# PCFG Grammar Generation for Malayalam

Dhanya L K
*Assistant Professor, MACFAST*

Rani Susan Oommen
*Assistant Professor MACFAST*

**Abstract- Grammar specification is an important component in natural language processing applications and machine translation. In this paper, a probabilistic context free grammar along with parse tree is generated for Malayalam language based using hybrid approach of rule based and statistical machine learning is presented. Generated transformation grammar is also represented with the probability(PCFG) calculated using treebank corpus .In this work, the accuracy depends on the accuracy of tagging phase ,here the system can achieve 96.21% in tagging phase.**

**Keywords – Machine Translation, Natural Language Processing,Parsing, Parse tree, Treebank Corpus.**

## I. INTRODUCTION

The grammar is a formal specification of the structure allowable in the language and the parsing technique is the method of analyzing an input sentence to determine its structure according to the grammar.
The major pipeline in the development of PCFG grammar generation includes transformation grammar generation, Treebank corpus generation and PCFG grammar generation. A hybrid approach based on rule based and statistical machine learning is proposed in this work. Malayalam is a language with rich morphology and high agglutination and is of free order. Literature shows that the rule based grammar refinement process is extremely time consuming and difficult and failed to analyze accurately a large corpus of unrestricted text. Hence, most modern parsers are based on statistical or at least partly statistical, which allows the system to gather information about the frequency with which various constructions occur in specific contexts. Any statistical approach requires the availability of aligned corpora which are: large, good-quality and representative.
Probabilistic context free is implemented in certain Indian languages Hindi, Bengali etc.But in Malayalam, unfortunately, there is no work available related to this technique

## II. LITERATURE SURVEY

Akshar Bharati and Rajeev Sangal developed grammar formalism, Paninian Grammar Frame- work" that has been successfully applied to all free word Indian languages. They have described a constraint based parser for the framework. Paninian framework uses the notion of karaka relations between verbs and nouns in a sentence. Experiments show that the Paninian framework applied to modern Indian languages performs better accuracy.

Selvam M and Thangarajan R have attempted to build phrase structured hybrid language model. In the development of hybrid language model, new part of speech tag set for Tamil language has been developed with more than 500 tags which have the wider coverage. A hybrid language model has been trained with the phrase structured Treebank using immediate head parsing technique. This paper discussed the disadvantages of CFG, as well as advantages of PCFG.
There are some works related to Parser for Indian Languages. For Kannada language, Penn Treebank based statistical syntactic parser is developed in 2010 [1]. The Penn Treebank structure was used to create the corpus for statistical syntactic parser. The proposed syntactic parser was implemented using supervised machine learning and probabilistic context free grammars approaches. Experiment shows that the performance of the proposed system is significantly good and has very competitive accuracy.

## III. PROPOSED SYSTEM

The proposed system will generate parse tree based on grammar specifications (PCFG) while entering a Malayalam (simple/complex) sentence in Malayalam script. It is implementing using hybrid approach, include rule based as well as statistical machine learning and probabilistic method. These approaches are Unicode-based and reduce the use of lexical dictionaries. The scope of the parser is not limited to the machine translation scenario. It can also adapted to many other NLP tasks for Malayalam language such as relationship extraction, anaphora resolution, semantic role labeling, named entity recognition etc. In this system .The work is performed in four phases, serves as separate modules and the result of each phase connects as the input to the subsequent phase. At the end of parsing phase parse tree as well as transformation grammar with probability (PCFG) also generated.
The pipeline starts with a tokenization that splits sentences and produce output as tokens. In tokenization, rules are applied for proper identification of tokens. The POS tagger will then produce appropriate Part Of Speech(POS) tag to tokens, and the chunking phase combine POS Tags to appropriate phrases as noun phrase, verb phrase etc. Finally the parser phase will generate parse tree. In addition to parse tree, transformation grammar with probability (PCFG) also generated during parsing phase. Subsequently transformation grammar and parse tree are generated for each sentence that helps to create Treebank corpus that may be further utilized for statistical parsing.

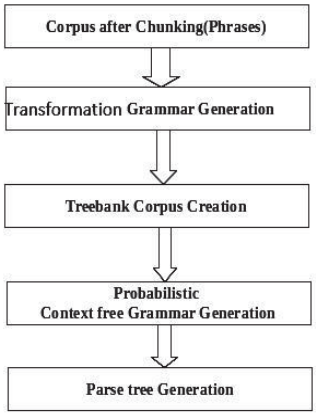



Fig 1: Pipeline of Proposed system

Tokenization phase uses reverse form of sandhi rules, described in Kerala Panineeyam [8]. The tokenizer developed acts as a preprocessor for morphological analyzer and POS tagger and is implemented based on Unicode. Since a single rule solves many different word combinations, it outperforms the current systems. This component is essentially required as the first preprocessing step for languages like Malayalam and any other agglutinative languages. Parts of Speech Tagging, a grammatical tagging, is a process of marking the words in a text as corresponding to a particular part of speech, based on its definition and context. This is the first step towards understanding any languages. The hierarchical tag set BIS tag set is used for tagging. Commonly chunking (shallow parsing) for Malayalam is implemented using statistical approach. But here rule based approach is used for implementing chunking. Nouns, Pronouns, Demonstratives, Quantifiers are grouped in to the phrase called Noun Phrase (NP). Finite verbs, Auxiliary verbs are grouped into Finite verb phrase (VGF) and Non finite verbs are grouped into Non finite Verb Phrase (VGNF) Adverbs and adjectives are grouped to for RBP and JJP respectively

*A .Transformation Grammar Generation*

CFG, sometimes called a phrase structure grammar plays a central role in the description of natural languages. In general a CFG is a set of recursive rewriting rules called productions that are used to generate patterns of strings. The transformation grammar for the given sentence "സീത സിനിമ കണ്ടു"is given below.

```
S          ---->    NP  VGF

NP         --->     N_NNP_S_F
NP         --->     N_NN_O_NU
VGF        --->     V_VM_VF

N_NNP_S_F  ---> സീത
N_NN_O_NU  ---> സിനിമ
V_VM_VF    ---> കണ്ടു
```

The different parts-of-speech tags and phrases associated with a sentence can be easily illustrated with the help of a syntactic structure. NLTK (Natural Language Tool Kit) is used for tree generation. There is one successful package called draw trees , by passing parsed output into this function will generate following parse tree. The parse tree for the given sentence "□□□ □□□□□ □□□□□"is given below.

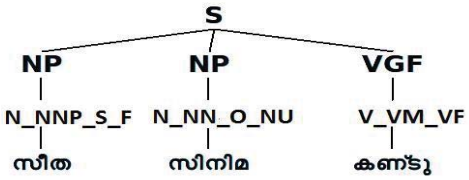



Fig 2: Example Parse Tree

In this work, grammar was generated using rule based method and that grammar has lot of applications. By leftmost derivations system can generate sentences from transformation grammar and this will help to calculate the probability of each rule in transformation grammar of a sentence. Transformation Grammar has three levels, root level grammar, which contain 'S' and phrases. „S" will represent sentence and second level will contain productions for phrases that is actually pos tags. The last level is the productions for tags that will reveal the words. In table 1,the real necessity of transformation grammar is explained, because it top down parser we have to generate sentence from grammar by leftmost derivation.

Table -1 Sentence Generation from Transformation Grammar

| Production Rules | Derivation | Rules Used |
|---|---|---|
| S----> NP VGF<br><br>NP---> N_NNP N_NN_O_NU<br>VGF ---> V_VM_VF<br><br>N_NNP--->സീത<br><br>N_NN_O_NU--->സിനിമ<br><br>V_VM_VF--->കണ്ടു | S---->NP VGF<br><br>S---->N_NNP N_NN_O_NU VGF<br>S----> സീത N_NN_O_NU VGF<br><br>S----> സീത സിനിമ VGF<br><br>S----> സീത സിനിമ V_VM_VF<br><br>S----> സീത സിനിമ കണ്ടു | <br><br>NP--->N_NNP N_NN_O_NU<br>N_NNP--->സീത<br><br>N_NN_O_NU--> സിനിമ<br><br>VGF---> V_VM_VF<br><br>V_VM_VF-->കണ്ടു |

*B. Treebank Corpus Generation*

Statistical Parsing (Probabilistic Parsing) needs parsed corpus (parsed corpus is also called tree bank corpus) for probability calculation, But for Malayalam unfortunately tree bank corpus is not available. So that In this work parsed 2000 sentences.

*C.PCFG Grammar Generation*

The problem of CFG is that it misses the probabilistic model which is needed in order to disambiguate between parses. A Probabilistic Context Free Grammar (PCFG) is a probabilistic version of a CFG where each production has a probability. The simplest way to gather statistical information about a CFG is to count the number of times each production rule is used in a corpus containing parsed sentences. This count is used in order to estimate the

probability of each rule being used.In our case, we estimate the rules probabilities using the relative frequency of the rule in the training set

$$P(A \rightarrow BC/A) = \text{freq}(A \rightarrow BC)/\text{freq}(A)$$

Once we have the probability of the production rules in a PCFG, the probability of a parse tree for a particular sentence can easily be calculated by multiplying the probabilities of the rules that built its sub-trees. The advantage of PCFG based syntactic parser model is that, for any two or more different sentences that have same pos tag sequence, but have different syntactic tree structure, then the sentence structure that has more probability would be considered or correctly parsed.

A stochastic context-free grammar (SCFG; also probabilistic context-free grammar, PCFG) is a context-free grammar in which each production is augmented with a probability. The probability of a derivation (parse) is then the product of the probabilities of the productions used in that derivation; thus some derivations are more consistent with the stochastic grammar than others. There are two ways to assign probability to a grammar [11].the simplest way is to use a corpus of already parsed sentences. Such corpus is called Treebank corpus.eg: Penn treebank corpus. Given a Treebank the probability of each expansion of a nonterminal can be computed by counting the number of times that expansion occurs and then normalizing

$$P(A \rightarrow B/A) = \text{count}(A \rightarrow B)/\text{count}(A)$$

When a Treebank is unavailable the counts needed for finding PCFG probabilities can be generated by first parsing a corpus. Total probability of trees calculated by multiplying the probabilities of all rules in a PCFG for a sentence. Here probability calculation is done by above equation that is Bayes'law. In probability theory and statistics, Bayes‟law is a result that is of importance in the mathematical manipulation of conditional probabilities. Bayes'law can be derived from more basic axioms of probability, specifically conditional probability.

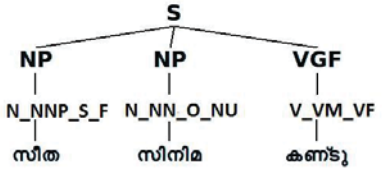
For the transformation grammar generation phase, last level (word level) probabilities cannot calculated by Bayes'law, so that here I calculated probability for POS tag based on total no of words in the corpus(112734 tokens).That probability list is given below **.**

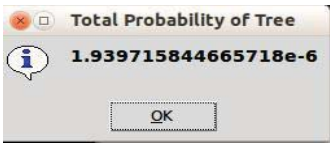
| | |
|---|---|
| Proper Noun | 0.141176887377 |
| Common Noun | 0.313060190598 |
| Location Noun | 0.0245066780033 |
| pronoun | 0.0133382931236 |
| Adjactive | 0.0551953803096 |
| Nonfinite Verb | 0.0691270769122 |
| Finite Verb | 0.0632904665746 |
| Auxilary Verb | 0.0365607397173 |
| Post Position | 0.0306621320013 |
| Adverb | 0.0996563573883 |
| Demonstratives | 0.0260566124632 |
| Particles | 0.011744074822 |
| Quantifier | 0.0794894958727 |
| Conjunction | 0.0139405533709 |

The probabilistic context free grammar for the given sentence “□□□ □□□□□ □□□□□”is given below.

| | | |
|---|---|---|
| S ----> | NP VGF | 0.113888888889 |
| NP ---> | N_NNP_S_F N_NN_O_NU | 0.0100607839027 |
| VGF---> | V_VM_VF | 0.605194805195 |
| N_NNP_S_F ---> | സീത | 0.141176887377 |
| N_NN_O_NU ---> | സിനിമ | 0.313060190598 |
| V_VM_VF ---> | കണ്ടു | 0.0632904665746 |

The total probability of tree is calculated by multiplying the probabilities of all rules.






## IV. PERFORMANCE EVALUATION

The performance of the system is mainly depends on the accuracy of tagging phase, so system was evaluated using tnt-diff module and the incorrect outputs were noticed. The system performance was considerably increased by adding the input sentences.

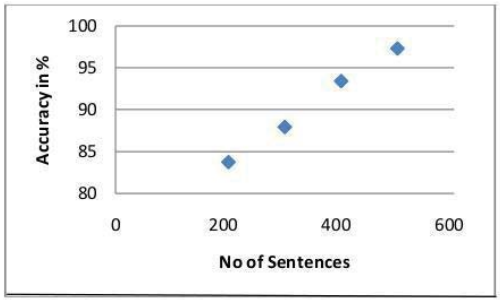



Fig 3: Performance graph using TnT-diff command

The precision for a class is the number of true positives (i.e. the number of items correctly labeled as belonging to the positive class) divided by the total number of elements labeled as belonging to the positive class (i.e. the sum of true positives and false positives, which are items incorrectly labeled as belonging to the class). Recall in this context is defined as the number of true positives divided by the total number of elements that actually belong to the positive class (i.e. the sum of true positives and false negatives, which are items which were not labeled as belonging to the positive. In information retrieval, a perfect precision score of 1.0 means that every result retrieved by a search was relevant (but says nothing about whether all relevant documents were retrieved) whereas a perfect recall score of 1.0 means that all relevant documents were retrieved by the search (but says nothing about how many irrelevant documents were also retrieved). The evaluation of the system is done with four corpora from different domain which contains various types of words.

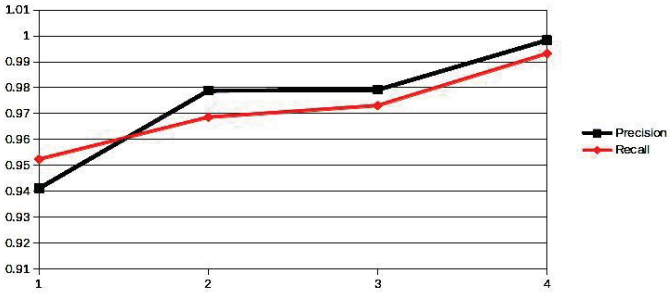



Fig 4: Performance graph using precision and recall

## V. CONCLUSION AND FUTURE WORK

The unavailability of large volume of corpus is a big handicap. The proposed PCFG grammar generator has been tested with 100 distinguished sentences, and the result obtained is promising and encouraging. Subsequently created Treebank corpus that may be further utilized for statistical parsing.

The accuracy of this work can be increased by expanding the size of the training corpus further. Increasing the annotated corpus size is meaningful only if there is a sufficiently large corpus of good quality available. It can also

be utilized to improve many other NLP tasks such as anaphora resolution, relationship extraction, named entity recognition. Parser can be used in the future to improve these NLP tasks in Malayalam. The accuracy of probability calculation can also be increased by expanding the no of sentences.

## REFERENCES


[1] Manju K, Soumya S, and Sumam Mary Idicula, "Development of a POS Tagger for Malayalam -An Experience," artcom, pp.709-713, 2009 International Conference on advances in Recent Technologies in Communication and Computing, 2009

[2] Latha R Nair,David Peter S,"Language Parsing and Syntax of Malayalam Language" , 2nd International Symposium on Computer, Communication, Control and Automation 3CA 2013

[3] Antony P J, Nandini. J. Warrier, Dr. Soman K P "Penn TreebankBased Syntactic Parsers for South Dravidian Languages using a Machine Learning Approach". International Journal of Computer Applications (0975 8887) Volume 7 No.8, October 2010.

[4] Latha R Nair, David Peter S, Renjith P Ravindran, A System for Syntactic Structure Transfer from Malayalam to English, International Conference on Recent Advances and Future Trends in Information Technology,p.110-122.

[5] Remya Rajan, Remya Sivan, Remya Ravindran, K.P Soman. Rule based machine translation from english to malayalam. In Conference Proceedings on International Conference on Advances in Computing, Control, and Telecommunication Technologies, pages 439441, 2009.

[6] Mary Priya Sebastian, G Santhosh Kumar,English to malayalam translation:a statistical approach. In Proceedings of the 1st Amrita ACM-W Celebration on Women in Computing in India, page 64. ACM, 2010.

[7] Letha R Nair,David Peter S,A System for Syntactic Structure Transfer from Malayalam to English,International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT2012)Proceedings published in International Journal of Computer Applications,Vol 3

[8] A.R.Raja Raja Varma, "Kerala Paaniniiyam", Ed. 8, D C Books, Kottayam, May 2006

[9] Aneena George,English To Malayalam Statistical Machine Translation System,International Journal of Engineering Research & Technology (IJERT),ISSN: 2278-0181,Vol. 2 Issue 7, July – 2013

[10] .Unnikrishnan P,A Novel Approach for English to South Dravidian Language Statistical Machine Translation System,(IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 08, 2010, 2749-2759