Automatic Extraction of Clause Relationships from a Treebank

Oldřich Krůza and Vladislav Kuboň

Charles University in Prague
Institute of Formal and Applied Linguistics
Malostranské nám. 25, Prague
Czech Republic
kruza@ufal.mff.cuni.cz, vk@ufal.mff.cuni.cz

Abstract. The paper concentrates on deriving non-obvious information about clause structure of complex sentences from the Prague Dependency Treebank. Individual clauses and their mutual relationship are not explicitly annotated in the treebank, therefore it was necessary to develop an automatic method transforming the original annotation concentrating on the syntactic role of individual word forms into a scheme describing the relationship between individual clauses. The task is complicated by a certain degree of inconsistency in original annotation with regard to clauses and their structure. The paper describes the method of deriving clause-related information from the existing annotation and its evaluation.

1 Introduction

One of the major factors which changed linguistics during the past twenty years was without a doubt a strong stress on building and exploiting large annotated corpora of natural languages. They serve nowadays as a primary source of evidence for the development and evaluation of linguistic theories and applications.

Although the corpora are extremely important source of data, they are not omnipotent. The more elaborated annotation scheme the authors use, the more problems with linguistic phenomena they have to solve. Creating a consistently annotated treebank requires making a large number of decisions about a particular annotation of particular linguistic phenomena. The more elaborated and detailed is the annotation, the easier it is to find phenomena which are annotated in a seemingly inconsistent way. If the annotation is really well-designed and consistent, then it should be possible to extract an information hidden in the corpus or treebank even in case that a particular phenomenon we are interested in was not annotated explicitly.

This paper describes an attempt to do precisely that - to extract an information which may be useful for research of a particular linguistic phenomenon from the treebank, where this phenomenon is not explicitly tagged. The axtracted information should provide a linguistic basis for research in the fields of natural language parsing and information retrieval (exploiting linguistically motivated

methods in information retrieval seems can recently be found in a number of papers, cf. e.g. [1]). The treebank under consideration is the Prague Dependency Treebank (PDT)[1], [3] a large and elaborated corpus with rich syntactic annotation of Czech sentences. The reason why we concentrate on this particular corpus is very simple - it is the only existing large scale syntactically annotated corpus for Czech. The phenomenon we are interested in are Czech complex sentences, the mutual relationship of their clauses and properties of those clauses. In the following sections we would like to present a brief description of the PDT, followed by a discussion of the annotation of clauses in complex sentences (and their parts - segments) in the PDT. Then we are going to describe an automatic method how to extract the required information from PDT (where it is not explicitly marked). In the last section we are going to present a discussion concerning the methods and results of an evaluation of the method presented in the paper.

2 The Prague Dependency Treebank

The Prague Dependency Treebank is a result of a large scale project started in 1996 at the Faculty of Mathematics and Physics at the Charles University in Prague. It is a corpus annotated on multiple levels - morphological, analytical and underlying-syntactic layer (for a description of the tagging scheme of PDT, see e.g. [5], [2], [7], [8], and the two manuals for tagging published as Technical Reports by UFAL and CKL of the Faculty of Mathematics and Physics, Charles University Prague (see [6]) and available also on the website http://ufal.mff.cuni.cz).

2.1 Clauses and segments

Unfortunately, although the annotation scheme of PDT allows for a very deep description of many kinds of syntactic relationships, there is no explicit annotation of the mutual relationships of individual clauses in complex sentences in the corpus. Even worse, even the units which the clauses consist from (segments - cf. the following informal definition) are not explicitly and consistently annotated. In order to make clear what we understand by the notion of a *segment* in the following text, let us mention that very informally a *segment* is a term describing individual part of a simple clause which is visually separated from other segments by some separator (conjunction, relative pronoun, bracket, punctuation mark etc.). The segments are of course smaller units than clauses, a clause consists from more than a single segment for example in case that it contains coordination or when it is divided into more parts for example by an embedded subordinated clause. A more detailed description of segments can be found e.g. in [9].

[1] http://ufal.mff.cuni.cz/pdt2.0/

2.2 Annotation of segments at the analytic level of the PDT

The identification of segments and their mutual relationship might be an important part of the process of the identification of whole clauses, so let us look at the annotation of segments in the PDT. More precisely, let us look at the annotation at the surface syntactic (analytic) level of the PDT, because it constitutes much more natural information source for our purpose than the deeper, tectogrammatical level (mainly due to the fact that the analytic trees of PDT correspond more directly to individual tokens (words, punctuation marks etc.) of the input sentence. The tectogrammatical level represents a meaning of the sentence, the correspondence to its surface form is not so straightforward as it is in the case of the analytic level.

Formally, the structure of a sentence at the analytic layer of PDT is represented as a dependency-based tree, i.e. a connected acyclic directed graph in which no more than one edge leads into a node. The edges represent syntactic relations in the sentence (dependency, coordination and apposition being the basic ones). The nodes – labelled with complex symbols (sets of attributes) – represent word forms and punctuation marks.

In particular, there are no nonterminal nodes in PDT that would represent more complex sentence units – such units are expressed as (dependency) subtrees, see also Figures 1, 2 and 3:

- Complex units that correspond to particular "phrases", as verb phrase, nominal phrase or prepositional phrase – such units are expressed as subtrees rooted with nodes labelled with respective governing word forms, e.g. governing verb (its "lexical" part in case of analytical verb form, or copula in verbal-nominal predicate, or modal verb), governing noun, or preposition (as a "head" of a whole prepositional phrase);
- Dependent clauses – in principle, they are rendered as subtrees rooted with a node labelled with the governing verb of dependent clause (e.g. for attributive dependent clauses), or with a node for subordinating conjunction (adverbial and content clauses);
- Coordinated structures and sentence units in apposition (whether they are sentence members or whole clauses) – they are represented by subtrees rooted with nodes that correspond to a coordinating conjunction or some formal flag for an apposition (e.g. comma, brackets).

The rules described above always concern syntactic relationships between a pair of tokens (words) in the sentence. They do not deal by any means with bigger units – not even in the case of coordination where it would be natural. Segments and clauses are not explicitly marked in the tree, nor is their mutual relationship. We can, of course, intuitively suppose that subtrees might be rooted with the governing node of the segment or a clause as a natural solution. Unfortunately, it turned out that this is not the case at the analytic level of PDT. Let us demonstrate this fact on constructions contained in brackets, visually very easily distinguishable sentential segments with a direct relationship to

the structure of individual clauses (a bracketed construction is one of the phenomena dividing a clause into more segments). Unlike punctuation marks, the brackets unambiguously show the beginning and the end of a text inserted into a sentence. It is therefore quite natural to expect this easily detectable segment to be annotated in a single consistent way. The following examples of structures assigned to segments in brackets have been borrowed from [4], see Fig. 1, 2 and 3.

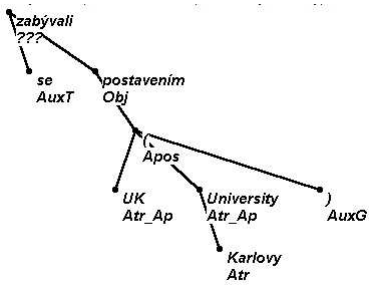

Fig. 1. PDT: Segment in parenthesis as an apposition: *zabývali se postavením UK (Univerzity Karlovy)* [(they) dealt-with *refl.* a_position of_UK (University of_Charles)]

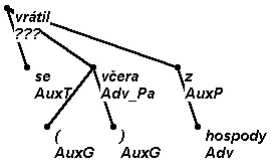

Fig. 2. PDT: Segments in parenthesis as a sentence member: *vrátil se (včera) z hospody* [(he)... returned *refl* (yesterday) from a_pub]

Let us point out that not only the annotation of a content of these parenthesis differs, but even the mutual position of both types of brackets in the tree is different. The situation is similar when we look at clauses, the main difference being the size - in the case of whole clauses the examples would be longer and even less transparent than the examples presented above for bracketed segments.

Although inconsistently annotated at the first sight, the trees contain an extremely valuable syntactic information which may serve as a basis for an automatic re-annotation procedure. Such a procedure should exploit the fact that in other respects, with regard to other syntactic phenomena, the corpus had been annotated with extreme care and with a great deal of consistency. Actually, the

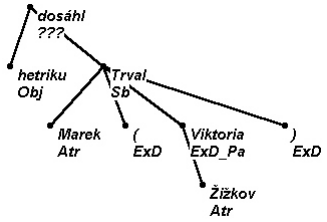

Fig. 3. PDT: Segments in parenthesis as an independent sentence part: *a hetriku dosáhl Marek Trval (Viktorie Žižkov)* [and hattrick$_{Obj}$ achieved Marek Trval$_{Subj}$ (Viktorie Žižkov)]

reason for inconsistency wrt. the annotation of obvious segments is the consistency achieved wrt. a different syntactic phenomenon, in this case the endeavor to annotate the apposition and coordination in a similar way, i.e. with the whole construction depending on a conjunction / appositional comma.

3 Definition of a Clause Based on the Analytic Layer of the PDT

We base our definition of clauses on the analytic level of PDT. This definition will serve as a basis for the algorithm of clause identification. By taking advantage of the analytic annotation, we could come up with a simple definition that only minimally refers to language intuition and meaning.

A clause is defined as a subtree of a predicate, the predicate inclusive. There are two exceptions to this general rule:

1. A subordinating conjunction governing the predicate belongs to the clause.
2. A clause whose predicate is in the subtree of another clause is not considered to be a part of the governing clause.

Since not every predicate is explicitly annotated as such in the analytic level of PDT, this amounts to

1. tokens explicitly marked as predicates (have analytic function "Pred"),
2. finite autosemantic verbs,
3. tokens that govern a node of the analytic function "AuxV" (this denotes predicates formed by compound verbs) and
4. tokens that are coordinated with a predicate.

There are exceptions and special cases to these rules:

– ad 2: Finite verbs that hold coordination or apposition are not considered to be predicates for our purposes. See subsection 3.3.

– ad 3: If the token governing an AuxV-node is a coordinating conjunction, then it is not considered a predicate. In such a case though, the coordinated tokens are considered as if they governed an AuxV-node and are thus recognized as predicates.

3.1 Sections of a sentence

The criteria introduced above state what is a clause and what tokens belong to one. This is one of the two goals of our algorithm. The second one deals with relations between and among clauses. These are basically dependency and coordination relations. Since clauses are not atomic objects and there are cases where a token belongs to more than one clause, we need to introduce a new, more general term: *sections*. The reason why we cannot use the notion of segment instead of a section is the different nature of both components of a clause. Segments are units distinguishable on the surface, they are defined for sentences in the form of sequences of word forms and punctuation marks. Sections, on the other hand, are defined as units distinguishable from the surface syntactic (analytic) representation of a sentence. Both terms refer to units which are similar but not identical.

A section of a sentence is defined by its *representative* and its *component*. The representative of a section is a token of the sentence or its technical root (every sentence in PDT has a technical root). Each token as well as the technical root represents no more than one section. The component of a section is a subset of the tokens of the sentence. The components of the sentence's sections constitute a perfect coverage of the sentence's tokens.

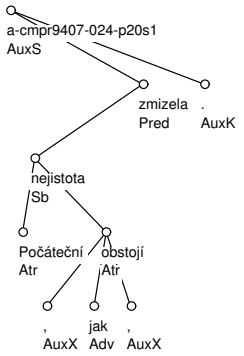

Fig. 4. Analytic tree of the sentence "Počáteční nejistota, jak obstojí, zmizela." [Initial uncertainty, how it-will-do, vanished.]

Typically, the representative of a section belongs also to its component. The exception is the technical root, which can represent a section but can never be in its component.

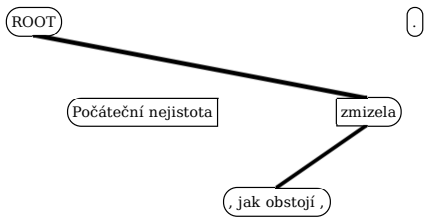

Fig. 5. Sections of the sentence from Figure 4. Each bordered shape marks a section. Each horizontal level contains one clause. The lines mark bloodline relations of clauses.

The component of a section forms a tree on the analytic level. The only exception is the section represented by the technical root, the component of which can be a forest.

Sections are of three types:

1. clause,
2. coordination and
3. adjunct.

Each clause constitutes a section of the type *clause*. Its representative is the finite verb that governs the clause and its component consists of the tokens that belong to the clause.

Coordination of clauses Whenever two or more clauses are coordinated, the coordination itself constitutes an extra section of the type *coordination*. Its representative is the coordinating conjunction (or punctuation token) that holds the coordination (i.e. it has the analytic function of "Coord" or "Apos" in case of appositions, which we treat equally to coordinations). The component of a coordination section is its representative and leaf tokens dependent on it that are not related to the coordinated clauses. That is:

1. other conjunctions, commas and other separators of the coordinated clauses,
2. other words of the conjunction in case of multi-word conjunctions,
3. auxiliary leaf tokens (those with analytic function beginning with "Aux").

The third case emerges when a coordination of clauses governs a phrase that effectively depends on all the coordinated clauses. Take the English example: *"John loves Mary but won't marry."* There are two finite verbs present: "loves" and "won't". So our above stated definition would recognize two clauses plus one coordination.

Clause 1 would certainly contain tokens "loves Mary", Clause 2 would contain "won't marry" and the coordination would only contain "but". So, where would "John" go? It's him who loves Mary and it's also him who won't marry the poor

girl. We could see this sentence as a coordination of clauses with the subject distributed: "John loves Mary" + "John won't marry".

To denote this type of relation, we give "John" (with his whole (empty) subtree) his own section of the type *adjunct*. Sections of this type are always formed by subtrees (dependent clauses excluding) of tokens that are not clauses and depend on a coordination of clauses but are not coordinated in it.

Tokens that do not fall into any section by the above criteria belong to the section represented by the technical root. Its type is set to *clause*, but that is a purely technical decision.

3.2 Inter-clausal Relations

The sections were defined with the intention to obtain a help when capturing relations among clauses. Notice that every section's component has a tree-like structure. The only exception again being the clause whose representative is the technical root. This means that every section has one root token. We can therefore define bloodline relations between sections like this:

Definition 1 (Parent section) Let D be a section whose representative is not the technical root. Let r be the root token of D. Let p be the analytic parent token of r. We call the section to which p belongs or which p represents the *parent* section of D. The root section is its own parent.

As in the real life, one child is sometimes quite unlike another, that is an experience of many human parents. It's the same here, so we differentiate several *types of children*. These are:

1. dependants,
2. members and
3. parts.

Every clause and every adjunct can only have child sections of the *dependant* type. Coordinations, on the other hand, can have children of any type.
Whenever a coordination has a child of the *member* type, it means that the child section is coordinated in the coordination.
Whenever a coordination has a child of the *dependant* type, it means that the child section is effectively dependent on all the sections coordinated in the parent coordination.
Whenever a coordination has a child of the *part* type, it means that the child section belongs to all the sections coordinated in the parent coordination. Children of the *part* type are exactly the sections of the type *adjunct*.

This approach allows to capture virtually any clause structure from the PDT, keeping information about tokens belonging to clauses, their dependencies and coordinations. The grammatical roles of clauses are easily extracted from analytic functions of their representatives.

Since the definitions mentioned above are all based on information available on the analytic and lower levels of PDT, the algorithm for extracting clause structure from the analytic annotation is a straight-forward rewrite of those definitions into a programming language.

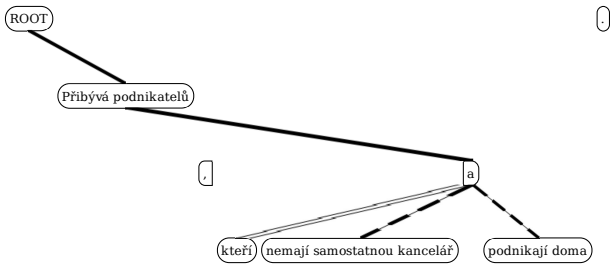

Fig. 6. Sections and their relations of the sentence "Přibývá podnikatelů, kteří nemají samostatnou kancelář a podnikají doma." [The-number-grows of-businessmen, who don't-have separate office and work at-home.] (The number of businessmen who have no separate office and work at home grows.) The main clause "Přibývá podnikatelů" (the number of businessmen grows) governs the coordination formed by the conjunction and the comma. There are two dependent clauses: "kteří nemají samostatnou kancelář" (who have no separate office) and "kteří podnikají doma" (who work at home). Their disjoint parts are marked as coordinated sections (*section type: clause, child type: member*) and their common word "kteří" (who) is marked as another section (*section type: adjunct, child type: part*).

3.3 Appositions held by a finite verb

The only phenomenon we know that our algorithm is not handling correctly concerns finite verbs that bear an apposition (or potentially coordination), that is, they have the analytic function of "Coord" or "Apos". Take the sentence "Do úplných detailů jako jsou typy obkladaček nelze jít." [Into sheer details like are types of-tiles is-not-possible to-go.] (We can't go into sheer details like the types of tiles.) Figure 7 shows its analytic tree. The clause that should apparently be recognized is formed by tokens "jako jsou typy obkladaček" [like are types of-tiles]. Notice that the tokens "úplných detailů" [sheer details], which do not belong to the inner clause, are in the subtree of the inner clause's founding verb "jsou" [are].

Here, the most profound rule our definition is based upon – that a clause is a subtree of its predicate – breaks the factual distribution of clauses. Even if we only tore off the clause itself (which is a well-formed tree), the parent clause would stop being connected. Our way of dealing with this is to simply ignore the presence of the inner clause and keep it as a part of the parent clause. This seems to be the best way to go, as it has virtually no negative consequences, it is very easy to detect and implement, and the phenomenon is not frequent.

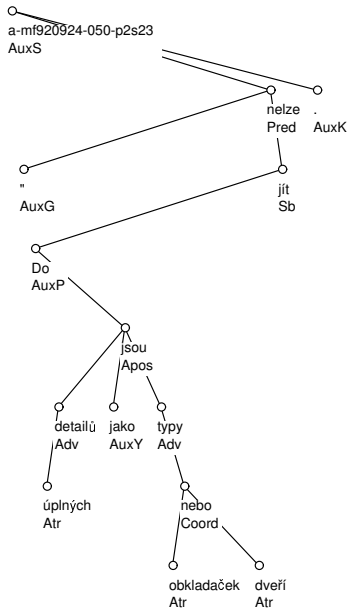

Fig. 7. Analytic tree of a sentence containing an apposition held by a finite verb

4 Evaluation

Applying the algorithm described above on the PDT, we get clauses marked up in the sentences. The process is deterministic, it reflects the annotation of individual nodes of an analytic tree of the PDT. It is also an application of a *definition*, not an attempt to model a given linguistic phenomenon. The data should then be used as gold standard for clause detection from the lower levels of annotation (e.g. morphological). It is clear that standard precision/recall evaluation is not useful in this case.

Instead, we have decided to try to count the sentences where the algorithm provides clauses in a different manner than we think a human would. The difference between automatic and man-made annotation is based upon the fact that our algorithm keeps clauses syntactically compact, while humans prefer to keep them linearly compact. These requirements go against each other mostly in the case where a coordination section has tokens inside some of the coordinated clauses. See Figure 8. Other cases include erroneously annotated trees in the corpus (garbage in – garbage out) and the presence of adjunct segments, which humans tend to connect to the adjacent clause only.

Table 1 presents the evaluation done on a large subset of the PDT. First row shows the number of sentences where a clause has alien tokens inside (precisely, where a clause is not bordered by conjunctions or punctuation). Second row shows the number of the problematic appositions whose governing token

is a verb. Row three shows the number of sentences manifesting both phenomena. Evidently, the number of sentences where the intuitive and the definition-conforming splits of tokens to clauses differ is significant. However, the number of sentences where the algorithm fails to *do the right thing* (row 2) is almost negligible.

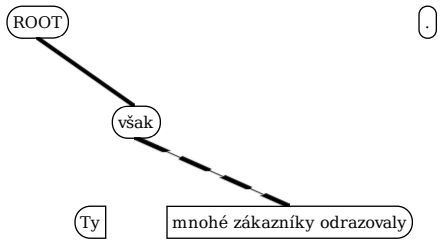

Fig. 8. Sections of the sentence "Ty však mnohé zákazníky odrazovaly." [Those however many clients$_{obj}$ discouraged.] (Those have, however, discouraged many clients.) Here, the sentence is marked up as one coordinated clause, governed by the coordination formed by the "však" (however) conjunction. The reason for this maybe surprising annotation is that the sentence is de facto coordinated with the previous one.

	Count	Ratio
Linearly incompact	7124	8.59%
Appositions	114	0.14%
incomp&appos	7225	8.71%
All	82944	100.00%

Table 1. Evaluation of the extraction of clauses from analytic trees.

5 Conclusions

The most valuable result of the experiment described in the paper is actually the treebank of Czech with automatically added annotation of mutual relationships among clauses in complex sentences. Our experiment shows that it is possible to reconstruct this information from the syntactic relationships among individual words. The data obtained as a result of application of our algorithm may serve for future experiments with complex sentences and clauses. Our algorithm provides

enough data not only for testing the theories, but also for the application of stochastic or machine–learning methods.

By ensuring that every section (and thus clause) is a tree on its own, our algorithm allows to develop a sort of a two-step parsing algorithm, where the first step will be the analysis of the structure of complex sentences and the second step the parsing of single clauses. The results of many existing parsers (cf. [10] etc.) show a dependency of parsing accuracy wrt. the length of input sentences, therefore dividing the complex sentence into shorter units and parsing them separately might help to overcome this natural behavior of existing parsers. This experiment is yet to be done, though.

Acknowledgments

This work was supported by the Grant Agency of the Czech Republic, grant No. 405/08/0681 and by the programme Information Society of the GAAV CR, grant No. 1ET100300517.

References

1. Fernández, M., de la Clergerie, E., Vilares, M.: Mining Conceptual Graphs for Knowledge Acquisition In: Proc. of ACM CIKM Workshop on Improving Non-English Web Searching (inews'08), ISBN 978-1-60558-253-5, Napa Valley, USA, 25-32 (2008)
2. Hajič, J., Hladká, B.: Probabilistic and Rule-Based Tagger of an Inflective Language – A Comparison In: Proceedings of the Fifth Conference on Applied Natural Language Processing. Washington D.C., 111-118 (1997)
3. Hajič, J., Hajičová, E., Pajas, P., Panevová, J., Sgall, P., Vidová-Hladká, B.: Prague Dependency Treebank 1.0 (Final Production Label). In: CD-ROM, Linguistic Data Consortium (2001)
4. Hajič, J., Panevová, J., Buráňová, E., Urešová, Z. Bémová, A.: Anotace Pražského závislostního korpusu na analytické rovině: pokyny pro anotátory. (in Czech) Technical Report No. 28, ÚFAL MFF UK, Prague, Czech Republic (1999)
5. Hajič, J.: Building a Syntactically Annotated Corpus: The Prague Dependency Treebank In Issues of Valency and Meaning, 106-132, Karolinum, Praha (1998)
6. Hajič, J., Panevová, J., Buráňová, E., Urešová, Z. and Bémová,A.: A Manual for Analytic Layer Tagging of the Prague Dependency Treebank, ISBN 1-58563-212-0 (2001)
7. Hajičová, E. : Prague Dependency Treebank: From Analytic to Tectogrammatical Annotations. In: Text, Speech , Dialogue, ed. by P. Sojka, V. Matoušek and I. Kopeček, Brno, Masaryk University, 45-50 (1998)
8. Hajičová, E. : The Prague Dependency Treebank: Crossing the Sentence Boundary. In: Matoušek, V., Mautner, P., Ocelíková, J. and Sojka, P. (eds.) Text, Speech and Dialogue, Berlin: Springer, 20-27, (1999)
9. Kuboň, V., Lopatková, M., Plátek, M., Pognan, P.: Segmentation of Complex Sentences, In: LNCS 4188, Text, Speech and Dialogue, TSD 2006, Springer Berlin / Heidelberg 151-158 (2006)
10. Zeman, D.: Parsing with a Statistical Dependency, PhD. Thesis, Charles university, Prague (2004)