NetGraph System

Searching through the Prague Dependency Treebank

Jiří Mírovský and Roman Ondruška

Abstract

The goal of one of the projects being solved at our Center is to build a corpus of Czech with a rich annotation scheme–the Prague Dependency Treebank. Text files containing these data in the form of labeled trees are being created automatically with manual corrections. With the increasing amount of annotated material searching a corpus becomes a complex task. The present paper describes the software system that was developed to contribute to the solution of this problem. Some requirements were established during the analysis of the design: the software has to be able to work in the Internet environment, to display tree structures in a graphical form, and the client part has to be hardware independent.

1 Introduction

The Prague Dependency Treebank (PDT) (Böhmova et al.), (Hajič, 1998) is a very large corpus of Czech texts with a rich annotation scheme. Its theoretical background is a dependency-based syntax, handling the sentence structure as concentrated around the verb and its valency members, but containing a further dimension, namely coordination and apposition. The nodes of the dependency tree are labeled by complex symbols, consisting of lexical, morphological and syntactic parts.

The PDT has a three-level scenario. Full morphological tagging is carried out on the lowest level (Hajič). The intermediate level (Hajič et al., 1997) deals with syntactic annotation using dependency syntax; it is called analytical level and is conceptually close to the syntactic annotation used in the Penn Treebank. The highest level of annotation is the tectogrammatical level (Hajičová et al., 2000), or the level of linguistic meaning.

The current version of the treebank is PDT 1.0 (CD-ROM), which contains $\approx 100,000$ sentences on the analytical level. A non-automatic searching for concrete dependencies through so large set is impossible. For this purpose we have developed a special software called Netgraph.

Netgraph is a multiuser system with a net architecture (Ondruška, 1998). This means that more than one user can access it at the same time and its components may be located in different nodes of the Internet (see Fig. 1). Netgraph generally consists of a server part, which mainly realizes the corpus searching itself, and a client part, which provides a user interface to the system. The client part, to be flexible, exists in two forms—as a Java2 application and as a Java2 applet (Horstmann et al., 1999).

After the client part is started, it allows the user to choose a subcorpus in which the searching is executed, to enter a query, and it offers the possibility to display the result of the query in a graphical form. A query has the form of a labeled tree structure, which is derived from the tree structure that represents sentences in the corpus files. The result of the query consists of trees from the subcorpus that contain the tree from the query as a subtree. The matching of the nodes evaluation is checked as well.

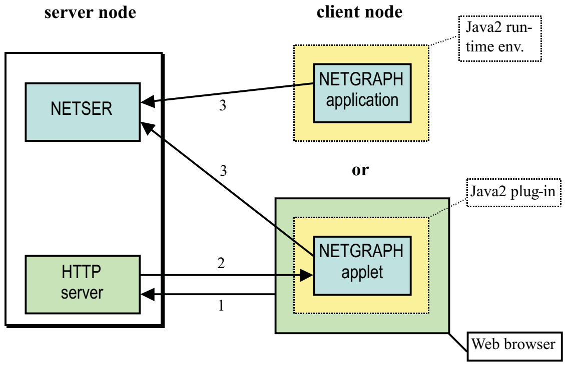

Figure 1: Start of Netgraph client. The arrows #1 and #2 represent the first two steps of the work with the Netgraph client as an applet. First, the user clicks in a web browser on a link to Netgraph client applet (the arrow #1). Then the applet is loaded into the web browser (the arrow #2) and connects to the program netser waiting for connections on the same server the applet has been loaded from (the lower arrow #3). Using the Netgraph client as an application, the user runs the client, then selects a server (the name and the port) and the client connects to the server (the upper arrow #3). In both of the cases, the Netgraph client needs Java2 as its running environment - either Java2 plug-in (the Netgraph applet) or Java2 run-time environment (the Netgraph application).

2 Searching in Netgraph

After the user is connected to the server, s/he has to go through three steps: to define a subcorpus to be queried, to define an object of a query, and to fetch and display the result of the query. All the steps can be passed using an easy-to-use graphical interface.

2.1 Subcorpus definition

The user can browse the directory structure of the corpus provided by the server and select files s/he wants to use for searching; this set can be saved to disk (and loaded back). The subcorpus may be also defined as the result of the previous query.

2.2 Query definition

By queries it is determined which trees will be included in the result of searching. The user defines a tree (see Fig. 2) s/he wants to be included as a subtree in each tree of the result. For defining such a tree, including the labels of its nodes, a graphical interface can be used. The graphically created tree is simultaneously displayed in a linear text form; the text form can also be edited directly.

If an unlabeled tree is used for a query then the searching process only considers the tree structure itself, the node matching is not checked in this case. However, the user usually de-

[lemma=chodit]
[origf=na]
[tag=N...4*]

Figure 2: Example of a simple query with attributes: *lemma*—an identifier of the underlying lexical unit, *origf*—an original word form as found in text, *tag*—a morphological category.

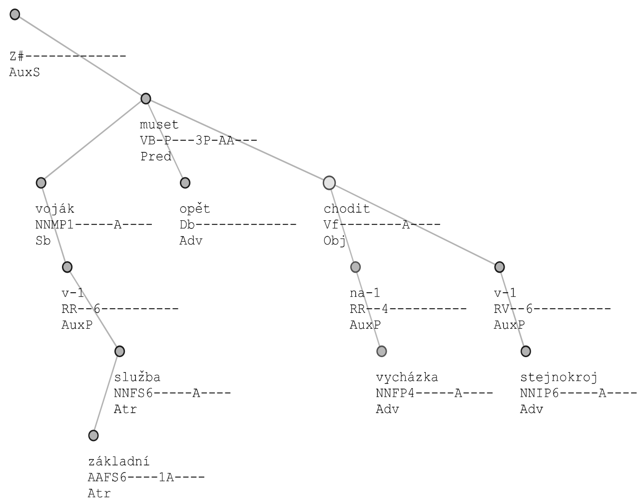

Figure 3: Example of tree depiction in Netgraph. The nodes in the tree represent words and their linguistic attributes and the edges represent analytical dependencies. Although the depicted tree is projective, this treatment also supports non-projective dependencies (present in Czech surface shapes of sentences and thus on the analytical level of PDT). The user can select attributes to be displayed by each node of the tree.

mands some restrictions on the node evaluation. In Netgraph one may enter them for every attribute by defining so called masks. The masks are expressions written into particular nodes of a query in brackets, separated by commas. Two special characters can be used in the mask definitions: the asterisk character `'*'` represents a sequence of characters and the dot `'.'` represents a single character.

Example: `[origf=.resident*]`

This mask indicates that the attribute *origf* can be of the value *'President'*, *'presidents'* and so on.

Another useful operation is the alternation—the logical conjunction, represented by the separator `'|'`.

Example:
```
[lemma=president,afun=Sb]|
[lemma=president,afun=Obj,tag=N...4*|N...6*]
```

This requires the lemma *president* as the *(subject or (object in (accusative or locative)))*. The analytical function is represented by the *afun* attribute.[1]

To define the query in more detail a system of meta attributes—attributes not really present in the corpus—can be used. There are two meta attributes at this point: *_transitive* – by defining this as true, nodes between this node and its parent (in query) are allowed (in result); *_optional* – if true, then the node may but need not be in the result. If any node is on its place, it must be the node itself or the root of its subtree in the query.

2.3 Result viewing

After the first tree matching the query is found, it is immediately displayed; the subtree matching the query is emphasized. The order of words is viewed from left to right. According to

[1]The exact description of the attributes meaning can be found in (CD-ROM).

that, the tree in Fig. 3, obtained as the result of the query from Fig. 2, represents the sentence: *"Vojáci v základní službě musejí opět chodit na vycházky ve stejnokrojích." (Soldiers in obligatory service must again go on walks in uniforms.)*

The order of nodes on the analytical level may be different from the order on the tectogrammatical level. Also, some nodes from the analytical level (esp. those representing function words and punctuation marks) should be hidden on the tectogrammatical level. Netgraph provides two modes of tree displaying according to these two levels.

3 Conclusion

Netgraph provides anyone interested with access to PDT simply and effectively. PDT itself is just an amorphous set of trees without any structure. With Netgraph a user can add the structure into PDT and obtain linguistic information that PDT contains. Netgraph also works corpus-language independently, so it is not restricted to Czech corpora. It only requires a corpus in fs (CD-ROM) format. Netgraph is stable, but is still being developed and other features are planned to be added in the future—for example relations among numeric values of attributes cannot be defined in a query yet. Some technical enhancements like XML support are planned too. The most recent version of Netgraph is available for downloading on the Netgraph home page (Mírovský).

Acknowledgments

This work has been supported by the Ministry of Education—project *Center for Computational Linguistics* (No. LN00A063).

References

Böhmová A., Hajič J., Hajičová E., Hladká B.: The Prague Dependency Treebank: Three-Level Annotation Scenario, In: *Treebanks: Building and Using Syntactically Annotated Corpora*, ed. by Anne Abeille. Kluwer Academic Publishers, in press[2]

Hajič J.: Building a Syntactically Annotated Corpus: The Prague Dependency Treebank, In: *Issues of Valency and Meaning*, ed. by E. Hajičová, pp.106-132, Karolinum, Praha 1998

Hajič J.: *Disambiguation of Rich Inflection—Computational Morphology of Czech.* Charles University Press - Karolinum, in press.

Hajič J. et al.: *A Manual for Analytic Layer Tagging of the Prague Dependency Treebank. ÚFAL Technical Report* TR-1997-03, Charles University, Czech Republic, 1997

Hajičová E., Panevová J., Sgall P.: *A Manual for Tectogrammatical Tagging of the Prague Dependency Treebank. ÚFAL/CKL Technical Report* TR-2000-09, 2000

CD-ROM PDT 1.0, available at http://shadow.ms.mff.cuni.cz/pdt/

Ondruška R.: *Tools for Searching in Syntactically Annotated Corpora*, Diploma Thesis, Charles University, Praha 1998

Horstmann C.S., Cornell G.: *Core Java2* Volume I-II, The Sun Microsystems Press, Prentice Hall, 1999

Mírovský J.: *Netgraph home page*[3]

[2] PDT documentation can be found at http://shadow.ms.mff.cuni.cz/pdt/Corpora/PDT_1.0/References/index.html

[3] http://shadow.ms.mff.cuni.cz/~mirovsky/netgraph/index.html