Yet Another Formalism for Morphological Paradigm

Marek Grac

Natural Language Processing Centre, Faculty of Informatics
Masaryk University, Brno
xgrac@fi.muni.cz

Abstract. Morphology is one of the few areas in the natural language processing where computers are good enough. Different approaches lead to different problems. For Slavonic languages rules and statistical methods are commonly used. Rule based methods are more precise but tend to fail when parsing unknown words. Hybrid technologies with statistical methods helps to solve this problem. It is also possible to solve this problem by extending existing rule-based resources. These resources can be used also for other linguistic research. This paper presents new formalism which is closer to human understanding of natural language morphology and its application in extending morphological dictionary.

Key words: morphology; morphological analysis

1 Introduction

Morphology was first area in natural language processing reaching maturity. Tokenization, splitting running text into tokens, is difficult problem for Arabic or Chinese but not for Slavonic languages. Morphological analysis was first real problem for them. Simple solutions suitable for English with its simple morphological system shown to be ineffective for Slavonic languages. For them morphological paradigm, contains information about lemma and possible word forms, needs higher level of abstract formalism.

2 Suffix Based Formalism

Rule based system for Slavonic languages are popular and widely used (e.g. [1]). These systems use different sets of grammatical tags and are not used for different languages. Most of them use suffix grammar to model existing paradigm. This approach is suitable for Slavonic languages because they use mostly postfix morphology with limited prefix morphology (verb and adjection negation *ne*, superlatives *naj*. Formalisms uses only few basic operations that can be performed at the end of word e.g. add/remove character. Such simple formalism helps us to create morphological analyzers that look simple and can be very fast (tens of thousands analysis per second). Disadvantage of this approach is that we have to define large number (hundreds) of patterns. In [2] we can find several tricks how to partially reduce their numbers in particular cases but it won't help too much.

Petr Sojka, Aleš Horák (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2009, pp. 9–12, 2009.

3 Linguists Defined Formalism

Morphology is interested not only for computer linguists but also for traditional linguists. Our attempts to formalize patterns described in linguistics resources ended with puzzled results. Authors of these books wrote them for readers with their language experience, so explained patterns uses words like 'sometimes' or 'mainly'. Ambiguity of these patterns is problem even for non-expert readers. This formalism when one pattern is described on several pages is not suitable for computers as we cannot parse them correctly. Main benefit of this approach is that we ends in small amount of patterns which are distinguishable from each other.

4 Yet Another Formalism

Our attempts to formalize knowledge in monographies about morphology lead us to create formalism which will be closer to traditional patterns but still unambiguous. In SBF we define pattern as a set of suffixes with tags. LDF extends this because it tends to use semantic characteristics or etymology. Unfortunately these information are not easily accessible for machine usage. We have found that only suitable feature of LDF patterns for us is condition constraining lemma. Usually lemma has to have defined suffix (this is covered by SBF, too) and after removing this suffix, it is possible to generate new conditions for this form. For Slovak and Czech language we found out that they belongs to these group:

- on N-th position is character X *b, ch*
- on N-th position is character which belongs to class X (e.g. soft consonant *š*, long vowel *á*
- on N-th position ends long/short syllable *domáci, cudzí*

Using set of conditions from previous list can rapidly reduce number of lemma which can potentionaly be part of pattern. Problems which does not have to be solved in SBF arise. Some characters contain more then one letter (e.g. ch) and others are specific for one languages (e.g. ř, dž). Also splitting word into character can be ambiguous (e.g. viachlas does not contain character ch). Also syllables can be different across languages. Splitting word into syllables is not a trivial problem. For our purpose we don't care about boundaries and we just have to found core of syllable and decide if it is short or long.

Each pattern tries to describe every accepted word form for lemma. Such word form can be divided in various ways. In SBF generation rules usually contain of what should be removed and suffix which will be added. We are following this simple method but generalize it a bit. Each word form consist of prefix, base and suffix. Each pattern can define several bases so in pattern generation we just point to them using identificators. Using several bases is based to LR (also applied in Slovak morphological database [3]). As we want do remove ambiguity, algorithm of creating base from lemma have to be disambiguate.

Algorithm for creating base can be part of lemma requirements and then it apply only to lemma which successfully based requirerements. Or it can be part of pattern directly if it will be applied to every lemma belonging to pattern. We are aware that extensive usage of bases can result in having empty suffixes. Usually for Czech or Slovak, we use one or two bases for nouns and up to five of them for verbs. Those bases can be created by operations. Several of them are language independent e.g. remove and add character on/to N-th position (operations: chop and append). It is possible to create user defined operations but it is not necessary in most of the cases.

Next part of pattern generation consists of rules. Each word form is defined with its prefix, pointer to base and suffix. It is possible to generate several word forms with same tag. Concatenating prefix, base and suffix does not have to result in final word form. In some cases we want to polish it a bit. For Slovak we want quite often to shorten last syllable if previous one is long e.g. domácí -> domáci. Such filters can be applied directly to word form. Second possibility are filters that are generally valid across language, they can be defined for language itself and does not have to be mentioned in patterns e.g. in Slovak medveď + e -> medvede.

In some cases we want to add a special pattern to lemma. We distinguish two such cases. First case is when we know that given word form can be used as lemma with defined pattern. Example of such case in Slovak is generation of deverbalism plávať -> plávanie (pattern vysvedčenie). Second case is different because in some cases we want to use pattern of lemma. Such case is best shown on pattern negation which adds prefix ne- and then copy existing pattern (we will have just one negation for each PoS).

5 Langusta Framework

In project Langusta we attempted to write an implementation of previous formalism. We have decided to use Java programming language and free BSD license. Result of our work is framework which consist of language independent parts which are inherited by language specific parts.

Defining new Slavonic language is very simple. We have to begin with alphabet definition which contains all characters and classes to which they belong. Second step is to modify general word to characters splitter to cover cases when there is an ambiguity. After these two steps we are able to start writing patterns. Usually after just few of them we will realize which operations and filters will be usefull to write. They have to follow appropriate interface and in the language definition we will add those Java objects to identificators used in pattern definiton. This part usually takes few days. Last and usually the more complex and time consuming part is to write pattern definition for flective parts of speech.

We are in a process of creating set of morpholigical patterns but preliminary results are promising. Our tool based on Trdlo framework allow us to work very fast 200–300 lemma/hour because it shows expert only those patterns that

are acceptable for given lemma. It means that even if we have around 50 verb patterns for Slovak in usual case only 2 or 3 them are shown.

6 Conclusion

In this paper we presented a new approach for formalisation of morphological pattern. Benefits of such approach where explained. Architecture of framework Trdlo was described and preliminary results are promising. In the future we would like to finish writing patterns for Slovak and Czech. Having such resource for several languages can be very usefull for comparative linguists. After we will annotate enough data we are going to try to use more automatic methods to determine pattern according to data found in non-tagged corpora.

Acknowledgments This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536 and in the National Research Programme II project 2C06009.

References

1. Sedláček, R.: Morfologický analyzátor češtiny. Master's thesis, FI MU, Brno (1999).
2. Garabik, R.: Levenshtein Edit Operations as a Base for a Morphology Analyzer. Computer Treatment of Slavic and East European Languages. Ed. R. Garabík. Bratislava: Veda (2005) 50–58.
3. Benko, V., Hašanová, J., Kostolanský, E.: Morfológia podstatných mien. Počítačové spracovanie prirodzeného jazyka. Pedagogická fakulta UK, Bratislava (1998).