VIBHAKTI PARSER: A TOOL FOR ENGLISH PLAIN TEXT

Beenu Yadav*

ABSTRACT

Resolving syntactic ambiguities is still one of the biggest problem in machine translation systems. The goal of designing a computer system, capable of conversing with people in their own natural language has been a dream of A. I., almost since the inception of digital computers in the 1940's. To be able to converse in human language is a basic pre-requisite of any intelligent assistant, because language serves as our basic vehicle for thought and communication. Producing an unambiguous parse is a major challenge for the parsers developed for English Language. Correct verb and noun group attachment poses the greatest hindrance in this regard.

Keywords: *Vibhakti, Vibhakti Parser, Case Roles.*

*Assistant Professor, College of Professional Education, Meerut.

1.0 INTRODUCTION

Communication is fundamental to the evolution and development of all kinds of living beings. With no disputes, languages should be recognized as the most amazing artifacts ever developed by mankind to enable communication. Computer has also become such a unique machine, due to its capacity to communicate with humans through languages. It is worth mentioning that the languages understood by computers and humans are quite different, yet people can communicate with computers.

Natural languages are communication tools which constantly evolve and therefore are highly complex in nature. The presence of ambiguities at all levels is the source of complexity of natural languages, and it constitutes a major problem for automatic language processing. For achieving this we design Vibhakti Parser.

2.0 DEFINING VIBHAKTI PARSER

The parser verifies the grammatical correctness of the input text and identifies the 'Vibhaktis' or 'Case Roles' in the input text. So we call it *"Vibhakti Parser"*. The Vibhakti Parser performs two functions.

- Parsing the text
- Identifying the Vibhaktis/Case Roles

2.1 Parsing the Text

To parse the text, parser uses language grammar rules [1, 3], which are defined as production rules. This parsing examines the syntax of the text and results that text is syntactically correct or incorrect.

Parser is a collection of rules for representation of sentences in the form of production rules. The Production rules can be written as,

<simple sentence> = <subject> < verb> <complement>

The Parser has production rules for all types of sentences such as Simple sentences, Compound sentences etc.

2.2 Identifying the Vibhaktis/Case Roles

Within a sentence different nouns are connected with verb through case relationship. To identify these case relations in each language vibhaktis are used. The Paninian Grammar Framework concerns the Sanskrit language [4, 5]. However, it prescribes a generic and language independent decomposition of any sentence into eight different information carrying vibhaktis. These vibhaktis or case roles are as follows:

1. **Kartaa/ Nominative**

   Doer of an activity or the subject.

2. **Karma/Accusative**

   Entity that is being acted upon or the object.

3. **Karan/Instrumental**

   Entity that is being employed to complete an act.

4. **Sampradan/Dative**

   The chief motivation behind the action of the beneficiary subject.

5. **Apadan/Ablative**

   Entity in Karma is separated as a consequence of the action.

6. **Sambandh/Genitive**

   The possessor of something in the sentence.

7. **Adhikaran/Locative**

   Place, time related to the entity at the time of action.

8. **Sambodhan/Vocative**

   Calling upon someone – hey etc.

For example, consider the sentence,

English*:* The student presented the seminar of his project with projector in seminar hall.

Hindi: Student ne Apne Project ka Seminar Kaksha mein Projector se seminar ko present kiya

In this sentence,

(i) Student – *Kartaa*

(ii) Seminar – *Karma*

(iii) Projector – *Karan*

(iv) His Project – *Sambandh*

(v) Seminar Hall – *Adhikaran*

3.0 SYNTACTIC PARSING

Syntactic parsing examines the sentence syntactically and results valid sentence, if sentence is syntactically correct else results invalid sentence. The language grammar rules, which are defined in the form of production rules, are used to parse the text [1, 2]. For representation of sentences, production rules are described in the parser. It includes representation for all types of sentences. Input sentences are parsed by defined sentence structure rules and when it sets to any one of the rules then that sentence is proved to be syntactically correct.

Examples:

1) **S1: Which book did you buy from book-store for C++?**

→ "Which" <noun phrase> "did" <subject> <predicate>

→ "Which" <noun> "did" <nominative personal pronoun> <predicate>

<predicate> → <V> <prep phr>*

→ <V1> <prep phr> <prep phr>

2) **S2: I called him but he gave me no answer.**

→ <Simple Sentence> <Conjunction> <Simple Sentence>

→ <I> <called him> <Conjunction> <he> <gave me no answer>

→ <subject1> <predicate1> <Conjunction> <subject2> <predicate2>

<subject1> → <nominative personal pronoun>

<predicate1> → <V> <complement>

→ <Vpast> <object>

<subject2> → <nominative personal pronoun>

<predicate2> → <V> <complement>

→ <Vpast> <indirect object> <object>

The abbreviations prep phr (prepositional phrase), post phr(postpositional phrase), V(verb), Vpast (verb in past form), V1 (verb in first form) [1] are used to represent the parts of speech in the above examples.

We have demonstrated the syntactic parsing with the help of few examples. The examples are all valid sentences as they match to any one of the defined production rules for syntactic parsing.

After the syntactic checking the vibhaktis/case roles of valid sentences are determined. The vibhakti parser follows different approach to handle non simple sentences because the Rule base for Vibhakti Parsing is made effectively for simple sentences. So before Vibhakti Parsing non simple sentences are remodeled to simple sentences. This approach is delineated in the next section.

4.0 REMODELING OF NON-SIMPLE SENTENCES

In order to examine Vibhaktis/case roles in Compound and Complex sentences, the sentences have to be transformed into simple sentences. To convert into simple sentence the conjunctions used in non simple sentences are removed [3]. Few conjunctions are 'and', 'but', 'because', 'although', 'otherwise', 'who', 'which', 'that' etc.

Before examining Vibhaktis in such sentences rules for the elimination of conjunctions are defined. Some of the rules are given below.

4.1 For Compound Sentence

Parser eliminates the conjunction and the sentence will be divided into simple sentences, which were joined by conjunction.

If C1 = Simple sentence1 + conjunction + simple sentence2

It can be modeled in two simple sentences as:

S1 = Simple Sentence1

S2 = Simple Sentence2

For Example,

C1 = Come soon otherwise you will get late.

→ Come soon + conjunction + you will get late

It can be modeled in two simple sentences as:

S1 = Come soon.

S2 = you will get late.

4.2 For Complex Sentence

The parser debars all the imminent conjunctions in a complex sentence and transforms it into simple sentences. The following examples demonstrate and give the comprehensible picture for the remodeling of complex sentences.

- **If CON is 'but':**

C1 = subject + predicate1 + 'but' + remaining sentence

→ subject + verb/VP + complement + 'but' + remaining sentence

There can be two cases:

a. if there is linking verb/auxV in verb/VP then (subject + linking verb/auxV) will be added to the remaining sentence.
b. Else only subject is added to the remaining sentence.

Thus the sentence formed is a compound sentence. For example:

C1 = He is poor but honest.

→subject (he) + linking verb (is) + complement (poor) + 'but' + remaining sentence

→subject (he) + linking verb (is) + complement (poor) + 'but' + subject (he) + linking verb (is) + remaining sentence

→He is poor but he is honest.

→ a compound sentence

It can be modeled as:

S1 = He is poor

S2 = He is honest.

- **If CON is 'who', 'which':**

If C1 = subject + predicate1 + 'who' + predicate2

Then

a. If Karam is present in predicate1, it becomes Kartaa of predicate2.
b. If Karam is not present in predicate1 then subject (Kartaa) of sentence becomes Kartaa of predicate2.

For Example,

C1 = Julia calls Alex who is sitting in the park.

→subject + verb + object + 'who' + predicate2

→subject + verb + object + 'who' + object + predicate2

It can be modeled as:

S1 = subject + verb + object → Julia calls Alex

S2 = object + predicate2 → Alex is sitting in the park.

C2 = The farmer is cutting corn which has ripened.

→subject + verb + object + 'which' + predicate2

→subject + verb + object + 'which' + object + predicate2

It can be modeled as:

S1 = subject + verb + object → the farmer is cutting corn

S2 = object + predicate2 → corn has ripened

Similarly such rules can be further appended to convert complex sentence into simple sentences.

5.0 VIBHAKTI PARSING

The Vibhakti Parser parses the syntactically correct sentence to identify the vibhaktis, states, verbs and others elements. The rule base is made for determination of each of them. After remodeling we apply the following rules and identify Vibhaktis, States, etc.

5.1 Rule Base

For identification of Vibhaktis/Case roles

1. Subject of the sentence is identified as Kartaa Vibhakti.
2. If the subject has pronoun then Parser replace it with the corresponding noun, it is identified as Kartaa Vibhakti.

3. Rest of the Vibhkatis are identified from complement of the sentence.
    a. If complement has an object(direct/indirect) then it is Karam Vibhakti.
    b. In case of pronoun object before determining Vibhakti, Parser substitutes it with its respective noun.
4. The vibhaktis are identified by preposition in the prepositional phrase.
5. In prepositional phrase if
    a. Preposition is **"** *Main verb+ to + NP "* → Karam Vibhakti
    b. Preposition is *"by, with, from"* → Karan Vibhakti
    c. Preposition is *"for, to + Vinf"* → Sampradaan Vibhakti
    d. Preposition is *"from*, by*"* → Apadaan Vibhakti
    e. Preposition is *"of, to*"* → Sambandh Vibhakti
    f. Preposition is *"at, in, on, above"* → Adhikaran Vibhakti

from* => 'from' when used with some special verbs that indicate separations such as fell, break or some phrases as fell down etc. then it is categorized as Apadaan Vibhakti else it is Karan Vibhakti.

by* => 'by' when used with some special verbs that indicate separations such as fell or some phrases as letting off etc. then it is categorized as Apadaan Vibhakti else it is Karan Vibhakti.

to* => 'to' when used in the form other than as explained in 'a' and 'c' then it is Sambandh Vibhakti.

We have categorized some prepositions for identifying Vibhaktis/Case roles. In a similar manner this categorization of prepositions can be enhanced by working on more prepositions such as compound prepositions, phrase prepositions.

For identification of Verbs

1. Verbs or verb phrases in the sentence represent actions.

For identification of States

1. Some sentences represent state rather than actions; the state is identified as property of the subject.

For identification of Other Elements

1. The conditional sentences impose restrictions on either the verbs or the property. The 'if' clause or 'when' clause of such sentences is added to all the relations.
2. The quantifiers are added as restrictions to the noun/noun phrase that will be further identified as concepts in the construction of ontology.

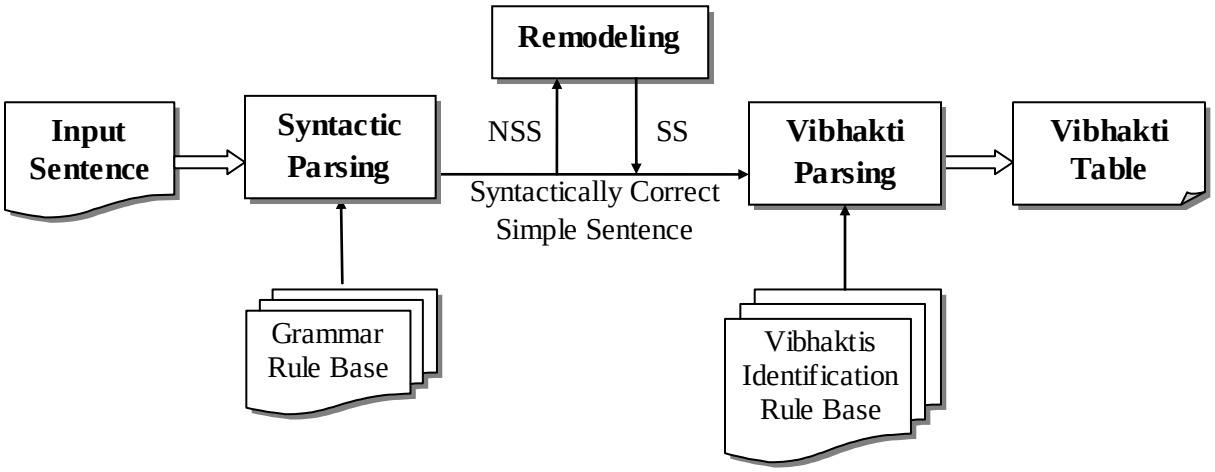
6.0 FORMATION OF VIBHAKTI TABLE

The Vibhakti Parser generates the Vibhakti Table of the input document on applying vibhakti parsing rules on syntactically correct simple sentences. Vibhakti Table has columns for Verb of the sentence, one for property of Kartaa in the sentence, seven for Vibhaktis/case roles of sentence. Using the above defined rules, Vibhakti Parser frames a Vibhakti Table for given text/document.

6.1 Steps for Framing Vibhakti Table

1. Each sentence is processed for syntactic correctness by using Production rules defined above in Syntactic Parsing section.
    a. If the parsed sentence (after remodeling, if any) is valid in grammatical sense then it undergoes Vibhakti Parsing.
    b. Else Syntactic Parsing is interrupted and the subsequent sentence is treated as the next input for parsing.
2. Each syntactically valid simple sentence is scanned for identifying noun phrases, verbs or prepositional phrases. As the Parser encounters any one of these then using Vibhakti Parsing rules, Vibhaktis/case roles, verbs and properties are determined.
3. The determined vibhaktis, verbs and properties are simultaneously fed into the respective cell of Vibhakti Table.

The pictorial representation of Vibhakti Parser can be delineated.



SS → Simple Sentence
NSS → Non Simple Sentence

Vibhakti Parser

6.2 Examples

Based on the method explained for identifying the vibhaktis in section 1.4, the sentences can be categorized mainly into three categories.

Category 1 includes those sentences which represent 'states' only and not 'action'.

Category 2 includes the simple sentences which represent action.

Category 3 is of non simple sentences which are remodeled first to form into simple sentences before identifying vibhaktis.

Parser traces the sentences to extract the Vibhaktis/Case roles, verbs and properties of the sentence. These determined vibhaktis, verbs and properties are then represented into tabular format that is Vibhakti Table which is shown below with the following examples.

Category 1

- The apple is sweet to eat.

Vibhakti/Case Role Table

S. No.	***Verb***	***Karta a***	***Kara m***	***Karan***	***Sampr adan***	***Apada n***	***Samb andh***	***Adhik aran***	***Prope rty***
1	Is	The apple							sweet to eat

Category 2

- The lecture was focused on the problem of unemployment.

Vibhakti/Case Role Table

S. No.	***Verb***	***Karta a***	***Kara m***	***Karan***	***Sampr adan***	***Apa dan***	***Samba ndh***	***Adhik aran***	***Prope rty***
1	Was focused	The lecture					of unempl oyment	on the proble m	

Category 3

- Ram is a good boy and follows the example of the great people.
    - Ram is a good boy
    - Ram follows the example of the great people.

Vibhakti/Case Role Table

S. No.	*Verb*	*Karta a*	*Kara m*	*Karan*	*Sampr adan*	*Apada n*	*Samb andh*	*Adhik aran*	*Prope rty*
1	Is	Ram							a good boy
2	Follows	Ram	the examp le				of the great people		

7.0 CONCLUSION

Thus the Vibhakti Parser checks the syntax and identifies vibhaktis/case roles of input document. The Vibhakti Parser builds the Vibhakti Table in the light of vibhakti parsing rules. All the rules used for this purpose has been explained. The contents of the Vibhakti Table will be used to determine semantic relationships between concepts, properties and constraints on relations [6]. Thus, the automatically generated Vibhakti Table is used to understand the English language automatically.

8.0 REFERENCES

1. Basic English Sentence Structures, http://www.scientificpsychic.com/grammar/enggram3.html.
2. Modern English Grammar, http://papyr.com/hypertextbooks/grammar/.
3. Wren and Martin, 1996, *"High school English Grammar and Composition"*.
4. Paninian Grammar Framework Applied to English, http://www.iiit.ac.in/~sangal/files/papers/pan_eng_98salr.pdf.
5. Sanskrit Grammar: Noun Cases, http://www.everything2.com/?node=Sanskrit+Grammar%3A+Noun+Cases.
6. Christiane Fellbaum, 1998, *"WordNet- An Electronic Lexical Database*". pp. 23-43, 69-99, 131-137.