GRAMMAR FOR CHECKING THE SYNTACTIC VALIDITY OF ENGLISH SENTENCE

P. G. Magdum, D. V. Kodavade

Abstract-- We describe a Context Free Grammar (CFG) for English language and hence we propose an English sentence syntax checker based on the grammar. Our approach is very much general to apply in English Sentences and the method is well accepted for parsing a language of a grammar. Using the proposed grammar sentence is derived by using top down parsing and parse tree is constructed for valid English sentences. If a natural language can be successfully parsed then grammar checking from this language becomes possible. The proposed scheme is based on Top down parsing method and we have avoided the left recursion of the CFG using the idea of left factoring.

Index terms- Context Free Grammar (CFG), Left Recursion, Top down and Bottom up Parsing.

I. INTRODUCTION

Parsing is the process of using grammar rules to determine whether a sentence is legal, and to obtain its syntactical structure. Tree structure provides two information viz. it divides the sentence into constituents (in English, these are called phrases) and it puts them into categories (Noun Phrase, Verb Phrase, etc). To process any natural language, parsing is the fundamental problem for both machines and humans. In general, the parsing problem includes the definition of an algorithm to map any input sentence to its associated syntactic tree structure [1]. A parser analyzes the sequence of symbols presented to it based on the grammar [2]. Natural language applications namely Information Extraction, Machine Translation, and Speech Recognition, need to have an accurate parser [3]. Parsing natural language text is more difficult than the computer languages such as compiler and word processor because the grammars for natural languages are complex, ambiguous and infinite number of vocabulary. For a syntax based grammar checking the sentence is completely parsed to check the correctness of it. If the syntactic parsing fails, the text is considered incorrect. On the other hand, for statistics based approach, Parts Of Speech (POS) tag sequences are prepared from an annotated corpus, and hence the frequency and the probability[4]. The text is considered correct if the POS-tagged text contains POS sequences with frequencies higher than some threshold [5]. Natural languages like English and even Hindi are rapidly progressing as far as work done in processing by computers is concerned.

In this paper, we proposed a context free grammar for the English language. We have adopted the top down parsing scheme and avoided the problem of left recursion using left factoring for the proposed grammar. English grammar has huge amount of forms and rules. We believe the proposed grammar can be applicable to any forms of English sentences and can be used as grammar checker.

A rule based English parser has been proposed in [1] that handles semantics as well as POS identification from English sentences and ease the task of handling semantic issues in machine translation. The system is based on analyzing an input sentence and converting into a structural representation. A parsing methodology for English natural language sentences is proposed in [1] and shows how phrase structure rules can be implemented by top-down and bottom-up parsing approach to parse simple sentences of English. A comprehensive approach for English syntax analysis was developed [1] where a formal language is defined as a set of strings. Each string is a concatenation of terminal symbols.

II. A PARSING SCHEME FOR ENGLISH GRAMMAR RECOGNITION

A parsing is an efficient way of implementing recursive decent parsing by handling the stack of activation record. The predictive parser has an input, a stack, a parse table and output. The input contains the string to be parsed or checked, followed by a $, the right end marker.

<table>
<thead>
<tr>
<th>Tag name(Symbol)</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determinator</td>
<td>A, an, the</td>
</tr>
<tr>
<td>Noun</td>
<td>Bus, home</td>
</tr>
<tr>
<td>Pronoun</td>
<td>He, she</td>
</tr>
<tr>
<td>Adjectives</td>
<td>Beautiful,</td>
</tr>
<tr>
<td>Adverb</td>
<td>Slowly, fast</td>
</tr>
<tr>
<td>Verb</td>
<td>Run, keep</td>
</tr>
<tr>
<td>Auxiliary verbs</td>
<td>Is ,was</td>
</tr>
<tr>
<td>Conjunction</td>
<td>And, but</td>
</tr>
</tbody>
</table>

Table 1: Tag set Description for English grammar

A. ENGLISH GRAMMAR DESIGN

Once constituents have been identified, the productions for Context Free Grammar (CFG) are developed for English
sentence structures. As English grammar has different forms, the same production term can be used only by reorganizing the in the grammar.

B. CONTEXT FREE GRAMMAR

A context-free grammar (CFG) is a set of recursive rewriting rules (or productions) used to generate patterns of strings.

A CFG consists of the following components:

- A set of terminal symbols, which are the characters of the alphabet that appear in the strings generated by the grammar.
- A set of nonterminal symbols, which are placeholders for patterns of terminal symbols that can be generated by the nonterminal symbols.
- A set of productions, which are rules for replacing (or rewriting) nonterminal symbols (on the left side of the production) in a string with other nonterminal or terminal symbols (on the right side of the production).
- A start symbol, which is a special nonterminal symbol that appears in the initial string generated by the grammar.

To generate a string of terminal symbols from a CFG, we:

- Begin with a string consisting of the start symbol;
- Apply one of the productions with the start symbol on the left hand size, replacing the start symbol with the right hand side of the production;
- Repeat the process of selecting nonterminal symbols in the string, and replacing them with the right hand side of some corresponding production, until all non-terminals have been replaced by terminal symbols.

C. LEFT FACTORING

The parser generated from this kind of grammar is not efficient as it requires backtracking. To remove the ambiguity from the grammar we have used the idea of left factoring and reconstruct the grammar productions. Left factoring is a grammar transformation useful for producing a grammar suitable for predictive parsing. The basic idea is that when it is not clear which of the productions are to use to expand a non terminal then it can defer to take decision until we get an input to expand it. In general, if we have productions of form

\[ A \rightarrow \alpha\beta_1 | \alpha\beta_2 \]

We left factored productions by getting the input \( \alpha \) and break it as follows

\[ A \rightarrow \alpha A', A' \rightarrow \beta_1 | \beta_2 \]

S \rightarrow NP.VP
NP \rightarrow a NP1.VP4|pronoun.NP4| the.NP6|an.NP7| propernoun.NP3| I|noun
NP1 \rightarrow noun|adjective.NP2
NP2 \rightarrow noun
NP3 \rightarrow conjunction.NP5| e
NP4 \rightarrow conjunction.NP5| noun|e
NP5 \rightarrow noun|propernoun|noun
NP6 \rightarrow propernoun.NP4| adjective.NP2
NP7 \rightarrow adjective1.NP2
VP \rightarrow verb1.VP'| verb2.vp' | aux31.VP3| aux32.VP6| aux21.VP4| aux22.VP9| aux11.VP5
VP \rightarrow aux11.VP7|adverb.VP6| adverb.VP6
VP \rightarrow NP1.VP2| adverb.VP2| PP.NP|e |pronoun
VP1 \rightarrow adjective.NP2
VP2 \rightarrow PP.NP| e
VP3 \rightarrow verb4.VP'| adverb.VP6| pronoun1.VP1
VP4 \rightarrow verb1.VP'| be.VP6| aux11.VP7| have.VP8
VP5 \rightarrow verb3.VP'| been.VP6
VP6 \rightarrow verb4.VP'
VP7 \rightarrow verb3.vp'
VP8 \rightarrow been.VP6
VP9 \rightarrow be.VP6
PP \rightarrow preposition

Fig 1. Left factored grammar

Some abbreviation used:-
Verb1: form of verb in present tense.
Verb2: form of verb in past tense.
Verb3: form of verb in future tense.
Verb4: form of verb in continuous tense.
Aux11: auxiliary verbs like have, has.
Aux12: auxiliary verbs like had.
Aux21: auxiliary verbs can, will, shall, may.
Aux22: auxiliary verbs could, would, should, might.
Aux31: auxiliary verbs like am, is, are.
Aux32: auxiliary verbs like was, were.
Pronoun1: pronoun like my, your.

III. PARSE TREE GENERATION

A parse tree for a grammar G is a tree where the root is the start symbol for G, the interior nodes are the non terminals of G and the leaf nodes are the terminal symbols of G. The children of a node T (from left to right) correspond to the symbols on the right hand side of some production for T in G. Every terminal string generated by a grammar has a corresponding parse tree and every valid parse tree represents a string generated by the grammar.

IV. EXPERIMENTAL RESULT

We will test two examples and check its syntax using grammar rules which are designed. Consider the first example as:
Fig. 2. Parse tree for English parser on input “the old man walking on the road”.

The second example as:

Fig 3. Parse tree for English parser on input “ram going to home”.

The above grammar gives result for many English sentences which had been examined.

V. CONCLUSION

In this paper we describe a context free grammar for English language and hence we check the syntactic validity of given sentence based on that grammar. Our approach is very much general to apply in English Sentences and the method is well accepted for syntactic validating a language of a grammar. The structural representation that has been built can cover the maximum simple sentences. But there are some sentences composed of how, why, what and starting with verb are beyond the scope of this paper. Also mixed sentences are of out of the discussion. But further increasing and modifying the production rule it can be possible to remove the above limitations. We believe the proposed method can be applied to check most of the English grammar to check syntax of English language.

REFERENCES

Journal Papers:

Books:

Prof. P. G. Magdum received his B.E. Computer Engineering and currently pursuing M.Tect. in Computer Science and Technology, Shivaji University, Kolhapur and his main research interests in Natural Language Processing (NLP). Currently he is servicing as a Assistant Professor in RMCET, Devrukh (Maharashtra).

Prof. D. V. Kodavade received his M. E. in Electronics Specialization in Computer. Currently pursuing his PhD. His main research interests in Natural Language Processing (NLP) AND ALGORITHMS. Currently he is servicing as Associate Professor and HOD, Department of Computer Science and Engineering in DKTE Ichalkaranji (Maharashtra)