Bottom Up Parsing

19CSE401 Compiler Design

Department of Computer Science

Bottom up Parsing

- Opposite of Top- down parsing.
- Creates parser tree from string until all the token are processed and the start symbol is reached.
- Reverse of the right-most direction.

- Consider the below grammar

  S → aABe

  A → Abc | b

  B → d

- Bottom- up parsing of the string : abbcde
- ➔ a b b c d e

  ➔ a A b c d e

  ➔ a A d e

  ➔ a A B e

  ➔ S

a b b c d e

Bottom-up Parsing

- Use explicit stack to perform a parse
- Simulate rightmost derivation (R) from left (L) to right, thus called LR parsing
- More powerful than top-down parsing
  - Left recursion does not cause problem
- Two actions
  - Shift: take next input token into the stack
  - Reduce: replace a string B on top of stack by a nonterminal A, given a production $A \rightarrow B$

Example of Shift-reduce Parsing

- Grammar

  $S' \rightarrow S$

  $S \rightarrow (S)S \mid \lambda$

- Parsing actions

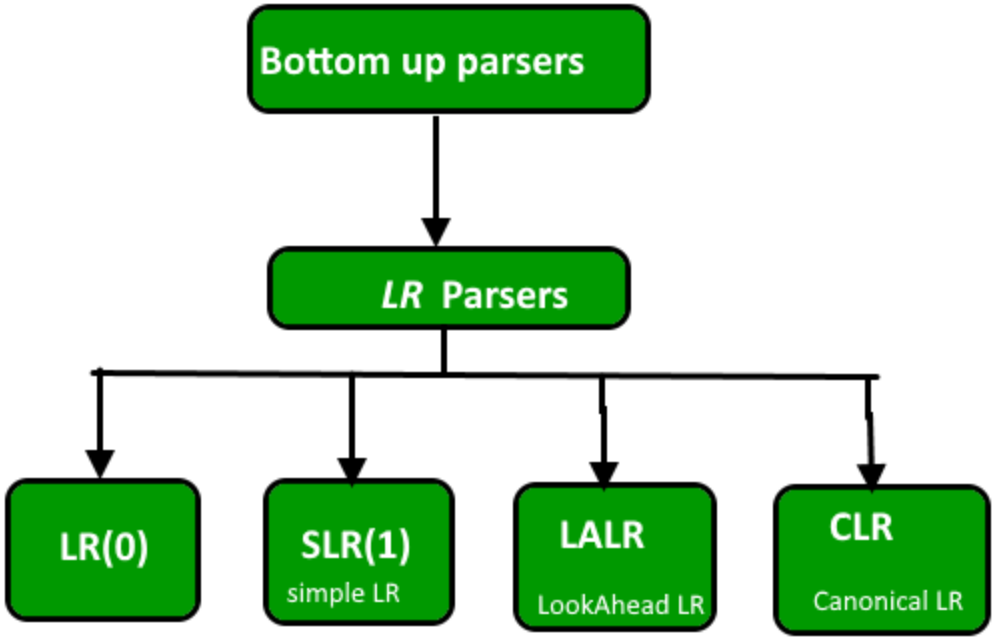
Stack	Input	Action
$	(( )) $	
$ (	( )) $	
$ ((	)) $	
$ ((S	)) $	
$ ((S)	) $	
$ ((S) S	) $	
$ (S	) $	
$ (S)	$	
$ (S) S	$	
$ S	$	

- Reverse of
- rightmost derivation
- from left to right

1 $\Rightarrow$ **(( ))**
2 $\Rightarrow$ **(( ))**
3 $\Rightarrow$ **(( ))**
4 $\Rightarrow$ **((S))**
5 $\Rightarrow$ **((S))**
6 $\Rightarrow$ **((S) S)**
7 $\Rightarrow$ **(S)**
8 $\Rightarrow$ **(S)**
9 $\Rightarrow$ **(S) S**
10 S' $\Rightarrow$ **S**

Shift-reduce parsers

- There are two possible actions:
  - shift and reduce
- Parsing is completed when
  - the input stream is empty and
  - the stack contains only the start symbol
- The grammar must be *augmented*
  - a new start symbol S’ is added
  - a production S’ $\rightarrow$ S is added
    - To make sure that parsing is finished when S’ is on top of stack because S’ never appears on the RHS of any production.


Bottom up parsers
LR Parsers
LR(0)
SLR(1)
simple LR
LALR
LookAhead LR
CLR
Canonical LR

Types of Bottom up

- **Bottom Up Parsers / Shift Reduce Parsers**
  - 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.**