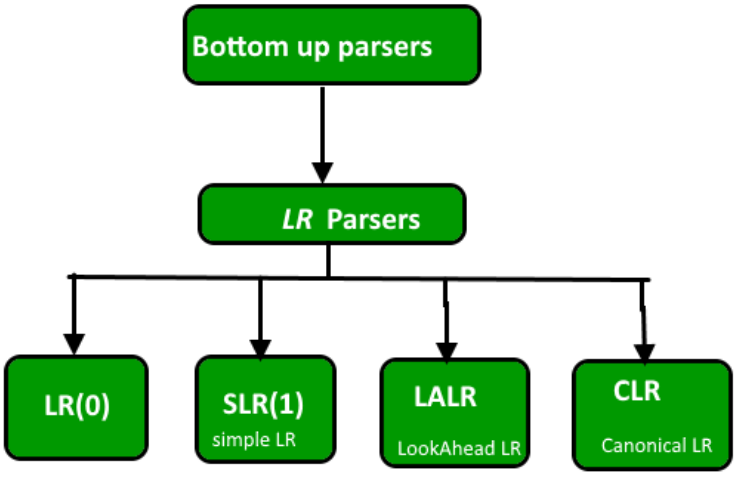
Experiment No:8

AIM: To implement Bottom Up Parser.

THEORY: Build the parse tree from leaves to root. Bottom-up parsing can be defined as an attempt to reduce the input string w to the start symbol of grammar by tracing out the rightmost derivations of w in reverse.

Classification of bottom up parsers



A general shift reduce parsing is LR parsing. The L stands for scanning the input from left to right and R stands for constructing a rightmost derivation in reverse. The bottom-up name comes from the concept of a parse tree, in which the most detailed parts are at the bottom of the (upside-down) tree, and larger structures composed from them are in successively higher layers, until at the top or "root" of the tree a single unit describes the entire input stream. A bottom-up parse discovers and processes that tree starting from the bottom left end, and incrementally works its way upwards and rightwards. [1] A parser may act on the structure hierarchy's low, mid, and highest levels without ever creating an actual data tree; the tree is then merely implicit in the parser's actions.

Here we describe a skeleton algorithm of an LR parser:

1. token = next_token()
2. repeat forever

   s = top of stack

3. if action[s, token] = “shift si” then
4. PUSH token
5. PUSH si

   token = next_token()

6. else if action[s, token] = “reduce A::= β “ then
7. POP 2 * |β| symbols

   s = top of stack

8. PUSH A
9. PUSH goto[s,A]
10. else if action[s, token] = “accept” then
11. return
12. else

error()

COMPUTING ENVRONMENT

Platform: ubuntu
Programming Language: C / C++ / Java

Expected OUTPUT

Enter Number of productions:4
Enter productions:
E->E+E
E->E*E
E->(E)
E->a
Enter Input:(a+a)*a

Stack	Input	Action
(	a+a)*a	Shifted
(a	+a)*a	Shifted
(E	+a)*a	Reduced
(E+	a)*a	Shifted
(E+a	)*a	Shifted
(E+E	)*a	Reduced
(E	)*a	Reduced
(E)	*a	Shifted
E	*a	Reduced
E*	a	Shifted
E*a		Shifted
E*E		Reduced
E		Reduced

String Accepted

Conclusion: Thus the Bottom up parser(Shift Redused Parser) is implemented.

Viva Voce Questions:

1. What is used in bottom-up parsing?

Answer: Bottom-up parsing can be defined as an attempt to reduce the input string w to the start symbol of grammar by tracing out the rightmost derivations of w in reverse.

2. Why Bottom-up parser is more powerful?

Answer:The LR parser is a non-recursive, shift-reduce, bottom-up parser. It uses a wide class of context-free grammar which makes it the most efficient syntax analysis technique.

3. What is the role of parser?

Answer: The parser obtains a string of tokens from the lexical analyzer and verifies that the string can be the grammar for the source language. It detects and reports any syntax errors and produces a parse tree from which intermediate code can be generated.

4. What is most common type bottom-up parser?

Answer: Shift-reduce parsing is the most commonly used and the most powerful of the bottom-up techniques.