ISSN: 0976-3031

Available Online at http://www.recentscientific.com

CODEN: IJRSFP (USA)

International Journal of Recent Scientific Research
Vol. 8, Issue, 6, pp. 17965-17970, June, 2017

International Journal of Recent Scientific Research

DOI: 10.24327/IJRSR

Research Article

SYNTAX AND SEMANTIC ANALYSIS OF DEVANAGARI HINDI

AlokKumar*., Saurabh and Mushahid Raza

Department of Computer Science and Engineering, University Institute of Engineering and Technology, C.S.J.M. University, Kanpur, India

ARTICLE INFO

Article History:

Received 06th March, 2017
Received in revised form 14th April, 2017
Accepted 23rd May, 2017
Published online 28th June, 2017

Key Words:

Syntax analysis of Hindi, Semantic analysis of Hindi, Semantical correctness/incorrectness of Hindi sentences, Probabilistic parsing of Hindi sentences, CYK algorithm for parsing Hindi, Natural language processing.

ABSTRACT

With the recent development of utf-8 encoding format, lot of focus has been shifted toward development of artificially intelligent machines that can handle Hindi language text. Lexical, Syntactic and Semantic analyses are the essential steps in processing applications involving any natural language. These act as major pre-processing tasks in applications like information extraction, text summarization, machine translation etc. Moreover these are also helpful to obtain better results as efficiency of such machines depends mostly upon percentage of ambiguities that had been resolved in Lexical, Syntax and Semantic processing phases of natural language itself. In this paper we present a method to perform syntax analysis of Hindi sentences via probabilistic parsing technique using CYK algorithm to build parse table and a method to detect semantical correctness/incorrectness of Hindi sentence by using morphological information of words provided by morphological analyzer tool of IIIT Hyderabad, India. Our work achieved overall accuracy of approximately 80% in resolving ambiguities from Hindi sentences at syntax processing stage and accuracy of approximately 89% in detecting semantical errors in semantic analysis phase (provided that sentence was syntactically correct).

INTRODUCTION

This research paper is part of an intellectual field of computer science called as Natural Language Processing (NLP). An abiding challenge within artificial intelligence has been to build machines that can understand human languages unambiguously. Resolving ambiguities is the main area to focus upon in processing natural languages because they are hardly precise or plainly spoken. Recent advances in NLP technologies have emerged more mature and robust methods for analysing unrestricted texts of human languages. Despite the fact that Hindi language has more than 500 million speakers only few works have been done in this field taking Hindi language as an input natural language. Our work is among few works that have made significant contribution towards resolving ambiguities occurring in syntactic and semantic structures of Hindi sentences. Our work focuses on assigning best syntactic structure to the sentences of Hindi language by using probabilistic parsing techniques and then checking semantical correctness (provided it is syntactically correct) of sentence by using morphological information of words occurring in the sentence. Our major contribution lies in the fact that entire processing of syntax and semantical analysis can be done without having any need of internet connection.

Entire paper is structured as follows: Section 2 gives enumerated narration of the proposed system. Section 3 shows delineative analysis of the results obtained. Section 4 knocks around related works in this area. Section 5 contains conclusion as well as some guidance for future works. Section 6 mentions references used to complete our work.

Proposed System

Entire system proposed by us can be divided into following phases:

1. Lexical analysis of Hindi text.
2. Constructing probabilistic context free grammar (PCFG) for Hindi.
3. Syntax analysis of Hindi.
4. Checking semantical correctness of sentence by using morphological information.

Lexical Analysis of Hindi

This phase processes input text at character level and recognizes valid Hindi words form input text. In order to handle Devanagari Hindi characters we have used UTF-8 encoding scheme. Major role of this phase is Lexical error detection and its correction. This phase initially collected Hindi text from different sources (Hindi newspaper websites, blogs in

*Corresponding author: **AlokKumar**
Department of Computer Science and Engineering, University Institute of Engineering and Technology, C.S.J.M. University, Kanpur, India

Hindi etc.) to create Hindi lexicon/dictionary. All misspelled words in an input text were identified with the help of created lexicon/dictionary. To assist users, system generates a list of most probable correct words for each misspelled word. Implementation of this phase was done by algorithms proposed by us [1]. To demonstrate processing of Lexical analysis phase we have used Hindi text editor developed by us as shown in figure 1. We added two new buttons "SYN" and "SEM" into existing text editor in order to demonstrate processing of syntax and semantic analysis phase as well.

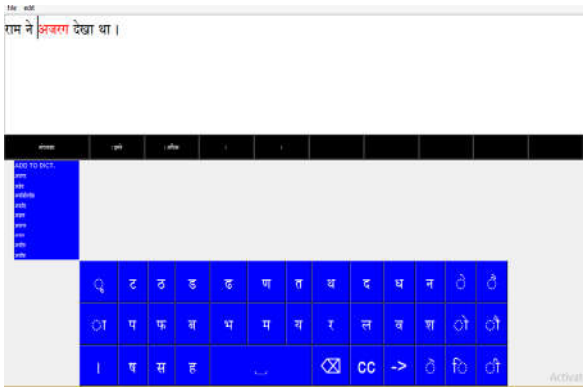


Figure 1 Lexical analysis of Hindi

Constructing Probabilistic Context Free Grammar for Hindi

Probabilistic context free grammar is a context free grammar in which every production rule has a probability value associated with it. Since pre-defined probabilistic context free grammar for Hindi is not available therefore we constructed a probabilistic context free grammar for it by generalizing production rules that are most commonly used in Hindi sentences.

We followed these sequence of steps to construct probabilistic context free grammar for Hindi:

Input: Correct Hindi sentences.
Output: Probabilistic context free grammar in CNF (FILE_4)

Tool Used: Offline POS (part of speech) tagger tool for Hindi language provided by NLTK.

1. Collect approximately 2000 Hindi sentences in a file (say FILE_1) from various Hindi corpus available online.
2. Associate part of speech tags with each word of a sentence stored in FILE_1.
3. Identify short phrases (Noun phrase (NP), Verb phrase (VP), Adjective phrase (JJP), Adverb phrase (RBP)) present in each sentence of FILE_1 by constructing regular expressions for defining different kind of phrases combination possible in Hindi language and store each structure of sentence (sentence along with POS tags and short phrases) in FILE_1.
4. Count frequency of each Hindi word present in FILE_1 and store word along with POS tag and frequency value in a file (say FILE_2).
5. Remove Hindi words from all 2000 structure of sentence present in FILE_1.
6. Add start symbol "S" with each structure of sentence present in FILE_1.
7. Call function CONSTRUCT_CFG to construct context free grammar (CFG) for Hindi and pass FILE_1 as input parameter.
8. Convert CFG production rules of FILE_3obtained after step vii into Chomesky Normal Form (CNF).
9. Evaluate frequency of each production rule present in FILE_3.
10. Append data of FILE_2 into FILE_3.
11. Evaluate probability value of each production rule contained in FILE_3 by using a formula: Probability of a production rule = Frequency of occurrence of production rule evaluated in Step viii given its left hand side variable divided by Total frequency of only those productions that have same left hand side variable.
12. Store production rules along with corresponding probability values in a file say (FILE_4).

Algorithm followed by function CONSTRUCT_CFG:

Input: FILE_1

Output: Context free grammar (FILE_3)

Global Variable: i (initially i=0)

Data Struture Used: Array of stacks

1. Take pointer to start index of FILE_1.
2. Read a line at current pointer position.
3. Split line into individual words/symbols.
4. PUSH the symbols on a STACK until an opening brace is encountered.
5. If opening is encountered for the first time, then PUSH it onto STACK. Else create a new STACK and copy element from top of STACK of predecessor stack at bottom of this newly created stack and then start pushing next symbols encountered onto this newly created stack.
6. If closing brace is encountered then pop everything from the STACK in which symbols are currently being pushed, store result in form of string in a new file (say FILE_3) and delete the STACK afterwards.
7. Add the upcoming symbols on the predecessor stack of recently deleted stack and If opening brace is encountered then goto Step v. Else if closing brace is encountered then goto Step vi.
8. If all symbols of a line have been read then move pointer to next index of FILE_1 and If EOF is reached then return FILE_3 else goto Step ii.

```
JJP ( JJP NN ) 0.0471731241952
VP ( PREP NVB ) 0.0259032294447
NP ( NNP PREP ) 0.0171432254184
A ( CC NP ) 0.15652173913
NP ( NLOC NN ) 0.00620851146104
VP ( RB VP ) 0.0149253731343
S ( A PUNC ) 0.172444444444
NP ( PRP NN ) 0.0201508088251
JJP ( INTF JJP ) 0.05700573569
VP ( NNP VFM ) 0.0189856761532
```

Figure 2 Screenshot of portion of FILE_4

We were successfully able to find generalized probabilistic context free grammar for Hindi (contained in FILE_4) using

above mentioned algorithms. Screenshot of portion of FILE_4 is shown below in figure 2.

Entire processing of above mentioned algorithms is explained below with the help of an example Hindi sentence:

1. Example sentence:
2. POS Tags: QF_
3. Short phrases: NP Deletion of Hindi words: NP (QF NN) NP (NN) VP (VFM VAUX) SYM.
4. Add start symbol "S": S (NP (QF NN) NP (NN) VP (VFM VAUX) SYM).
5. Construction of context free grammar: From output of Step iv we wanted to extract:

    A. NP (QF NN)
    B. NP (NN)
    C. VP (VFM VAUX)
    D. S (NP NP VP SYM)

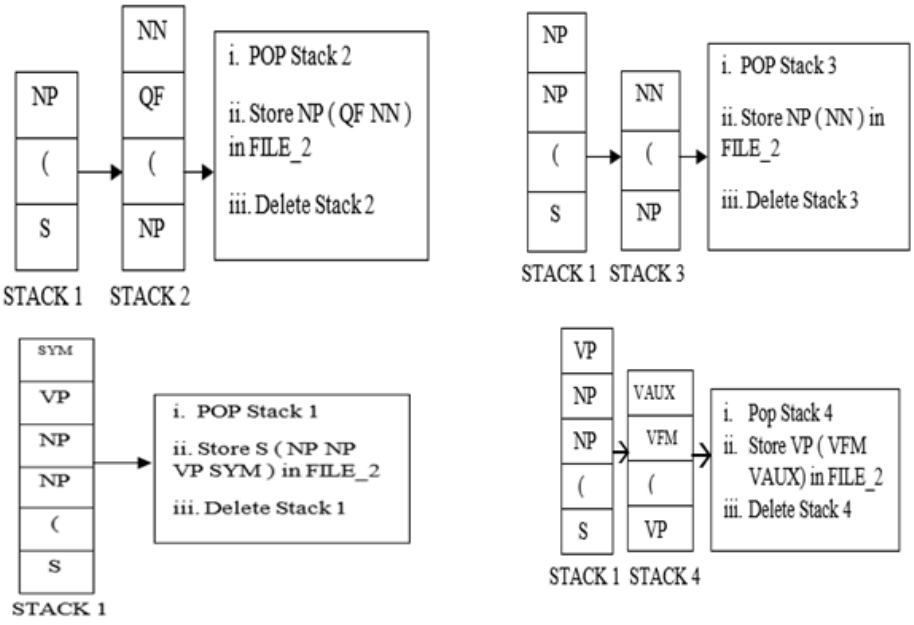
These components were part of context free grammar of Hindi being constructed.

We assumed following things in order to construct context free grammar:

1. All abbreviations used for phrases and start symbol "S" are non-terminals.

In place of arrow (→) denoting right hand side, Right hand side of production is contained inside parenthesis.

2. Based on above assumptions our CONSTRUCT_CFG algorithm works as follows:



We used CYK algorithm to generate parse trees and since CYK uses context free grammar in CNF therefore conversion from CFG to CNF in above algorithm became necessary part of our work. Moreover conversion to CNF form introduced 1407 new non terminals.

Evaluation of probabilities for each production rule present in FILE_3 can be understood by going through example shown below in TABLE 1.

Table 1

CNF rules	Frequ-ency	Total frequ-ency	Probab-ility
NP (NN PRP)	200	NP=200+500 =	200/700=0.28571
NP (NP VP)	500	700	500/700=0.71428
VP (VFM VAUX)	220	VP= 220+240=460	220/460= 0.47826
VP (VAUX VAUX)	240		240/460= 0.52173

Syntax Analysis of Hindi

Syntax analysis mechanism performs two things:

1. Checks whether sentence is syntactically correct or not.
2. Derives best possible parse tree if sentence is syntactically correct.

We derived parse trees along with their corresponding probability value for a given Hindi sentence by designing an algorithm that makes use of probabilistic context free grammar (FILE_4) developed in Section 2.2 and matrix data structure for creating parse tables. Steps of algorithm are given below:

Input: Hindi sentence

Output: Best possible parse tree along with its probability value.

1. Pass sentence through offline POS tagger tool of NLTK.
2. Execute CYK algorithm to construct parse table matrix (say PARSE_TABLE) and a table that stores information of how parse table is being formed (say INFO_TABLE) by sending POS tags of sentence evaluated in Step i and a file (FILE_4) of PCFG constructed in section 2.2 as input parameters.
3. If location (o, n) of parse table matrix contains start symbol 'S' then goto step iv. Else display syntactically incorrect sentence.
4. Derive all possible parse trees for input Hindi sentence by back tracking PARSE_TABLE using the information present in INFO_TABLE.

5. Evaluate probability of each parse tree derived in step iv by multiplying the probability values of each of the PCFG production of FILE_4 used in derivation of given parse tree.
6. Find parse tree having highest probability value and display it along with its probability value.

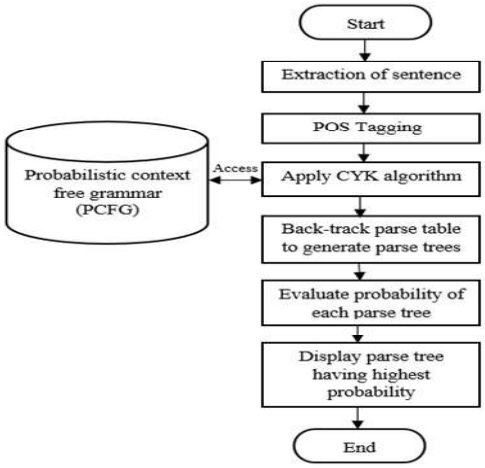


Figure 3 Syntax analysis of Hindi

If parse tree derived for sentenceis S (NP then probability of parse tree (step v) is evaluated as shown in Table 2.

Table 2

Production rules	Probability Value	Probability of parse tree
S (NP VP)	0.50	
NP (NN NN)	0.40	
VP (VM VAUX)	0.30	
NN	0.40	0.50*0.40*0.30*
NN	0.40	.40*0.40*0.60*0.60 =
VFM	0.60	0.003456
VAUX	0.60	

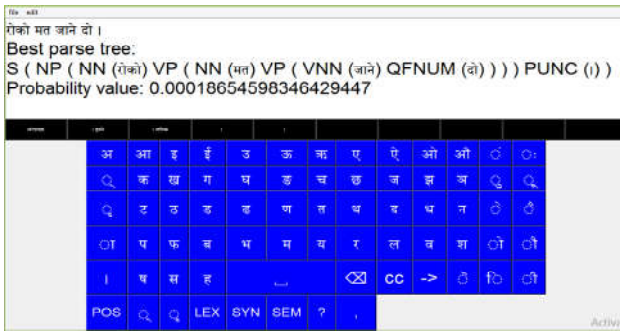


Figure 4 Screenshot of text editor showing best parse tree along with its probability.

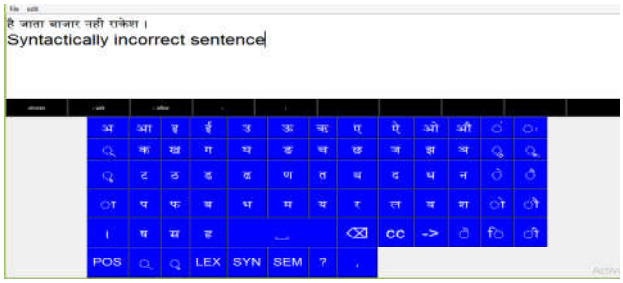


Figure 5 Screenshot of text editor showing display message for syntactically incorrect sentence

Screenshot of text editor showing best parse tree along with its probability and result of parsing is shown in figure 4 and figure 5.

Semantic Analysis/Checking Semantical Correctness Of Sentence

For checking semantical errors in a sentence we required morphological information of each word present in a sentence. Morphology is the branch of NLP that deals with identification, analysis and internal structure of words of a given language. We used morphological analyzer tool for Hindi developed by IIIT Hyderabad, India. For a given Hindi sentence this morphological analyzer produces eight fields as output for each word present in a sentence, they are:

1. Root: Root of word.
2. Cat: Category of word (e.g. Noun=n, Pronoun=pn, verb= v etc.).
3. Gen: Gender of word (e.g. Masculine =m, Feminine= f, Neuter=n etc.).
4. Num: Number of the word (e.g. Singular=sg, Plural=pl etc.).
5. Per: Person of the word (e.g. 1st person =1, 2nd person=2 etc.).
6. Case: Case of the word (e.g. direct=d, oblique=o etc.)
7. Tam: Case marker for noun or tense aspect mode for verb of the word.
8. Suff: Suffix of the word.

Ex- For word morphological analyzer of IIIT Hyderabad produces a set of these eight values: <,n,m,pl,3,d,0, 0>. Flow chart for Semantic analysis of Hindi text is shown in figure 6.

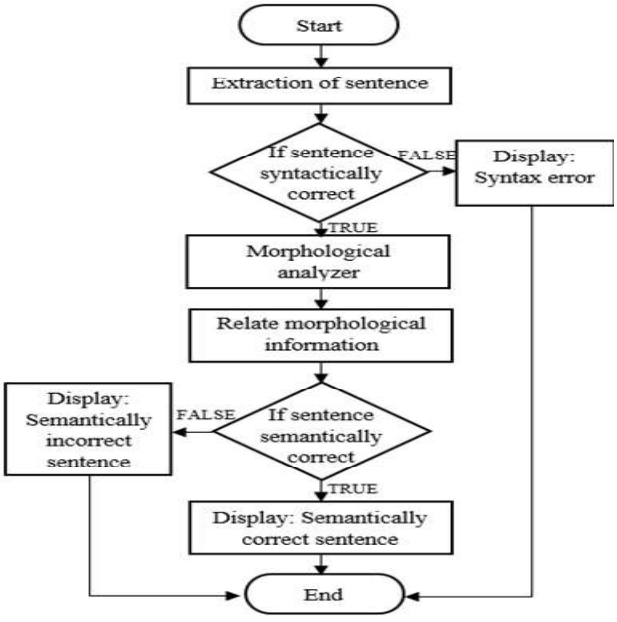


Figure 6 Semantic analysis of Hindi

Using the information provided by morphological analyzer we were able to detect semantical errors in the sentences and the reasons behind those errors based on rules defined by us. These rules wanted a given Hindi sentence to fulfil certain set of conditions in order to be semantically correct, some of such rules are mentioned below:

1. Verb and auxiliary verb must have same gender.

2. Noun/Pronoun and verb must have same gender if there is only one subject in the sentence.
3. If there are multiple subjects in the sentence separated by comma or conjunction then Verb must be of plural category.
4. Noun and adjective must have same gender.

Using rules as mentioned above sentence was detected syntactically correct but semantically incorrect by our system as it failed to satisfy rule i since Verb possessed feminine gender whereas Auxiliary verb possessed masculine gender (see figure 7), sentence was detected syntactically correct but semantically incorrect as it failed to satisfy rule iv since Noun possessed masculine gender whereas adjective possessed feminine gender, sentence was detected syntactically correct but semantically incorrect as it failed to satisfy rule ii since subject i.e. Noun possessed masculine gender whereas Verb possessed feminine gender and sentence was detected syntactically correct but semantically incorrect as it failed to satisfy rule iii since more than one subjects were being referenced in the sentence but verb used was of singular nature.

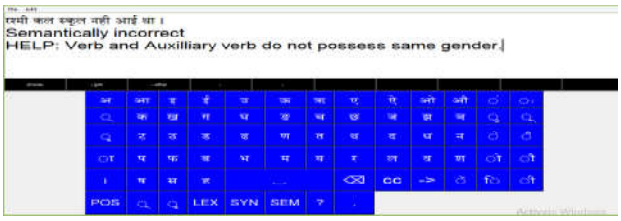


Figure 7 Screenshot of text editor showing semantic analysis

RESULT ANALYSIS

We measured efficiency of our design by taking1500 Hindi sentences and passed them through the syntax analyzer designed by us, among them frequency of occurrence of different kinds of Hindi phrases in the parse trees generated are shown in TABLE 3.

Table 3

Part of speech phrase	Frequency
Noun Phrase (NP)	2533
Verb Phrase (VP)	2376
Adverb Phrase (RBP)	1839
Adjective Phrase (JJP)	1587

Table 4 shows that method proposed by usto derive parse trees for Hindi sentences obtained approximately 83% accuracy when sentences taken were unambiguous whereas it achieved accuracy of approximately 77% when ambiguous sentences were chosen as an input.

Table 4

Sentencetype	Tested Sentences	NO. OF Times Parse Trees Generated
Unambiguous	1350	1127
Ambiguous	150	115

We took 1000 syntactically correct Hindi sentences and passed each one of them through our semantic analyzer and it correctly detected semantical correctness/incorrectness 894 times thereby achieving an accuracy of approximately 89% in detecting semantical errors.

It was seen, when more and more number of ambiguous words were present in the sentence this accuracy percentage was dropping because in such cases incorrect morphological information was being generated by the morphological analyzer tool used by us.

Related Research Work

Alok kumar, Saurabh Sharma, Mushahid Raza [1] designed a method to detect and correct Lexical errors in Hindi words by taking help of a dictionary containing meaningful Hindi words. They created dictionary of Hindi words using two basic information retrieval techniques, Web crawling and web scrapping. Error recovery mechanism designed by them suggested most probable words suggestion for an erroneous words to the user.

Harsh Verma [2] designed a PCFG model for Hindi by training it on an annotated corpus for Hindi (obtained from NLPAI) containing about 600 sentences. He further reduced PCFG in CNF (Chomsky Normal Form) by standard tree binarization algorithm.

Akanksha Gehlot, Vaishali Sharma *et al*.[3], designed a system that translated the document from Hindi to English by using transfer based approach. Their system took an input text, checked its structure through parsing using CYK algorithm and then reordering rules were used by them to generate the text in target language.

Akshar Bharati *et al*.[4], proposed a parser for parsing Indian languages. Parser worked on constraint based hybrid approach and they further showed detailed description of role of hard constraints and soft constraints to build an accurate and powerful parser.

B.Venkata Seshu Kumari and R. Rajeswara Rao[5] proposed a method of improving parsing of Hindi and Telugu language by combining Malt parser (Nivre *et al*., 2007a), and MST parser (McDonald *et al*., 2006).

Nitin Hambir and Ambrish Srivastav[6] developed a parser for Hindi language. Parser generated a parse table using CYK algorithm which took as an input, a sentence along with PCFG-CNF form rules which mainly covered the assertive sentences of Hindi language.

Latha R Nair and David peter S[7]constructed syntax rules for parsing sentences of Malayalam language by following procedure of POS tagging, Chunking and setting hierarchical dependency rules for chunk tags in Malayalam language.

Sambhav Jain *et al*.[8], integrated extraneous knowledge from Hindi word net in order to obtain better results while parsing. The knowledge of semantics was extracted from Hindi Word Net using concept Hierarchy technique.

Seema Mahato and Dr.Ani Thomas[9] proposed an automated essay grading system to overcome the issues involved evaluating grammatical and semantic error and to overcome influence of local and regional languages in Hindi essays.

CONCLUSION AND FUTURE WORKS

Syntax and Semantic analysis are the two important phases of the natural language processing because ability of an intelligent machine to take most appropriate action depends entirely on

number of ambiguities that had been resolved and number of error that had been detected and corrected in syntax and semantic processing stages itself. Our design to resolve such ambiguities and errors for Hindi language produced satisfactory results with an average accuracy of more than 80%.Efficiency of deriving correct parse trees for ambiguous sentences can be further increased by taking more number of sentences (we took 2000 sentences) to generalize the probabilistic context free grammar. In order to get better results while detecting semantical correctness/incorrectness in the sentences having more number of ambiguous words, some other morphological analyzer tool for Hindi(we used tool of IIIT Hyderabad, India) can be used to obtain better morphological information. Our work can also be extended to build the system that can translate Hindi sentences into some other language.

References

1. *Alok Kumar, Saurabh Sharma, Mushahid Raza. Lexical Analysis of Devanagari Hindi Language.*
2. Harsh Verma. Probabilistic Context Free Grammar for Hindi.
3. AkankshaGehlot, VaishaliSharma, ShashipalSingh, Ajai Kumar. Hindi to English Transfer Based Machine Translation System.
4. AksharBharati, Samar Husain *et al*. Constraint Based Hybrid Approach to Parsing Indian Languages.
5. B.VenkataSeshu Kumari, R. Rajeswara Rao. Improving Indian Language Dependency Parsing by combining Transition based and Graph based Parsers.
6. Nitin Hambir, Ambrish Srivastav. Hindi Parser-based on CKY algorithm.
7. Latha R Nair and David peter S. Language Parsing and Syntax of Malayalam Language.
8. SambhavJain, Naman Jain *et al*., Exploring Semantic Information in Hindi Word Net for Hindi Dependency parsing.
9. Seema Mahato, Dr. (Mrs.) Ani Thomas. Lexico-Semantic analysis of essays in Hindi language.
10. Abney, Partial Parsing via Finite-State Cascades. Natural Language Engineering, 2(4):337-344.
11. Attardi, G. and F. Dell'Orletta. 2008. Chunking and Dependency Parsing. LREC Workshop on Partial Parsing: Between Chunking and Deep Parsing. Marrakech, Morocco.
12. Shubhangi Bhatambrekar, and Niket Tajne.Context Free Grammar (CFG) for MODI Script.
13. Nisheeth Joshi, ItiMathur. Evaluation of Computational Grammar Formalisms for Indian Languages.
14. Pawan Deep Singh, Archana Kore *et al*. Hindi Morphological Analysis and Inflection Generator for English to Hindi Translation.
15. Pallavi Bagul, Prachi Mahajan *et al*. Semantic analyzer for Marathi Text.
16. Robert F. Simmons and John F. Burger. A Semantic Analyzer for English Sentences.
17. Sandeep Kumar Vishwakarma and Chanchal Kumar Vishwakarma. A Graph Based Approach to Word Sense Disambiguation for Hindi Language.

How to cite this article:

AlokKumar., Saurabh and Mushahid Raza.2017, Syntax And Semantic Analysis of Devanagari Hindi. *Int J Recent Sci Res.* 8(6), pp. 17965-17970.

****