Feature Unification for Morphological Parsing in Bangla

Sajib Dasgupta
Department of CSE,
BRAC University, Bangladesh.
sajib44new@bracuniversity.net

Dr. Mumit Khan
Department of CSE,
BRAC University, Bangladesh.
mumit@bracuniversity.net

Abstract

This paper describes a Feature Unification Based Word Grammar model for the morphological parsing of Bangla words. While normal morphological parsing strategy is adequate to decompose a word into morphemes, it is not able directly to compute the part of speech of a derivationally complex word or return a word's inflectional features--precisely the information required for syntactic parsing. These deficiencies have now been remedied by adding a unification-based word grammar component which can provide parse trees and feature structures. In addition to that, feature unification lessens the number of lexicon classes (less space) and actually reduces the complexities regarding morphotactic analysis.

INTRODUCTION

Normal morphological parsing strategy decomposes a word into morphemes given lexicon list, proper lexicon order and different spelling change rules. But this is not enough to compute the part of speech of a derivationally complex word or return a word's inflectional features. In this paper we will discuss about feature based morphological parsing for Bangla which gives us parts of speech and other morphological features in addition to the morpheme division. **[2][8]** At first we give an idea of normal morphological parsing, then we discuss on feature based morphological parsing and in the end we shed light on the comparisons between the two approaches.

NORMAL MORPHOLOGICAL PARSER

In the normal morphological parser or generator there are actually 3 components: (1) Lexicon (2) Morphotactics (3) Orthographic Rules. **[5]**

1. Lexicon

The list of stems and affixes, together with basic information about them (whether a stem is a Noun stem or a Verb stem, etc.). Every lexicon is of a certain class.

Example: Here is an example

hAt[1] (হাট)
Class: Verb_Stem or Root
Feature: Parts of Speech = Verb

All the lexicons in a certain class is stored in a FSA (Finite State Automata).

2. Morphotactics

The model of morpheme ordering that explains which classes of morphemes can follow other classes of morphemes inside a word. For example, the rule that the Bengali Tense_Person_Affixes follow the Verbs rather than preceding it. Normally morphotactics is implemented using Finite State Automata (FSA). For example the following FSA can be a representation of morphotactic analysis for Bangla:

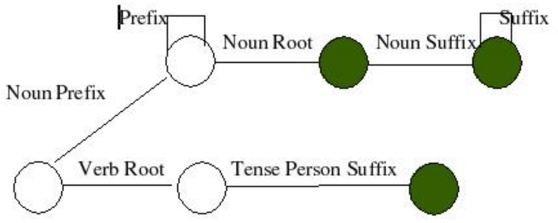


Figure 1: FSA representing morphotactics

3. Orthographic Rules

These spelling rules are used to model the changes that occur in a word, usually when two morphemes combine. For example root word hAt (হাট) is changed into hEt (হেট) when added with verb suffix to form a word hEtECI (হেটেছি):

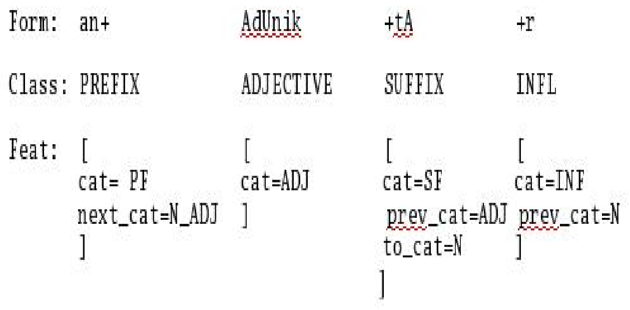
PC_KIMMO version 1 implements this parsing strategy. **[1] [12]**

1 Through out this paper we have used English alphabet to represent Bangla characters. For example "আ" is "a", "া " is "A", "ি " is "I", "ক" is "k", "খ" is "K", "য়" is "y", "্ "(hasanta) is "~" etc.
We have also assumed that the words are given in **Unicode Format (vowel comes after consonant)**. For example খেয়েছি is represented as KEyECI.

FEATURE BASED MORPHOLOGICAL PARSING

This is a morphological parser which uses a unification based chart parser given a proper word grammar. *It does so by adding an extra analytical component Word Grammar in addition to the three components described previously in the normal parsing strategy.*

Just as a sentence parser produces a parse tree with words as its leaf nodes, a word parser produces a parse tree with morphemes as its leaf nodes. When we parse a sentence, it is normally already tokenized into words (since we put white space between words); but when we parse a word, we must first tokenize it into morphemes. This tokenizing is done by the morphotactic and orthographic rules and lexicon. When a surface word is submitted to a Recognizer, the rules and lexicon analyze the word into a sequence of morpheme structures (or possibly more than one sequence if more than one analysis is found). A morpheme structure consists of a lexical form, its gloss, its category, and its features. For example, the word **anAdUnIktAr (অনাধুনিকতার)** is tokenized into this sequence of morpheme structures.



Form:	an+	AdUnik	+tA	+r
Class:	PREFIX	ADJECTIVE	SUFFIX	INFL
Feat:	[cat= PF next_cat=N_ADJ]	[cat=ADJ]	[cat=SF prev_cat=ADJ to_cat=N]	[cat=INF prev_cat=N]

Figure 2. Morpheme structure

Here cat, next_cat, to_cat, prev_cat all are **feature variables** and PF (prefix), ADJ (adjective), N_ADJ (both noun and adjective), N(noun), SF (suffix), INF(inflection) are **features**. The descriptions of the features are as belows:

cat:
It specifies the category of a lexicon.
It can be N, ADJ, V, P, ...

next_cat:
It specifies the lexical category of the stems to which it can attach as a prefix.
It can be N, ADJ, V, P, ...

prev_cat:
It specifies the lexical category of the stems to which it can attach as a suffix.
It can be N, ADJ, V, P, ...

This analysis (all the tokens) is then passed to the word grammar which returns the parse tree and feature structure. Word grammar portion actually contains rule list showing how to form a word and all the feature constraints. **[8][5]** *We can use a chart parser to get a parse tree. For every node in the parse tree we have to ensure that no feature constraint is violated***.** Features of a certain node are actually those features which are derived from the features of the child nodes. So for a node in the parse tree we have to do two things :

(1) Feature Unification

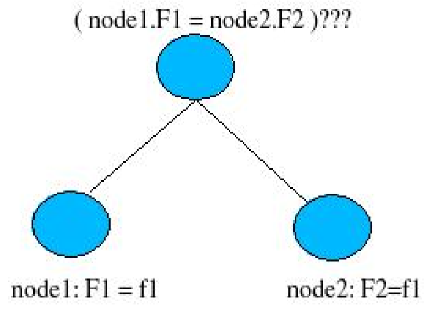


Figure 3. Feature Unification

Feature unification is to see whether the feature constraint specified in the parent node prevails if we have the features from the child nodes. For example in the above picture in the parent node we have to see whether feature F1 of node1 is equal to the feature F2 of node2. If it is not true then this parse tree formation is false.

(2) Feature Collection

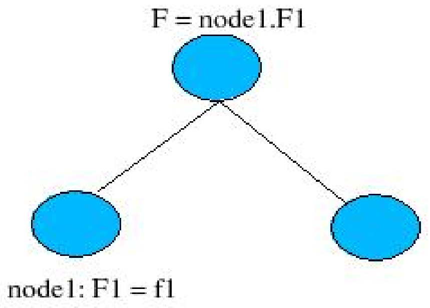


Figure 4. Feature Collection

It is to collect features from the child nodes. For example in the above picture in the parent node feature F is equal to the feature F1 of node1.

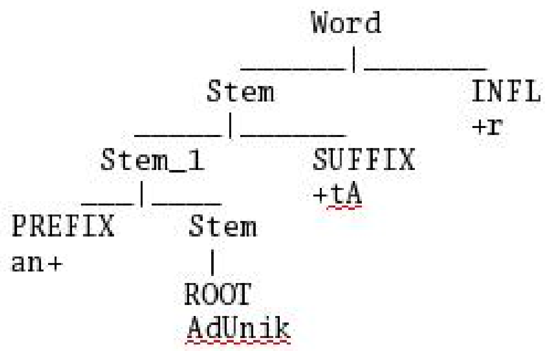
So for the Bangla if we define a word grammar like this [PCKIMMO Version2]: **[2][7][13]**

```
Word    =  Stem INFL
   <Stem cat> = <INFL prev_cat >  //feature unification
   <Word cat> = <Stem cat>                //feature collect

Stem     =  Stem_1 SUFFIX
   <Stem_1 cat> = <SUFFIX prev_cat >
   <Stem cat>    = <SUFFIX to_cat>

Stem_1 =  PREFIX ADJECTIVE
   <PREFIX next_cat> = <ADJECTIVE cat >
   <Stem_1 cat>          = < ADJECTIVE cat>
```

Then after the chart parsing and feature unification we get the following parse tree and feature structure:



```
Word:
[
  cat= N
]
```

Fig 5: Parse tree and feature structure for anAdUnIktAr(অনাধুনিকতার).

Here we can see that after the final parsing the top node Word has feature cat=N which specifies that the final word's category is NOUN although its root word **a*dUnIk*** is actually ADJECTIVE. This is because the SUFFIX **tA** is added with the ADJECTIVE and changes it into NOUN. This feature constraint specified above in the word grammar is specified once again as belows:

```
Stem     =  Stem_1 SUFFIX
  <Stem_1 cat> = <SUFFIX prev_cat > //unification
  <Stem cat>    = <SUFFIX to_cat>   //feature collect
```

This states that prev_cat feature of SUFFIX has to be same with the cat feature of Stem_1 and cat feature of Stem is equal to the to_cat feature of SUFFIX. For the word ***anAdUnIktAr***, ***anAdUnIk*** is Stem_1 and **tA** is SUFFIX. And after the normal parsing [as shown in Figure 2] we get the lexicon **tA(তা)** as

```
Form       tA(তা )
class      SUFFIX
fearture   [
              cat=SF
              prev_cat=ADJ
                to_cat= N
              ]
```

Which confirms that **tA** should be added with the category ADJ (ADJECTIVE) and after the addition new category will be N(NOUN). For the above word, as ***anAdUnIk*** is of category ADJ feature unification succeeds**. [11][9]**

The above feature unification also rules out any other combination like

anAdr + tA = anAdrtA   X

because **anAdr(অনাদর)** is NOUN and **tA(তা)** cannot be added with the NOUN.

THE REASONS BEHIND FEATURE BASED MORPHOLOGICAL PARSING

There are several reasons why we should use feature based morphological parsing for Bangla words: **[10]**

(1)The word grammar component can deduce the lexical category (part-of-speech) of a word:

It is not easy to determine the part-of-speech of derived words whose parts of speech is different from the parts of speech of the the root word. For example **adUnIk (অনাধুনিক)** is ADJECTIVE but **adUnIkAyn(adUnIk + Ayn)** is NOUN. Because the suffix **ayn (আয়ন),** when added with the ADJECTIVE root, changes it into NOUN. **[11]** So just dividing a word into its morpheme structure is not enough. We have to collect all the lexical features which are relevant to a certain word. This is only can be done in Feature Based Morphological Parsing.

As we showed in the example of **anAdUnIktAr** given above in Figure 5**,** we can see that after the parsing and feature unification we get a feature structure which specifies that the word **anAdUnIktAr** is of category (part of speech) NOUN.

(2)The word grammar component offers a more powerful model of morphotactics.

Morphotactics, as we said earlier, explains of morpheme ordering that explains which classes of morphemes can follow other classes of morphemes inside a word. Normally a Bangla word follows the following morpheme order:

Word = PREFIX* ROOT SUFFIX* ....(1)

In this notation an asterisk(*) indicates zero or more occurrence of an element. Thus a word consists of an obligatory root (or indivisible stem) preceded by zero or more prefixes and followed by zero or more suffixes. This accounts for all inflectional or derivational structure.

*This obviously is a rather coarse analysis of morphotactic structure, and as such greatly over recognizes***.** While it enforces the relative order of prefixes, roots, and suffixes, it does not enforce any order among prefixes or suffixes. For example,

adUnIktA = ROOT + SUFFIX = adUnIk + tA
adrtA = ROOT + SUFFIX = adr + tA
...............................(2)

both follows the same morpheme structure as in (1). But we know first one is correct but second one not.

We can solve this by creating more classes depending on the ROOT category (parts of speech) and more PEEFIX and SUFFIX subsets depending on the which one is added with which ROOT class. For example this can be a morpheme structure which can handle the above over recognition.

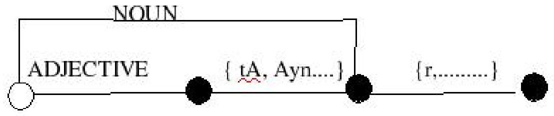


Figure 6: Changed morpheme structure.

So SUFFIX **tA** is added with only ROOT of category ADJECTIVE.

*But there are two problems associated with this kind of morphotactics based on classification***.**

Problem 1

This kind of classification based morphotactics makes the finite state (FSA, which is used to represent the morphotactics) look cumbersome. For example the finite state to handle the VERB morphology can be like Figure 7: **[4]**

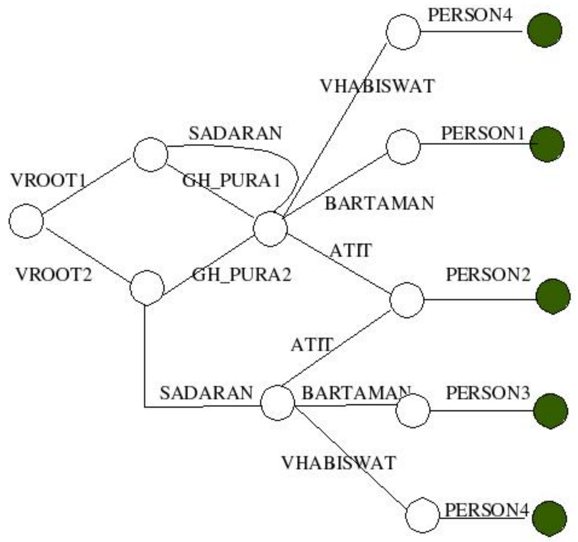


Figure 7. FSA for verb.

Problem 2

The second problem is that different suffix and prefixes wants ROOT to be classified on different category. For example In the example given in (2) we classify ROOT based on its parts of speech(NOUN or ADJECTIVE). Again the number SUFFIX (gUlO, rA,.....**)** wants the ROOT to be classified on SEMANTIC information like whether the root is PERSONAL ROOT (mAnUsh) or MATERIAL ROOT(bI). This makes morphotactics look even more complex.

*All these problems can be solved if we use feature based morphological analysis***.** For example the FSA for representing the VERB morphology in Fig 7 will be as simple as this:

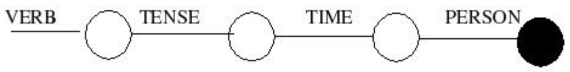


Figure 8: Simple FSA for verb.

For this just we have to define all the features and different features constraints which will specify which class of morpheme will add with which class. This feature constraints are actually handled during the parsing through feature unification as described previously. For example here are three lexicons:

lexicon : KA (খা)
category: VERB
feature : LAST_CHAR=V

lexicon : hAt (হাট)
category: VERB

feature : LAST_CHAR=C

lexicon : c~C (চ্ছ)
category: TENSE
feature : LAST_CHAR=V

lexicon : C (ছ)
category: TENSE
feature : LAST_CHAR=C

So, when we parse **KAc~CIlAm(খাচ্ছিলাম)** feature unification of **KA+ c~C** (VERB + TENSE) occurs successfully. Same is true for **hAtCIlAm**. But if we parse **KACIlAm(**খাছিলাম) feature unification of **KA+ C** (VERB + TENSE) does not occur.

It is true that as there are less classes more lexicons will remain in a certain class. So searching time should be high. For example for ROOT we have to search all the NOUN, ADJECTIVE etc. But if we use FSA to represent a class then searching time will be optimal irrespective of less class members or more class members. It is described in more detail in the next section.

(3)Feature Based Morphology Uses the FSA more optimally

We use FSA for representing a lexicon in morphology. We know in FSA we can search a string in the most optimal time (order(length of the string)) given enough space. *So space is a big issue in morphology not time if we use FSA. We can show that for Bangla if we use feature based morphological analysis we will require less space than we use classification based complex morphotactics.* **[3][5]**

As we said previously, without feature based parsing we have to classify the ROOT depending on parts of speech (NOUN, VERB,..........). In that case we will have a lexicon list like this:

NOUN: krA, klA, krAt, krtl................
VERB: kr, krA, jA, ...

So Noun FSA will look like this:

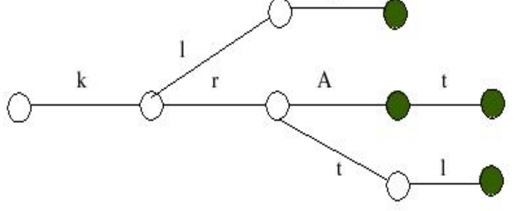


Figure 9: NOUN FSA.

And VERB FSA will look like this:

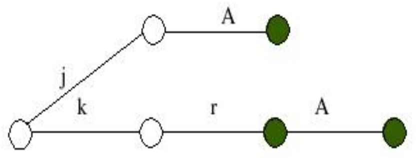


Figure 10: VERB FSA.

So if we use two classes NOUN and VERB, we need total (9+6) = 15 nodes and (8+5)=13 edges.

But if we use Feature Based Word Grammar process we will have only one class which contains all the NOUN, VERB etc. Then the FSA will look like this:

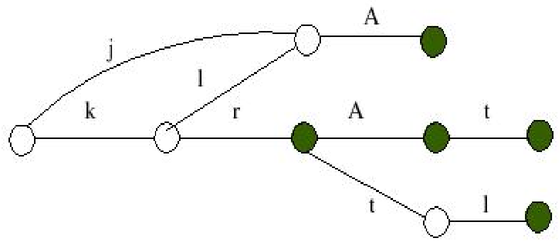


Figure 11: ROOT FSA.

So here we see that if we use only ROOT class we need total 9 nodes and 9 edges which is less than the previous need. The only extra information we have to store here is that in the final states we have to store whether the lexicon is NOUN or VERB.

(4) The word grammar component can provide a full feature specification for a word

Besides lexical category, a word grammar can also determine all features of a word that are relevant to syntactic parsing, such as tense, number, gender, and case. For example if we give a word like **krECI(করেছি)** to the parser it generates the following feature structure:

```
Word:
[
    cat= V
    time=PR tense=PER
    person=1st
]
```

which specifies that **krECI** is of root VERB, tense PERFECT (PURAGHATITA) time PRESENT (BARTAMAN) and person 1st (UTTAM).

CONCLUSION

So we can see that feature based morphological parsing is the best solution towards morphological parsing of Bangla word. Although the analysis we put up here on Bangla is quite elementary, future researchers can take it as a stepping stone towards building a complete morphological parser for Bangla.

ACKNOWLEDGEMENTS

This work has been partially supported through PAN Localization Project (www.PANL10n.net) grant from the International Development Research Center, Ottawa, Canada, administered through Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Pakistan. We would like to give them special thanks. We also acknowledge the constant advices and support of all the members of the BRAC University specially Ms. Nupur, Mr. Sheemam Monjel, Mr. Dewan Shariar Hossain Pavel, Mr. Asif Iqbal Sarkar and Mr. Kamrul Haider.

REFERENCES

[1]Antworth, Evan L. “PC-KIMMO: a two-level processor for morphological analysis.”, Occasional Publications in Academic Computing No. 16. Dallas, TX: Summer Institute of Linguistics (1990).

[2]Antworth, Evan L. “Morphological Parsing with Unifcation-based Word Grammar.”, A paper presented at North Texax Natural Language Processing Workshoup (May 23, 1994).

[3] Andrew Spencer and Arnold M. Zwicky, "The Handbook of Morphology", Blackwell Publishers, 2001.

[4] B.B. Chaudhuri, N. S. Dash, and P. K. Kundu, "Computer Parsing of Bangla Verbs", Linguistics Today, Vol. 1, No. 1, pp. 64-86 (1997).

[5]Daniel Jurafsky and James H. Martin, "Speech and Language Processing: An Introduction to Nataural Language Processing, Computational Linguistics, and Speech Recognition", Prentice Hall, (2000).

[6]G. Edward Barton, Jr."The Computational Complexity of Two-Level Morphology", November 1985, MIT, Artificial Intelligence Laboratory, Cambridge, Mass. "ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-856.pdf"

[7]Koskenniemi, Kimmo. “Two-level morphology: a general computational model for word-form recognition and production.”, Publication No. 11. Helsinki: University of Helsinki Department of General Linguistics (1983).

[8]Shieber, Stuart M. 1986. “An introduction to unification-based approaches to grammar.”, CSLI Lecture Notes No. 4. Stanford, CA: Center for the Study of Language and Information.

[9]Pabitra Sarkar, "Bangla Rupthatter Bhumica", 1997.

[10]P. Sengupta and B.B. Chaudhuri, "Morphological processing of Indian languages for lexical interaction with application to spelling error correction", Sadhana, Vol. 21, Part. 3, pp. 363-380 (1996).

[11]Suniti Kumar Chottapaday “Vasha Prokash Bangla Bakaran”.

[12]PCKIMMO is available at www.sil.org/pckimmo.

[13]Documentation of PCKIMMO is available at www.sil.org/pckimmo/v2/doc.