FI MU

Faculty of Informatics
Masaryk University

Automatic Processing of Czech Inflectional and Derivative Morphology

by

Radek Sedláček
Pavel Smrž

FI MU Report Series **FIMU-RS-2001-03**

Copyright © 2001, FI MU **June 2001**

Automatic Processing of Czech Inflectional and Derivative Morphology

Radek Sedláček, Pavel Smrž

May 29, 2001

Abstract

This paper deals with the effective implementation of the new Czech morphological analyser `ajka` which is based on the algorithmic description of the Czech formal morphology. First, we present two most important word-forming processes in Czech — inflection and derivation. A brief description of the data structures used for storing morphological information as well as a discussion of the efficient storage of lexical items (stem bases of Czech words) is included too. Then, we describe the morphological analysis algorithm in details and finally, we bring some interesting features of the designed and implemented system `ajka` together with current statistic data.

1 Introduction

Typically, morphological analysis returns the base form (lemma) and associates it with all the possible POS (part-of-speech) labels together with all grammatical information for each known word form. In analytical languages a simple approach can be taken: it is enough to list all word forms to catch the most of morphological processes. In English, for example, a regular verb has usually only 4 distinct forms, and irregular ones have at most 8 forms. On the other hand, the highly inflectional languages like Czech or Finnish present a difficulty for such simple approaches as the expansion of the dictionary is at least an order of magnitude greater[1] [4]. Specialised finite-state compilers have been implemented [1], which allow the use of specific operations for combining base forms and affixes, and applying rules for morphophonological variations [3]. Descriptions of morphological analysers for other languages can be found in [8, 11].

Basically, there are three major types of word-forming processes can be distinguished — inflection, derivation, and compounding. Inflection refers

[1] As our effective implementation of spell-checker for Czech based on finite state automata suggests, it does not necessarily mean that no application can take advantage of a simple listing of word forms in highly inflecting languages.

to the systematic modification of a stem by means of prefixes and suffixes. Inflected forms express morphological distinctions like case or number, but do not change meaning or POS. In contrast, the process of derivation usually causes change in meaning and often change of POS. Compounding deals with the process of merging several word bases to form a new word.

Czech belongs to the family of inflectional languages which are characterised by the fact that one morpheme (typically an ending) carries the values of several grammatical categories together (for example an ending of nouns typically expresses a value of grammatical category of case, number and gender). This feature requires a special treatment of Czech words in text processing systems. To this end, we developed a universal morphological analyser which performs the morphological analysis based on dividing all words in Czech texts to their smallest relevant components that we call *segments*. The notion of segment roughly corresponds to the linguistic concept *morpheme*, which denotes the smallest meaningful unit of a language.

Presented morphological analyser consists of three major parts: a formal description of morphological processes via morphological patterns, an assignment of Czech stems to their relevant patterns and a morphological analysis algorithm.

The description of Czech formal morphology is represented by a system of inflectional patterns and sets of endings and it includes lists of segments and their correct combinations. The assignment of Czech stems to their patterns is contained in the Czech Machine Dictionary [10]. Finally, algorithm of morphological analysis using this information splits each word into appropriate segments.

The morphological analyser is being used for lemmatisation and morphological tagging of Czech texts in large corpora, as well as for generating correct word forms, and also as a spelling checker. It can also be applied to other problems that arise in the area of processing Czech texts, e.g. creating stop lists for building indexes used in information retrieval systems.

2 Czech Inflectional Morphology

The main part of the algorithmic description of formal morphology, as it was suggested in [10], is a pattern definition. The basic notion is a *morphological paradigm* — a set of all forms of inflectional word expressing a system of its respective grammatical categories.

As stated in [5], the traditional grammar of Czech suggests much smaller paradigm system than there is existing in reality. For this reason we decided to build quite large set of paradigm patterns to cover all variations of Czech from the scratch. Fortunately, we had not been limited by technical restrictions,[2] thus we could follow the straightforward approach to the linguistic adequacy and robust solution.

[2] Hajič [5], e.g., indicates that his system is limited to 214 paradigm patterns.

The detailed description of all variations in Czech paradigms enables us to define application dependent generalisations of the pattern system. For example, if we do not need to take into consideration archaic word forms for a specific application, the number of paradigm pattern (that reaches 1500 in its fully expanded form) can be automatically reduced considerably.

Noun paradigms consist of word forms in particular cases of singular and plural. Verbs have more paradigms — for present tense, for imperative forms, etc.

For example, the nouns *hora* (mountain), *slza* (tear) and *řeka* (river) display the following forms in the paradigms:

Nom.	Gen.	Dat.	Acu.	Voc.	Loc.	Ins.
hora	hory	hoře	horu	horo	hoře	horou
hory	hor	horám	hory	hory	horách	horami
slza	slzy	slze	slzu	slzo	slze	slzou
slzy	slz	slzám	slzy	slzy	slzách	slzami
řeka	řeky	řece	řeku	řeko	řece	řekou
řeky	řek	řekám	řeky	řeky	řekách	řekami

As we can see, the corresponding word forms in the paradigms have the same ending. That is why we can divide the given word form into two parts: a *stem* and an *ending*. We then obtain the following segmentation:

hor-{a,y,u,o,ou} hoř-{e,e}
hor-{y,_,ám,y,y,ách,ami}
slz-{a,y,e,u,o,e,ou}
slz-{y,_,ám,y,y,ách,ami}
řek-{a,y,u,o,ou} řec-{e,e}
řek-{y,_,ám,y,y,ách,ami}

In the paradigm for the word *hora*, there are two alternative stems (*hor* and *hoř*); in the paradigm for the word *slza*, there is only one stem *slz*; and in the paradigm for the word *řeka*, there are again two alternative stems *řek* and *řec*. We can also identify four different ending sets S1={a,y,u,o,ou}, S2={e,e}, S3={y,_,ám,y,y,ách,ami} and S4={a,y,e,u,o,e,ou}, but it is clear that S4=S1+S2.

This observation leads us to a system of ending sets. We make distinction between two types of sets — *basic* and *peripheral* ending sets. The basic ones contain endings that have no influence to the form of the stem, while endings from the peripheral ending sets cause changes in the stem. In our case, sets S1 and S3 are basic and S2 is peripheral, because the ending *e* causes alternation change of the last letter *r* to *ř* in the stem *hor* and, similarly, *k* to *c* in the stem *řek*.

Moreover, we can put all the endings from set S1 and S3 into one (newly created) set, say S5, because they are both basic and are common for stems *hor*, *slz* and *řek*. Now we can shortly write previous paradigms in the following way:

```
hor-S5 hoř-S2
slz-S5+S2
řek-S5 řec-S2
```

Every ending carries values of grammatical categories of the relevant word form. For example, all endings in previously defined sets are characterised as endings of nouns in feminine gender. Endings from sets S1 and S2 originate from the singular paradigm, the others (from S3) express plural. Thus, the set S5 now includes endings from both singular and plural paradigm and this information must be preserved and stored in the system, so we decided to use the following data structure for storing ending sets:

```
S5=[1FS.](a,1)(y,2)(u,4)(o,5)(ou,7)
   [1FP.](y,1)(_,2)(ám,3)(y,4)(y,5)
         (ách,6)(ami,7)
S2=[1FS.](e,3)(e,6)
```

An ending set is denoted by its unique identifier (S5, S2) and consists of collections of pairs (an ending, a value of the appropriate grammatical category) in parenthesis. Each block of these pairs begins with a grammatical tag in brackets. This tag encodes values of grammatical categories that are common to all endings in the block. We can see that S5 has two blocks — the first one (for singular) begins with `[1FS.]` and contains five pairs, the second (for plural) starts with `[1FP.]` and has seven pairs. S2 includes only one block of two pairs — the first pair determines an ending in dative and the second specifies an ending for the locative of a noun. A more detailed description of the structure of grammatical tags can be found in [13].

Since endings, grammatical tags and values of grammatical categories repeat in the definitions of sets, we store them in unique tables and use references to these tables in the definitions of sets.

In the next step, we perform further segmentation of stems into a *stem base* and an *intersegment*. The stem base is the part that is common to all word forms in the paradigm (it doesn't change) and the intersegment is a final group of the stem which forms changes. We obtain the following segmentation of stems:

```
ho-{r,ř}
slz-{_}
ře-{k,c}
```

Since stems *hor* and *řek* can be followed by the endings from the set S5, while stems *hoř* and *řec* can be followed by endings from the set S2, and stem *slz* accepts endings from both S5 and S2, we have to store the information about the only possible combinations of stem bases, intersegments and endings in our system in the form of a *pattern* definition. To this end, we use the following data structure:

```
hora+<r>S5<ř>S2
slza+<_>S5,S2
řeka+<k>S5<c>S2
```

A pattern is denoted by its unique identifier (*hora*, *slza*, *řeka*) and it consists of blocks that are prefixed with an intersegment visually closed in <>. The special character "_" stands for an empty intersegment. Each block then contains a list of identifiers of sets. Identifiers of sets are visually separated by a comma in lists. Again, since intersegments and lists of identifiers repeat in the definitions of patterns, we store them in unique tables and use references to these tables in the definitions.

3 Derivative Processes

As have been shown in the previous paradigms, the morphological process of inflection is captured by means of paradigms in our system. Compounding does not play crucial role in Czech morphology if compared with other languages, e. g. German [7]. Therefore, the description of derivative processes remains untouched so far. It will be discussed in this section.

The process of morphological derivation of new words, primarily with distinct POS categories, is considered as a higher degree of morphological process, in the level above the inflection. Indeed, for example, a particular class of deverbative adjectives can be derived from the derivation paradigm of transitive verbs. A hierarchical system of morphological paradigms has been implemented as a tool able to capture different levels of the Czech morphology.

Hierarchical patterns are constructed fully automatically from the binding defined on the level of basic forms connecting always one lemma with another one by a specified type of a link. If a process could be described as a n-ary relation, it would be partitioned into $n-1$ binary relations. This partitioning is much more flexible and allows automatic generalisations of derivation relations. To demonstrate the derivation binding on the level of lemmata, we present the following example with participles:

```
počítat--(DEVESUBST)--počítání          count--counting
počítat--(DEVEADJPAS)--počítaný         count--counted
počítat--(DEVEADJPASSHORT)--počítán     count--is counted
počítat--(DEVEADJACTIMPF)--počítající   count--is counting
```

From the given example it follows that each link connects one base form of a word with another one and names such relation. If the label of a base form is unambiguous and therefore it can be used as an primary identifier it is sufficient to specify only these labels in the binding process. If the label itself can be ambiguous, the pairs of lemma and the relevant inflectional pattern are connected. However, even this approach is not able to represent completely the dependency of the relation on a particular sense of a word. For example, the relation "possessive adjective" applies only for the reading of *jeřáb* denoting a bird. This is the reason why we have implemented the system connecting pairs of triplets of (sense-id, lemma, paradigm) by a named relation.

Indexing techniques and dictionary methods [6] used in our implementation allow an efficient retrieval of related lemmata. It is also possible quickly

return a chosen base form for a set of related words — the feature which is highly favourable in several applications, e.g. in the area of information retrieval or indexing Internet documents.

The system of base form binding is not limited to the basic derivative processes described above. The same principle e.g. depicts two types of relation in the level under the basic derivation, namely original / adapted orthography and inflectional / non-inflectional doublets in the case of loanwords. The former can be demonstrated by the example of a link between *gymnasium* and *gymnázium* (in the actual version of our morphological analyser we use even more elaborated assignment of these doublet types in the form of basic type of relation and more specific subtype). A link between the word *abbé* assigned to the paradigm *abbé* (non-inflectional) and *Tony* (inflectional) is the example of inflectional / non-inflectional doublet. It is of course possible to model such relations on the basic level of inflectional paradigms as a word-form homonymy. However, it would lead to the mixture of unrelated forms and would complicate special types of analyses, e.g. a style-checker analysis, that could be very interesting.

There are other relations that connect lemmata above the level of basic derivative processes. We take advantage of the standard process and are able to uniformly describe such different relations as diminutives (and its degree):

`vůz--(DIMIN:1)--vozík`
`vůz--(DIMIN:2)--vozíček`,

aspectual relations of verbs:

`říci--(ASPPAIR)--říkat`,

iterative relations of verbs (together with “degrees”):

`chodit--(ITER:1)--chodívat`
`chodit--(ITER:2)--chodívávat`,

the relations between an animate noun and derived possessive adjective:

`otec--(MASCPOSS)--otcův`,

the process of creation feminine from masculine nouns:

`soudce--(MASC2FEMI)--soudkyně`,

or synonyms and antonyms:

`kosmonaut--(SYNO)--astronaut`
`mladý--(ANTO)--starý`.

The last class of links brings us directly to other relations that can be found in semantic nets like Wordnet [9]. Typical relations of hyperonym / hyponym, part / whole (meronyms) etc. are modelled on the higher level, the level based on synonyms, to be able to link groups of synonyms (that are called synsets in the context of Wordnet).

The possibility of building complex structures of links, e.g. relations of relations, is also employed in connecting roots of loanwords to their Czech equivalents. Similarly to [12], we are therefore able to relate words derived from the Greek root *kard* with the group of Czech words derived from the Czech root *srd*, e.g. *osrdečník, kardiostimulátor, srdce, kardiologie*.

In the first paragraph of this section we have been speaking about hierarchical morphological patterns. So far, however, we have presented only several types of relations connecting particular words. It is justified by the fact that the patterns are considered only as the linguistic interpretation of statistics obtained from relational data. It is only the matter of the point of view where to place a threshold on the frequency of concrete relation behind which we will interpret data as an exception or peculiarity and which relation will form a particular derivative pattern.

4 Morphological Analysis Algorithm

The basic principle of the algorithm for morphological analysis of Czech word forms is based on the segmentation described in Section 2. The result is that every Czech word form W can be divided into four segments — a prefix P, a stem S, an intersegment I and an ending E. Thus, we obtain the following equation $W = P + S + I + E$. The aim of the algorithm for morphological analysis is to find such a segmentation, i.e. to identify these four segments in a given word form. The separated ending then determines values of grammatical categories.

The algorithm of the analysis consists of the following steps:

1. The input of the algorithm is a word form $W = a_1 a_2 \ldots a_n$.
2. Try to separate prefix nej+ne, nej or ne from the beginning of the word W and obtain $W_1 = S + I + E = a_1 a_2 \ldots a_n$.
3. Try to identify stem base between all possible candidates $S_i = a_1 a_2 \ldots a_i$, where $1 \leq i \leq n$.
4. The word segmentation $P + S + I + E$ and the values of grammatical categories are the output.

The first step of the analysis is a prefix separation. The algorithm tries to separate only the prefix nej+ne or the superlative prefix nej or the negative prefix ne (in this order). After the prefix separation, the stem base S stands in the front position of the word form $W_1 = S + I + E$.

At this moment, the identification of the stem base S is being performed in a character-by-character manner. Let us suppose that the word form W_1 can be written as a sequence of characters $a_1 a_2 \ldots a_n$, where $n \geq 0$. If W is a correct Czech word form and the prefix P was separated correctly, then the stem base S is one of the strings $S_i = a_1 a_2 \ldots a_i$, where i satisfies $0 \leq i \leq n$. Because of possible homonymy, it is necessary to prove all possible candidates S_i.

The process of identifying the candidate S_i as the right stem base of the word form W_1 consists of three steps:

- Look at the candidate S_i in the dictionary of stem bases.
- Try to match the rest $a_{i+1}a_{i+2}\ldots a_n$ of the word form W_1 with one of the combinations $I + E$ allowed in the pattern definition.
- Check if the prefix P was separated correctly.

At first, the candidate S_i is being looked up in the dictionary of stem bases. If the candidate S_i is found in the dictionary, the algorithm checks whether the rest $a_{i+1}a_{i+2}\ldots a_n$ of the word form W_1 is one of the correct combinations $I+E$ that are included in the definition of some of the patterns that the stem base S_i belongs to. With the ending E we obtain appropriate grammatical information as well.

It is necessary to check all patterns and all possible combinations $I + E$ in their definitions, because we request all possible values of grammatical categories on the output. However it is clear that if the first character a_{i+1} does not match with the first character of the intersegment I or with the ending E (when the intersegment is empty), further comparisons of characters $a_{i+2}, \ldots, a_n$ are useless and the algorithm does not need to perform a non-trivial number of them. This is true especially in the case of unsuccessful match with the intersegment, because all comparisons of all endings in all ending sets can be then missed. Finally, the algorithm checks whether the separation of the prefix P was correct, which it means that the separation is consistent with the information stored in the Czech Machine Dictionary.

If there is no collision in one of the three steps, the word form W is accepted and correctly analysed. Furthermore, all grammatical information describing this word form is available and can be sent to the output in the form of grammatical tags.

The effectiveness of the algorithm depends mainly on the speed of looking up the candidates S_i in the dictionary. There is a relation between the candidate S_i and S_{i+1} such that the candidate S_i is a prefix of the candidate S_{i+1}. That is why we use a trie structure for storing stem bases of Czech word forms. In this case, if the previous candidate S_i was not found in the trie structure, then trying to find the following candidate S_{i+1} is useless, because it is sure that it can not be there. If the candidate S_i was found, then the search algorithm can use this fact and can continue in finding the string $a_{i+1}a_{i+2}\ldots a_n$ from the place where it has stopped looking at the previous candidate. For detailed information about searching in the trie structure, see [6].

Memory requirements are one of the main disadvantages of the trie structure. We tried to solve this problem by implementing the trie structure in the form of the minimal finite state automaton. The incremental method of building such an automaton was presented in [2] and is fast enough for our purpose. Moreover, the memory requirements for storing the minimal automaton are significantly lower (see Table 1).

5 Czech Morphological Analyser `ajka`

The implementation of the presented analyser is based on the algorithm of morphological analysis described in Section 4. Because we decided to use dictionaries, the main part of morphological information is included in data files.

There are two binary files that are essential for the analyser. One of them contains definitions of sets of endings and morphological patterns stored in data structures described in Section 2. The source of this binary file is a text file with definitions of ending sets and patterns. The second is a binary image of the Czech Machine Dictionary and contains stem bases of Czech words and auxiliary data structures. We developed a program `abin` that can read both of these text files and efficiently store their content into appropriate data structures in destination binary files.

The first action of the analyser is loading these binary files. These files are not further processed, they are only loaded into memory. The main reason for this solution is to allow as quick a start of the analyser as possible. The next actions of the analyser are determined by steps of the morphological analysis algorithm (see Section 4).

Features and behaviour of the analyser are more important information for potential users. The analyser works in two modes. If the name of a text file for analysis was written on the command line, then the analyser works in a batch mode. The text file is supposed to contain text, one word per line. Otherwise, if there is no name of the text file, the analyser works in an interactive mode.

To control the analyser in the interactive mode is very easy. User simply inputs a word form to be analysed after the prompt "`ajka>`". The output format is influenced by the working mode. User can choose a normal mode or a brief mode or a mode that makes the analyser to generate all possible derived word forms. User terminates the analyser by typing the special character "#".

Another feature of the analyser is a possibility to select various forms of the basic word form (lemma).

Finally, user can have more versions of binary files that contain morphological information and stem bases and can specify which pair should be used by the analyser. Users can take advantage of this feature to "switch on" analysis of colloquial Czech, domain-specific texts etc.

The power of the analyser can be evaluated by two features. The most important thing is number of words that can be recognised by the analyser. This number depends on the quality and richness of the dictionary. Our database contains 223,600 stem bases and `ajka` is able to analyse (and, conversely, generate) 5,678,122 correct Czech word forms. The second feature is the speed of analysis. In the brief mode, `ajka` can analyse more than 20,000 words per second on PentiumIII processor with the frequency of 800MHz. Some other statistic data, such as number of segments and size of binary files, is shown in the following Table 1.

Table 1: Statistic data

#intersegments	779
#endings	643
#sets of endings	2,806
#patterns	1,570
#stem bases	223,600
#generated word forms	5,678,122
#generated tags	1,604
speed of the analysis	20,000 words/s
dictionary	1,930,529 Bytes
morph. information	147,675 Bytes

6 Applications

The main present application of the analyser is morphological tagging of Czech corpus texts. The task of tagging is to assign relevant grammatical information to every word form. This grammatical information is known after the analysis and thus it can be sent to the output in the form of a grammatical tag.

Automatic synthesis is the reverse process of using the algorithmic description of Czech morphology. It is possible to generate a set of all possible word forms and their grammatical categories simply by applying patterns as rules determining the only correct endings of the word form.

Automatic morphological analysis and synthesis is a key process for lemmatisation. Morphological synthesis allows us to generate all possible forms. Amongst them, there is a special dedicated word form, lemma. A process of lemmatisation is in such a case reduced into the problem of choosing the required word form. (for example a nominative of singular for nouns).

The analyser can be used as a spelling checker as well. If the word segmentation mechanism is unable to split a given word into segments, usually it is a construct containing a spelling mistake.

7 Conclusion

We have described two-phase ternary segmentation of Czech word forms for the need of automated morphological analysis by a computer. We have briefly explained the data structures representing Czech formal morphological processes (a system of sets of endings, pattern definitions and hierarchical patterns) as well as the data structures used for storing stem bases and information from the Czech Machine Dictionary. Finally, we have shown the steps of the morphological analysis algorithm in detail, mentioned some features of the analyser `ajka` and given several examples of practical applications where it is being used.

The morphological analyser `ajka` has been tested on large corpora containing 100,000,000 positions. Based on the test results, the definitions of sets of endings and patterns as well as the Czech Machine Dictionary are being extended by some missing, mostly foreign-language stem bases and their appropriate patterns and endings. In its current state, `ajka` can be used for morphological analysis of any raw Czech texts.

The analyser `ajka` can readily be adapted to other inflectional languages that have to deal with morphological analysis. In general, only the language-specific parts of the system, i.e. definitions of sets of endings and the dictionary, which are stored as text files, have to be replaced for this purpose.

References

[1] Kenneth R. Beesley and Lauri Karttunen. Finite-state non-concatenative morphotactics. In *Proceedings of the Fifth Workshop of the ACL Special Interest Group in Computational Phonology*, 2000.

[2] Jan Daciuk, Richard E. Watson, and Bruce W. Watson. Incremental construction of acyclic finite-state automata and transducers. In *Finite State Methods in Natural Language Processing*, Bilkent University, Ankara, Turkey, June – July 1998.

[3] G. Grefenstette et al. *Recognizing Lexical Patterns in Text*. Kluwer Academic Publishers, 1st edition, 2000.

[4] Jan Hajič. *Unification Morphology Grammar*. Ph.D. Thesis, Faculty of Mathematics and Physics, Charles University, Prague, 1994.

[5] Jan Hajič. *Disambiguation of Rich Inflection (Computational Morphology of Czech)*. Charles University Press, 1st edition, 2000. In preparation.

[6] Donald E. Knuth. *The Art of Computer Programming: Sorting and Searching*, volume 3. Addison Wesley, 2nd edition, 1973.

[7] Gabriele Kodydek. A word analysis system for German hyphenation, full text search, and spell checking, with regard to the latest reform of German orthography. In *Proceedings of the Third Workshop on Text, Speech and Dialogue — TSD 2000*, 2000.

[8] Wolfgang Lezius, Reinhard Rapp, and Manfred Wettler. A freely available morphological analyzer, disambiguator and context sensitive lemmatizer for German. In *Proceedings of the COLING-ACL*, 1998.

[9] G. A. Miller. Five papers on Wordnet. Technical report, Princeton, 1993.

[10] Klára Osolsobě. *Algorithmic Description of Czech Formal Morphology and Czech Machine Dictionary*. Ph.D. Thesis, Faculty of Arts, Masaryk University Brno, 1996. In Czech.

[11] S. Murat Oztaner. A word grammar of Turkish with morphophonemic rules. Master's thesis, Middle East Technical University.

[12] Emil Páleš. *Sapfo — Paraphraser of Slovak.* Veda, Bratislava, 1994.

[13] Radek Sedláček. Morphological analyser of Czech. Master's thesis, Faculty of Informatics, Masaryk University Brno, 1999. In Czech.

**Copyright © 2001, Faculty of Informatics, Masaryk University.
All rights reserved.**

**Reproduction of all or part of this work
is permitted for educational or research use
on condition that this copyright notice is
included in any copy.**

Publications in the FI MU Report Series are in general accessible via WWW and anonymous FTP:

`http://www.fi.muni.cz/informatics/reports/`
`ftp ftp.fi.muni.cz (cd pub/reports)`

Copies may be also obtained by contacting:

**Faculty of Informatics
Masaryk University
Botanická 68a
602 00 Brno
Czech Republic**