Implementation of Malayalam Morphological Analyzer Based on Hybrid Approach

Vinod P M, Jayan V, Bhadran V K
Language Technology Centre
CDAC Thiruvananthapuram
{vinodpm, jayan, bhadran}@cdac.in

Abstract

The Malayalam Morphological analyzer, which described in this paper, is developed based on the hybrid approach, i.e. combining methodologies of both paradigm and suffix stripping approaches. Lttoolbox, an important module in the Apertium package, acts as the back bone of this system. The analyzer program in Lttoolbox tokenizes the text in surface forms (lexical units as they appear in texts) and delivers, for each surface form, one or more lexical forms consisting of lemma, lexical category and morphological inflection information. This system also borrows the concepts of suffix stripping approach that would help a lot to improve the accuracy. The main objective of this system is to help the language students as well as common people.

Keywords: Apertium, Lttoolbox, paradigm approach, suffix stripping, hybrid approach.

1. Introduction

Malayalam language is one among the four major Dravidian languages in south India and also one among the 22 scheduled languages in India. It is mainly spoken by the people of Kerala state and the union territories of Lakshadweep and Mahe. Around 35.9 million people are using this language.

Developing a full fledged Morphological analyzer is very difficult due to the rich morphology and agglutinative nature of Malayalam language. The major problems of Malayalam morphology are wide range of inflections, multiple suffixes and tendency of adjacent words to concatenate etc. The multiple inflections can be solved by following the paradigm approach and Lttoolbox helps to implement the paradigm approach. In order to handle other problems we need a powerful suffix stripping module. The proposed system [1] follows a hybrid approach for developing the morphological analyzer, i.e., the combination of paradigm and suffix stripping approaches.

Lttoolbox is available with the Apertium toolkit, which is an open source shallow-transfer machine translation system originated within the project “Open-Source Machine Translation for the Languages of Spain” [2]. Lttoolbox can be customized to any language by including the required lexical dictionary. Agglutinative languages require some additional modules for better and accurate word processing.

2. Related Works

There are many works carried out in the field of Malayalam Morphological Analyzer, but no complete system is available for common people. References [3], [4] and [5] are some important works related to Malayalam Morphological Analyzer. Two common approaches identified towards the development of Morphological analyzer are suffix stripping and paradigm approaches. Reference [6] mentioned about a hybrid approach for developing the Malayalam Morphological Analyzer and its comparison with the above mentioned approaches.

3. Hybrid approach architecture

Lttoolbox plays an important role in our system. The lt-proc program processes the surface form in single pass. Multiple suffix words can be processed on iterative basis. So we added post processing and suffix stripping modules to handle multiple suffix problems.

As an example consider the case of a pronoun concatenate with a post position and it seems like a single word unit. In this situation the lt-proc cannot recognize the surface form even though both of these words are present in the dictionary. The post processing and suffix stripping modules help us to handle this situation by identifying the post position and provides a space between them. Then once again process them with the help of lt-proc program. Whenever the same surface form comes twice as the input of the post processing module, then it is considered as unknown.

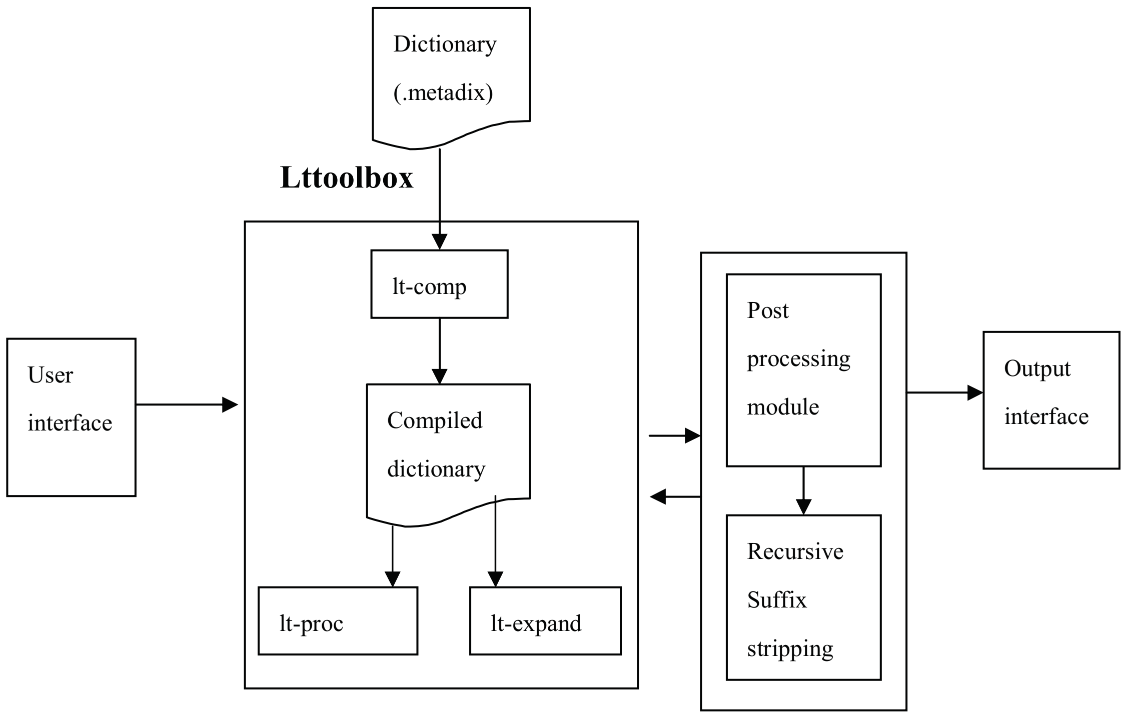

Figure 1. Architecture of Malayalam Morphological Analyzer.

4. Lttoolbox

Lttoolbox [1]can be used for lexical processing, morphological analysis and generation etc. Here we are using Lttoolbox for morphological analysis. Splitting the surface form into its lemma and the grammatical information is known as analysis process. For example the word ‘cats’ splits into its lemma ‘cat’ and grammatical information <noun><plural>. The reverse process is the generation process. The Lttoolbox doing this processing with the help of provided lexical dictionary.

Ltoolbox makes use of the finite state transducer (FST) approach for doing lexical processing. The class of FST used in Lttoolbox is ‘letter transducer’. Lt-comp, lt-proc and lt-expand are the three programs provided by the Lttoolbox package. These programs are used for compiling, morphological analyzing/generating and expanding the dictionary respectively.

The first part of developing a morphological analyzer is the creation of lexical dictionary which is also called morphological analyzer specification file. Monolingual dictionary, bi-lingual dictionary and post generation dictionary are the three types of dictionaries used in the Lttoolbox. The dictionary files providing a facility to define and use paradigms which helps to share the same inflection pattern. The dictionary files are mainly found with “.dix” or “.metadix” extensions. We are using monolingual metadictionary [2] (‘.metadix’) since it provides some extra features like handling variable lemma and argument passing etc. Dictionary files for some languages are available from the Apertium incubator. [3]

4.1 Dictionary structure

The dictionary format is XML, which become very powerful in linguistic data representation and exchange. The dictionaries follow a typical block structure and the important blocks in the dictionary structure are,

- An alphabet definition: The list of alphabets used in the dictionary file.
- Definition of symbols: It contains the grammatical symbols that are present in the file.
- Definition of paradigms: Paradigm definitions to be used in the dictionary sections or in other paradigms.
- One or more sections with conditional tokenization.
- One or more sections with unconditional tokenization.

4.2 Paradigm Approach

A paradigm definition defines an inflection paradigm in the dictionary file. It groups the words which are having similar inflection pattern. The paradigm can be viewed as small dictionaries which specify regularities in the lexical processing of the dictionary entries. To specify these regularities, each paradigm has lists of entries <e> like the ones in the dictionary, that is, it has the same structure as a dictionary section <section>; therefore, paradigm entries consist of a pair (<p>) with left side (<l>) and right side (<r>). These elements can contain text or grammatical symbols <s>. Some times a paradigm definition contains entries of

[1] http://wiki.Apertium.org/wiki/Lttoolbox

[2] http://wiki.Apertium.org/wiki/Metadix

[3] http://wiki.Apertium.org/wiki/Incubator

another paradigm. An example of paradigm definition is shown below.

```
<pardef n="walk/ed__verb">
      <e><p>
              <l>ed</l>
              <r>ed<s n="verb"/><s n="past"/></r>
          </p></e>
   <e> <p>
              <l></l>
              <r><s n="verb"/><s n="present"/></r>
         </p></e>
</pardef>
```

5. Malayalam Morphological Dictionary

The main part in the development of a morphological analyzer using Lttoolbox is creating the morphological dictionary. Since Malayalam is morphologically rich with wide range of inflections [7] and [8], lot of attention is needed in creating the dictionary. We choose metadictionary in order to make use of its additional features over the normal dictionary.

The paradigm facility helps a lot to handle the inflections of the words. In order to cover all cases we created 24 noun paradigms, 56 verb paradigms, 12 adjective paradigms etc [9], [10], [11] and [12]. The alphabets used in the dictionary are in wx notation.[4]

5.1 Noun Paradigm

To understand the paradigm design the surface form of a word can be considered as two parts such as root and suffix. By examining the suffix portion of the noun word the grammatical information like case, gender and number can be obtained. The main task is identifying the suffix and root of the particular surface form. With the help of the compiled dictionary file the Lttoolbox will perform the morphological analysis provided the particular noun entry (consists of the word and the paradigm number) is present in the dictionary. While examining the noun suffixes we identify some common properties like plural markers, case markers etc. For example consider the noun 'മ രം' (maraM) and 7 case markers comes in the suffix part are,

മ രം Nominative case (maraM -> no suffix)

മ രത്തെ Accusative case (marawwe -> -e)

മ രത്തിന് Dative case (marawwin ->-in)

മ രത്തോട് Sociative case (marawwOt ->-Ot)

[4] Phonetic Notations for Malayalam alphabets and which is used for the linguistic processing.

മ രത്തിൽ	Locative case (marawwil_ ->-il_)
മ രത്താൽ	Instrumental case (marawwAl_ ->-Al_)
മ രത്തിന്റെ	Genitive case (marawwinZe ->-inZe)

Similarly the plural marker comes with the word ‘maraM’ is ‘kaL_’. The variant form of plural markers comes in the Malayalam morphology are ‘kaL_’, ‘mAZ_’ and ‘aZ_’. Examples for them are ‘മ രങ്ങൾ’ (maraffaL_), ‘രാജാക്കന്മ ാർ’ (rAjAkkan_mAZ_) and ‘മ നു ്യർ’ (manuRyaZ_) respectively.

```
<pardef n="mara/M__n">

<e><p>
      <l/>
      <r>M<s n="N_NOUN"/><s n="SG"/><sa/></r>
    </p></e>
<e><p>
      <l>ffaL_</l>
      <r>M<s n="N_NOUN"/><s n="PL"/><sa/> + <s n="kaL_"/></r>
   </p></e>
:
:
</pardef>
```

The tags used in the paradigm definition are given in Table 1. The ‘sa’ tag is used for passing some arguments while adding an entry of that particular paradigm. The name of the paradigm is "mara/M__n" (where n denotes noun) and which is specified in the ‘pardef’ tag. We can give name according to our convenience. The possible inflections of the word ‘maraM’ are maraM, maraffal_, marawwe, maraffaLe, marawwin, maraffaLkk, marawwOt, maraffaLOt, marawwil_, maraffaLil_, marawwAl_, maraffaLAl_, marawwinZe and maraffaLute.

This paradigm "mara/M__n" can be used for the words which have the same inflection pattern. The paradigm entries are mentioned in the main section in the dictionary. One entry for this paradigm is,

```
<e lm="vanaM">
            <i>vana </i>
                    <par n="mara/M__n" sa="NTR"/> </e>
```

Table 1. List of some noun tags.

Tag	Description
N_NOUN	Nominative noun
SG	Singular
PL	Plural

MF	Masculine/Feminine
NTR	Neuter
M	Masculine
F	Feminine

While specifying the entries the attribute ‘lm’ (lemma name) contains the surface form of the word. The ‘<i>’ element contains the root of the particular word and which may or may not be the actual root word according to the Malayalam language. The actual root word will be obtained after the processing. The next tag contains the paradigm name and the additional information that we are passing to the paradigm definition. With the help of the specified paradigm name the particular surface form is processed. When we give a word to the ‘lt-proc’ program it will check whether that word starts with any of the ‘<i>’ element in the paradigm entries. If a match occurs the corresponding paradigm is used for processing that word. If the particular inflection is present in the paradigm the program will give a result.

5.2 Verb Paradigm

“The morphology of the verb in Malayalam is somewhat complex, and more research is needed before a definitive statement can be made about, firstly, what different aspectual values can be combined, and, secondly, what restrictions there are on the combination of different aspectual values with different modal forms. A hypothesis one might put forward for testing is that all morphological combinations are possible that are semantically interpretable and compatible.” [13]. Malayalam verbs always have complex and multiple suffixes. Much of the complexity is resolved in the 53 verb classifications done for computational purpose by R Ravindra Kumar [9]. With the help of this classification we have created 56 verb paradigms. In order to know the paradigm classifications consider the first verb paradigm which groups the root words that are ends in “Y”(ഴ്). In this class “unnu”, “uM” and “wu” are the present future and past tense marker respectively. The present and future suffixes are concatenated with out any addition and deletion from the root. To add the past tense suffix “wu” a “u” should be added to the root word and then the tense suffix will be added.

E.g.

Present (pr)	“unnu”	uYunnu	ഉഴുന്നു
Future (fu)	“uM”	uYuM	ഉഴും
Past (pa)	“wu”	uYuwu	ഉഴുതു

To get a causative form, causative suffix “kk” will be added to the root word except in past. In past the causative suffix will be “cc” and past tense suffix is “u”. While adding “kk” or “cc” the link morph “uvi” will be added to the root.

E.g. uYuvikkunnu, uYuvikkuM, uYuviccu. (ഉഴുവിക്കുന്നു , ഉഴുവിക്കും , ഉഴുവിച്ചു)

The double causative forms can be generated by adding the causative marker “ppi” and “kk”. The link morph ‘uvi” will be added at the end of root verb before adding the double causative suffixes.

E.g. uYuvippikkunnu, uYuvippikkuM, uYuvippiccu.

(ഉഴു ി ിക്കുന്നു, ഉഴു ി ിക്കും, ഉഴു ി ി ു)

A noun form can be derived from the root word by adding the suffix “al_” at the end of root word.

E.g. uYal_ (ഉഴൽ)

Analogously paradigms are created for adjectives and pronouns also. But post positions and adverbs have no paradigm classes and they are specified in the main section of the dictionary file. Examples for adverb entries are,

```
<e> <p>
        <l>ennAl_</l>
        <r>ennAl_<s n="ADV"/> </r>
    </p></e>
<e> <p>
        <l>ivite</l>
        <r>ivite<s n="ADV"/> </r>
    </p></e>
```

6. Post Processing and Suffix Stripping Modules

The main problem associated with the Malayalam morphological analyzer is the identification of words which are formed by combining multiple words. These kinds of words are commonly called the complex words. The Lttoolbox cannot handle the complex words in Malayalam. So the presence of complex words and wide range of inflections makes our task very difficult. In order to overcome this situation we introduced the post processing module and the suffix stripping module. The task of post processing module is to identify and extract the words which are not accepted or unidentified by the processing module in Lttoolbox. And the suffix stripping module has a good collection of commonly used link morphs, post positions and suffixes. With the help of this collection the suffix stripping module can separate the word and its suffixes from the surface form. Recursive suffix stripping method is using for Malayalam. The sandhi rules are also taken into consideration during the splitting process.

6.1 Recursive Suffix Stripping Algorithm

We have been using a good collection of suffixes, postpositions and link morphs to perform the suffix stripping. The accuracy of splitting is very much depending up on these collections as well as the word splitting. A good sandhi splitter will improve the performance to a great extend.

1) Check for any link morphs present in the surface form. If no go to step 3
2) Split the word based on the information obtained from the previous step and check the prefix part for validity using lt-proc. If valid remove the prefix part from the word and stores in a word list. Then consider the remaining portion for further processing. If not valid go to step 3.
3) Check for a highest matching suffix. If not found any match then add the word to the word list. If found go to step 2.
4) Word list gives the splitted form of the complex word.

For example consider the complex word “ഓടിക്കൊണ്ടിരിക്കുകയായിരുന്നു”

(OtikkoNtirikkukayAyirunnu). This verb form contain the link morph 'കൊണ്ട് ' (koNt). The suffix stripping module processes this word in a step by step fashion as follows.

Step 1: [ഓടിക്കൊണ്ടിരിക്കുകയായിരുന്നു]

Step 2: ഓടി (കൊണ്ട്) ! രിക്കുകയായിരുന്നു

Step 3: ഓടി + കൊണ്ട് +[! രിക്കുകയായിരുന്നു]

Step 4: ഓടി + കൊണ്ട് + ! രിക്കുക (ആയിരുന്നു)

Step 5: ഓടി + കൊണ്ട് + ! രിക്കുക + ആയിരുന്നു

Here the word within the square bracket denotes the unknown word obtained from Lttoolbox. And the simple bracketed portion denotes the link morph of suffix which is identified by the suffix stripping module. The splitting is done based on the words shown with in the simple brackets. In each step the unknown word (if present) is processed with the suffix stripping module. The decomposed form of the complex word is generated in the fourth step. This outcome is given to the output interface. One important thing is that the tense of the complex word is identified from the last portion (here it is ആയിരുന്നു (Ayirunnu) which denotes past tense).

7. Evaluation and Results

Table 2. Dictionary entries.

Category	**Number of Paradigms**	**Number of entries**
Noun	24	25267
Adjectives	12	17213
Verb	56	9070
Pronoun	-	150
Adverb	-	2423
Postpositions	-	120

The important functions of the morphological analyzer are finding the exact lemma, segmenting the suffix part and identifying the POS tags. The evaluation is done based on the above 3 criteria's [14]. Since it is very hard to evaluate the accuracy of the system automatically, we compare the results with human intuitions and Malayalam lexicon books. The dictionary contains around 54,240 entries we randomly take 10000 words for the evaluation purpose. Each result is evaluated with respect to the 3 functionalities of the morphological analyzer. The average of 3 cases is considered as the total accuracy of the system.

Root identification accuracy	**94%**
POS identification accuracy	**85%**
Suffix segmentation accuracy	**72%**
Accuracy of MA (average of above 3)	**83.67%**

Output for the word “ഓടിക്കൊണ്ടിരിക്കുകയായിരുന്നു” (OtikkoNtirikkukayAyirunnu.) is given below.

```
ഓടി
----------------
 Root: ഓട്
 Verb     Mun_Vineyaccam
 Suffix:  (!  )

 Root: ഓട്
 Verb     Past  Transitive
 Suffix:  (!  )

 Root: ഓട്
 Verb     Past  Intransitive
 Suffix:  (!  )

ക്കൊണ്ട്
--------------------
 Postposition (കൊണ്ട്)

!  രിക്കുക
-------------------
 Present  (!  രിക്കുക)

 Root: !  ർ
 Verb   Nat_Vineyaccam
 Suffix:  (!  ) (!  ക്ക്) (!_ക)

യായിരുന്നു
---------------
 Past  (ആയിരുന്നു)
```

8. Conclusion and Future Work

Malayalam is a morphologically rich and agglutinative Indian language. So it is very difficult to develop a computer system for Malayalam. The major problems which we have to face during the initial stages are multiple suffixes, high inflections, tendency of adjacent words to join together etc. But the hybrid approach is very effective for developing a morphological analyzer for Malayalam language. The accuracy of the system is mainly depends on the morphological dictionary and the suffix list used. The efficient handling of unknown words also improves the quality.

Morphological analyzer is a main and important module of the Parsing system. This work can be extended to develop a Malayalam parser.

References

[1] Vinod P M, Jayan V, Sulochana K G, "Malayalam Morphological Analyzer: A Hybrid Approach with Apertium Lttoolbox," *Proceedings of ICON-2011: 9th International Conference on Natural Language Processing*, Macmillan Publishers, India, Page: 219-224, 2011.

[2] Mikel L. Forcada, Boyan Ivanov Bonev, Sergio Ortiz Rojas, Juan Antonio Pérez Ortiz, Gema Ramíŕez Sánchez, Felipe Sánchez Martínez, Carme Armentano-Oller, Marco A. Montava, Francis M. Tyers. "Documentation of the Open-Source Shallow-Transfer Machine Translation Platform Apertium", 2010.

[3] Rajeev. R. R, Elizabeth Sherly, "Morph Analyser for Malayalam Language: A suffix stripping approach," *Proceedings of 20th Kerala Science Congress,* Thiruvananthapuram, 2008

[4] Jisha P. Jayan, Rajeev R.R, Dr. S Rajendran, "Morphological Analyser and Morphological Generator for Malayalam-Tamil Machine Translation," *International Journal of Compter Applications*, Volume 13, No.8, 2011.

[5] Saranya S. K, "Morphological analyzer for Malayalam Verbs," unpublished, 2008.

[6] Jisha P. Jayan, Rajeev R. R., S. Rajendran. "Morphological Analyzer for Malayalam-A comparison of Different Approaches", *IJCSIT*, Vol. 2: 155-160, 2009.

[7] A.R.Raja Raja Varma. 2000. "Keralapaanineeyam", D. C Books Kottayam-12.

[8] Suranad Kunjan Pillai, "Malayalam Lexicon", The University of Kerala, 2000.

[9] R. Ravindra Kumar, K. G. Sulochana , V. Jayan. "Computational Aspect of Verb Classification in Malayalam", *Information Systems for Indian Languages Communications in Computer and Information Science*, Volume 139, Part 1, 15-22, 2011.

[10] Sunil R, Nimtha Manohar, V. Jayan, K. G. Sulochana, "Morphological Analysis and Synthesis of Verbs in Malayalam", *ICTAM,* 2012

[11] Sunil R, Nimtha Manohar, V. Jayan, K. G. Sulochana, "Noun Classification in Malayalam for Natural Language Computing Applications", *NCILC*,2012.

[12] Nimtha Manohar , Sunil R, V. Jayan, K. G. Sulochana, "Malayalam Adjective and Pronoun classification for Computational Applications", *NCILC,* 2012.

[13] R.E. Asher, T.C. Kumari. “Malayalam”, Routledge London and New York, 1997.

[14] Huihsin Tseng, Keh-Jiann Chen. “Design of Chinese morphological analyzer", *Proceedings of the first SIGHAN workshop on Chinese language processing* - Volume 18, 1-7, 2002.