Development of a Rule Based Learning System for Splitting Compound Words in Malayalam Language

Latha R. Nair
Division of Computer science Cochin University of Science and Technology

S. David Peter
Division of Computer science Cochin University of Science and Technology

Abstract—**Morphological analyzers are essential for any type of natural language processing works. As Malayalam like other Dravidian languages is an agglutinative language it needs a compound word splitter as a preprocessor. An algorithm has been developed and successfully used for splitting the compound words. 90% success has been established in initial scrutiny of around 4000 compound words. The splitter can be used for developing and implementing a full fledged morphological analyzer.**

Keywords-Dravidian languages, Malayalam, morphology, natural language processing, sandhi splitter, parts-of-speech tagger, finite automata.

I. INTRODUCTION

Morphological variations for words occur in Malayalam due to inflections, derivations and word compounding. Malayalam is an agglutinative language where words of different syntactic categories are combined to form a single word. Formation of new words by combining a noun and a noun, noun and adjective, verb and noun, adverb and verb, adjective and noun and in some cases all the words of an entire sentence to reflect the semantics of the sentence are very common[1]. The complexity of compounding in Malayalam language can be understood from the following example.

സീതയാരാനെയക്കണ്ടു gets split into
സീത ഒരു ആന എ കണ്ടു

A word by word translation of the above sentence becomes:
"Seetha an elephant saw" and the correct translation is
Seetha saw an elephant.

Such sentences are common in Malayalam language. This kind of compounding is found in plenty in media passages and in poems. The constituent morphemes are to be separated for the source language for all the Natural Language Processing (NLP) operations like Part of Speech (POS) tagging [2,3] machine translation and question answering. The splitter developed splits a compound word into morphemes. As there is a possibility of splitting a compound word in different ways it generates all possible splits for a compound word. The splitter uses a lexicon trie which reduces the look up time for the morphemes. It uses rote learning to cut down search space for the algorithm by storing intermediate results. The algorithm uses a depth first strategy with backtracking to find the splits. The splitter splits almost all kinds of word compounding in Malayalam.

II. LANGUAGE CHARACTERISTICS

In Malayalam Keralapanini [4] has adopted a classification for word compounding based on the changes occurring to the phonemes in the boundary of the two words: Lopa sandhi (elision), Aagama sandhi (addition), aadesha sandhi (substitution) and dwitwa sandhi (gemination). In addition since Malayalam has adopted a lot of words from Sanskrit, the Sanskrit sandhis like guna sandhi, vridhi sandhi, ayaadi sandhi and yan sandhi are commonly found in Malayalam [4, 5]. When two vowels come in the boundary either lopa sandhi or aagama sandhi occurs. Dvitwa sandhi occurs when a vowel and a consonant come in the junction. Aadesha sandhi usually occurs when two consonants or one consonant and a vowel comes adjacent to each other before being compounded.

III. RELATED WORK

The morphological analyzer reported previously is seen to be working only for compound word free sentences [2, 3]. A Moore model has been proposed for compound word splitting in Sanskrit as part of morphological parsing [6]. The need for sandhi splitting for machine translation is discussed in the paper. A Sanskrit sandhi splitter (vowel sandhi) tool has been reported as developed by Computational linguistics R&D, special centre for Sanskrit studies, Jawaharlal Nehru University, New Delhi. It handles only vowel compounding in Sanskrit. Corpus driven method for compound splitting using a parallel corpus has been reported where the correct split determination and its impact on statistical machine translation is discussed using both a German corpus and a parallel corpus of German to English translations [7]. Compound splitting and recombination for reducing the vocabulary has also been reported earlier [7].

978-1-4244-9477-4/11/$26.00 ©2011 IEEE

IV. COMPOUND WORD SPLITTER

The schematic diagram of the compound word splitter is given in Figure 1.

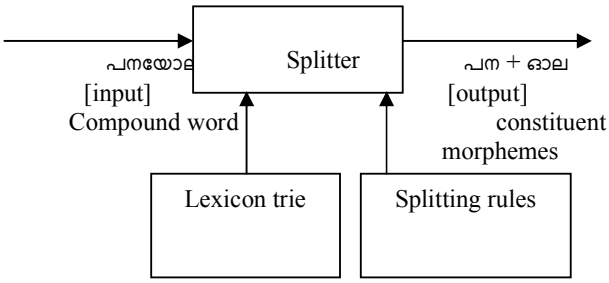

Fig 1. Block diagram of sandhi splitter

The splitter makes use of a lexicon consisting of morphemes belonging to each lexical category. The lexicon was created as a trie. A sample lexicon is as shown in figure 2. The circle shown with an arrow is the start state. The double circles are the accepting states. The transition symbols are the alphabets of Malayalam language. Separate tries are kept for each lexical category like noun, verb, adjective, adverb, noun affixes, and verb suffixes. The tags reduce the number of unwanted splits and also reduce the ambiguity when there exists multiple splits for the same compound word.

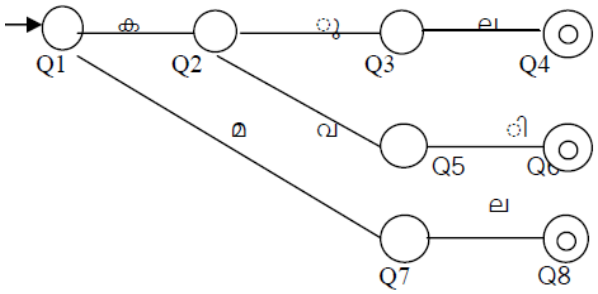

Fig. 2. Lexicon trie

V. ALGORITHM

Input to the algorithm for splitting compound word is a valid sentence and output is a sequence of morphemes for the sentence. As two level finite automata is not enough for highly agglutinative language like Malayalam a recursive depth first search with backtracking algorithm has been utilized [8,9]. The algorithm uses the sandhi tables which contain the replacement rules at each splitting boundary, lexicon trie with the words belonging to various lexical categories and a temporary array for storing partial results.

The Algorithm is detailed as follows:

Split (input: string)

Step1: If input is a valid word
Output=input
Return output.

Step2: While not end of input do steps 3-8

Step3: While there are unselected rules at the current syllable boundary do steps 3-7

Step4: Select an unselected rule from the rule table which satisfies the preconditions at the syllable boundary.

Step5: Apply the rule and split the input into two parts first and second

Step6: If the first is a valid word and If second word is present in the partial array attach the split result with first to form output goto step 8 else goto step 7

Step7: Temp=split (second) // recursive call to the same function with the second part //
If there are valid splits in temp form partial results in output array by combining first and each entry of the temp array.

Step 8: Advance to next syllable.

Step 9: return output.

Step10: Stop.

This algorithm works in a self recursive or depth first manner. First it checks whether the string can be accepted as such without splitting. In that case it is returned as such. Otherwise it tries to find all the splits for the string within the while loop in step2. The while loops in step 3 tries to apply multiple splits at the current syllable position. The string is split into two parts first and second following the standard sandhi splitting rules of the language (Table 1). If the first part is a valid word then the algorithm proceeds recursively with the second as the input string. Temp array is used to store the splits for the recursive call to second which is then combined with the first of the current call and stored in the array output. When the procedure ends the output array will contain all the possible splits. Rote learning is the process by which we store the computed values to reduce search time [9].

TABLE 1. RULES FOR WORD SPLITTING

Sl.no	Condition 1 On the current syllable	Condition2	String to be added to end of first word	Beginning of the second word	Example
1	All (C+SS)		C + ്	Vowel(SS)	കാടായി = കാട് + ആയി
2	ല്ല + SS	W1=അല്ല/ ഇല്ല	ല്ല	Vowel (SS)	അല്ലെന്ന് = അല്ല + എന്ന്
3	All	WC1=Verb / W1=ഒരു	C+(ു/ും)	Vowel(SS)	ഒരാന= ഒരു + ആന
4	യ + SS	W1= ആയി / പോയി	യ+ി	Vowel(SS)	ആയില്ല = ആയി + ഇല്ല
5	C+ (ു/ൂ)	C2= വ	S1	Vowel(SS2)	മധുവിനെ = മധു + ഇനെ
6	C+vowel sounds other than (ു/ൂ)	C2=യ	S1	Vowel(SS2)	സീതയുടെ = സീത + ഉടെ
7	അ,ഇ,എ	C2= വ	S1	Vowel(SS2)	അവർ = അ + അർ
8	C+ ാ		C	-	കലാവാസന = കല + വാസന
9	C+SS	C2= മ	C + ം	Vowel(SS2)	പണമില്ല = പണം + ഇല്ല
10	C+SS	C2=ത്ത	C + ം	Vowel(SS2)	മരത്തിൽ = മരം + ഇൽ
11	C	Gem(TC2)		Sin(C2)	പടിപ്പുര = പടി + പുര
12	C	Gem(C2)	C + ം	Sin(C2)	പണപ്പെട്ടി = പണം + പെട്ടി
13	ങ,ഞ,ണ്ട,ന്ത,മ്പ, ങ്ങ		ം	ക,ച,ട,ത, പ,ക	വരങ്കാലം= വരും +കാലം
14	റ് / ട്+ SS		റ്, ട്	Vowel(ss)	നാട്ടിൽ = നാട് +ഇൽ
15	C+ ാ		C	C2=അ/ആ	
16	ഗ,ദ,ധ,ന,മ്മയ,ര,ല ,വ,ഹ + ോ		C + സ്	S2	മനോദുഖം= മനസ് +ദുഖം
17	നീ	W1=നീ	നി:	S2	നീരസം = നി:+ രസം
18	ദു/ നി:	W1=ദു/ നി: C2=ര		Vowel(ss2)	ദുരർഥം=ദു: +അർഥം
19	C+ോ		C+ സ്	S2	മനോദുഖം= മനസ് +ദുഖം
20	അ,ഇ,എ	Gem(C2)	S1	Sin(C2)	അക്കാലം = അ + കാലം

SS (vowel symbol) – ാ, ി, ീ, ു, ൂ, ൃ, െ, േ, ൈ, ോ, ൗ, ം

C- Consonant C1- Ending consonant of the first word

C2- Starting consonant of the second word.

Sin(C) – singular form of the geminated character. e.g.: sin (ക്ക) – കS1- the ending syllable of the first word

S2-the starting syllable of the second word, TC-Tense characters (ക, ച, ട, ത, പ)

W1-Fisrt word , Gem(C2) – a geminated character

The split tree for the compound word കരതലാമലകം is as shown in figure 3. The number of recursive calls is reduced by storing intermediate results. After finding തല + ആമലകം as one split for the word ,when the system backtracks to find other splits for തലാമലകം, it is split into തലം and ആമലകം and since a split for ആമലകം was already generated the search stops there and the splits for ആമലകം is attached to it. So the sub tree with the root ആമലകം won't be traversed again thus improving the speed of the system.

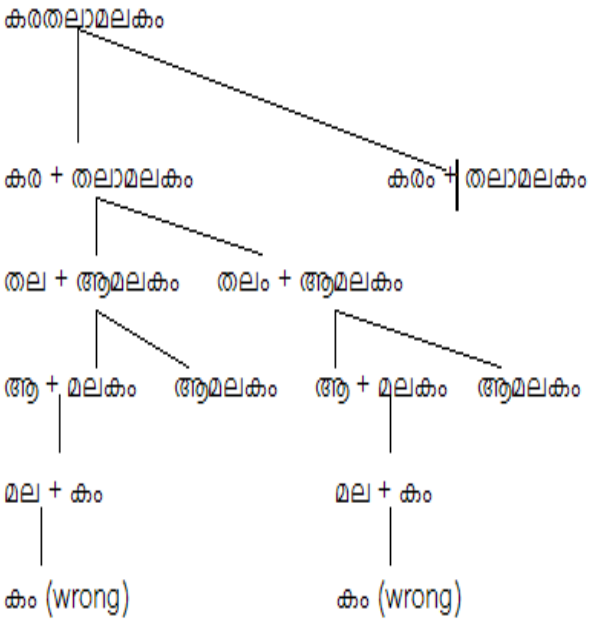

Figure 3. Depth first traversal for the splits.

This algorithm generates all possible splits for each compound word. A set of compound words were subject to random analysis using the algorithm. A lexicon size of 1000 words which includes most of the commonly occurring nouns, verbs, modifiers, verb and noun suffixes was utilized for analyzing the performance of the developed splitter algorithm.

VI. RESULTS AND DISCUSSIONS

The results of the analysis of a few compound words using the algorithm are detailed in table 2 .
The splitter generated all the possible splits for each of the compound words. The system generated three kinds of output

TABLE 2. TABULATED RESULTS

	Compound word/splits		Compound word/splits
1	കരതലാമലകം	6	പടക്കത്തിന്
	കര (land)+ തല (head)+ ആമലകം(aamla)		പട(army) +കത്ത(letter)+ഇന്(case suffix)
	കര (land)+ തല (head)+ ആമലകം(amla)		പട (picture)+ കത്ത്(lett er)+ ഇന്(case suf)
	കര(hand)+ തല(plane)+ ആമലകം(ahna)		പടക്കം(crackers)+ ഇന്(case suffix)
	കര(hand)+ തല(head)+ ആമലകം(amla)		
2	പകരാവി	7	പടച്ച്
	പകര + ആവി		പട + ച്ച്
3	അവനാരെന്ന്	8	കലാത്
	അവ (those) +നാര്(fibre)+ എന്ന്(that)		കല (pot)+ ആത് (verb suffix)
	അവൻ(he)+ ആര്(who) + എന്ന്(that)		കല(mark)+ മാൻ(deer)
4	നല്ലാള	9	പറയാനായി
	നല്ല(good)+അവ(that)+ ആള(friend)		പറ(measure)+ ആന (elephant)+ ആയി
	നല്ല(good)+ താള(shampoo)		പറ(talk)+ ആത് (verb suffix)+ ആയി
		10	കരിമ്പനക്
5	ആനക്കൊമ്പുകൊണ്ടുള്ള		കരി(black) +പന(palm)+ കൊ(suffix)
	ആന(elephant)+ഉം (conjunct)+ ആട(goat)+ഉം (conjunct)+ പാടി(jump)		കരിമ്പ(sugarcane)+ നടക്ക(move)

1. Correct unique split in which only a single split is obtained (sl no5)
2. Correct multiple splits in which more than one split is obtained and all the splits are correct (sl. no 3 and 4). The correct ones can be found only after more contextual analysis.
3. Multiple splits which contain both correct and syntactically incorrect splits (sl no. 6).

The statistics related to splitting a package of lexicon size of 1000 words is as shown in Table 3.

TABLE 3. PERFOMANCE STATISTICS

Lexicon size(words)	1000
Compound words tested	4000
Correct unique split	90%
Correct multiple split	8%
Multiple splits with incorrect split	2%

The results of compound word splitting thus arrived at can be subsequently forwarded to a morph analyzer for detailed analysis.

The following merits have been observed for the developed algorithm.

1. It uses the minimum number of compounding rules which reduces the computing time.
2. The use of a trie model for lexicon storage which reduces the word lookup time.
3. Since intermediate split results are stored, the system takes less time for finding all splits. Here before a recursive call a check is made whether a split was generated for the substring. In that case the split is taken directly from the stored array. In this way the system implements rote learning which is used to cut down the search paths [9].
4. It takes into account of almost all sandhi rules in Malayalam and Sanskrit.
5. The use of morphemes instead of surface forms for the words reduces the lexicon size.
6. It generates all possible spits for a compound word.
7. It uses best set of splitting rules to minimize the number of splits. In example 9, the first split can be eliminated by using the morphological tags of ആന and ആയി and the rule that the deergha sandhi occurs only between nouns.

VII. CONCLUSION

Morphological processing is an important component in any NLP application like machine translation, story understanding system and question answering system. This demands an in-depth study of the morphology of the language used. Malayalam, a prominent language in the family of Dravidian languages, was taken for case study and a detailed study of the morphology of the language was made. The compound word splitter developed can be used as a preprocessing step in NLP tasks like translation, question answering etc to split the compound words in the source text. The system can also be used as a tool for learning compound word splitting in Malayalam. Spell checker efficiency can be improved using this compound splitting method. The system can process almost all the morphology due to inflections and derivations of verbs and nouns and word compounding. The results obtained are encouraging and the work can be extended to other Dravidian languages since they exhibit structural homogeneity.

REFERENCES

[1] Anand Kumar M, Dhanalakshmi V, Soman K.P, Rajendran S, "A Sequence Labeling Approach to Morphological Analyzer for Tamil Language", IJCSE, Vol. 02, No. 06, 2010, pp.2201-2208

[2] Sumam mary Idicula, Peter S. David, "A morphological processor for Malayalam language", South Asia Research, vol.27 (2), 2007, pp.173-186.

[3] Srivastava,M., Mohapatra,B., Bhattacharya,P., Nitin Agrawal,N. and Smriti Singh, "Morphology based Natural Language tools for Indian Languages",2004 .cse.iitb.

[4] Varma A.R.R.R, "Kerala Paniniyam", DC. Books, Kerala, India, 2000.

[5] Narayana Pilla K.S, "Aadhunika malayala vyakaranam", Kerala Bhasha Institute, 2nd Edition, 2003,pp.30-35.

[6] Aparna,S. and Ingle,M., " Morphological parsing of 'sandhi' based words in Sanskrit", Language in India., Vol.4, 2004.

[7] Philipp Koehn,and Kevin Knight(2003)- "Empirical methods for compound word splitting", Proceedings of the tenth conference on European chapter of the ACL, vol.1.

[8] Jurafski,D. and Martin, J.H. , "Speech and natural language processing", Pearson Education,New Delhi, India, 2008.

[9] Elaine Rich, Kevin Knight, Sivasankar B Nair, "Artificial Intelligence", Tata McGraw-Hill, 3rd Edition,2009, pp.348-349.

[10] Jisha P Jayan, Rajeev R.R, s Rajeendran "Morphological Analyser for Malayalam-A Comparison of Different Approaches", IJCSIT, vol 2, No 2, December 2009, pp 155-160.

[11] Guy De Pauw, Gilles-Maurice de Schryver, "Improving the Computational Morphological Analysisof a Swahili Corpus for Lexicographic Purposes", Thirteenth International Conference of the Association for Lexicography, Lexikos 18 ,AFRILEX-reeks/series 18, 2008,pp,.303-31.