Ambiguity of Context Free Grammar Using the CYK Algorithm

R.C. Dharmik
Asso. Professor,
Deptt. of IT, YCCE, Nagpur
{raj_dharmik@yahoo.com}

R. S. Bhanuse
Asstt. Professor,
Deptt. of IT, YCCE, Nagpur
{roshanbhanuse15@gmail.com}

A.D. Gaikwad
Asstt. Professor,
Deptt. of IT, YCCE, Nagpur
{amolgaikwad.ag@gmail.com}

Abstract

The syntax analysis phase of a Compiler is to check syntactic structure of Programming Language construct using Context Free Grammar. Either by using Top-Down or Bottom-Up parsing technique to parse string of a given language. The string of a Language is successfully parsed by parser of Context Free Grammar then that string is syntactically correct. In this paper CYK algorithm is membership algorithm which gives string is member of language generated by Context Free Grammar or not. We have found out the Context Free Grammar is ambiguous or not using CYK algorithm.

Keyword: Syntax analysis, Context Free Grammar, Parsing, CYK algorithm, ambiguous

1. Introduction

The Context Free Grammar is represented by G= (V, T, P, S) Where V is s finite set of Variable/Non-terminal symbols, T is a finite set of terminal symbols, P is a set of rules/productions and S is a start symbol of a grammar. The Context Free Grammar is used to recognize the programming language construct using top-down and bottom-up parsing techniques. Basically the CYK algorithm is used to check or test the string whether the string belongs to the given language or not (i.e. the given string, is the member of the given language). We can describe membership of a string w in a CFL L. There is an efficient technique based on the idea of "Dynamic Programming "which may known as "Table Filling Algorithm" or "Tabulation". This algorithm known as CYK Algorithm (**i.e. Cocke-Younger-Kasami**) [1].

The algorithm works only if the grammar is in Chomsky normal form **(CNF)** and succeeds by breaking one problem into a sequence of smaller one. Compare at most n pairs of previously computed sets [2]:

$$X_{i,j} = (X_{i,i} , X_{i+1, j}), \ (X_{i,i+1} , X_{i+2, j}), (X_{i,i+2} , X_{i+3, j}), \text{-----} , (X_{i,j-1} , X_{j, j})$$

CYK Triangular table:

$X_{1,5}$				
$X_{1,4}$	$X_{2,5}$			
$X_{1,3}$	$X_{2,4}$	$X_{3,5}$		
$X_{1,2}$	$X_{2,3}$	$X_{3,4}$	$X_{4,5}$	
$X_{1,1}$	$X_{2,2}$	$X_{3,3}$	$X_{4,4}$	$X_{5,5}$
W1	W2	W3	W4	W5

978-1-4673-9214-3/16/$31.00 © 2016 IEEE

2. Ambiguous Context Free Grammar

The Context Free Grammar is said to be ambiguous if there is more than one way to generate given string of a language from a given grammar or having two derivation/parse trees for the string generating from the given grammar. Let the Context Free Grammar is G = (V, T, P, S)

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow id$

The string of a language is w= id + id * id

Using left most derivation for generation of string from the given grammar is [2]

1) $E \rightarrow E + E$
   $\rightarrow id + E$
   $\rightarrow id + E * E$
   $\rightarrow id + id * E$
   $\rightarrow id + id * id$

2) $E \rightarrow E * E$
   $\rightarrow E + E * E$
   $\rightarrow id + E * E$
   $\rightarrow id + id * E$
   $\rightarrow id + id * id$

There are two ways to generate the string "id + id * id" from a given grammar, then the given grammar is said to be ambiguous.

Fig 1 and Fig 2 are the two Derivation Trees/Parse Trees for generating the same given string from the given grammar

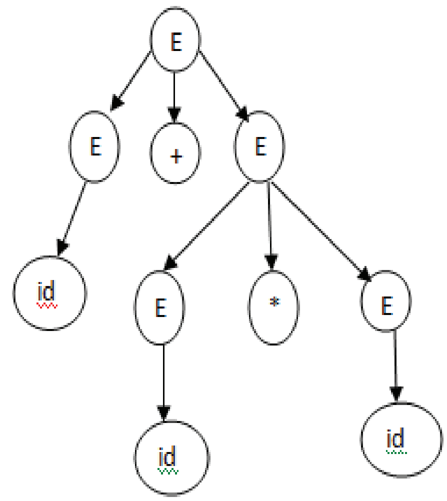

Fig 1

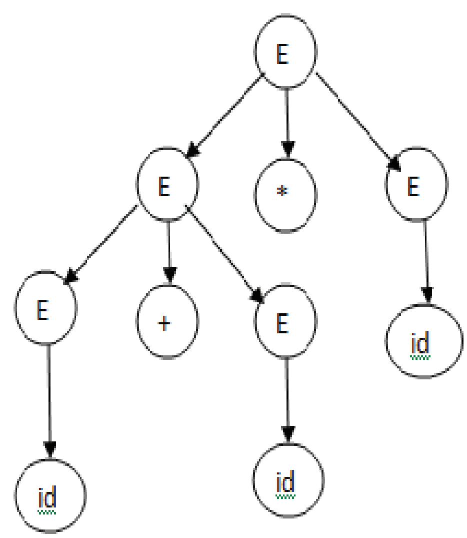

Fig 2

The Context Free Grammar is ambiguous.

2.1 Removal of Ambiguity of CFG

The ambiguous grammar is converted into unambiguous means only one way to generate the given string of language from the given grammar or having only one derivation tree for a string from the given grammar[2].

$E \rightarrow E + E$

The two same non-terminal symbols appears on right hand side of a production, replace one non-terminal by any other new non-terminal symbol T, we get production as

$E \rightarrow E + T \mid T$

Each non-terminal symbol E is replace by T

$E \rightarrow E * E$

$T \rightarrow T * T$

again two same non-terminal symbols T appears on right hand side of a production, replace one non-terminal by any other new non-terminal symbol F, we get production as

$T \rightarrow T * F \mid F$

$E \rightarrow id$

Non-terminal E is replace by T and T is replace by F, we get production as

$F \rightarrow id$

The unambiguous Context Free Grammar is

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow id$

2.2 Chomsky Normal Form of Grammar

The Context Free Grammar is said to be Chomsky Normal Form of Grammar only if the right hand side of a production must have two non-terminal symbols or single terminal symbol then that grammar is CNF grammar

$S \rightarrow AB \mid a$

For every CFG, there is an equivalent grammar G in Chomsky Normal Form[3]

Construction of grammar in CNF

Step 1: Eliminate Null Productions and Unit Productions

Step 2: Eliminate terminals on right hand side of productions as follows

i) All the productions in P of the form $A \rightarrow a$ and $F \rightarrow BC$ are included

ii) Consider $A \rightarrow w1w2$ ----w_n will some terminal on right hand side then w_i is replace by any new non-terminal symbol, add new production as $X \rightarrow wi$
Repeat same for all terminal symbols

Step 3: Restricting the number of non-terminal symbol on the right hand side as follows

i) Consider $A \rightarrow A1A2$------An
Introduce new non-terminal Symbol $T \rightarrow A1A2$, only two Non-terminals on the right hand side of production.

3. CYK Algorithm

The **Cocke–Younger–Kasami algorithm** (CYK) is a parsing algorithm for Context Free Grammar. The structure of the rules/productions of a CFG is in a Chomsky Normal Form (CNF). CYK uses a dynamic Programming or table filling algorithm. The CYK algorithm is used to find whether the given string is a member of grammar [1][2].

Let w be the string then w is in L (G)

begin

For i = 1 to n do

Vi1 = {A → a is a production and the ith symbol of x is a}

For j = 2 to n do

For i = 1 to n-j+1 do

begin

Vij = Ø;

For k = 1 to j-1 do

Vij = Vij Ú {A → BC is a production, B is in Vik and C is in Vi+k, j-k}

end

4. Ambiguity of CFG using CYK

Example 1:

The Context Free Grammar is

$E \rightarrow E + E$

$E \rightarrow E * E$

$E \rightarrow id$

String: "id + id * id"

Converted into CNF Grammar is

$E \rightarrow E\ R_3$

$E \rightarrow E\ R_4$

$R_3 \rightarrow R_1\ E$

$R_4 \rightarrow R_2\ E$

$R_1 \rightarrow +$

$R_2 \rightarrow *$

$E \rightarrow id$

{E,E} $X_{1,5}$				
{Ø} $X_{1,4}$	{ R_3 } $X_{2,5}$			
{E} $X_{1,3}$	{Ø} $X_{2,4}$	{E} $X_{3,5}$		
{Ø} $X_{1,2}$	{ R_3 } $X_{2,3}$	{Ø} $X_{3,4}$	{ R_4 } $X_{4,5}$	
{E} $X_{1,1}$	{ R_1 } $X_{2,2}$	{E} $X_{3,3}$	{ R_2 } $X_{4,4}$	{ E} $X_{5,5}$
id	**+**	**id**	*****	**id**

$$X_{1,2} = (X_{i,i}, X_{i+1,j}) = (X_{1,1}, X_{2,2}) = \{E\}\{R_1\}$$

$$= \{E\ R_1\} = \{Ø\}$$

$$X_{2,3} = (X_{2,2}, X_{3,3}) = \{R_1\}\{E\} = \{R_1\ E\} = \{R_3\}$$

$$X_{3,4} = (X_{3,3}, X_{4,4}) = \{E\}\{R_2\} = \{E\ R_2\} = \{Ø\}$$

$$X_{4,5} = (X_{4,4}, X_{5,5}) = \{R_2\}\{E\} = \{R_2\ E\} = \{R_4\}$$

$$X_{1,3} = (X_{1,1}, X_{2,3})\ U\ (X_{1,2}, X_{3,3})$$

$$= (\{E\}\{R_3\})\ U\ (\{Ø\}\{E\} = \{E\ R_3\}\ U\ \{E\}$$

$$= \{E\}$$

$$X_{2,4} = (X_{2,2}, X_{3,4})\ U\ (X_{2,3}, X_{4,4})$$

$$= (\{R_1\}\{Ø\})\ U\ (\{R_3\}\{R_2\})$$

$$= \{R_1\}\ U\ (\{R_3\ R_2\})$$

$$= \{Ø\}$$

$$X_{3,5} = (X_{3,3}, X_{4,5})\ U\ (X_{3,4}, X_{5,5})$$

= ({E} {R_4}) U ({ Ø } {E})

= { E R_4 , E}

= { E }

$X_{1,4}$ = ($X_{1,1}$, $X_{2,4}$) U ($X_{1,2}$, $X_{3,4}$) U ($X_{1,3}$, $X_{4,4}$)

= ({E}, { Ø }) U ({ Ø }, { Ø }) U ({E} , {R_2})

= { E } {E, R_2 } = { Ø }

$X_{2,5}$ = ($X_{2,2}$, $X_{3,5}$) U ($X_{2,3}$, $X_{4,5}$) U ($X_{2,4}$, $X_{5,5}$)

= ({R_1}, { E }) U ({ R_3 }, { R_4}) U ({ Ø } , {E})

= { R_1 E , R_3 R_4 , E } = { R_3}

$X_{1,5}$ = ($X_{1,1}$, $X_{2,5}$) U ($X_{1,2}$, $X_{3,5}$) U ($X_{1,3}$, $X_{4,5}$) U ($X_{1,4}$, $X_{5,5}$)

= ({E} { R_3 }) U ({ Ø }, {E}) U ({ E } , {R_4} U { Ø} {E})

= { ER_3 , E, ER_4 , E }

= { E, E}

Grammar is Ambiguous as it contains two times Start Symbol 'E' in the Cell X_1, n ('n' is length of a string)

Example 2:

Context Free Grammar is

S → iC t S | i C t S e S | a

C → b

String: " i b t i b t a e a"

CNF Grammar is

S → R_4 R_5 | R_4 R_7 | a

R_4 → R_1 C

R_5 → R_2 S

R_6 → R_3 S

R_7 → R_5 R_6

R_1 → i

R_2 → t

R_3 → e

C → b

{S,S} $X_{1,9}$								
{ Ø } $X_{1,8}$	{ Ø } $X_{2,9}$							
{S} $X_{1,7}$	{ Ø } $X_{2,8}$	{ R_5 R_7} $X_{3,9}$						
{ Ø } $X_{1,6}$	{ Ø } $X_{2,7}$	{ Ø } $X_{3,8}$	{S} $X_{4,9}$					
{ Ø } $X_{1,5}$	{ Ø } $X_{2,6}$	{ R_5} $X_{3,7}$	{ Ø } $X_{4,8}$	{ Ø } $X_{5,9}$				
{ Ø } $X_{1,4}$	{ Ø } $X_{2,5}$	{ Ø } $X_{3,6}$	{S} $X_{4,7}$	{ Ø } $X_{5,8}$	{ R_7} $X_{6,9}$			
{ Ø } $X_{1,3}$	{ Ø } $X_{2,4}$	{ Ø } $X_{3,5}$	{ Ø } $X_{4,6}$	{ Ø } $X_{5,7}$	{ Ø } $X_{6,8}$	{ Ø } $X_{7,9}$		
{ R_4} $X_{1,2}$	{Ø} $X_{2,3}$	{ Ø } $X_{3,4}$	{ R_4} $X_{4,5}$	{ Ø } $X_{5,6}$	{ R_5} $X_{6,7}$	{ Ø } $X_{7,8}$	{ R_6} $X_{8,9}$	
{ R_1} $X_{1,1}$	{C} $X_{2,2}$	{ R_2} $X_{3,3}$	{ R_1} $X_{4,4}$	{C} $X_{5,5}$	{ R_2} $X_{6,6}$	{S} $X_{7,7}$	{ R_3} $X_{8,8}$	{S} $X_{9,9}$
i	b	t	i	b	t	a	e	a

Grammar is Ambiguous as it contains two times Start Symbol 'S' in the Cell X_1, n ('n' is length of a string)

5. Conclusion

The CYK algorithm is used to check the given string of a language is member of a grammar. The given string is parsed using dynamic programming or table filling algorithm. If start symbol of a grammar is appeared in top cell of first column of a triangular table then the string is member of language generated by a grammar. CYK algorithm is only the membership algorithm. In this paper, we have found out the given Context Free Grammar is ambiguous or not using CYK algorithm. If the start symbol of a grammar is appeared two times in top cell of first column of a triangular table ($X_{1, n}$) then the given Context Free Grammar is ambiguous.

6. References

1. Shamshad Ali, "CYK Algorithm", International Journal of Scientific Research Engineering & Technology (IJSRET), Volume 1 Issue 5 pp 001-004 August 2012
2. Hopcroft, Ullman, "Introduction to Automata Theory, Languages and Computation", Pearson Education.
3. K.L.P. Mishra and N. Chandrasekaran, "Theory of Computer Science: Automata, Languages and Computation", PHI
4. Yuqiang Sun, Lei Zhou, Qiwei He, Yuwan Gu, Liang Jia, "Algorithm of Word-Lattice Parsing Based on Improved CYK-algorithm", 2010 International Conference on Web Information Systems and Mining
5. Nathan Bodenstab, "Efficient Implementation of the CYK Algorithm".
6. Xinying Songy, Shilin Dingx , Chin-Yew Linz, "Better Binarization for the CKY Parsing"
7. Zsolt Tóth, László Kovács, "CFG Extension for META Framework", INES 2012 , IEEE 16th International Conference on Intelligent Engineering Systems , June 13–15, 2012, Lisbon, Portugal
8. Xiao Yang, Jiancheng Wan, Ling Zhang, "Arithmetic Computing Based Chinese Automatic Parsing Method", Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing
9. Huong Thanh Le, Lam Ba Do, Nhung Thi Pham," Efficient Syntactic Parsing with Beam Search".
10. Gend Lal Prajapati, Aditya Jain, Mayank Khandelwal, Pooja Nema, Priyanka Shukla, "On The Inference of Context-Free Grammars Based On Bottom-Up Parsing and Search", Second International Conference on Emerging Trends in Engineering and Technology, ICETET-09
11. Paweł Skórzewski, "Effective natural language parsing with probabilistic Grammars", Proceedings of the International Multiconference on Computer Science and Information Technology pp. 501–504ISBN 978-83-60810-27-9ISSN 1896-7094
12. Richard E. Stearns and Harry B. Hunt, "On the equivalence and Containment Problems for unambiguous Regular Expression, Grammar and Automata,CH1695-6/81/0000/0074$OO.75 1981 IEEE.