic.

Zdeněk Žabokrtský, Jan Ptáček, and Petr Pajas. 2008. TectoMT: Highly Modular MT System with Tectogrammatics Used as Transfer Layer. In *Proceedings of the 3rd Workshop on SMT, ACL*.

Bibliography

[cnk, 2005] (2005). Český národní korpus - syn2005. Ústav Českého národního korpusu FF UK, Praha, http://www.korpus.cz .

[Boguslavsky et al., 2004] Boguslavsky, I., Iomdin, L., and Sizov, V. (2004). Multilinguality in ETAP-3: Reuse of Lexical Resources. In Sérasset, G., editor, *COLING 2004 Multilingual Linguistic Resources*, pages 1–8, Geneva, Switzerland. COLING.

[Bojar, 2008] Bojar, O. (2008). *Exploiting Linguistic Data in Machine Translation*. PhD thesis, ÚFAL, MFF UK, Prague, Czech Republic.

[Bojar and Hajič, 2008] Bojar, O. and Hajič, J. (2008). Phrase-Based and Deep Syntactic English-to-Czech Statistical Machine Translation. In *ACL 2008 WMT: Proceedings of the Third Workshop on Statistical Machine Translation*, pages 143–146, Columbus, OH, USA. Association for Computational Linguistics.

[Bojar and Žabokrtský, 2009] Bojar, O. and Žabokrtský, Z. (2009). Building a Large Czech-English Automatic Parallel Treebank. *Prague Bulletin of Mathematical Linguistics*, 92. (in print).

[Bolshakov and Gelbukh, 2000] Bolshakov, I. A. and Gelbukh, A. F. (2000). The Meaning-Text Model: Thirty Years After. International Forum on Information and Documentation.

[Brants, 2000] Brants, T. (2000). TnT - A Statistical Part-of-Speech Tagger . In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, pages 224–231, Seattle.

[Brown et al., 1993] Brown, P. E., Della Pietra, V. J., Della Pietra, S. A., and Mercer, R. L. (1993). The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics.*

[Callison-Burch et al., 2008] Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2008). Further Meta-Evaluation of Machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106, Columbus, Ohio. Association for Computational Linguistics.

[Callison-Burch et al., 2009] Callison-Burch, C., Koehn, P., Monz, C., and Schroeder, J. (2009). Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece. Association for Computational Linguistics.

[Cinková et al., 2006] Cinková, S., Hajič, J., Mikulová, M., Mladová, L., Nedolužko, A., Pajas, P., Panevová, J., Semecký, J., Šindlerová, J., Toman, J., Urešová, Z., and Žabokrtský, Z. (2006). Annotation of English on the tectogrammatical level. Technical report, Institute of Formal and Applied Linguistics, Faculty of Mathematics and Physics, Charles University in Prague.

[Collins, 1999] Collins, M. (1999). *Head-driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, Philadelphia.

[Cuřín et al., 2004] Cuřín, J. et al. (2004). Prague Czech - English Dependency Treebank, Version 1.0. CD-ROM, Linguistics Data Consortium, LDC Catalog No.: LDC2004T25, Philadelphia.

[Curry, 1963] Curry, H. B. (1963). Some logical aspects of grammatical structure. In *Proceedings of the Twelfth Symposium in Applied Mathematics*, pages 56–68, American Mathematical Society.

[Cuřín, 2006] Cuřín, J. (2006). *Statistical Methods in Czech-English Machine Translation*. PhD thesis, Charles University in Prague, Prague.

[Diligenti et al., 2003] Diligenti, M., Frasconi, P., and Gori, M. (2003). Hidden Tree Markov Models for Document Image Classification. *IEEE Transactions on pattern analysis and machine intelligence*, 25(4):519–523.

[Džeroski et al., 2006] Džeroski, S., Erjavec, T., Ledinek, N., Pajas, P., Žabokrtský, Z., and Žele, A. (2006). Towards a Slovene Dependency Treebank. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 1388–1391, Paris, France.

[Filippova and Strube, 2008] Filippova, K. and Strube, M. (2008). Sentence Fusion via Dependency Graph Compression . In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP 08)*, pages 177–185, Honolulu, Hawaii.

[Fox, 2005] Fox, H. (2005). Dependency-Based Statistical Machine Translation. In *Proceedings of the 2005 ACL Student Workshop*.

[Gaifman, 1965] Gaifman, H. (1965). Dependency Systems and Phrase-Structure Systems. In *Information and Control*, pages 304–337.

[Hajič, 2002] Hajič, J. (2002). Tectogrammatical Representation: Towards a Minimal Transfer in Machine Translation. In Frank, R., editor, *Proceedings of the 6th International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+6)*, pages 216—226, Venezia. Universita di Venezia.

[Hajič, 2004] Hajič, J. (2004). *Disambiguation of Rich Inflection (Computational Morphology of Czech)*. Karolinum, Charles Univeristy Press, Prague, Czech Republic.

[Hajič et al., 2006] Hajič, J. et al. (2006). Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.

[Hajič et al., 1999] Hajič, J., Panevová, J., Buráňová, E., Urešová, Z., and Bémová, A. (1999). Anotace Pražského závislostního korpusu na analytické rovině: pokyny pro anotátory. Technical Report 28, ÚFAL MFF UK, Prague, Czech Republic.

[Hajič et al., 2004] Hajič, J., Smrž, O., Zemánek, P., Pajas, P., Šnaidauf, J., Beška, E., Kráčmar, J., and Hassanová, K. (2004). Prague Arabic Dependency Treebank 1.0.

[Hajič, 2004] Hajič, J. (2004). *Disambiguation of Rich Inflection – Computational Morphology of Czech*. Charles University – The Karolinum Press, Prague.

[Hajič et al., 2009a] Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., and Zhang, Y. (2009a). The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

[Hajič et al., 2009b] Hajič, J., Cinková, S., Čermáková, K., Mladová, L., Nedolužko, A., Pajas, P., Semecký, J., Šindlerová, J., Toman, J., Tomšů, K., Korvas, M., Rysová, M., Veselovská, K., and Žabokrtský, Z. (2009b). Prague English Dependency Treebank 1.0. In *Linguistic Data Consortium (LDC)*, number LDC2004T25. Institute of Formal and Applied Linguistics, MFF UK, Prague.

[Hajič et al., 2006] Hajič, J. et al. (2006). Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.

[Helbig and Schenkel, 1969] Helbig, G. and Schenkel, W. (1969). *Wörterbuch zur Valenz und Distribution deutscher Verben*. VEB BIBLIOGRAPHISCHES INSTITUT, Leipzig, Germany.

[Hoang et al., 2007] Hoang, H., Birch, A., Callison-burch, C., Zens, R., Aachen, R., Constantin, A., Federico, M., Bertoldi, N., Dyer, C., Cowan, B., Shen, W., Moran, C., and Bojar, O. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computation Linguistics (ACL), Demonstration Session*, pages 177–180.

[Kirschner, 1987] Kirschner, Z. (1987). APAC 3-2: An English to Czech Machine Translation System. In *Explizite Beschreibung der Sprache und automatische Textbearbeitung*. MFF UK, Praha.

[Klimeš, 2006] Klimeš, V. (2006). *Analytical and Tectogrammatical Analysis of a Natural Language*. PhD thesis, Faculty of Mathematics and Physics, Charles University, Prague, Czech Rep.

[Koehn and Hoang, 2007] Koehn, P. and Hoang, H. (2007). Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876.

[Kos and Bojar, 2009] Kos, K. and Bojar, O. (2009). Evaluation of Machine Translation Metrics for Czech as the Target Language.

[Kravalová, 2009] Kravalová, J. (2009). Využití syntaxe v metodách pro vyhledávání informací (Using syntax in information retrieval). Master's thesis, Faculty of Mathematics and Physics, Charles University in Prague.

[Kravalová and Žabokrtský, 2009] Kravalová, J. and Žabokrtský, Z. (2009). Czech Named Entity Corpus and SVM-based Recognizer. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 194–201, Suntec, Singapore. Association for Computational Linguistics.

[Kučová et al., 2003] Kučová, L., Kolářová, V., Žabokrtský, Z., Pajas, P., and Čulo, O. (2003). Anotování koreference v Pražském závislostním korpusu. Technical Report TR-2003-19, UFAL/CKL MFF UK, Prague.

[Kučová and Žabokrtský, 2005] Kučová, L. and Žabokrtský, Z. (2005). Anaphora in Czech: Large Data and Experiments with Automatic Anaphora. *LNCS/Lecture Notes in Artificial Intelligence/Proceedings of Text, Speech and Dialogue*, 3658:93–98.

[Linh and Žabokrtský, 2007] Linh, N. and Žabokrtský, Z. (2007). Rule-based Approach to Pronominal Anaphora Resolution Applied on the Prague Dependency Treebank 2.0 Data. In Branco, A., McEnery, T., Mitkov, R., and Silva, F., editors, *Proceedings of the 6th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2007)*, pages 77–81, Lagos (Algarve), Portugal. CLUP-Center for Linguistics of the University of Oporto.

[Lopatková et al., 2008] Lopatková, M., Žabokrtský, Z., and Kettnerová, V. (2008). *Valenční slovník českých sloves*. Nakladatelství Karolinum, Praha.

[Lopez, 2007] Lopez, A. (2007). A Survey of Statistical Machine Translation. Technical report, Institute for Advanced Computer Studies, University of Maryland.

[Maamouri et al., 2003] Maamouri, M., Bies, A., Jin, H., and Buckwalter, T. (2003). Arabic Treebank: Part 1 v 2.0.

[Machová, 1977] Machová, S. (1977). Functional generative description of Czech. Dependency phrase structure model. In *Eksplicitnoje opisanije jazyka i avtomatičeskaja obrabotka tekstov*, pages 81–232. MFF UK, Praha.

[Marcus et al., 1994] Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1994). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

[Mareček, 2008] Mareček, D. (2008). Automatic Alignment of Tectogrammatical Trees from Czech-English Parallel Corpus. Master's thesis, Charles University, MFF UK.

[Mareček et al., 2008] Mareček, D., Žabokrtský, Z., and Novák, V. (2008). Automatic Alignment of Czech and English Deep Syntactic Dependency Trees. In Hutchins, J. and Hahn, W., editors, *Proceedings of the Twelfth EAMT Conference*, pages 102–111, Hamburg. HITEC e.V.

[Mareček and Kljueva, 2009] Mareček, D. and Kljueva, N. (2009). Converting Russian Treebank SynTagRus into Praguian PDT Style. In *Proceedings of the RANLP 2009 (International Conference on Recent Advances in Natural Language Processing)*, Borovets, Bulgaria.

[McDonald et al., 2005] McDonald, R., Pereira, F., Ribarov, K., and Hajič, J. (2005). Non-projective dependency parsing using spanning tree algorithms. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada.

[Menezes and Richardson, 2001] Menezes, A. and Richardson, S. D. (2001). A best-first alignment algorithm for automatic extraction of transfer mappings from bilingual corpora. In *Proceedings of the workshop on Data-driven methods in machine translation*, volume 14, pages 1–8.

[Mikulová et al., 2005] Mikulová, M., Bémová, A., Hajič, J., Hajičová, E., Havelka, J., Kolářová-Řezníčková, V., Kučová, L., Lopatková, M., Pajas, P., Panevová, J., Razímová, M., Sgall, P., Štěpánek, J., Urešová, Z., Veselá, K., and Žabokrtský, Z. (2005). Anotace Pražského závislostního korpusu na tektogramatické rovině: pokyny pro anotátory. Technical report, ÚFAL MFF UK, Prague, Czech Republic.

[Minnen et al., 2000] Minnen, G., Carroll, J., and Pearce, D. (2000). Robust Applied Morphological Generation. In *Proceedings of the 1st International Natural Language Generation Conference*, pages 201–208, Israel.

[Novák, 2008] Novák, V. (2008). *Semantic Network Manual Annotation and its Evaluation* . PhD thesis, Faculty of Mathematics and Physics, Charles University in Prague, Prague.

[Novák and Žabokrtský, 2007] Novák, V. and Žabokrtský, Z. (2007). Feature Engineering in Maximum Spanning Tree Dependency Parser. In Matoušek, V. and Mautner, P., editors, *Lecture Notes in Artificial Intelligence, Proceedings of the 10th International Conference on Text, Speech and Dialogue*, Lecture Notes in Computer Science, pages 92–98, Pilsen, Czech Republic. Springer Science+Business Media Deutschland GmbH.

[Němčík, 2006] Němčík, V. (2006). Anaphora Resolution. Master's thesis, Faculty of Informatics, Masaryk University, Brno.

[Och and Ney, 2003] Och, F. J. and Ney, H. (2003). A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

[Oliva, 1989] Oliva, K. (1989). A Parser for Czech Implemented in Systems Q. In *Explizite Beschreibung der Sprache und automatische Textbearbeitung*. MFF UK, Praha.

[Panevová, 1980] Panevová, J. (1980). *Formy a funkce ve stavbě české věty*. Academia, Praha.

[Panevová et al., 2001] Panevová, J., Hajičová, E., and Sgall, P. (2001). Manuál pro tektogramatické značkování (III. verze, prosinec 2001). Technical Report TR-2001-12, ÚFAL/CKL MFF UK, Prague.

[Petkevič, 1987] Petkevič, V. (1987). A new dependency based specification of underlying representations of sentences. *Theoretical Linguistics*, 14:143–172.

[Petkevič and Skoumalová, 1995] Petkevič, V. and Skoumalová, H. (1995). Vocalization of Prepositions. In Petkevič, V., editor, *Linguistic Problems of Czech. Final Research Report for the JRP PECO 2824 project.*, pages 147–157. Univerzita Karlova v Praze, Matematicko-fyzikální fakulta.

[Piasecki, 2007] Piasecki, M. (2007). Multilevel Correction of OCR of Medical Texts. *Journal of Medical Informatics and Technologies*, 11:263–274.

[Popel, 2009] Popel, M. (2009). Ways to Improve the Quality of English-Czech Machine Translation. Master's thesis, Faculty of Mathematics and Physics, Charles University in Prague.

[Ptáček, 2008] Ptáček, J. (2008). Two Tectogrammatical Realizers Side by Side: Case of English and Czech. In *Fourth International Workshop on Human-Computer Conversation*, pages 1–4, Bellagio, Italy. University of Sheffield, [http://www.companions-project.org/events/200810_bellagio.cfm].

[Quirk et al., 2005] Quirk, C., Menezes, A., and Cherry, C. (2005). Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *ACL*, pages 271–279, Ann Arbor, Michigan.

[Razímová and Žabokrtský, 2005] Razímová, M. and Žabokrtský, Z. (2005). Morphological Meanings in the Prague Dependency Treebank 2.0. *LNCS/Lecture Notes in Artificial Intelligence/Proceedings of Text, Speech and Dialogue*, 3658:148–155.

[Romportl, 2008] Romportl, J. (2008). *Zvyšování přirozenosti strojově vytvářené řeči v oblasti suprasegmentálních zvukových jevů*. PhD thesis, Faculty of Applied Sciences, University of West Bohemia, Pilsen, Czech Republic.

[Rouš, 2009] Rouš, J. (2009). Probabilistic Translation Dictionary. Master's thesis, Faculty of Mathematics and Physics, Charles University in Prague.

[Ševčíková-Razímová and Žabokrtský, 2006] Ševčíková-Razímová, M. and Žabokrtský, Z. (2006). Systematic Parameterized Description of Pro-forms in the Prague Dependency Treebank 2.0. In *Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories (TLT)*, pages 175–186.

[Sgall, 1967] Sgall, P. (1967). *Generativní popis jazyka a česká deklinace.* Academia, Praha.

[Sgall et al., 1986] Sgall, P., Hajičová, E., and Panevová, J. (1986). *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects.* D. Reidel Publishing Company, Dordrecht.

[Smrž et al., 2008] Smrž, O., Bielický, V., Kouřilová, I., Kráčmar, J., Hajič, J., and Zemánek, P. (2008). Prague Arabic Dependency Treebank: A Word on the Million Words. In *Proceedings of the Workshop on Arabic and Local Languages (LREC 2008)*, pages 16–23, Marrakech, Morocco.

[Spoustová et al., 2007] Spoustová, D., Hajič, J., Votrubec, J., Krbec, P., and Květoň, P. (2007). The Best of Two Worlds: Cooperation of Statistical and Rule-Based Taggers for Czech. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing, ACL 2007*, pages 67–74, Praha.

[Thurmair, 2004] Thurmair, G. (2004). LREC 2004 Workshop on the Amazing Utility of Parallel and Comparable Corpora. In *Proceedings of LREC-2004, Workshop: The amazing utility of parallel and comparable corpora*, pages 5–9.

[Vauquois, 1973] Vauquois, B. (1973). Les systèmes informatiques et l'analyse de textes. In *Colloque de Strasbourg sur l'analyse des corpus linguistiques. Problèmes et méthodes de l'indexation maximale.*

[Čmejrek et al., 2003] Čmejrek, M., Cuřín, J., and Havelka, J. (2003). Czech-English Dependency Tree-Based Machine Translation. In *Proceedings of EACL 2003, 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 83–90, Budapest, Hungary.

[Šindlerová et al., 2007] Šindlerová, J., Mladová, L., Toman, J., and Cinková, S. (2007). An Application of the PDT-scheme to a Parallel Treebank. In *Proceedings of the 6th International Workshop on Treebanks and Linguistic Theories (TLT 2007)*, pages 163–174, Bergen, Norway.

[Žabokrtský, 2005] Žabokrtský, Z. (2005). Resemblances between Meaning-Text Theory and Functional Generative Description. In *Proceedings of Second International Conference on Meaning-Text Theory*, Moscow.

[Žabokrtský and Kučerová, 2002] Žabokrtský, Z. and Kučerová, I. (2002). Transforming Penn Treebank Phrase Trees into (Praguian) Tectogrammatical Dependency Trees. *Prague Bulletin of Mathematical Linguistics.*

[Žabokrtský and Smrž, 2003] Žabokrtský, Z. and Smrž, O. (2003). Arabic Syntactic Trees: from Constituency to Dependency. In *Proceedings of the 10th Conference of the European Chapter of the Association of Computational Linguistics, ACL 2003*, Budapest.

[Žabokrtský, 2005] Žabokrtský, Z. (2005). *Valency Lexicon of Czech Verbs (PhD thesis)*. PhD thesis, Charles University, Prague, Czech Rep.

[Žabokrtský et al., 2002] Žabokrtský, Z., Džeroski, S., and Sgall, P. (2002). A Machine Learning Approach to Automatic Functor Assignment in the Prague Dependency Treebank. In González Rodríguez, Manuel//Paz Suárez Araujo, C., editor, *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, volume 5, pages 1513—1520. ELRA.

[Žabokrtský and Popel, 2009] Žabokrtský, Z. and Popel, M. (2009). Hidden Markov Tree Model in Dependency-based Machine Translation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 145–148, Suntec, Singapore. Association for Computational Linguistics.

[Zeman et al., 2005] Zeman, D., Hana, J., Hanová, H., Hajič, J., Hladká, B., and Jeřábek, E. (2005). A Manual for Morphological Annotation, 2nd edition. Technical Report 27, ÚFAL MFF UK, Prague, Czech Republic.

[Zeman and Žabokrtský, 2005] Zeman, D. and Žabokrtský, Z. (2005). Improving Parsing Accuracy by Combining Diverse Dependency Parsers. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT)*, pages 171–178, Vancouver, B.C., Canada, Oct. 9-10. Association for Computational Linguistics.

[Zhang et al., 2007] Zhang, M., Jiang, H., Aw, A. T., Sun, J., Li, S., and Tan, C. L. (2007). Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of The Machine Translation Summit, The 11th Machine Translation Summit*, pages 535–542.