Bottom up Parsing

Bottom up ~~Parsing~~ Parser builds a derivation by working from input sentence back towards the start symbols. Rightmost derivation in reverse order is done in bottomup parsing.

eg:- $S \rightarrow aABe$
$A \rightarrow Abc \mid b$
$B \rightarrow d$

Input String : abbcde

Rightmost Derivation

$S \rightarrow aA\underline{B}e$
$\rightarrow a\underline{A}de \; [B \rightarrow d]$
$\rightarrow a\underline{A}bcde \; [A \rightarrow Abc]$
$\rightarrow abbcde \; [A \rightarrow b]$

Parsing using Bottom up

Input	Production Used
$a\underline{b}bcde$	
$a\underline{Abc}de$	$A \rightarrow b$
$aA\underline{d}e$	$A \rightarrow$ ~~aAd~~ Abc
$aABe$	$B \rightarrow d$
S	$S \rightarrow aABe$

Getting Start Symbol 'S' from Input String. So Input String is acceptable

Dr. Rahul Bhandari
Dept. of CSE

~~Shift Reducing~~

Shift Reduce Parsing

It uses a stack to hold grammar symbols and input buffer to hold string parsed, because handles always appear at the top of the stack i.e. there is no need to look deeper into it.

It has four actions -

① Shift - Next word is shifted into stack until a handle is formed.

② Reduce - Right end of handle is at top of stack. locate left end of handle within the stack. Pop handle off stack and push appropriate LHS.

③ Accept - Stop parsing on successful completion of parse and report success.

④ Error - Call an error reporting/recovery routine.

eg:- Grammar: $E \rightarrow E+E \mid E*E \mid (E) \mid id$

Input String: id + id + id

Stack	input	Action
$	id+id+id$	Shift
$id	+id+id$	Reduce by E→id
$E	+id+id$	Shift
$E+	id+id$	Shift
$E+id	+id$	Reduce by E→id
$E+E	+id$	Reduce by E→E+E
$E	+id$	Shift
$E+	id$	Shift
$E+id	$	Reduce by E→id
$E+E	$	Reduce by E→E+E
$E	$	Accept [Because Starting Symbo