Oriental Journal of Computer Science & Technology Vol. **3**(1), 89-93 (2010)

Subanta pada analyzer for Sanskrit

SMITA SELOT[1], A.S. ZADGAONKAR[2] and NEETA TRIPATHI[3]

[1]Department of CA, SSCET, Bhilai (India). [2]CV Raman University, Bhilai (India).
[3]GD Rungta College, Bhilai (India).

(Received: April 30, 2010; Accepted: May 15, 2010)

ABSTRACT

Natural language processing has wide coverage in application areas like machine translation , text to speech conversion , semantic analysis , semantic role labeling and knowledge representation .Morphological and syntactic processing are components of NLP which process each word to produce the syntactic structure of the sentence, with respect to its grammar .Semantic analysis follows syntactic analysis. One of the key task in morphological analysis is identifying the correct root word from its inflected form.In Sanskrit language, these inflected word follow the rules which are used to separate the root word from its suffix.These extracted suffix carry sufficient amount of syntactic and semantic information with them .To develop such word splitter, rules called sandhi rules given in the grammar of the Sanskrit language has been used.The challenge in the problem lies is in identifying the junction point or breaking point of the word as multiple junction can be obtained within the word .Developed system maintains database of all possible suffix , which are then used for splitting the word . The algorithm for the same is presented in the paper with the solutions to problem faced while developing the module.

Key words: Vibhakti ,Semantic analysis, vibhakti-karka mapping and splitter.

INTRODUCTION

Sanskrit ,an ancient language in India, is known as the language with strong computational grammar associated with it hence an efficient language for computer processing[1,2]. People with good background of the language , grammar and programming concept will be able to explore its computational aspect, which is a rare combination[2]. Though the work is being carried out at IIIT Hyderabad, JNU, Dehli, but still very few people come forward due to complexity associated with it.

In Indian languages, sandhi means joining of two words to obtain a new word and sandhi viched means splitting of words in two parts. Sandhi based splitters are significant in the areas of natural language processing where syntactic and semantic information are key issues. Sanskrit is a heavily inflected language, and depends on nominal and verbal inflections for communication of meaning. A fully inflected unit is called pada ,inflected nouns are called - subanta padas and inflected verbs are called tridanta pad. Hence identifying and analyzing these inflections are critical to any further processing of Sanskrit.Such process has been described as complex segmentation problem requiring lexicon database of root words for morphological generator[2]. Here we minimize the use of dictionary of root words as suffix and sandhi rules are used to solve the problem.

While developing the system for knowledge representation in Sanskrit , a partial parser was developed to perform semantic analysis using case based Panini Grammar[11,12]. In this parser, need for sandhi analysis was felt as most of words in a sentence has embedded suffix within them, which assist in semantic analysis. After analysing the sandhi process, the focus was on identifying the correct position within the word where splitting will take place , so that , root word and suffix are generated . This paper presents an algorithm for splitting the word within the sentence

giving possible answer, problem encountered and its solution .It is divided into following section :- section one gives the design of the database maintained for storing all the suffixes of verb and noun and pronoun., second section describes the algorithm for processing the words of a sentence and splitting it by mapping the respective suffix from database. Third section focuses on the problem encountered while developing the system and fifth is the analysis of the same.

Database Design of the System

For extraction of semantic of a sentence suffix analysis is required which can be done in following ways - one is using transition network[11] and the other is using the database of suffix of all categories of word - noun pronoun and verb[12,14]. Following databases are maintained for the system:

Database for verb suffix
Database for noun suffix
Database for pronoun

A Number scheme for suffixes

Five digit number scheme is used for the storing the verb suffix in the database where each digit signifies the syntactical parameter[12]]. In Sanskrit grammar, verbs are classified into ten groups called gan, It occupies the most significant digit in the number scheme as shown in fig1.When a root word joins with the suffix , some changes takes place at the junction .With respect to these changes verbs are classified into different gan. It will take values in the range 0-9. Second position from left is occupied by pad and there are three pad-Atmnepad , parsmaipad and ubhaypad.Those verbs whose outcome is for another person, they fall under Parasmaipad and those verb whose outcome is for oneself come under Atmnepad. Words whose outcome is for both, otherperson and oneself , they come under ubhaypad.Next position is for tenses, as there are ten tenses, range of values for these are from 0-9.With three person and three number , value for them is in range 1-3.

Category	Gan	pad	Tense & mood	Person	Number
Range	1-9	1-3	0-9	1-3	1-3

Fig. 1: Five digit number schemefor verb suffix

If the word bhavati is mapped in database , the word will have the suffix *ti* which will give us the value 11011 meaning *bhavAdigan , Atmanepad, latlakAr, pratham puruSh and ekvachan.* Similar structure for noun is given in fig 2.

Category	x ending	Gender	Vibhakti	Number
Range	1-8	1-3	1-3	1-3

Fig. 2: Five digit number schemefor noun suffix

Classification of noun is done with respect to the ending of the word with particular vowel a *A i I* . Eight such endings are identified and number scheme is designed for the same. As number of pronoun are limited ,complete word along with their respective gender, vibhakti and number are stored. Four digit representation scheme is designed for pronoun as given in fig 3.

Category	word	Gender	Vibhakti	Number
Range	1-8	1-3	1-3	1-3

Fig. 3: Four digit number scheme for pronoun

Eight pronouns and their declensions are stored in the database with the structure given in the fig-3.The overall objective is to use these suffixes in the extraction of semantic analysis of the sentence. When the suffixes are mapped in these databases, the order of mapping followed is pronoun, noun and then verb. Some problems are encountered while performing the mapping of suffixes with databases. Multiple suffixes of varying length, multiple suffixes of same length are mapped to single word .Solution of these problems are given along with the algorithm in the following section.

Algorithm For Sandhi Analysis

After storage of all suffixes in the databases, word in declined form is taken as input and processed. Each word(w) is made up of root(r) word and suffix(s).

Word(w)=root_word(r) + suffix(s)

Suffixes have different forms depending

upon the type of ending words,gender etc for noun. For example all a ending words have suffixes which are shown in the table 1.With each suffix, its number as per representation scheme is associated - fig1.This number will be refered as key in subsquent presentations.

Table 1: suffix for a ending words with

H (1111)	au (1112)	AH (1113)
Am (1121)	au (1122)	An (1123)
en (1131)	AbhyAm (1132)	ebhy H (1133)
At (1141)	AbhyAm (1142)	ebhy H (1143)
asya (1151)	yoH (1152)	ANAm (1153)
e (1161)	yoH (1162)	eShu (1163)

All the words within a given sentence is processed and splitted to find the root word and their corresponding suffix . If word is rAmbhyAm , it is mapped with all the suffix in noun database , mapped suffixes are stored in nmap_data list and their respective key values are stored in key list . If nmap_data contains more than one element , then the suffix with maximum length is considered as final suffix and its key as final key value.From the key most significant digit gives the type of ending which is stored in k1.Next digit gives the gender, which is stored in k2 .If the value of k1 is 1 then after removing the mapped suffix, a is added to the word ,forming the root word.Similary, depending upon the value of k1 either a,A,i,I etc is added at the junction and root word is created .However if k1 is 1 but k2 is 2 , it means the word is in feminine gender , hence A is added to make root word. Table 2 summarizes the character to be added based on the values of k1 and k2. Again there may be more then one values in nmap_data as in word *rAmAbhyAm, AbhyAm* is longest mapped suffix , but there are multiple occurrence of AbhyAm in the database.Then the key values k1 and k2 are analysed to find possible set of solutions.

Table 2: characters to be added to make root word

Value of k1	Value of k2	Character to be added
1	1	a
1	2	A
1	3	a
2	1	i
2	2	I
3	1	U
3	2	U
4	1	R
5	1	T
6	1	N
7	1	N
8	2	I

For example if the word is rAmH then
Nmap_data={'H'} and
Key={1111} where k1=1 and k2 = 1 ,therefore it is identified as a ending word. If we spilt the word with respect to suffix H we get root word as rAm which becomes rAma after adding a and suffix obtained as H .
This is simple example , let us examine some more cases
When word= *rAmAbhyAm* then
Nmap_data={ *'AbhyAm' 'AbhyAm' 'AbhyAm' 'yAm' 'yAm' 'AbhyAm' 'AbhyAm' 'AbhyAm' 'AbhyAm'*}
Key={1232 1242 1252 2271 8271 1332 1342 1352 1132}
This is the case of multiple matched suffix where nine suffixes are mapped with the given word.Out of all these, suffixes with maximum length are selected .So now the set is reduced to
Nmap_data=={ *'AbhyAm' 'AbhyAm' 'AbhyAm' 'AbhyAm' 'AbhyAm' 'AbhyAm' 'AbhyAm'*}
Key={1232 1242 1252 1332 1342 1352 1132}
Logic selects the the longest suffix and if k1 and k2 are analysed ,it is observed that all the values are from a ending word as k1=1 and gender can be masculine,feminine or neuter as k2 takes values 1,2 and 3 .Hence possible solution for the word rAmAbhyAm is as follows:

rAmAbhyAm=rAma + AbhyAm
rAmAbhyAm=rAmA + AbhyAm

Here two root words are generated and to find the correct root word , we need to look up in the dictionary.Root word found in the dictionary is the correct root word.Algorithm for splitting the word , basically subanta pad is as follows:

Step1: Input word w.
Step2: Match w in noun databse NDB to find suffix sfx
Step3: If match found
(i) Store the suffix to set nmap_data
(ii) Store their respective number in set key
(iii) Filter nmap_data and key so that it contains only longest suffix and its number respectively
(iv) Extract the first and second most significant digit from the number and store it in k1 and k2 respectively.
Step3: Depending on the value of k1 and k2 split the word as root word and suffix.
Add the character as per table 2 at the end of the root word
Step4:Display the possible results.
Step5:If more than one solution is obtain , map the root word in the dictionary.Word whose root word is found in the dictionary are the valid root words and final answer.

This algorithm was tested on the words taken from anuvAd chandrikA and rachnAnuvAdkamaudi .All types of paradigm with a e i I ending words were tested using the algorithm.Algorithm works for subanta pad only as rules for tridant pad is different, another module is needed for the same.

Performance of the System

Algorithm is developed in MATLAB 7.0 and the dictionary is not used in the implemented algorithm ,hence last step (step 5) is ignored .After testing nealry 1000 words ,following results were obtained:

Table 3: results of sandhi madule

S.No	Type of words	Accuracy
1	a ending-masculine	99%
	a ending-feminine	62%
	a ending-neuter	99%
2	i ending-masculine	79%
	i ending-feminine	99%
3	u ending-masculine	92%
	u ending - feminine	98%
4	R ending -masculine	87%
5	t ending-masculine	95%
6	inat masculine	54%
7	annat imasculine	95%
8	I ending-feminine	70%

The low result of inat ending words is due to the fact that endings are quite similar to the i ending words hence as per algorithm most of the inat ending declined word is broken as i ending word .However if the dictionary is used to check the correct root word , the wrong roots will be omitted or cancelled and performance of the system will further increase. This paper presents the algorithm for splitting the word basically noun, based on the suffix stored.For single occurrence of the suffix , output is accurate , but for multiple occureence , more than one solutions are obtained . This algorithm suggest the splitting of the word as an individual entity , hence at times, multiple answers are obtained .When same word appears in a sentence , splitters can be modified to obtain the correct answer as relation of the current word with other words therby helping in selecting the correct answer.

CONCLUSIONS

These result are useful in the application were semantic analysis is the key issue .While processing a senetnec in a NLP based system , each word is analysed for syntactic and semantic analysis. As most the synacto-semantic relations in Sanskrit are given by word endings(declensions) , this module will be of great significance in the applications like machine translation , semantic role labelling etc.

REFERENCES

1. Akshar Bharti , Vineet Chaitanya and Rajeev Singhal "Natural Language Processing : A panian persentive"IIT, Knapur, PHI 1995-(Book)
2. G'erard Huet "Towards Computational Processing of Sanskrit " INRIA.
3. Dr Kapildev Dwivedi "RachanAnuvAdakumaudi"vishavavidyAlaya prakAshan, Varanasi (Book)
4. Nills J Nilson "Principles of Artificial Intelligence" (Book).
5. Akshar Bharati,Amba Kulkarni,Department of Sanskrit StudiesUniversity of Hyderabad "Sanskrit and Computational Linguistics" at the First International Sanskrit Computational Symposium, (2007).
6. Subhash C Kak "The panian approach to natural language processing " *International journal of approximate reasoning.* **1**(1).
7. R. Briggs "Knowledge representation in Sanskrit and Artificial Intelligence " AAI , AI magzine.
9. Ms Smita Selot, Dr Jyoti Singh "Infromation retrieval in an Intelligent system using panani grammar" at Technovision -2207 -a National Conference at SSCET, Bhilai.
10. Ms Smita Selot Dr Jyoti Singh "Knowledge representation and Information Retrieval in Panini Grammar Framework " International Conference ICSCIS-07 2007 at Jabalpur, MP.
11. Smita Selot,Neeta tripathi AS zadgaonkar " Transition network for processing of Sanskrit text for identification of case endings" *icfai journal of computer science*, **3**(4): (2009).
12. Smita Selot , Neeta Tripathi , AS zadgaonkar "Case frame generation for Sanskrit language -A knowledge representation tool" *International journal of applied engineering research*.**4**(8): (2009).