Santu Jana /Roll :cse21/L-146/ Uni roll:10700121127

ASSIGNMENT

1. Which Parser is used for the implementation of recursive descent parsing? Draw the model diagram for that parser. Construct the parsing table for the grammar

E →E+T|T

*T →T*F|F*

F →(E)|id

Also construct the LR (0) Parsing table of the above grammar.

Answer

A grammar is said to be left recursive if it has a non-terminal A such that there is a derivation

A=>A α for some string α. Top-down parsing methods cannot handle left-recursive grammars.

Hence, left recursion can be eliminated as follows:

If there is a production A → A α | β it can be replaced with a sequence of two productions

A → β A' A' → α A' | ε

Without changing the set of strings derivable from A.

Consider the following grammar for arithmetic expressions:

E → E+T | T

T → T*F | F

F → (E) | id

This is not LR(0) Grammar.

First eliminate the left recursion for E as

E → TE'

E' → +TE' | ε

Then eliminate for T as

T → FT'

T'→ *FT' | ε

Thus, the obtained grammar after eliminating left recursion is

E → TE'

E' → +TE' | ε

T → FT'

T' → *FT' | ε

F → (E) | id

	FIRST	FOLLOW
E	{(, id}	{ $,) }
E'	{+, ε }	{ $,) }
T	{(, id}	{ +, $,) }
T'	{*, ε }	{ +, $,) }
F	{(, id}	{ *, +, $,) }

Answer CONT…

	INPUT SYMBOLS					
	+	*	(	)	id	$
E			E →TE’		E →TE’	
E’	E’→+TE’			E’→ ε		E’→ ε
T			T → FT’		T → FT’	
T’	T’ → ε	T’ → *FT’		T’ → ε		T’ → ε
F			F → (E)		F → id	

Stack implementation:

stack	Input	Output
$E	id+id*id $	
$E’T	id+id*id $	E → TE’
$E’T’F	id+id*id $	T → FT’
$E’T’id	id+id*id $	F → id
$E’T’	+id*id $	
$E’	+id*id $	T’ → ε
$E’T+	+id*id $	E’ → +TE’
$E’T	id*id $	
$E’T’F	id*id $	T → FT’
$E’T’id	id*id $	F → id
$E’T’	*id $	
$E’T’F*	*id $	T’ → *FT’
$E’T’F	id $	
$E’T’id	id $	F → id
$E’T’	$	
$E’	$	T’ → ε
$	$	E’ → ε

2. *S→iEtSS'|a*

S'→Es|ε

E→b

Is this grammar LL (1). Describe it.

		FIRST	FOLLOW
S		{i ,a}	[$,b}
S'		{b, ε }	{$,b}
E		{b}	{t}

Parsing Table:

Non-terminal	i	t	a	b	s	$
S	*S→iEtSS'*		S->a			
S'				S'-> ε S'->Es		S'-> ε
E				E->b		

M[S' ,E]=2

So this grammar is not an LL(1) grammar. BeauseM[S', E] has two entries.