جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

قسم الامن السيبراني

DEPARTMENT OF CYBER SECURITY

SUBJECT: COMPUTATION THEORY
CLASS: 3rd

LECTURER: MSC :MUNTATHER AL-MUSSAWEE

LECTURE: (9)

TOP – DOWN/ BOTTOM – UP PARSING

Lecture Seven

Top – Down/ Bottom – Up Parsing

There are two type of derivation:

1- **Top – down:** In this type of derivation, the string will derive from start symbol to leaves such as:

Example: S ⟶ ( ) | (S) | SS

Is the string "(( )( )( ))" accept or not?

S ⟶ (S)
⟶ (SS)
⟶ (SSS)
⟶ (( )SS)
⟶ (( )( )S)
⟶ (( )( )( )) Accept

2- **Bottom – Up:** In this type of derivation, the string will derive from leaves to start symbol. There are many type of methods to derive one of these called ***shift – reduce***.

Shift – Reduce

Shift Reduce parser attempts for the construction of parse in a similar manner as done in bottom up parsing i.e. the parse tree is constructed from leaves (bottom) to the root (up). This parser requires some data structures such as, an input buffer for storing the input string and a stack for storing and accessing the production rules.

Basic Operations

Shift: This involves moving of symbols from input buffer onto the stack.

Reduce: If the handle appears on top of the stack then, replace one element from the left side of derivation grammar with one or more elements from the right side.

Accept: If only start symbol is present in the stack and the input buffer is empty then, the parsing action is called accept. When accept action is obtained, it is means successful parsing is done.

Error: This is the situation in which the parser can neither perform shift action nor reduce action and not even accept action.

✍ دائماً اول عملية نقوم بتنفيذها هي (shift).

Example: E ⟶ E + E | E – E | E ∗ E | E/E | (E) | -E | id

Is the string "a + b ∗ c" accept or not?

id + id ∗ id

Stack	Input Buffer	Parsing Action
$	id + id ∗ id$	Shift
$id	+ id ∗ id$	Reduce E ⟶ id
$E	+ id ∗ id$	Shift
$E +	id ∗ id$	Shift
$E + id	∗ id$	Reduce E ⟶ id
$E + E	∗ id$	Reduce E ⟶ E + E
$E	∗ id$	Shift
$E∗	id$	Shift
$E ∗ id	$	Reduce E ⟶ id
$E ∗ E	$	Reduce E ⟶ E ∗ E
$E	$	Accept

Example: Let G be the grammar below:

S ⟶ aAcBe

A ⟶ Ab | b

B ⟶ d

Is the string "abbcde" accept or not?

Stack	**Input Buffer**	**Parsing Action**
$	abbcde$	Shift
$a	bbcde$	Shift
$ab	bcde$	Reduce A ⟶ b
$aA	bcde$	Shift
$aAb	cde$	Reduce A ⟶ Ab
$aA	cde$	Shift
$aAc	de$	Shift
$aAcd	e$	Reduce B ⟶ d
$aAcB	e$	Shift
$aAcBe	$	Reduce s ⟶ aAcBe
$S	$	Accept

Example: Consider the following grammar:

$S \rightarrow (L) \mid a$

$L \rightarrow L , S \mid S$

Parse the input string "(a , (a , a))" using a shift-reduce parser.

Stack	**Input Buffer**	**Parsing Action**
$	(a , (a , a)) $	Shift
$ (	a , (a , a)) $	Shift
$ (a	, (a , a)) $	Reduce S → a
$ (S	, (a , a)) $	Reduce L → S
$ (L	, (a , a)) $	Shift
$ (L ,	(a , a)) $	Shift
$ (L , (	a , a)) $	Shift
$ (L , (a	, a)) $	Reduce S → a
$ (L , (S	, a)) $	Reduce L → S
$ (L , (L	, a)) $	Shift
$ (L , (L ,	a)) $	Shift
$ (L , (L , a	)) $	Reduce S → a
$ (L , (L , S	)) $	Reduce L → L , S
$ (L , (L	)) $	Shift
$ (L , (L)	) $	Reduce S → (L)
$ (L , S	) $	Reduce L → L , S
$ (L	) $	Shift
$ (L)	$	Reduce S → (L)
$ S	$	Accept

Homework:

1- Let G be the grammar below:

E ⟶ E o E | -E | (E) | id

o ⟶ - | + | ∗ | /

Is the string "A + B ∗(C – D/K)" accept or not?

2- Let G be the grammar below:

E ⟶ E + E | E – E | E ∗ E | E/E | E↑E | (E) | -E | id

Is the string "id ↑ id ↑ (id + id/(id – id)" accept or not?