Building Big Czech Corpus
Collecting and Converting Czech Corpora

Pavel Hančar

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

Abstract. This paper describes creating of a big Czech corpus from many Czech corpora kept on the NLP Centre server. It describes new tools developed for this purpose, difficulties which may come up and a way how solve them.

1 Introduction

A corpus is a large collection of texts in a certain language. NLP Centre has many Czech corpora, but doesn't have any big one. A very valuable aspect of corpora is a morphological tagging, but this is missing in current data. So the task is to collect all Czech corpora (let's call them input corpora), to do the tagging, and to compile the result for corpus manager.

Corpora contains two kinds of data: the text itself in a vertical form and information about the text (metadata). The vertical form means, that every word or punctuation mark has it's own line (there are some special cases e.g. "..." is one position, not three). Documents, paragraphs, etc. are determined by XML marks [1,2].

Generally, the data are kept as text files, but there is more possibilities how to do this. Most of the input corpora consist of many vertical files, where each of them has it's metadata file. However the format we need is whole corpus in one vertical file with metadata in heading of each document. It is format used by the compiler ENCODEVERT. This program compiles corpora to binary files usable for the corpus manager MANATEE.

2 Corpus conversion

Conversion means concatenation of vertical files (one can contain more documents), looking for metadata of each document and creating document heading tag. This task has these specifics:

Language: Some corpora contain a few documents in different language, than the most common is. In that case the information about the document language is mentioned in metadata, and along this data field the documents are filtered.

Petr Sojka, Aleš Horák (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2008, pp. 94–97, 2008. © Masaryk University, Brno 2008

Multivalues in metadata: Some fields in metadata files can have more values. The specification [3] allows only the one-line notation:

```
M_Date: 2000-05-07; 2000-06-22
```

But in real data is possible to see this:

```
M_Date: 2000-05-07
M_Date: 2000-06-22
```

Also in XML it's not possible to use more attributes with the same name. So the right way for ENCODEVERT is `m_date="2000-05-07;2000-06-22"`. Naturally we must be careful in the case of language attribute if we need only one language.

Syntactic errors: There are syntactical errors in verticals and in metadata. For example this can appear in vertical as one position:

```
...stalo
```

Another problem are XML tags divided character after character to more lines, or bad order of XML tags eventually their bad occurrence. In metadata e.g. one data field on more lines can appear.

2.1 Implementation

The implementation on the NLP Centre's server is a set of Python and Shell scripts available in `/corpora/priprava_dat/ib047_99-07/scripts`.

vertjoin.py `Usage: vertjoin.py [-l lang] DIRECTORY [OUTPUT_FILE]`
Main script walking through directory tree with the root `DIRECTORY`, looking for `*.vert` and corresponding `*.meta` files and concatenating them on the standard output or to the `OUTPUT_FILE`.
Script expects verticals with document tags named as `doc` with obligatory attribute `id` corresponding to field `Doc` in metadata.
`vertjoin.py` also implements two methods to repair easy syntactic errors:

normal_meta A method for metadata which removes a possible backslash on the and of line and joins a data field written on two lines.

normal_vert A method for verticals which removes empty lines, strips possible white-spaces around the position, divides more words on one line, ensures the "..." not to be in the same position with some word and puts together short broken tags (means the tags without attributes e.g. </p>).

predesamb.sh `Usage: predesamb.sh INPUT_FILE`
Script repairing main syntactic errors of verticals. It's pipeline of smaller scripts. These scripts repair errors concerning concrete tags. It looks like this:

```
cat $1 |p_doc.py |p_p.py|p_tag.py|note_p.py|\
more_closing_tags.py|gg.py|g_tag_g.py|sed 's/</*q>/"/g'
```

Last `sed` command replaces <q> and </q> by symbol ". The `q` tag is specified in [2], but it's not accepted by ENCODEVERT.

3 Corpora tagging

A very important aspect of corpora is the morphological tagging. It makes corpora being especial tools even in Google ages. NLP Centre has it's own tagger for Czech language named DESAMB. It's developed by Pavel Šmerk and based on partial syntactic analyser DIS/VADIS developed by Eva Mráková-Žáčková.

DIS/VADIS is written in Prolog, which is a disadvantage, because it's quite slow. Rest of the DESAMB is written in Perl, but DIS/VADIS slows down whole process of tagging. So a future Pavel Šmerk's plan is to rewrite DIS/VADIS also to Perl.

Next disadvantage of Prolog is probably a faulty allocation of memory. It seems, that DESAMB is not able to process big corpora, because then the Prolog fails due to lack of memory. This problem appeared on verticals about 20 million positions long, but in this case, the second aspect was quite complicated structure of vertical (many tags, tables, long sentences, etc.).

So it's needed to divide verticals into more smaller parts before processing. A script `divide.py` implements this function, but it probably won't be included to DESAMB, because future Perl implementation of DESAMB doesn't need that.

divide.py `Usage: divide.py [-l lines] [-t doc_tag] FILE1 [FILE2 ...]`

Where `lines` is count of lines which is possible to shorten by one of letters `KkMm` (eg. `30K` means thirty thousands, `2m` means two million). The default value is 1 million. Default value of `doc_tag` is `"doc"`.

This script divides verticals – after counting `count_of_lines` – on the nearest document border. Output files are written in current directory having original name extended by three-figure number of part (`FILE1.001`, `FILE1.002`, ...).

The script can have more input files and also it can read from standard input.

Nevertheless, the tagging of corpora after dividing them is also slow. It becomes evident on the 20 million positions long verticals . It was tested on five such corpora and usually one of the corpora took about one day on one computer. But there was the especial one, processed about five days. Probable cause of this is count of long sentences (including also enumerations or tables without punctuation marks). All the corpora have a few sentences over 500 words long, but the 5-days one has about 460 of these sentences.

Last but not least current DESAMB has problems with parsing of corpora e.g. considering XML tags to be a word. The question is, if it is only because of complicated source code of parser, or if it can't be better because of too expressive syntax of corpora with vague definition.

A meaningful goal seems to be an improving tagger so, that its output would be usable for ENCODEVERT. But nowadays DESAMB would cause many warnings in ENCODEVERT, which can be prevented by a script `postdesam.sh`.

postdesam.sh A script repairing syntactic errors on the output of DESAMB. It also removes <s> and </s>, that are tags added in DESAMB to determinate sentence borders. Main part of the script is a pipeline consisting of `sed` substitute commands:

```
cat "$in"| sed '/^<\/*s> *$/d'| sed 's/<\/*s>//g' |\
 sed 's/^$/<l>/g' | sed 's/\(^<doc.*>\)\s*<doc.*$/\1/g'|\
 sed 's/<\(\/*[^<>][^<>]*\)>[\t ]*<\/*[^<>][^<>]*>[\t ]k?/<\1>/g'\
 > "$out"
```

Maybe it prevents more bugs than needed, because some substitutes were added during of changes in DESAMB.

4 Conclusion

This paper describes, some problems with building big corpora and shows a way how to solve them. Described way has its first result. It is a 80 million positions Czech corpus consisting of data collected by students in Pavel Rychly's course IB047 Introduction to Corpus Linguistics and Computer Lexicography.

Future plans are clear – to collect next data to the corpus. Hopefully it will be easier than the first 80 million, because now the tools are ready and other corpora probably contain more consistent data than corpora created by many students.

Acknowledgments. This work has been partly supported by the Academy of Sciences of Czech Republic under the project 1ET200610406.

References

1. Jak vytvořit korpus (2000) `http://nlp.fi.muni.cz/cs/Jak_vytvorit_korpus`.
2. Popis vertikálu (2000) `http://nlp.fi.muni.cz/cs/Popis_vertikalu`.
3. Popis metainformací (2000) `http://nlp.fi.muni.cz/cs/Popis_metainformaci`.