Rule based Word Morphology Generation Framework

Ramchandra P. Bhavsar[1], B. V. Pawar[1]

[1] Department of Computer Science, North Maharashtra University
Jalgaon, Maharashtra 425 001, India

Abstract

Lexical resources are very crucial for any NLP research as well as applications. The richness of a Lexical resource depends not only on the coverage of words but also on the coverage of their morphological forms. As keying in all word forms(morphology) for each word can be monotonous and mechanical exercise, hence automatic generation of the word morphology should be important agenda for Lexicon creation tools, in general and more specific for Indian languages, which are morphologically very rich as compared to English. In our ongoing effort of developing Hindi to Marathi Machine Translation (MT) system, we have developed a rule based framework for generating morphology for a given root (stem) word. The framework relies on rules for generating morphology. We have devised a generic format of rule, which caters for necessary grammatical information as required for generating word morphology for specified language. The framework can generate both inflectional as well as derivational morphology of a given word s.t. satisfaction of rule constraints. The rule management, rule selection and choice of selecting correct word forms are some of the important features of the framework.

Keywords:

Word morphology, inflectional morphology, derivational morphology, morphology generation framework, morphology rule.

1. Introduction

The advent of Internet has resurged interests in NLP due to which rich lexical resources are in high demand. Lexicon is one of the important resources for NLP research and applications. A lexicon generally contains possible grammatical information about a given word like its lexical category, gnp*(gender, number, person)* features and semantic features etc. Richness of lexicon depends not only on coverage of words, but also on the coverage of their morphological forms. Keying in all morphological forms of a given word in computer can become monotonic and mechanical very soon. The issue becomes severe for languages which are morphologically rich like languages of the Indian language family. Hence automatic generation of word forms based on inflectional and derivational morphology should be an important agenda for the lexicon creation tools. Also as the storage is getting cheaper than computing power these days, hence storing possible forms of a given root word is not a bad idea. It additionally helps to speed up the overall processing time, as the time required for morphological analysis can be saved. In the context of web based application, speed matters over storage because a web application is constrained by other operational parameters like response time etc. Optimized data design as discussed in [09] can be used to avoid redundancy in storage. Morphology is an important branch of Linguistics, which has attracted special attention of researchers after the evolution of Computational Linguistics and NLP applications on computers. Word morphology refers to obtaining different word forms of a root word due to change in grammatical attributes like gnp, case or change in lexical category by attaching affixes to the root word. There are two types/forms of morphology viz. inflectional and derivational morphology. In inflectional morphology, new word forms are obtained by inflecting the root words by changing their grammatical features like gender, number, person(gnp) and case. While in derivational morphology new word forms of different lexical category are derived from root word viz. adjectives from noun, nouns from verb etc. [3]. In nutshell inflectional or derivational morphology can be obtained by conjoining prefixes/suffixes to the root word (stem). In some cases few characters at the beginning or end of the word, are required to be trimmed before conjoining new prefixes/suffixes (mostly with inflectional morphology). Word ending symbol(s) plays crucial role (for Indian languages), for deciding trimming operation. For certain set of words, morphological generation is not possible through any rule based approach and they are out of scope of this paper. E.g. गाय, बैल etc. Description on obtaining the exact morphological forms for classes of words belonging to various lexical categories can be found in linguistic literature [5], [6], [7], [8]. From computational point paradigm based approach has been discussed in [1],[2] and [4]. After going through linguistic and computational aspects as discussed in [5], [6], [7], [8] and [1],[2], we came up with an idea of building a rule based generic framework for generating the inflectional and derivational morphology. This framework has been built as part of lexicon creation tool for creating Lexicon for the

proposed Hindi-Marathi MT system. The following sections of the paper discuss the objectives, approach and difficulties, implementation and testing of the framework.

2. Approach

Our approach is inspired from the paradigm based approach as discussed in [1], [2] and [4]. We have tried to extend the paradigm based approach in the sense that it uses rules coded using rule format as discussed in 3.1. Our approach follows the lexical representation as discussed in 2.1. In our approach the rule format is used to represent inflectional as well as derivational rules. Implementation idea has been inspired from ERP systems, which are highly configurable and have open system architecture, which allows easy integration of different functional modules. The rule format designed here is flexible enough to include rules for different languages. The objective behind this framework is to develop a rule driven framework in which morphology rules are stored in database table in the generic format such that rules of different languages should be accommodated seamlessly. After rules are created, appropriate rules are applied to root(stem) word depending on its' lexical category, word ending symbol, case and *gnp* feature. After going through the linguistic literature on the topic [5], [6], [7], [8], it is found that that a rule described in the literature may or may not be applicable to all words in that lexical category but is applicable to only subset of words with some common traits or subclass under that lexical category, catering for this difficulty only lead to further clue for designing the lexical organization. The following section describes the lexical organization adopted for this framework.

2.1 Lexical representation

Considering the above issues, three levels deep hierarchy for representing lexical categories (as depicted in **fig.1**) has been proposed and implemented. In this hierarchy lexical category is at the root level, while subcategory of the lexical category and subtype are at second and third level, respectively. E.g. Noun lexical category can be sub categorized as Common noun, Proper noun, Abstract noun etc. and each sub category can be further classified into subtypes under that sub category; e.g. Common Nouns for instance can be classified as person, location, botanical entity etc. To specify a lexical category of the word; along with its' subcategory and subtype, a notation of the form A.B.C has been devised, where A, B, and C are the lexical category, subcategory and subtype respectively. E.g. *Noun.CommonNoun.botanical* represents a common noun of subtype botanical. This way, subcategories and subtypes for each lexical category have been created. **Table 1**. shows the summary of the same as of today for Hindi as well as Marathi Lexicon. The design is adaptive so that new lexical category or subcategory or subtype can be defined with ease and becomes part of the master data of the framework. Hence the numbers given in following table should not be treated as final.

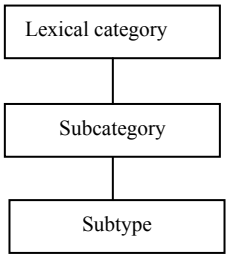

Fig. 1 Lexical Hierarchy

Table 1. Source Lexicon summary

Lexical Category	Subcategory	Subtype
Noun	03	21
Adjective	03	12
Verb	02	06
Pronoun	07	05
Adverb	04	--

3. Implementation

After deciding the approach and basic lexical representation, implementation strategy was worked out. The formal Rule specification, rule application algorithm and output specification are the important aspects of the implementation strategy. The framework has been implemented on Microsoft .NET platform 2.0 using VB.NET & MS SQL server 2005 RDBMS.

3.1 Rule specification

Design of rule format caters for different operational as well grammatical attributes required to generate word forms. Basically rule specification has two sides viz. source and target, source refers to input word for the rule, while target refers to new word formed after applying rule on source word. Rule includes lexical category of the source word, word ending symbol(s), *gnp* feature, case, tam(tense, aspect, mood) for verbs, semantic features, characters to be trimmed from source word, prefix/suffix to be conjoined, *gnp* feature of target word, case, semantic features, etc. the detailed rule specification is described in **Table 2.**

Use of wild card ('*') notation, reversible application of rule are the prominent features of the rule specification, which are useful in rule selection and application. With this representation of rule, we can generate the various forms of word by changing either of the values of *gnp*, case or TAM (in case of verbs). Mostly gender, number and case play important role for generating word

morphology for Indian Languages. The case field can also be used to specify *vibhakti markers* for languages like Marathi, in which *vibhakti* marker is conjoined with noun to form *Saamanyarupa*. Though some fields like semantic feature, Person etc. as such don't play major role in forming new word form, still they have been included because, they are required for full specification of the target word form and rule selection. The same rule format can be used to denote the derivational morphology, in which we can derive word form with different lexical category than source i.e. deriving adjectives from nouns and nouns from adjectives etc.

The framework allows rule management by facilitating addition, deletion and updation of rules. Rules can be extracted from literature and framed in to above format. The two count fields are helpful to decide strike rate (% of success) of given rule and useful in rule selection or ordering of rules. The example rule(**Fig. 2**) is used to used to create feminine singular direct case form for, 'आ' ending Hindi common nouns used to denote 'persons' (व्यक्तीवाचक).

Illustration For Hindi word लडका(boy) which is masculine, singular, direct case, if we apply example rule discussed in **Fig. 2**, we get लडकी which is feminine, singular, direct case with same semantic feature as source word counterpart.

Same way, we can also frame Derivational rules from the linguistic literature [5], [6], [7], [8] for deriving various lexical categories from other categories.

Table 2. Rule specification

Rule Fields	***Description***
Language	The Natural Language, which this rule is representing
Morphology Type	Whether rule represents Inflectional or Derivational morphology rule? **I**-Inflectional, **D**-Derivational
Affix type	Type of affix i.e. prefix or suffix
Source Word category	Lexical category of input word Eg. Noun, Adjective, etc.
Target Word category	Lexical category of new word Eg. Noun, Adjective, etc.
Source lexical category symbol	Category symbol of input word specified in A.B.C notation. Wild card notation,'*' can be used to represent all values for either subcategory or subtype. Eg. Noun.AbstractNoun.feeling, Noun.CommonNoun.*, Noun.*.* etc.
Target lexical category symbol	Category symbol of new word specified in A.B.C notation. Wild char notation * can be used as described above.
Source word ending symbol(s)	Word ending symbol(s) for source word
Target word ending symbol(s)	Word ending symbol(s) for target word.
Source Semantic Features	The semantic features of the input word to be matched while extracting rules.
Target Semantic Features	The proposed semantic features for the target word.
Source word prefix/suffix	Prefix/suffix to be attached to input word after trimming.
Target word prefix/suffix	Prefix/suffix to be attached to target word after trimming. This is used to obtain the input word and used if given rule is to be applied in reverse order i.e. source and target sides of rule are swapped.
Source GNP, Case	GNP, Case attribute of the input word; required to match with input word, if one of the parameter is to be ignored while matching then wild char expression * can be used.
Target GNP, Case	GNP, Case of the generated new word, the same are used for matching while deriving transitive rules from the current rule and if reverse operation is to be applied. If one of the parameter is to be ignored while matching then wild card expression * can be used.
Source TAM	Used only for verb stems.TAM features to be matched with verb stems, if one of the parameter is to be ignored while matching then wild card expression * can be used.
Target TAM	TAM features of the generated verb form, same are used for matching while deriving transitive rules from the current rule and if reverse operation is to be applied on current rule. If one of the parameter is to be ignored while matching then wild card expression * can be used.

Source Trimming characters	No. of characters to be trimmed from the stem word, before applying the rule(used only during forward i.e. source to target; application of rule). The characters are trimmed either from front side or rear side of source word depending upon the *Affix Type* field of rule i.e. prefix or suffix. -ve → removes no. of character(s) from input word, 0 → do nothing, +ve → add no. of character(s) to input word
Target Trimming characters	No. of characters to be trimmed from the target word(applicable, only if rule is applied in reverse) -ve → removes no. of character(s) from the target word, 0 → do nothing, +ve → add no. of character(s) to the target word. The characters are trimmed either from front side or rear side of target word depending upon the *Affix Type* field of rule i.e. prefix or suffix
Reverse Operation	Whether this rule can be applied in reverse manner i.e. target side of rule is treated as source and source as target.
Success count	Count representing successful application(s) of rule.
Failure count	How many times rule could not generate desired output on application

Affix	**Morph Type**	**Lex. category**	**Category Notation**	**Semantic Feature**	**Aff**	**TC**	**GNPC**	**Rev**
Suffix	Inflec – tionl	sNoun	Nn.CmnNn.p	Animate Human	◌ா	-1	MS*D	T
		tNoun	Nn.CmnNn.p	Animate Human	◌ீ	-1	FS*D	

Aff-Affix, **TC**- trimming characters, **GNPC**- Gender, Number, Person, Case, **Rev**- Reverse operation, **sNoun**- Source side noun, **tNoun**-Target side noun **MS*D**- Masculine, Singular, any person, Direct case, **T**-true

Fig.2 Sample Rule

3.2 Algorithm

The algorithm devised for generating new word form has primarily four parts: *rule selection*, *rule application order*, *rule application* and *rule output*. The later is described in Section **3.3**, The formal specification of the algorithm is as follows:

Input→ Stem word, lexical category in A.B.C. notation, GNP Case features,semantic Features (animate-animal,animate-human ,abstract, honorific, locative, Instrumental), Rule data base.

Output→ Possible word forms based on qualified rules.

1. *Rule extraction*
   a. Get all rules matching lexical category, GNP,Case, semantic feature, word ending symbol(s) of input word, mark them as 'original', and order the rules by, exact match, wild card match and valid count.
   b. Get additional rules by transitivity based upon above fetched rules i.e. find new set of rules where target side features are on source side in the rule, by matching above(original) rule's lexical category, GNP, Case TAM, recursively for each rule above. Mark them as 'transitive' rule, and order the rules by, exact match, wild card match and valid count.
   c. Rule sets obtained in a. and b. are merged retaining their order.
2. *Rule Application*

   For each rule in rule set

   i) Form morphed word from the rule specification by trimming the required characters from input word and conjoining the affix to input word.
   ii) Initialize other features of morphed word as specified in rule. Add morphed word to output if not already added by other rule.
3. From the output list, save correct word forms with their full specification. Update the

```
        success and failure count for the
        rules depending upon whether the
        word form was saved or not by
        user. Facility for editing is
        also provided (if required), in
        which case success count is not
        updated.

Stop after all rules are applied.
```

3.3 Output specification

The framework generates number of word forms for the matching rules extracted from database, all attributes related to lexical category, category symbol, *gnp* feature, case, tam(in case of verb), semantic features for newly formed word are specified in the output. In spite of stringent rule specification and application procedures, a rule may not fit for a given input stem word even though it qualifies hence for such exceptions; generated word form might not be correct. Thus all output words are shown to user in a grid for his perusal, after which he/she can choose the correct word forms for saving to lexicon for further use. The success or failure count for a rule is updated depending on selection of word form generated by that rule.

4. Testing

So far we have tested the framework for both Hindi & Marathi languages for inflectional as well as derivational morphology. Extensive Rules for inflectional morphology have been created for lexical categories like noun, verb and adjectives for Hindi as compared to Marathi, where we are in initial stage of creating rules. Derivational rules for deriving adjective from noun and vice-versa, nouns from verbs also have been created. Both types of rules have been successfully applied to sample words. **Table 3** shows the summary of rules in the database.

Table 3. Rule Summary

Language	*Morphology Type*	*Lexical Category*	*Total Rules*
Hindi	*Inflectional*	*Noun* *Adjective* *Verb*	*51* *14* *28*
	Derivational	*Noun→ Noun* *Noun→ Adjective* *Adjective→ Noun* *Verb→Noun*	*09* *19* *05* *03*
Marathi	*Inflectional*	*Noun* *Adjective*	*76* *10*
	Derivational	*Noun→ Noun* *Noun→ Adjective* *Adjective→ Noun* *Verb→ Noun*	*15* *05* *15* *04*

5. Features and Applications

The framework has following features/applications:

i) It can be integrated with lexicon creation tool to generate inflectional as well as derivational word morphology. Since the framework automates the process of word generation and its grammatical specification, the time and efforts required, to key-in possible word forms for given root word , are saved.
ii) Rule creation process is user friendly and use of wild card notation ('*') makes the rules flexible and powerful.
iii) Like in [2], data is stored in Unicode format, hence font related dependencies are completely avoided.
iv) Can be used for creating dictionaries/language tutoring tools.
v) Rules can also be used for morphological analysis.
vi) Framework generates statistical data about usage of rules, which can be useful to understand language phenomenon by the linguists.

6. Conclusions

The framework has been successfully tested to generate prefix and suffix based inflectional and derivational morphology of declinable words of Hindi and Marathi Language. The framework uses Unicode to store data, which makes it easy to use the data on different platforms. Selection of Morphology rules through transitivity helps to generate multiple word forms in one go. The framework also provides the facilities for rule management. The rule creation process is simple and user friendly. Use of this framework can ease in lexicon creation process. Lastly the rules for English and other Indian languages can also be framed and tested using this framework as part of future work.

Acknowledgments

We are thankful to Dr. Hemant Darbari, Executive Director, C-DAC, Pune for his moral support, valuable inputs for this work.

References

[1] Akshar Bharati, Vineet Chaitanya and Rajeev Sangal. 1995. Natural Language Processing: A Paninian Perspective, Prentice-Hall of India, New Delhi. ISBN: 81-203-0921-9

[2] V. Goyal and G. S. Lehal, " Hindi Morphological Analyzer and Generator, ICETET Proceedings of the 2008 IEEE International Conference on Emerging Trends in Engineering and Technology. IEEE Computer Society Press, Washington, DC, USA (2008), pp1156-1159.

[3] Hausser, Roland, Foundations of Computational Linguistics, Human-Computer Communication in Natural Language. 2nd Edition, part-III, Chap.13, Sprinter, Berlin, New York , 2001, ISBN: 978-3-540-42417-8.

[4] P.H. Mattews, Foundations of language, Vol. 1, No. 4, pp. 268-289, Nov., 1965, Springer.

[5] Omkar Kaul, Modern Hindi Grammar, Dunwoody Press, Springfield, USA, 2008 ISBN978-1-931546-06-5.

[6] Rajashree Pandharipande, Marathi, Rutledge New York, 1997, ISBN 0-415-00319-9.

[7] Pandit Kamata Prasad Guru(1920), Hindi Grammar (in Hindi), NagariPracharani Sabha, 11th edition, Varanasi(UP), India.

[8] M. K. Damale[1911], Scientific Marathi Grammar (in Marathi), Damodar Sawalaram and company, Pune(MS), India (1965 edition).

[9] R. P. Bhavsar , B. V. Pawar, "Lexical Organization: A concern for NLP", in the proceedings of NCAC-2011 at North Maharahstra University, Jalgaon(MS) 24-25 Feb 2011, ISBN:978-81-910591-2-0, pp.289-292

Ramchandra P. Bhavsar did his Master of Computer Application(MCA) in 1995 from the Department of Computer Science, North Maharashtra University, Jalgaon(MS), India. He started his career with the Department of Computer Science as Lecturer then he joined the Applied Artificial Intelligence(AAI) group at the Center for Development of Advance Computing(C-DAC), Pune, worked with AAI, C-DAC, Pune as MTS(Member Technical Staff) from 1997-2000, where he worked on English to Hindi MT system named MANTRA (Machine Assisted Translation). After that he joined the Computer Centre at North Maharashtra University, Jalgaon as System Analyst and became Reader in Computer Science in 2007 in the Department of Computer Science. The MANTRA project was nominated for Computer World Smithsonian Award in 1999.He is member of Linguistic Society of India. His research interests; include Machine Translation, NL parsing, Morphology and IT systems, Internet Computing.

B. V. Pawar did his B.E. in Production Engineering from V.J.T.I. Mumbai in 1986, his M.Sc.in Computer Science from University of Mumbai in 1988 and his Ph.D. in Computer Science from North Maharashtra University, Jalgaon in 2000.Currently he is working and Professor and Head in the Department of Computer Science, North Maharashtra University, Jalgaon. He has supervised 05 Ph. D. theses. His research Interests include Natural Language Processing, Web Technologies, Information Retrieval, Web Mining etc.