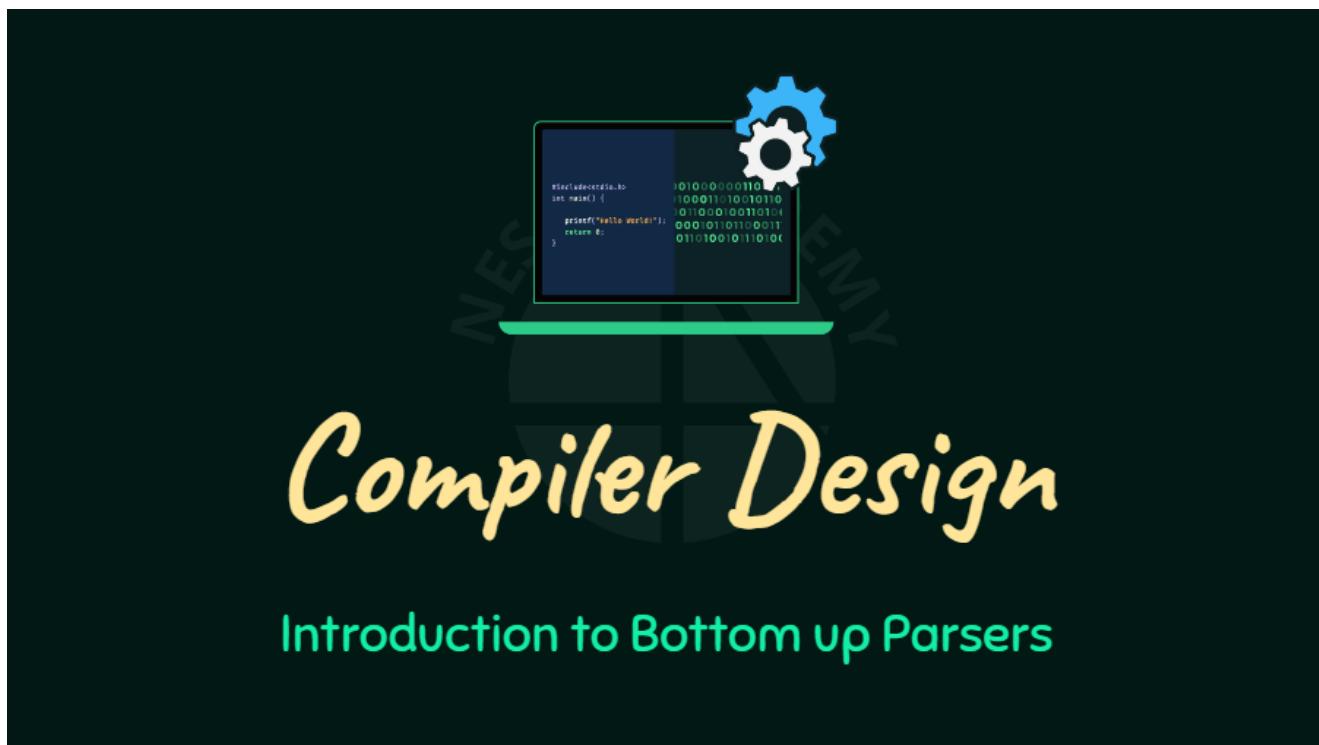


# Bottom-Up Parsers | Neso Academy

🔗 [nesoacademy.org/cs/12-compiler-design/ppts/04-bottom-up-parsers](https://nesoacademy.org/cs/12-compiler-design/ppts/04-bottom-up-parsers)



Bottom-up Parsers Neso Academy CHAPTER-4



Compiler Design Introduction to Bottom up Parsers



## Outcome

- ☆ Understanding the Bottom up approach of parsing.
- ☆ Revisiting the Classification of Parsers.

Outcome ☆ Understanding the Bottom up approach of parsing. ☆ Revisiting the Classification of Parsers.

### Generation of Parse Tree - Bottom up approach:

$$\begin{aligned}S &\rightarrow aABe \\A &\rightarrow Abc \mid b \\B &\rightarrow d\end{aligned}$$

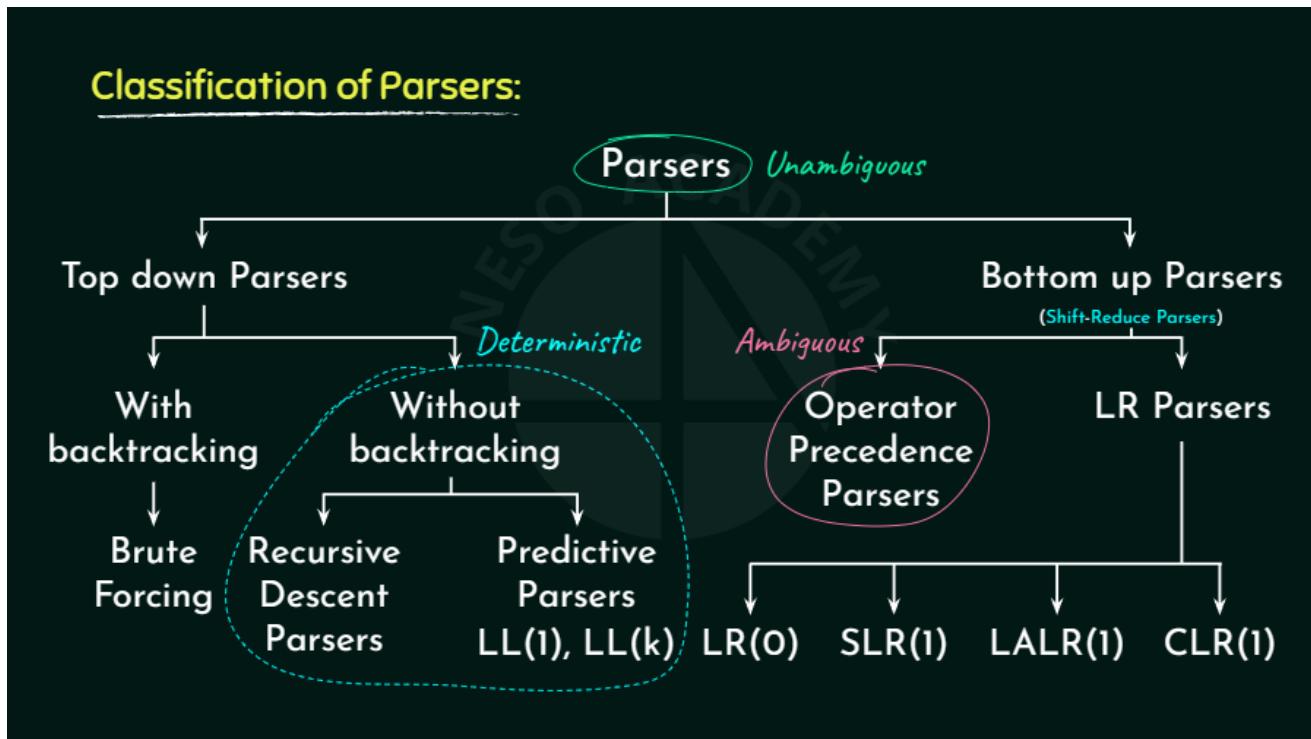
Decision: When to reduce.



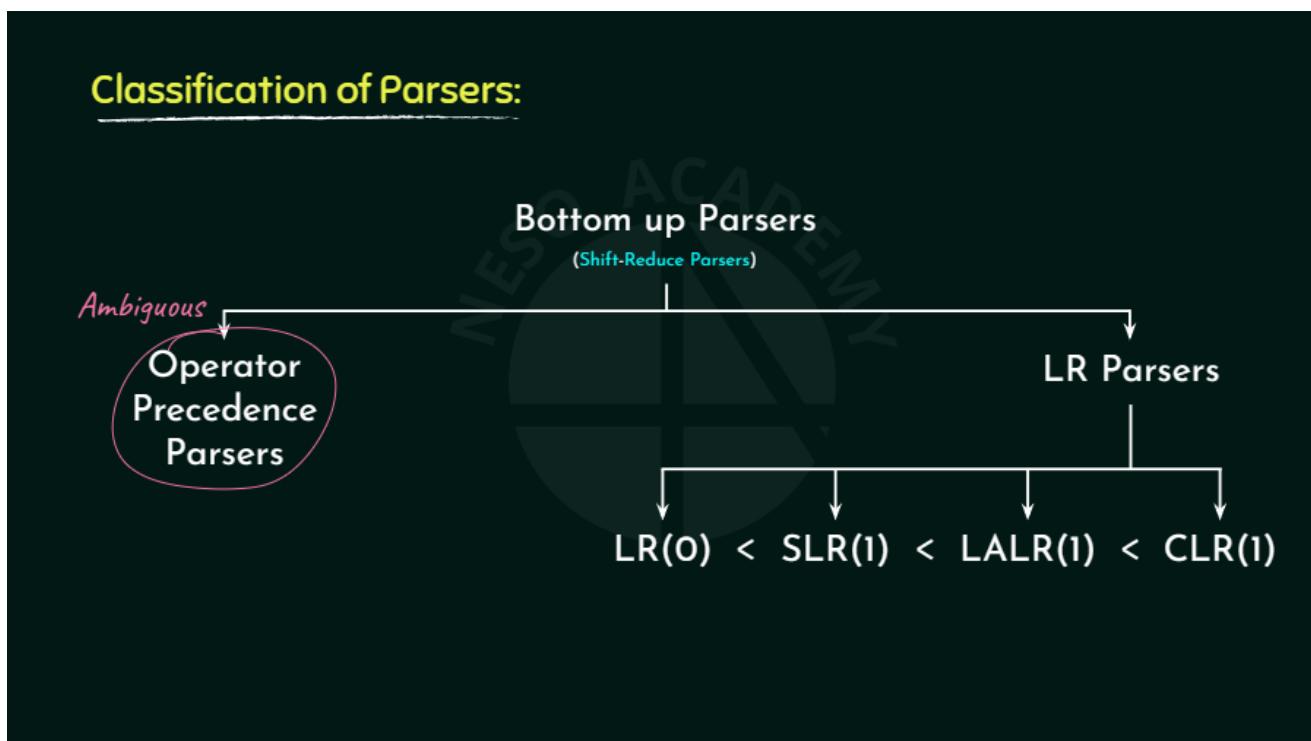
$$\begin{aligned}S &\Rightarrow aABe \\&\Rightarrow aAde \\&\Rightarrow aAbcde \\&\Rightarrow aabcde\end{aligned}$$

(Right most Derivation) - In reverse.

Generation of Parse Tree - Bottom up approach:  
 $S \rightarrow aABe$   $A \rightarrow Abc \mid b$   $B \rightarrow d$   
 $S \Rightarrow aABe \Rightarrow aAde \Rightarrow aAbcde \Rightarrow aabcde$   
(Right most Derivation)- In reverse. Decision: When to reduce.  
aabbcdedAASB



Classification of Parsers:  
 Parsers  
 Top down Parsers  
 Bottom up Parsers (Shift-Reduce Parsers)  
 With backtracking  
 Without backtracking  
 Brute Forcing  
 LR Parsers  
 Operator Precedence Parsers  
 LR(0)  
 CLR(1)  
 LALR(1)  
 SLR(1)  
 Recursive Descent Parsers  
 Predictive Parsers  
 LL(1), LL(k)  
 Unambiguous  
 Ambiguous  
 Deterministic



Ambiguous  
 Classification of Parsers:  
 Bottom up Parsers (Shift-Reduce Parsers)  
 LR Parsers  
 Operator Precedence Parsers  
 LR(0)  
 CLR(1)  
 LALR(1)  
 SLR(1)<<<



# Compiler Design

## Operator Precedence Parsers

Compiler Design Operator Precedence Parsers



### Outcome

- ☆ Understanding the Operator Grammar.
- ☆ How to convert a CFG in Operator Grammar.
- ☆ Generation of Operator Relation Table.
- ☆ Operator Precedence Parsing.

Outcome  
☆Understanding the Operator Grammar.  
☆How to convert a CFG in Operator Grammar.  
☆Generation of Operator Relation Table.  
☆Operator Precedence Parsing.

## Operator Precedence Parser:

1. It is mainly used for mathematical expressions.
2. It processes Operator Grammar.
3. It can handle Ambiguous Grammars.
4. It uses Operator Relation table.

Operator Precedence Parser:  
1. It is mainly used for mathematical expressions.  
2. It processes Operator Grammar.  
3. It can handle Ambiguous Grammars.  
4. It uses Operator Relation table.

## Operator Grammar:

-- Used to define Operators.

Restrictions:

1. No adjacent non-terminals.
2. No epsilon( $\epsilon$ ) productions.

$$\begin{array}{l} E \rightarrow \underline{EA}E \mid id \\ A \rightarrow + \mid * \end{array} \quad \Rightarrow \quad E \rightarrow E + E \mid E * E \mid id$$

Operator Grammar:-- Used to define Operators. Restrictions:  
1. No adjacent non-terminals.  
2. No epsilon( $\epsilon$ ) productions.  
 $E \rightarrow EAE \mid id$   $A \rightarrow + \mid * \rightarrow E \rightarrow E + E \mid E * E \mid id^*$

## Conversion to Operator Grammar:

$$S \rightarrow SAS \mid a$$

$$A \rightarrow bSb \mid b$$

$$S \rightarrow SbSbS \mid SbS \mid a$$

$$A \rightarrow bSb \mid b$$

$S \rightarrow SAS \mid a$   $A \rightarrow bSb \mid b$  Conversion to Operator Grammar:  
 $bSbSISbSI \ a \rightarrow SA \rightarrow bSb \mid bS$

## Construction of Operator Relation Table:

$$E \rightarrow E + E \mid E * E \mid id$$

- $id \rightarrow$  Highest Precedence
- $\$ \rightarrow$  Lowest Precedence

	Precedence	Associativity
$*, /, %$		Left to right
$+, -$		Left to right

	id	+	*	\$
id	--	>	>	>
+	<	>	<	>
*	<	>	>	>
\$	<	<	<	--

 > +  
 Higher Precedence due to Associativity

++Higher Precedence due to Associativity Construction of Operator Relation Table:  
 $E \rightarrow E + E \mid E * E \mid id \mid *, /, \% \mid +, -$   
 •  $id \rightarrow$  Highest Precedence  
 •  $\$ \rightarrow$  Lowest Precedence  
 $\rightarrow id + id * id + id + id / id + id \% id + id + id + id$

## Operator Precedence Parsing:

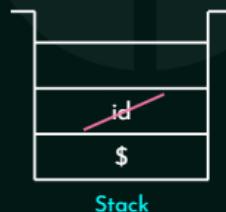
$$E \rightarrow E + E \mid E * E \mid id$$

*Rules:*

1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.
2. If TOS > LookAhead, POP(**Reduce**).



id	+	*	\$
id	--	>	>
+	<	>	<
*	<	>	>
\$	<	<	<
			--



Stack Operator Precedence Parsing: $E \rightarrow E + E \mid E * E \mid id^*$

Rules: 1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack. 2. If TOS > LookAhead, POP(**Reduce**). \$id

## Operator Precedence Parsing:

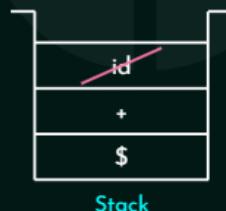
$$E \rightarrow E + E \mid E * E \mid id$$

*Rules:*

1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.
2. If TOS > LookAhead, POP(**Reduce**).



id	+	*	\$
id	--	>	>
+	<	>	<
*	<	>	>
\$	<	<	<
			--



Operator Precedence Parsing: $E \rightarrow E + E \mid E * E \mid id^*$

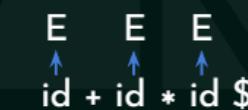
Rules: 1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack. 2. If TOS > LookAhead, POP(**Reduce**). \$Stack+idE

## Operator Precedence Parsing:

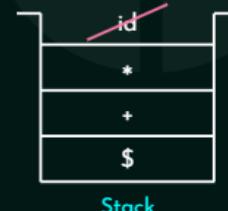
$$E \rightarrow E + E \mid E * E \mid id$$

*Rules:*

1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.
2. If TOS > LookAhead, POP(**Reduce**).



	id	+	*	\$
id	--	>	>	>
+	<	>	<	>
*	<	>	>	>
\$	<	<	<	--



Operator Precedence Parsing: $E \rightarrow E + E \mid E * E \mid id$

Rules:1.If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.2.If TOS > LookAhead, POP(**Reduce**).\$Stack+\*EidE

## Operator Precedence Parsing:

$$E \rightarrow E + E \mid E * E \mid id$$

*Rules:*

1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.
2. If TOS > LookAhead, POP(**Reduce**).

	id	+	*	\$
id	--	>	>	>
+	<	>	<	>
*	<	>	>	>
\$	<	<	<	--

Operator Precedence Parsing: $E \rightarrow E + E \mid E * E \mid id$

Rules:1.If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.2.If TOS > LookAhead, POP(**Reduce**).\$Stack+\*EE\*E

8/121

## Operator Precedence Parsing:

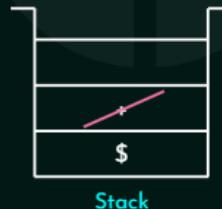
$$E \rightarrow E + E \mid E * E \mid id$$

Rules:

1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.
2. If TOS > LookAhead, POP(**Reduce**).



	id	+	*	\$
id	--	>	>	>
+	<	>	<	>
*	<	>	>	>
\$	<	<	<	--



Operator Precedence Parsing:  $E \rightarrow E + E \mid E * E \mid id$

Rules:  
1. If TOS < LookAhead, PUSH(**Shift**) LookAhead in Stack.  
2. If TOS > LookAhead, POP(**Reduce**).  
\$Stack+EE\*E+E



# Compiler Design

## Improved Operator Precedence Parser



## Outcome

- ☆ Disadvantage of using Operator Relation Table.
- ☆ Construction of Operator Function Table.

Outcome ☆ Disadvantage of using Operator Relation Table. ☆ Construction of Operator Function Table.

### Operator Precedence Parsing:

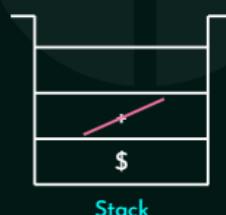
$$E \rightarrow E + E \mid E * E \mid id$$

Rules:

1. If TOS < LookAhead, PUSH(Shift) LookAhead in Stack.
2. If TOS > LookAhead, POP(Reduce).



	id	+	*	\$
id	--	>	>	>
+	<	>	<	>
*	<	>	>	>
\$	<	<	<	--



Operator Precedence Parsing:  $E \rightarrow E + E \mid E * E \mid id$

Rules: 1. If TOS < LookAhead, PUSH(Shift) LookAhead in Stack.

2. If TOS > LookAhead, POP(Reduce).

Stack + E \* E + E

## Operator Relation Table:

$E \rightarrow E + E \mid E * E \mid id$

#terminals: n

	id	+	*	\$
id	--	>	>	>
+	<	>	<	>
*	<	>	>	>
\$	<	<	<	--

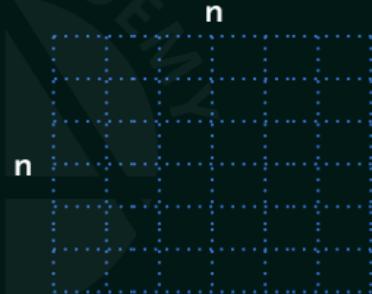


Table Size:  $O(n^2)$

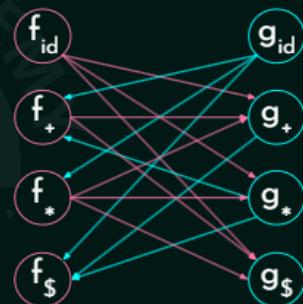
Operator Relation Table:  $E \rightarrow E + E \mid E * E \mid id \quad id + \$ \Rightarrow \dots$  #terminals: n  
Table Size:  $O(n^2)$

## Construction of Operator Function Table:

$E \rightarrow E + E \mid E * E \mid id$

Diagram illustrating the construction of the Operator Function Table. The table is shown with rows and columns labeled by terminals. A red oval encloses the first column (id), and a red arrow labeled 'f' points from it to the first row.

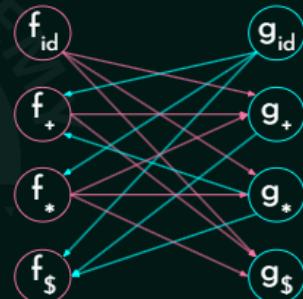
	id	+	*	\$
id	--	>	>	>
+	<	>	<	>
*	<	>	>	>
\$	<	<	<	--



Construction of Operator Function Table:  $E \rightarrow E + E \mid E * E \mid id \quad id + \$ \Rightarrow \dots$   
 $\dots \leftarrow f_id f + f * f \$ \leftarrow g_id g_+ g_* g \$$

## Construction of Operator Function Table:

$E \rightarrow E + E \mid E * E \mid id$



Construction of Operator Function Table: $E \rightarrow E + E \mid E * E \mid id$

$id^*fidf+g+gidf+*f\$g+*g$fidg+*f+g+f\$f+g+f\$f+*g+*f+g+f\$g+*f+g+f\$f+*g+*f+g+f\$g+f\$$

## Construction of Operator Function Table:

$E \rightarrow E + E \mid E * E \mid id$



	id	+	*	\$
f	4	2	4	0
g	5	1	3	0

Construction of Operator Function Table: $E \rightarrow E + E \mid E * E \mid id$

$id^*fidg+*f+g+f\$f+g+f\$f+*g+*f+g+f\$g+*f+g+f\$f+*g+*f+g+f\$g+f\$42400135fid+\$g^*$

## Construction of Operator Function Table:

#terminals: n

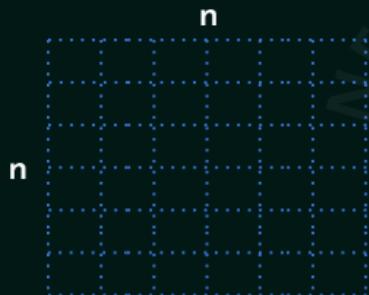


Table Size:  $O(n^2)$

	id	+	*	\$
f	4	2	4	0
g	5	1	3	0

Table Size:  $O(2n)$

Construction of Operator Function Table:#terminals: n nnTable Size:  $O(n^2)$

fid+\$g\*42400135Table Size:  $O(2n)$

The slide features a central title 'Compiler Design' in a large, stylized font. Above the title is a small icon of a computer monitor displaying code and a gear, symbolizing compilation. Below the title, the text 'Operator Precedence Parsers – Solved Problem' is displayed in a large, bold, light blue font.

Compiler Design Operator Precedence Parsers - Solved Problem



## Outcome

- ★ Construction of Operator Function Table for a given CFG.

Outcome★Construction of Operator Function Table for a given CFG.

Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:

$$\begin{aligned} P &\rightarrow SR \mid S \\ R &\rightarrow bSR \mid bS \\ S &\rightarrow WbS \mid W \\ W &\rightarrow L*W \mid L \\ L &\rightarrow id \end{aligned}$$

**Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:**

$P \rightarrow SR \mid S$  *Paragraph: Sentence followed by Recursive\_#Sentences or a Sentence.*

$R \rightarrow bSR \mid bS$  *Recursive\_#Sentences: a blank followed by a Sentence & Recursive\_#Sentences*

$S \rightarrow WbS \mid W$  *or*

$W \rightarrow L \cdot W \mid L$  *a blank followed by a Sentence.*

$L \rightarrow id$

*Sentence: a Word followed by a blank & a Sentence or a Word.*

*Word: a Letter concatenated with a Word or a Letter.*

*Letter: Identifier.*

**Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:**  
 $P \rightarrow SR \mid S$  *Paragraph: Sentence followed by Recursive\_#Sentences or a Sentence.*  
 $R \rightarrow bSR \mid bS$  *Recursive\_#Sentences: a blank followed by a Sentence & Recursive\_#Sentences or a blank followed by a Sentence.*  
 $S \rightarrow WbS \mid W$  *Sentence: a Word followed by a blank & a Sentence or a Word.*  
 $W \rightarrow L \cdot W \mid L$  *Word: a Letter concatenated with a Word or a Letter.*  
 $L \rightarrow id$  *Identifier.*

**Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:**

$P \rightarrow SR \mid S$

$R \rightarrow bSR \mid bS$

$S \rightarrow WbS \mid W$

$W \rightarrow L \cdot W \mid L$

$L \rightarrow id$

Q: Construct the Operator Function Table for Operator Precedence Parsing from the following  
CFG:  $P \rightarrow SR \mid S$   $R \rightarrow bSR \mid bSS \rightarrow WbS \mid WW \rightarrow L+W \mid LL \rightarrow id^*$

Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:

$P \rightarrow SR \mid S$

$R \rightarrow bSR \mid bS$

$S \rightarrow WbS \mid W$

$W \rightarrow L*W \mid L$

$L \rightarrow id$

$P \rightarrow SbSR \mid SbS \mid S$

Q: Construct the Operator Function Table for Operator Precedence Parsing from the following  
CFG:  $P \rightarrow SR \mid S$   $R \rightarrow bSR \mid bSS \rightarrow WbS \mid WW \rightarrow L+W \mid LL \rightarrow id^*$   $P \rightarrow SbP \mid SbS \mid S$

Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:

$P \rightarrow SR \mid S$

$R \rightarrow bSR \mid bS$

$S \rightarrow WbS \mid W$

$W \rightarrow L*W \mid L$

$L \rightarrow id$

$P \rightarrow SbP \mid SbS \mid S$

$S \rightarrow WbS \mid W$

$W \rightarrow L*W \mid L$

$L \rightarrow id$

Q: Construct the Operator Function Table for Operator Precedence Parsing from the following  
 CFG:  $P \rightarrow SR \mid S$   $R \rightarrow bSR \mid bSS \rightarrow WbS \mid WW \rightarrow L+W \mid LL \rightarrow id^*P \rightarrow SbPI$   $SbSI \mid SS \rightarrow WbS \mid WW \rightarrow L+W \mid L^*L \rightarrow id$

**Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:**

$P \rightarrow SbP \mid SbS \mid S$

$S \rightarrow WbS \mid W$

$W \rightarrow L^*W \mid L$

$L \rightarrow id$

	$id$	$b$	*	\$
$id$	--	>	>	>
$b$	<	<	<	>
*	<	>	<	>
\$	<	<	<	--

Q: Construct the Operator Function Table for Operator Precedence Parsing from the following  
 CFG:  $W \rightarrow L+W \mid LP \rightarrow SbPI$   $SbSI \mid SS \rightarrow WbS \mid W^*L \rightarrow idididb\$b\$^{**-->>><<<<><><<<--}$

**Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:**

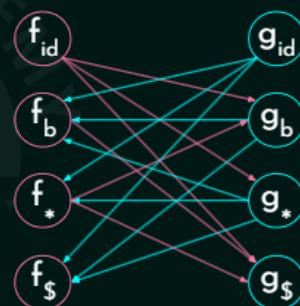
$P \rightarrow SbP \mid SbS \mid S$

$S \rightarrow WbS \mid W$

$W \rightarrow L^*W \mid L$

$L \rightarrow id$

$f$	$id$	$b$	*	\$	$g$
$id$	--	>	>	>	
$b$	<	<	<	>	
*	<	>	<	>	
\$	<	<	<	--	



Q: Construct the Operator Function Table for Operator Precedence Parsing from the following  
 CFG:  $W \rightarrow L + W \mid LP \rightarrow SbP \mid SbS \mid SS \rightarrow WbS \mid W^*L \rightarrow idididb\$b\$^{**-->>><<<<>><<<--}$   
 $fifdfbgbgdf+*f\$g+*g\$g f$

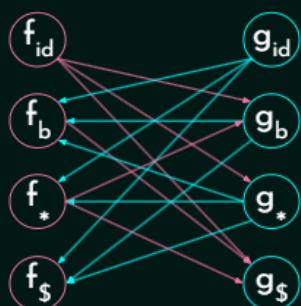
Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:

$P \rightarrow SbP \mid SbS \mid S$

$S \rightarrow WbS \mid W$

$W \rightarrow L * W \mid L$

$L \rightarrow id$



	id	b	*	\$
f	5	1	3	0
g	4	2	4	0

fidb\$g\*g\*Q: Construct the Operator Function Table for Operator Precedence Parsing from the following CFG:  $W \rightarrow L + W \mid LP \rightarrow SbP \mid SbS \mid SS \rightarrow WbS \mid W^*L \rightarrow idididb\$b\$^{**-->>><<<<>><<<--}$   
 $fifdfbgbgdf+*f\$g+*g\$fidg+*gbfbgdf+*f+*g\$gbfbg\$51300244$



# Compiler Design

## Introduction to LR Parsers



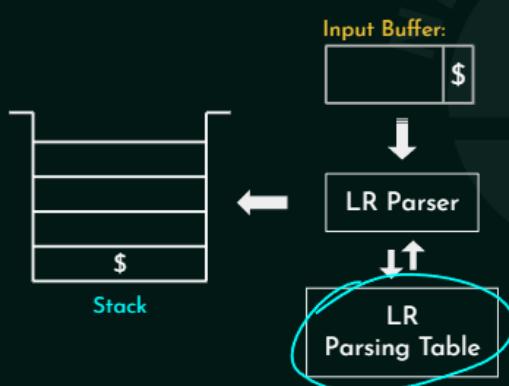
## Outcome

- ☆ Understanding the Organization of LR Parsers.
- ☆ What are LR(0) items?
- ☆ CLOSURE and GOTO properties.
- ☆ Derivation of Canonical collection of LR(0) items.

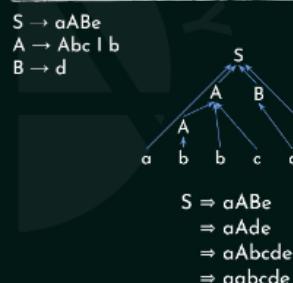
Outcome  
 ☆ Understanding the Organization of LR Parsers.  
 ☆ What are LR(0) items?  
 ☆ CLOSURE and GOTO properties.  
 ☆ Derivation of Canonical collection of LR(0) items.

## LR Parsers:

— Use Right most derivation - In reverse.  
 — Scan from Left to right.



### Generation of Parse Tree - Bottom up approach:

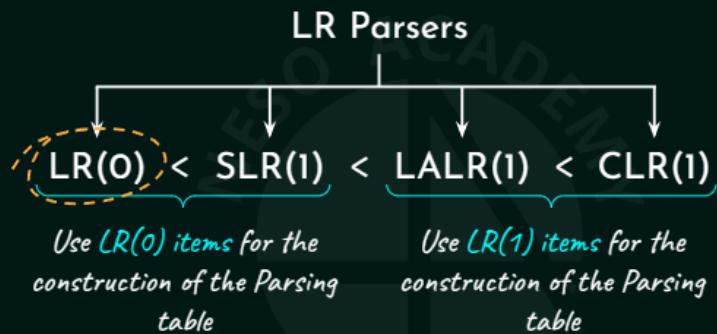


Decision:  
 When to reduce.

(Right most Derivation) - In reverse.

LR Parsers: \$  
 LR Parser  
 LR Parsing Table  
 Input Buffer  
 Stack  
 Scan from Left to right.  
 Use Right most derivation - In reverse.

## LR Parsers:



LR Parsers: LR Parsers  
LR(0) CLR(1) LALR(1) SLR(1) <<< Use LR(0) items for the construction of the Parsing table  
Use LR(1) items for the construction of the Parsing table

## LR(0) Parser:

1. Conversion to Augmented Grammar.
2. Use CLOSURE and GOTO properties.

$$\begin{array}{l} S' \rightarrow S \\ S \rightarrow AA \\ A \rightarrow aA \mid b \end{array}$$

LR(0) items

$$S \rightarrow .AA$$

The compiler hasn't read anything yet.

1. Conversion to Augmented Grammar.  
2. Use CLOSURE and GOTO properties.  
 $S' \rightarrow S$   
 $S \rightarrow AA$   
 $A \rightarrow aA \mid b$   
LR(0) items  
LR(0) Parser:  
 $S \rightarrow .AA$   
The compiler hasn't read anything yet.

### LR(0) Parser:

1. Conversion to Augmented Grammar.
2. Use CLOSURE and GOTO properties.

$S' \rightarrow S$   
 $S \rightarrow AA$   
 $A \rightarrow aA \mid b$

LR(0) items

$S \rightarrow A.A$

*The compiler has read till here.*

1. Conversion to Augmented Grammar.  
2. Use CLOSURE and GOTO properties.  
 $S' \rightarrow SS \rightarrow AA$   
 $A \rightarrow aA \mid b$   
LR(0) items  
 $S \rightarrow A.A$   
The compiler has read till here.

### LR(0) Parser:

1. Conversion to Augmented Grammar.
2. Use CLOSURE and GOTO properties.

$S' \rightarrow S$   
 $S \rightarrow AA$   
 $A \rightarrow aA \mid b$

LR(0) items

$S \rightarrow AA.$

*The compiler has read the entire RHS.*

1. Conversion to Augmented Grammar.  
2. Use CLOSURE and GOTO properties.  
 $S' \rightarrow SS \rightarrow AA$   
 $A \rightarrow aA \mid b$   
LR(0) items  
 $S \rightarrow AA.$   
The compiler has read the entire RHS.

## LR(0) Parser:

1. Conversion to Augmented Grammar.
2. Use CLOSURE and GOTO properties.

Whenever there is a '.' to the left of a non-terminal, include all its productions in the set adding a '.' preceding the rules.

$S' \rightarrow S$   
 $S \rightarrow AA$   
 $A \rightarrow aA \mid b$

$S' \rightarrow .S$   
 $S \rightarrow .AA$   
 $A \rightarrow .aA \mid .b$

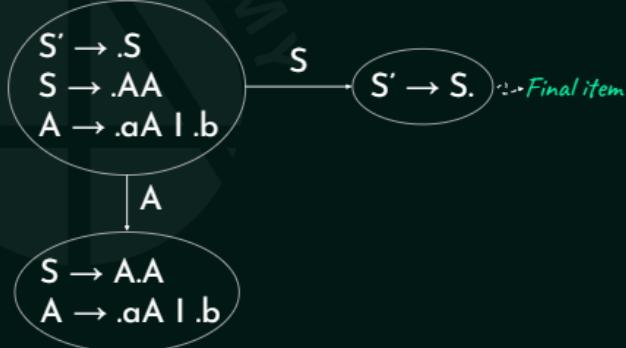
1. Conversion to Augmented Grammar.  
 2. Use CLOSURE and GOTO properties.  
 $S' \rightarrow SS \rightarrow AA A \rightarrow aA \mid b$
- LR(0) Parser: Whenever there is a '.' to the left of a non-terminal, include all its productions in the set adding a '.' preceding the rules.  
 $S' \rightarrow .SS \rightarrow .AA A \rightarrow .aA \mid .b$

## LR(0) Parser:

1. Conversion to Augmented Grammar.
2. Use CLOSURE and GOTO properties.

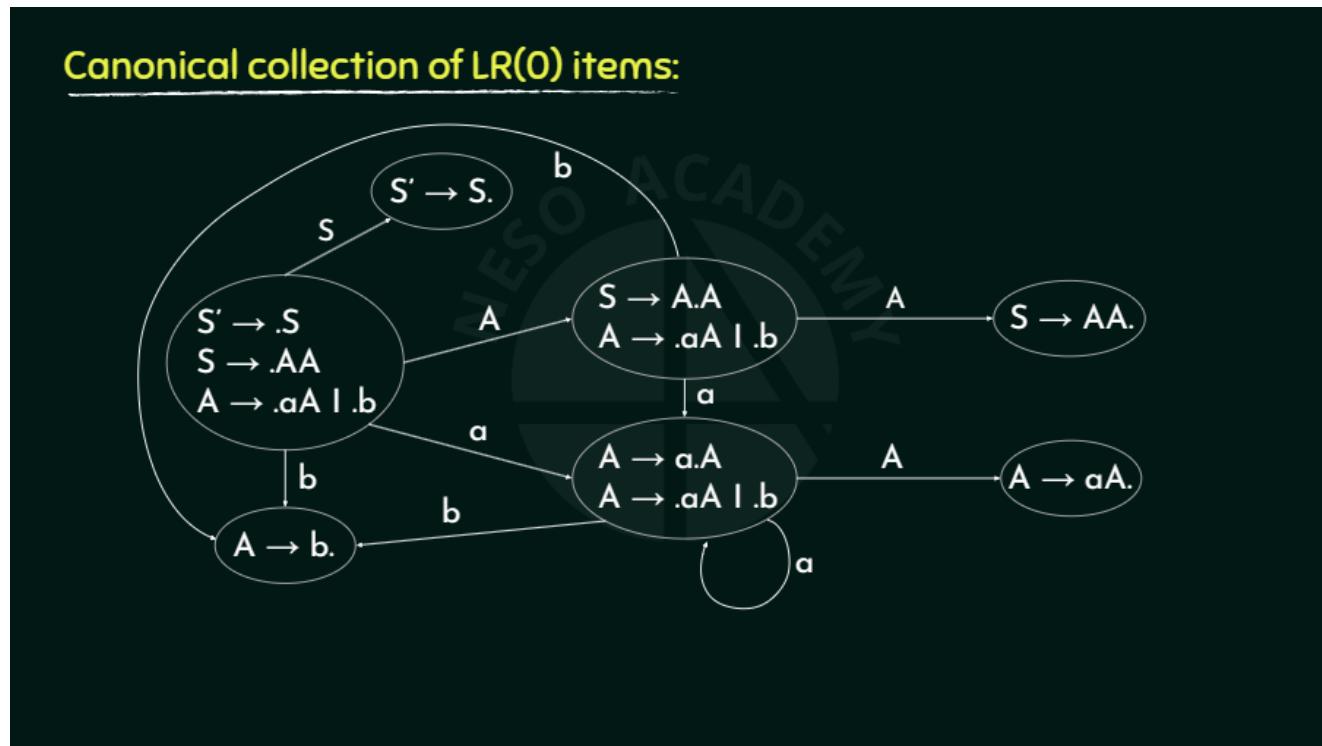
On a Canonical collection of LR(0) items we apply GOTO for every non-terminal by moving the '.' to the right of the same, thus the new collection is obtained.

$S' \rightarrow S$   
 $S \rightarrow AA$   
 $A \rightarrow aA \mid b$



1. Conversion to Augmented Grammar.  
 2. Use CLOSURE and GOTO properties.  
 $S' \rightarrow SS \rightarrow AA A \rightarrow aA \mid b$
- LR(0) Parser: On a Canonical collection of LR(0) items we apply GOTO for every non-terminal by moving the '.' to the right of the same, thus the new collection is

obtained.  $S' \rightarrow .SS \rightarrow .AA A \rightarrow .aA I .bS' \rightarrow S.S$  Final item  $S \rightarrow A.A A \rightarrow .aA I .bA$



Canonical collection of LR(0) items:  
 $A \rightarrow b.A \rightarrow a.A A \rightarrow .aA I .b S \rightarrow AA.A \rightarrow aA.S' \rightarrow .SS \rightarrow .AA A \rightarrow .aA I .b S' \rightarrow S.S$   
 $SS \rightarrow A.A A \rightarrow .aA I .b AAAababba$



Compiler Design LR(0) Parsers - Part 1

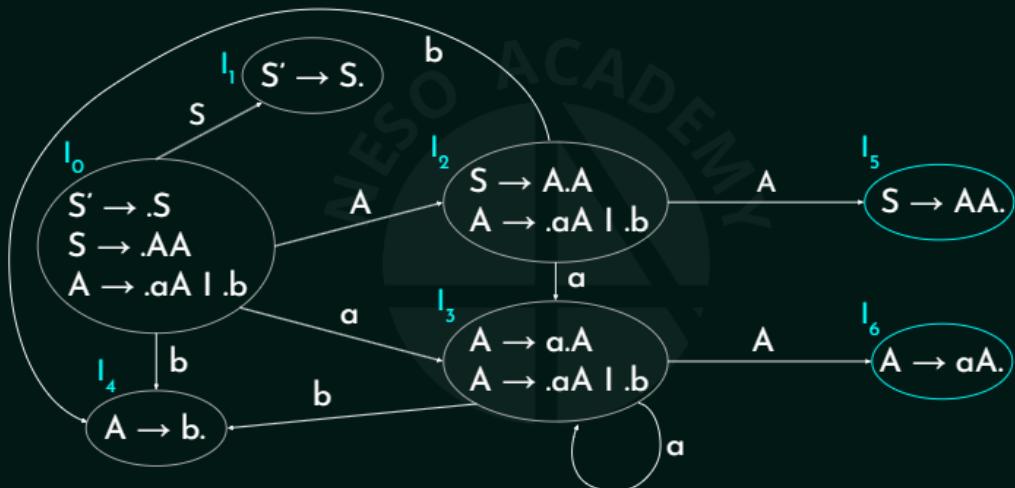


## Outcome

- ★ Construction of LR(0) Parsing table.

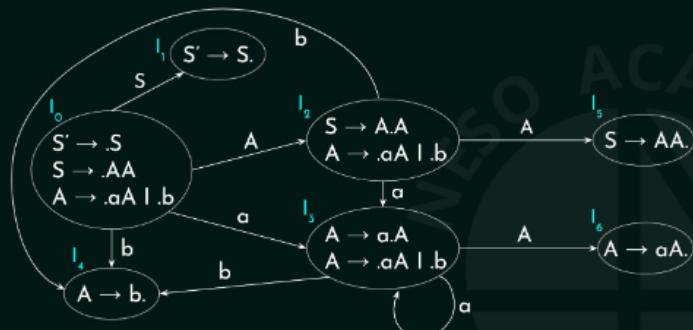
Outcome★Construction of LR(0) Parsing table.

### Canonical collection of LR(0) items:



Canonical collection of LR(0) items:  
 $A \rightarrow b$ .  
 $A \rightarrow a$ .  
 $A \rightarrow .aA \mid .b$   
 $S \rightarrow AA$ .  
 $A \rightarrow aA$ .  
 $S' \rightarrow .S$   
 $S \rightarrow .AA$   
 $A \rightarrow .aA \mid .b$   
 $A \rightarrow aA \mid b$   
 $A \rightarrow b$

## Construction of LR(0) Parsing Table:



- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$   
 Action GO TO States Construction of LR(0) Parsing Table: 0123456S3AcceptS42S3S4S3S4r<sub>iii</sub>r<sub>ii</sub>r<sub>ii</sub>r<sub>ii</sub>r<sub>ii</sub>r<sub>ii</sub>r<sub>ii</sub>r<sub>ii</sub>56ab\$SA

# Compiler Design

## LR(0) Parsers – Part 2

Compiler Design LR(0) Parsers - Part 2

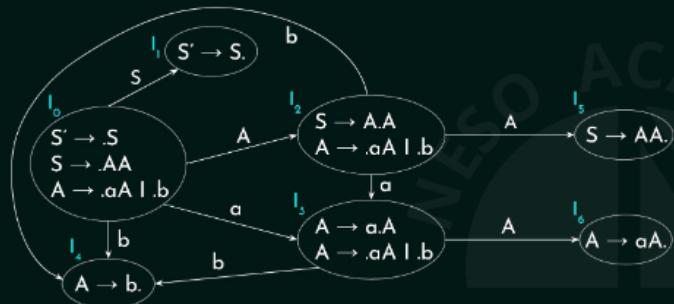


## Outcome

- ☆ LR(0) Parsing Procedure.

Outcome☆LR(0) Parsing Procedure.

### Construction of LR(0) Parsing Table:



- $S \rightarrow AA$
- $A \rightarrow aA$
- $A \rightarrow b$

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$   
 Action GO TO States Construction of LR(0) Parsing Table:  
 0 1 2 3 4 5 6 S3 Accept S4 S3 S4 S3 S4 r<sub>iii</sub> r<sub>iii</sub> r<sub>iii</sub> r<sub>i</sub> r<sub>i</sub> r<sub>ii</sub> r<sub>ii</sub> \$ SA

## LR(0) Parsing:

$$\begin{array}{l} S \rightarrow AA \\ A \rightarrow aA \\ A \rightarrow b \end{array}$$

Stack

4
b
3
a
3
a
0

$S_n$ : 1. Push the i/p  
2. Push the State 'n'

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

StackS → AAA → aA A → bLR(0) Parsing:1ActionGO

TOStates0123456S3AcceptS42S3S4S3S4riiiiiiiiririririri56ab\$SAa a b b \$a03a34bSn :  
1.Push the i/p 2.Push the State 'n'

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $\bar{A} \rightarrow b$

$$|\text{RHS}| = 1 \times 2 = 2$$

A vertical stack structure with 7 slots. The slots are labeled from bottom to top as 0, a, 3, a, 3, b, and 4. A pink line connects the top four slots (4, b, 3, a).

a a b b \$

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$a03a34bIRHSIRHS= 1x 2= 21ActionGO TOStates0123456S3AcceptS42S3S4S3S4riiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$

6
A
3
a
3
a
0

Stack



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1				Accept	
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stack i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  LR(0) Parsing: a a b b \$ a03a3AA61Action GO TO States 0123456 S3AcceptS42S3S4S3S4riiiiiiiiiiiiiiiiiiiiiiiiiii56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$

6
A
3
a
3
a
0

Stack



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1				Accept	
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

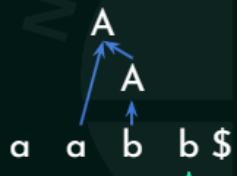
Stack i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  LR(0) Parsing: a a b b \$ a03a3AA61RHSIRHS= 2x 2= 41Action GO TO States 0123456 S3AcceptS42S3S4S3S4riiiiiiiiiiiiiiiiiiiiiiiiiii56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$

```

    [ ] ---+---+---+---+---+
    |       |       |       |       |
    +-----+-----+-----+-----+
      6     A     3     a     0
  
```



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$a0A3A6A 1ActionGO  
TOStates0123456S3AcceptS42S3S4S3S4riiiiiiiiiiiiiiiiiiiiiiiii56ab\$SA

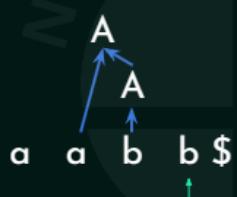
## LR(0) Parsing:

- i.  $S \rightarrow AA$   
ii.  $A \rightarrow aA$  |RHS| = 2 x 2 = 4  
iii.  $A \rightarrow b$

```

graph TD
    Stack[Stack]
    S1[ ] --- Stack
    S2[ ] --- Stack
    S3[ ] --- Stack
    S4[ ] --- Stack
    S5[ ] --- Stack

```

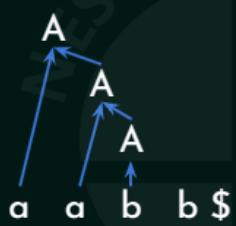
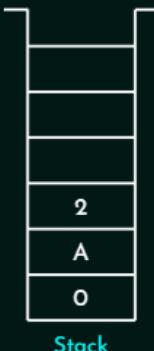


States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$a0A3AIRHSIRHS= 2x 2= 46A  
 1ActionGO TOStates0123456S3AcceptS42S3S4S3S4riiiiiiiiiiriririririririririi56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$

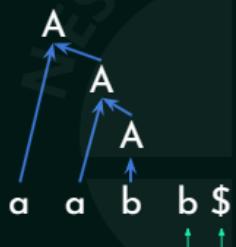
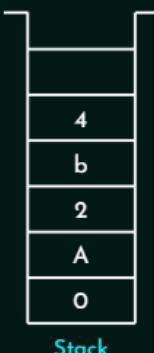


States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$A0AA A 21ActionGOToStates0123456S3AcceptS42S3S4S3S4riiiriiriiiriririirii56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$



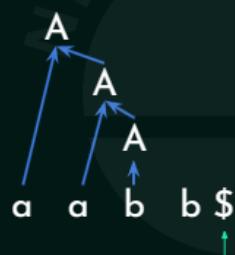
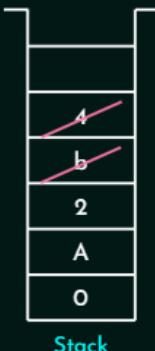
States	Action			GO	TO
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$A0AA A 2b41ActionGOToStates0123456S3AcceptS42S3S4S3S4riiiiiiiiiiiiiiiiiiiiiii56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$

$$|\text{RHS}| = 1 \times 2 = 2$$

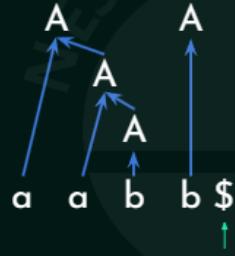


States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$A0AA A 2b4IRHSIRHS= 1x 2= 21ActionGO TOStates0123456S3AcceptS42S3S4S3S4riiiiiiririririririi56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$

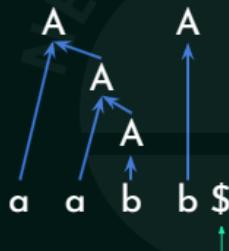


States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$A0AA A 2AA51ActionGOToStates0123456S3AcceptS42S3S4S3S4riiiiiiiiiiiiiiiiiiiiiii56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$  |RHS| = 2 x 2 = 4  
 ii.  $A \rightarrow aA$   
 iii.  $A \rightarrow b$

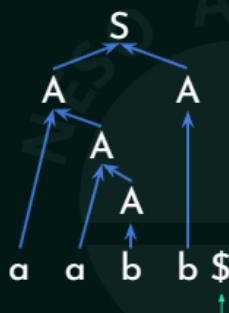


States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$A0AA A 2AA5IRHSIRHS= 2x 2= 41ActionGO TOStates0123456S3AcceptS42S3S4S3S4riiiiiiiiririririririiii56ab\$SA

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$S0AA A A1S1ActionGOTOSStates0123456S3AcceptS42S3S4S3S4riiiriiiriiiriririririi56ab\$SA



# Compiler Design

## SLR(1) Parsers

Compiler Design SLR(1) Parsers



### Outcome

- ☆ Problem with LR(0) Parsing Procedure.
- ☆ Difference between LR(0) and SLR(1) Parsers.
- ☆ Construction of SLR(1) Parsing Table.

Outcome  
☆ Problem with LR(0) Parsing Procedure.  
☆ Difference between LR(0) and SLR(1) Parsers.  
☆ Construction of SLR(1) Parsing Table.

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$

```

    Stack
    +---+
    |   4|
    +---+
    |   b|
    +---+
    |   3|
    +---+
    |   a|
    +---+
    |   3|
    +---+
    |   a|
    +---+
    |   0|
    +---+
  
```



$S_n$ : 1. Push the i/p  
2. Push the State 'n'

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

## StackLR(0) Parsing: 1 Action GO

TOStates0123456S3AcceptS42S3S4S3S4riiiiiiiiririririririi56ab\$SAa a b b \$a03a34bSn :

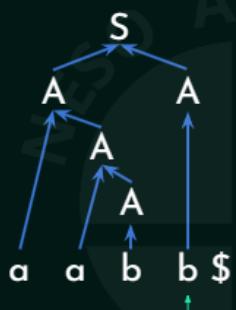
1.Push the i/p 2.Push the State 'n' i.S → AAii.A → aA iii.A → b

## LR(0) Parsing:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$

Stack

6
A
3
a
3
a
0

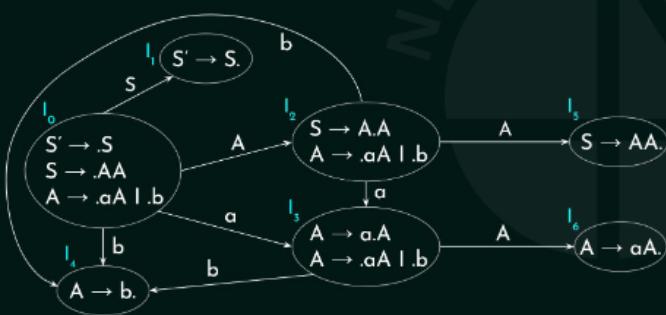


States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

Stacki.S → AAii.A → aA iii.A → bLR(0) Parsing:a a b b \$a03a3AA61ActionGO

### SLR(1) Parsing Table:

- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$  FOLLOW(A) = {a, b, \$}
- iii.  $A \rightarrow b$



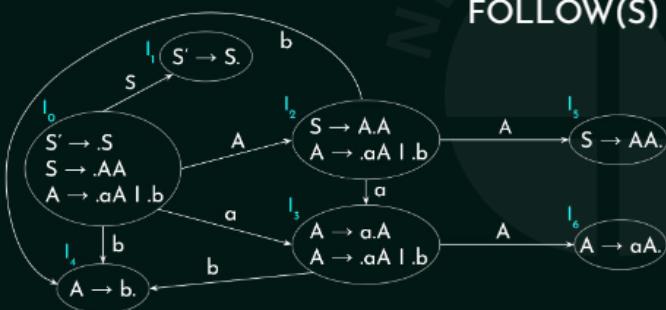
States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5			$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  SLR(1) Parsing Table: 1 Action GO

TO States 0 1 2 3 4 5 6 S 3 Accept S 4 2 S 3 S 4 S 3 S 4 r<sub>iii</sub> r<sub>ii</sub> r<sub>i</sub> 5 6 ab \$ SAriiiiiiiii FOLLOW(A) = {a, b, \$}

### SLR(1) Parsing Table:

- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$  FOLLOW(A) = FIRST(A)  
U  
FOLLOW(S)



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4					
5					
6					

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  SLR(1) Parsing Table: 1 Action GO

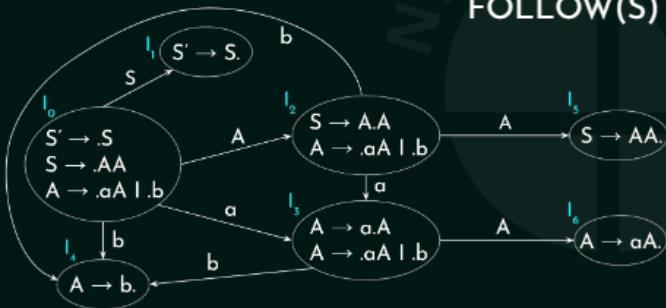
TO States 0 1 2 3 4 5 6 S 3 Accept S 4 2 S 3 S 4 S 3 S 4 5 6 ab \$ SA FOLLOW(A) = FIRST(A) U FOLLOW(S)

### SLR(1) Parsing Table:

i.  $S \rightarrow AA$

ii.  $A \rightarrow aA$

iii.  $A \rightarrow b$       FOLLOW(A) = {a, b}  $\cup$  FOLLOW(S)



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4					
5					
6					

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  SLR(1) Parsing Table: 1 Action GO

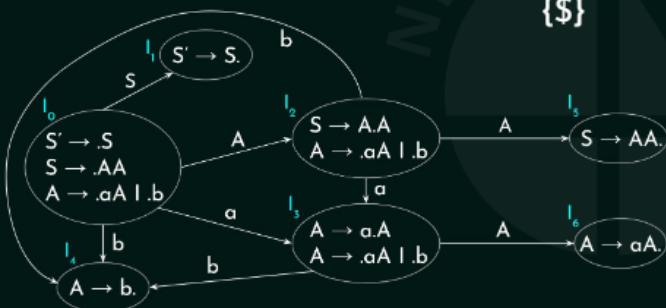
TO States 0 1 2 3 4 5 6 S 3 Accept S 4 2 S 3 S 4 S 3 S 4 5 6 ab \$ SA FOLLOW(A) = {a, b}  $\cup$  FOLLOW(S)

### SLR(1) Parsing Table:

i.  $S \rightarrow AA$

ii.  $A \rightarrow aA$

iii.  $A \rightarrow b$       FOLLOW(A) = {a, b}  $\cup$  {\$}



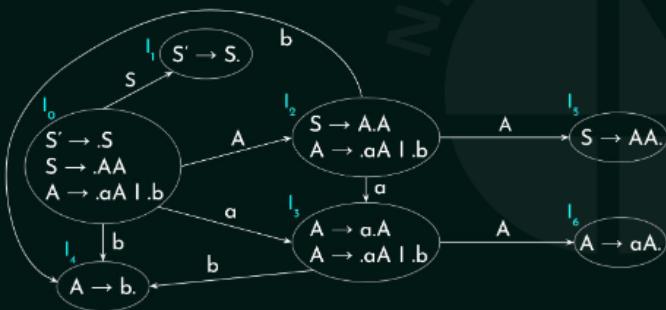
States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4					
5					
6					

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  SLR(1) Parsing Table: 1 Action GO

TO States 0 1 2 3 4 5 6 S 3 Accept S 4 2 S 3 S 4 S 3 S 4 5 6 ab \$ SA FOLLOW(A) = {a, b}  $\cup$  {\$}

### SLR(1) Parsing Table:

- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$      $\text{FOLLOW}(A) = \{a, b, \$\}$



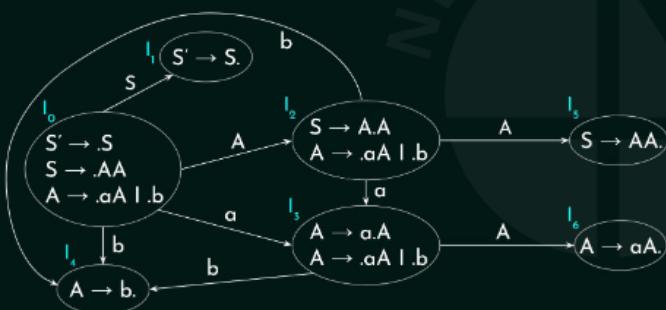
States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5					
6					

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  SLR(1) Parsing Table: 1 Action GO

TO States 0123456 S3 Accept S42S3S4S3S456ab\$ SAriiiriirii FOLLOW(A) = {a, b, \$}

### SLR(1) Parsing Table:

- i.  $S \rightarrow AA$      $\text{FOLLOW}(S) = \{\$\}$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5					
6					

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$  SLR(1) Parsing Table: 1 Action GO

TO States 0123456 S3 Accept S42S3S4S3S456ab\$ SAriiiriirii FOLLOW(S) = {\$}

## Comparison:

- i.  $S \rightarrow AA$
  - ii.  $A \rightarrow aA$
  - iii.  $A \rightarrow b$

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

## LR(0) Parsing Table

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5			$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

## SLR(1) Parsing Table

Comparison:i.S → AAii.A → aA iii.A → b1ActionGO

TOStates0123456S3AcceptS42S3S4S3S4riiiiiiiiririririririi56ab\$SALR(0) Parsing

Table1ActionGO TOStates0123456S3AcceptS42S3S4S3S4riiiii56ab\$SAriiiiriiiriSLR(1)

## Parsing Table



# Compiler Design

## Conflicts in LR(0) and SLR(1) Parsers

## Compiler Design Conflicts in LR(0) and SLR(1) Parsers



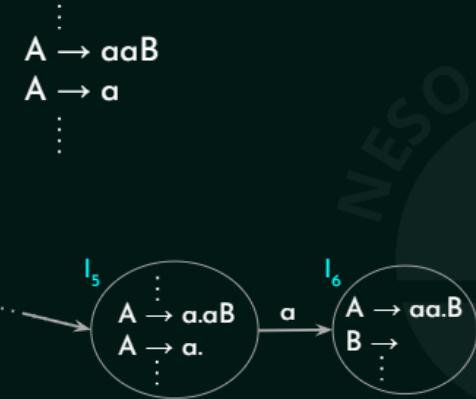
## Outcome

- ☆ S-R and R-R conflicts in LR(0) Parsing table.
- ☆ S-R and R-R conflicts in SLR(1) Parsing table.

Outcome  
☆ S-R and R-R conflicts in LR(0) Parsing table.  
☆ S-R and R-R conflicts in SLR(1) Parsing table.

### LR(0) Parsing Table:

- iii.  $A \rightarrow aaB$
- iv.  $A \rightarrow a$



LR(0) Parsing Table

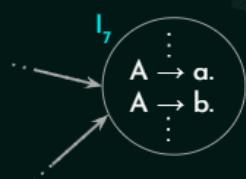
States	Action		
	a		
5	$S_6/r_{iv}$	$r_{iv}$	$r_{iv}$
...	...	...	...

Shift - Reduce  
Conflict

iii.  $A \rightarrow aaB$  iv.  $A \rightarrow a$   
LR(0) Parsing Table:  
aA  $\rightarrow aa.B$  B  $\rightarrow .$   
I6 LR(0) Parsing Table  
Action States S6 a riv riv 5 A  $\rightarrow a.aB$  A  $\rightarrow a.$  I5 riv Shift - Reduce Conflict

## LR(0) Parsing Table:

- iii.  $A \rightarrow a$
- iv.  $A \rightarrow b$



LR(0) Parsing Table

States	Action	
	a	b
7	$r_{\text{iii}} / r_{\text{iv}}$	$r_{\text{iii}} / r_{\text{iv}}$

Reduce - Reduce  
Conflict

LR(0) Parsing Table: iii.  $A \rightarrow a$  iv.  $A \rightarrow b$   
LR(0) Parsing Table Action States a 7  
a.  $A \rightarrow b$ . l7riiirivriiirivriiiriv  
b. Reduce - Reduce Conflict

## LR(0) Parsing Table:

- iii.  $A \rightarrow aaB$
- iv.  $A \rightarrow a$

Not LR(0) Grammar

- iii.  $A \rightarrow a$
- iv.  $A \rightarrow b$

LR(0) Parsing Table

States	Action		
	a		
5	$S_6 / r_{\text{iii}}$	$r_{\text{iii}}$	$r_{\text{iii}}$

Shift - Reduce  
Conflict

LR(0) Parsing Table

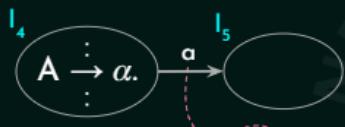
States	Action		
	a		
7	$r_{\text{iii}} / r_{\text{iv}}$	$r_{\text{iii}} / r_{\text{iv}}$	$r_{\text{iii}} / r_{\text{iv}}$

Reduce - Reduce  
Conflict

LR(0) Parsing Table: iii.  $A \rightarrow a$  iv.  $A \rightarrow b$   
LR(0) Parsing Table Action States a 7  
a.  $A \rightarrow b$ . l7riiirivriiirivriiiriv  
b. Reduce - Reduce Conflict  
iii.  $A \rightarrow aaB$  iv.  $A \rightarrow a$  Not LR(0) Grammar

## SLR(1) Parser:

Shift - Reduce Conflict:



$$\text{FOLLOW}(A) = \{ \dots \}$$

Reduce - Reduce Conflict:

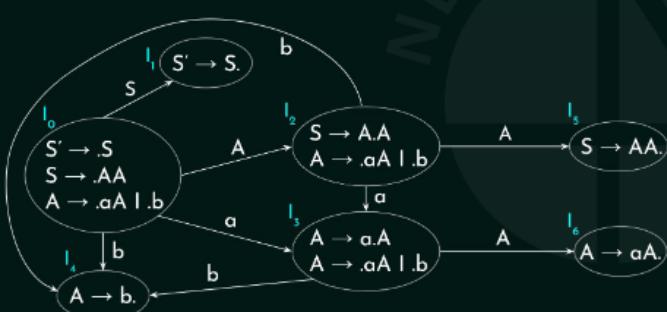


$$\text{FOLLOW}(A) \cap \text{FOLLOW}(B) \neq \emptyset$$

$\neq \emptyset$  SLR(1) Parser:  
 Shift - Reduce Conflict:  $A \rightarrow \alpha.$  | 4 | 5 | a  
 Reduce - Reduce Conflict:  $A \rightarrow \alpha.$  | B  $\rightarrow \beta.$   
 $\text{FOLLOW}(A) = \{ \dots \}$   $\text{FOLLOW}(A) \cap \text{FOLLOW}(B) \neq \emptyset$

## LR(0) Parsing Table:

- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1					Accept
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

LR(0) Parsing Table

i.  $S \rightarrow AA$  ii.  $A \rightarrow aA$  iii.  $A \rightarrow b$   
 LR(0) Parsing Table:  
 States 0 1 2 3 4 5 6  
 Action S  $S_3$   $S_4$   $r_{iii}$   $r_i$   $r_{ii}$   
 GO TO A 2 5 6  
 S 1 Accept  
 LR(0) Parsing Table



# Compiler Design

## Determining the type of Grammar – Set 1

Compiler Design Determining the type of Grammar - Set 1



### Outcome

- ☆ Understanding how to determine the type of a given grammar.

Outcome ☆ Understanding how to determine the type of a given grammar.

**Q:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$S \rightarrow dA \mid aB$$

$$A \rightarrow bA \mid c$$

$$B \rightarrow bB \mid c$$

**Q:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow dA \mid aB$   $A \rightarrow bA \mid c$   
 $B \rightarrow bB \mid c$

**Q:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

**Sol.**

	FIRST	FOLLOW
$S \rightarrow dA \mid aB$	{d, a}	{\$}
$A \rightarrow bA \mid c$	{b, c}	{\$}
$B \rightarrow bB \mid c$	{b, c}	{\$}

	a	b	c	d	\$
S	$S \rightarrow aB$			$S \rightarrow dA$	
A		$A \rightarrow bA$	$A \rightarrow c$		
B		$B \rightarrow bB$	$B \rightarrow c$		

LL(1) Parsing Table

LL(1) Parsing Table  
leads to:  
 $S \rightarrow aB \mid bB$   
 $A \rightarrow bA \mid c$   
 $B \rightarrow bB \mid c$

**Q:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow dA \mid aB$   $A \rightarrow bA \mid c$   
 $B \rightarrow bB \mid c$

**Q:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

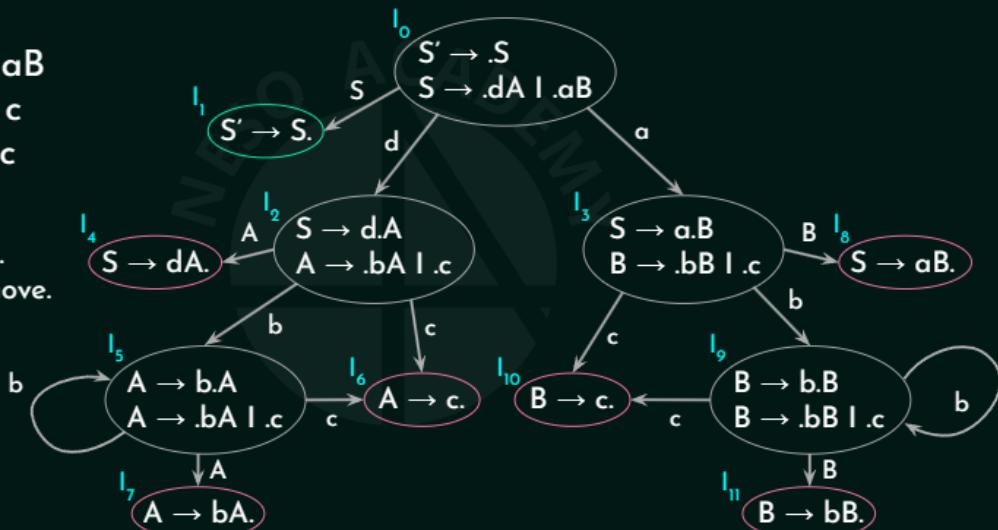
**Sol.**  $S' \rightarrow S$

$$S \rightarrow dA \mid aB$$

$$A \rightarrow bA \mid c$$

$$B \rightarrow bB \mid c$$

- 1 final item.
- No Shift move.



I9I5I3I2I0Q: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow SS \rightarrow dA \mid aB \quad A \rightarrow bA \mid cB \rightarrow bB \mid c$   
**Sol.**  $S' \rightarrow .SS \rightarrow .dA \mid .aB \rightarrow d.AA \rightarrow .bA \mid .cS \rightarrow a.BB \rightarrow .bB \mid .cA \rightarrow b.AA \rightarrow .bA \mid .cB \rightarrow b.BB \rightarrow .bB \mid .cS' \rightarrow S.I1S \rightarrow dA.I4A \rightarrow c.I6A \rightarrow bA.I7S \rightarrow aB.I8B \rightarrow c.I10B \rightarrow bB.I11SdaAbccbABbccBb$   
• 1 final item.  
• No Shift move.

**Q:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

**Sol.** i.  $S \rightarrow AA$

ii.  $A \rightarrow aA$

iii.  $A \rightarrow b$

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

LR(0) Parsing Table

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5				$r_i$	
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

SLR(1) Parsing Table

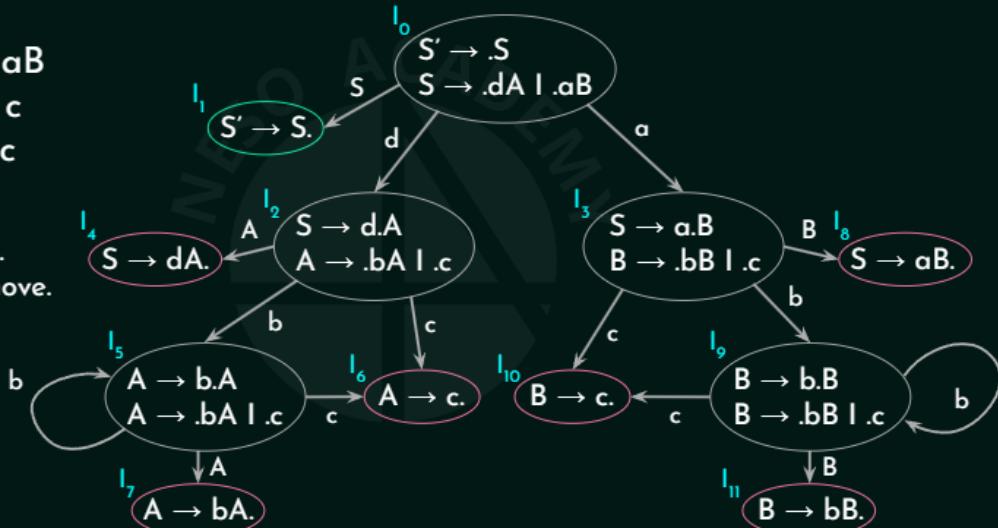
Q:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 Sol.i.  $S \rightarrow AA$   
 ii.  $A \rightarrow aA$   
 iii.  $A \rightarrow b1$   
 Action GO

TOS States 0123456 S3 Accept S42 S3 S4 S3 S4 riiiiiiriiiriiiriiirii56ab\$ SALR(0) Parsing  
 Table 1 Action GO TOS States 0123456 S3 Accept S42 S3 S4 S3 S4 riiiri56ab\$ SAriiiiriiirii SLR(1)  
 Parsing Table

Q: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow S$   
 $S \rightarrow dA \mid aB$   
 $A \rightarrow bA \mid c$   
 $B \rightarrow bB \mid c$

- 1 final item.
- No Shift move.



I9I5I3I2I0Q:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow SS$   
 $\rightarrow dA \mid aB$   
 $A \rightarrow bA \mid cB \rightarrow bB \mid c$   
 Sol.  $S' \rightarrow .SS \rightarrow .dA \mid .aBS \rightarrow d.AA \rightarrow .bA \mid .cS \rightarrow a.BB \rightarrow .bB \mid .cA \rightarrow b.AA \rightarrow .bA \mid .cB \rightarrow b.BB \rightarrow .bB \mid .cS' \rightarrow S.I1S \rightarrow dA.I4A \rightarrow c.I6A \rightarrow bA.I7S \rightarrow aB.I8B \rightarrow c.I10B \rightarrow bB.I11S$   
 daAbccbABbccBb  
 • 1 final item.  
 • No Shift move.



# Compiler Design

## Determining the type of Grammar – Set 2

Compiler Design Determining the type of Grammar - Set 2

### Outcome

- ☆ Two solved problems on how to determine the type of a given grammar.

Outcome ☆ Two solved problems on how to determine the type of a given grammar.

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$S \rightarrow A \mid a$$

$$A \rightarrow a$$

Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1): $S \rightarrow A \mid a$   $A \rightarrow a$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.

	FIRST	FOLLOW
$S \rightarrow A \mid a$	{a}	{\$}
$A \rightarrow a$	{a}	{\$}

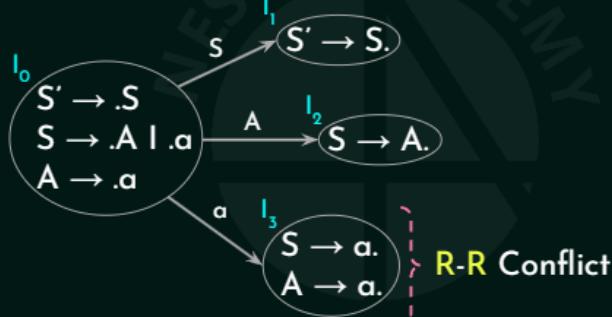
	a	\$
S	$S \rightarrow A$	
	$S \rightarrow a$	
A	$A \rightarrow a$	

LL(1) Parsing Table

Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
TableaSA\$ $S \rightarrow A \mid a$   $A \rightarrow a$  FOLLOW{\$}FIRST{a}{\$}{a}S  $\rightarrow AS \rightarrow aA \rightarrow a$  Sol.

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow S$   
 $S \rightarrow A \mid a$   
 $A \rightarrow a$



$$\text{FOLLOW}(S) \cap \text{FOLLOW}(A) = \{\$\} \neq \emptyset$$

I3I0Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
Sol.  $S' \rightarrow SS \rightarrow A \mid a$   
 $A \rightarrow a$   
Conflict  $\text{FOLLOW}(S) \cap \text{FOLLOW}(A) = \{\$\} \neq \emptyset$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$S \rightarrow A \mid a$   
 $A \rightarrow a$

Ambiguous

a



Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow A \mid a$   
 $A \rightarrow a \mid S$   
Ambiguous

**Q2:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$S \rightarrow (L) \mid a$$

$$L \rightarrow L, S \mid S$$

Q2:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow (L) \mid a$   $L \rightarrow L, S \mid S$

**Q2:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

**Sol.**

	FIRST	FOLLOW
$S \rightarrow (L) \mid a$	$\{(, a\}$	$\{ \$, , , )\}$
$L \rightarrow L, S \mid S$	$\{(, a\}$	$\{ , , )\}$

	(	)	,	a	\$
S	$S \rightarrow (L)$			$S \rightarrow a$	
L	$L \rightarrow S$			$L \rightarrow S$	

LL(1) Parsing Table

{\$Q2:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow (L) \mid a$   $L \rightarrow L, S \mid S$  FOLLOWFIRST{ $(, a\} \{ , , )\} \{ (, a\}$ }  
Sol., , , }LL(1) Parsing Table(aSL\$,,)S → (L)L → SL → SS → a

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$S \rightarrow (L) \mid a$$

$$L \rightarrow LS \mid S$$

### Top down Parser:

In order to construct Top down parsers the Context Free Grammars should **not** have,

1. Left Recursion,
2. Non-determinism.

Q2:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow (L) \mid a$   
 $L \rightarrow LS \mid S$

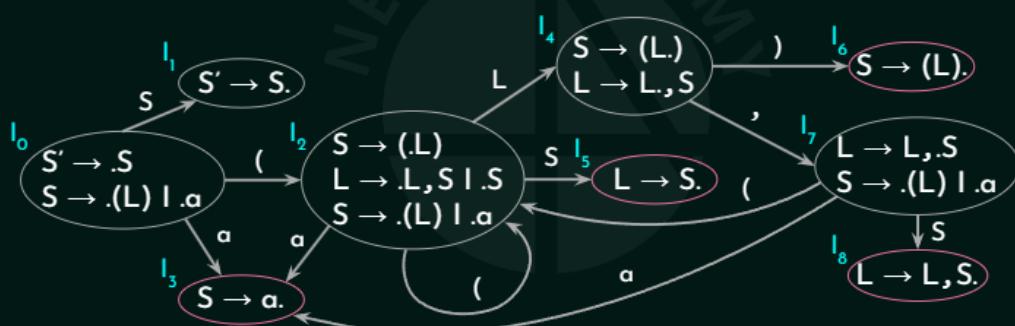
Top down Parser: In order to construct Top down parsers the Context Free Grammars should not have,  
 1.Left Recursion, 2.Non-determinism.

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow S$

$$S \rightarrow (L) \mid a$$

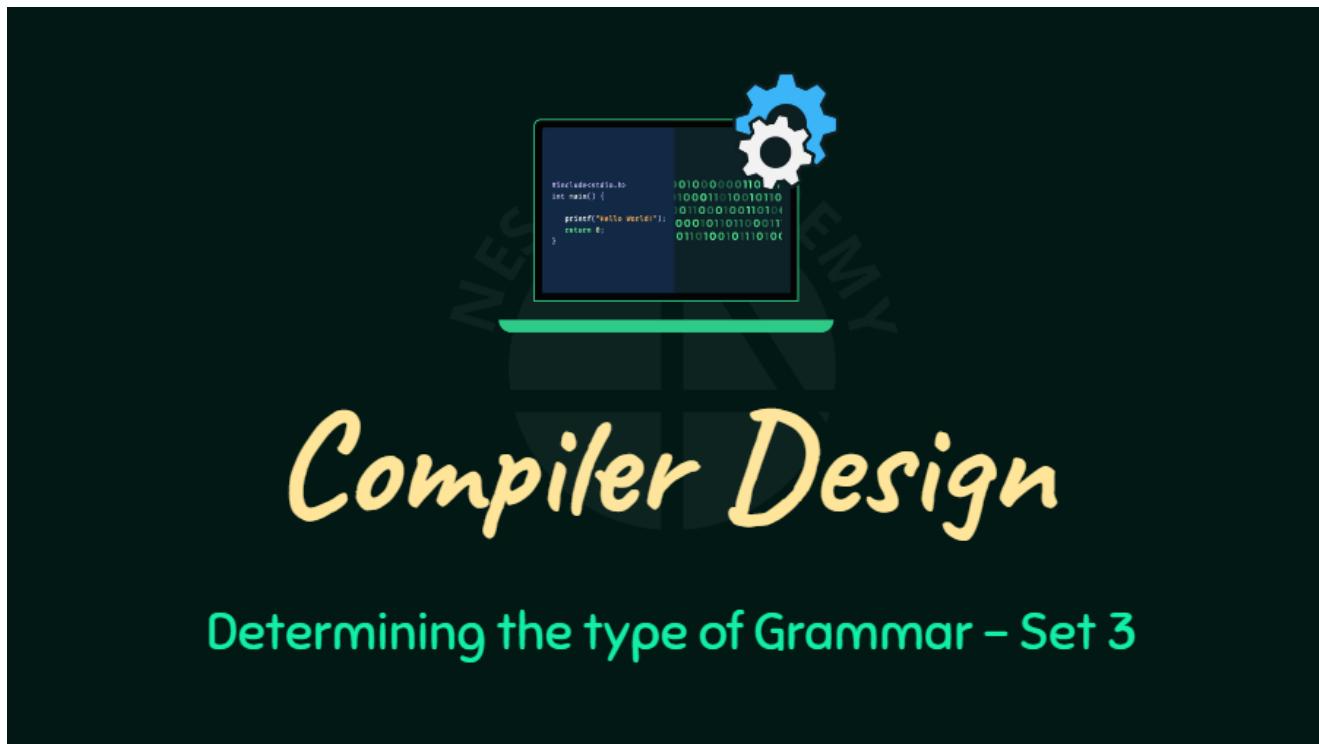
$$L \rightarrow LS \mid S$$



- 1 final item.
- No Shift move.

I7I2I0Q2:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 Sol.  
 $S' \rightarrow SS$   
 $\rightarrow (L) \mid a$   
 $L \rightarrow LS \mid S$   
 $SS' \rightarrow .SS \rightarrow .(L) \mid a$   
 $S \rightarrow (.L)L \rightarrow .L.S \mid .SS \rightarrow .(L) \mid a$   
 $S \rightarrow a.S \rightarrow (L)L \rightarrow L.S \mid L.SL \rightarrow L.SS \rightarrow .(L) \mid a$   
 $S' \rightarrow S$ .I1S(I3aI4L  $\rightarrow S$ .I5LSS  $\rightarrow (L).I6a()$ ,SL  $\rightarrow L.S$ .I8(a●1 final

item.●No Shift move.



# Compiler Design

## Determining the type of Grammar – Set 3

Compiler Design Determining the type of Grammar - Set 3



### Outcome

- ☆ Solved problem on how to determine the type of a given grammar.

Outcome☆Solved problem on how to determine the type of a given grammar.

Q: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$E \rightarrow E + T \mid T$$

$$T \rightarrow i$$

Q: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$\underline{E} \rightarrow \underline{E} + T \mid T$$

$$T \rightarrow i$$

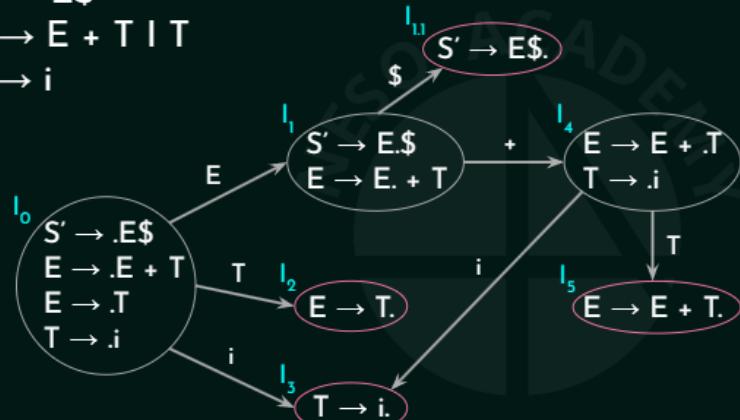
Q:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $E \rightarrow E + T \mid TT \rightarrow i$

Q: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$\text{Sol. } S' \rightarrow E\$$$

$$E \rightarrow E + T \mid T$$

$$T \rightarrow i$$



- 1 final item.
- No Shift move.

I4I1Q:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T \mid T$   
 $T \rightarrow i$

Q: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$E \rightarrow T + E \mid T$$

$$T \rightarrow i$$

Q:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $E \rightarrow T + E \mid T$   $T \rightarrow i$

Q: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.

	FIRST	FOLLOW
$E \rightarrow T + E \mid T$	{i}	{\$}
$T \rightarrow i$	{i}	{+, \$}

	+	i	\$
E		$E \rightarrow T + E$	
T		$E \rightarrow T$	

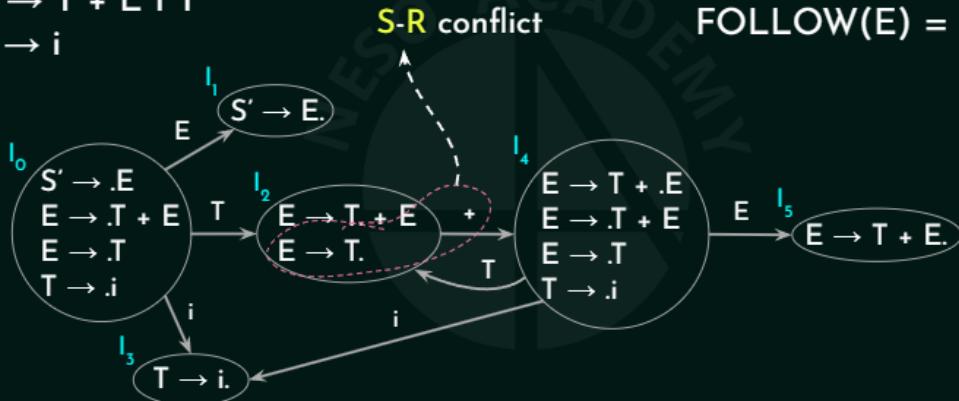
LL(1) Parsing Table

Q:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $E \rightarrow T + E \mid T$   $T \rightarrow i$   
 $\text{FOLLOW}(\$) = \{i\}$   $\text{FIRST}\{i\} = \{+, \$\}$

Sol.  $S' \rightarrow E$

$$\begin{aligned} E &\rightarrow T + E \mid T \\ T &\rightarrow i \end{aligned}$$

$$\text{FOLLOW}(E) = \{\$\}$$



Q:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow EE \rightarrow T + E \mid T$   $T \rightarrow i$

Sol.  $S' \rightarrow .EE \rightarrow .T + E$   $E \rightarrow .T$   $T \rightarrow .i$

$E \rightarrow T + .E$   $E \rightarrow .T + E$   $E \rightarrow .T$   $T \rightarrow .i$

$I_0: S' \rightarrow .EE$ ,  $E \rightarrow .T + E$ ,  $E \rightarrow .T$ ,  $T \rightarrow .i$

$I_1: S' \rightarrow .EE$ ,  $E \rightarrow T + .E$ ,  $E \rightarrow .T + E$ ,  $E \rightarrow .T$ ,  $T \rightarrow .i$

$I_2: S' \rightarrow .EE$ ,  $E \rightarrow T + .E$ ,  $E \rightarrow .T + E$ ,  $E \rightarrow .T$ ,  $T \rightarrow .i$

$I_3: S' \rightarrow .EE$ ,  $E \rightarrow T + .E$ ,  $E \rightarrow .T + E$ ,  $E \rightarrow .T$ ,  $T \rightarrow .i$

$I_4: S' \rightarrow .EE$ ,  $E \rightarrow T + .E$ ,  $E \rightarrow .T + E$ ,  $E \rightarrow .T$ ,  $T \rightarrow .i$

$I_5: S' \rightarrow .EE$ ,  $E \rightarrow T + .E$ ,  $E \rightarrow .T + E$ ,  $E \rightarrow .T$ ,  $T \rightarrow .i$

$I_6: S' \rightarrow .EE$ ,  $E \rightarrow T + .E$ ,  $E \rightarrow .T + E$ ,  $E \rightarrow .T$ ,  $T \rightarrow .i$



# Compiler Design

## Determining the type of Grammar – Set 4

Compiler Design Determining the type of Grammar - Set 4



### Outcome

- ☆ Two solved problems on how to determine the type of a given grammar.

Outcome ☆ Two solved problems on how to determine the type of a given grammar.

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$E \rightarrow E + T \mid T$$

$$T \rightarrow TF \mid F$$

$$F \rightarrow F^* \mid a \mid b$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$\underline{E} \rightarrow E + T \mid T$$

$$T \rightarrow TF \mid F$$

$$F \rightarrow F^* \mid a \mid b$$

$$\text{FOLLOW}(E) = \{+, \$\}$$

$$\text{FOLLOW}(T) = \{a, b, +, \$\}$$

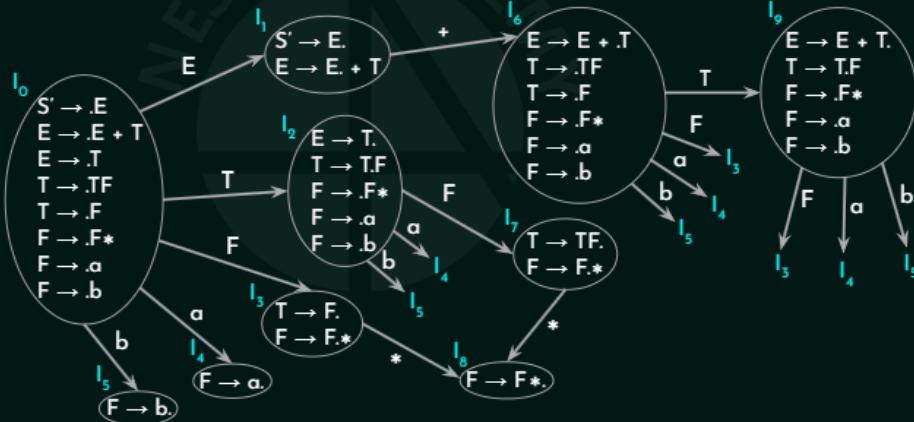
$$\text{FOLLOW}(F) = \{*, a, b, +, \$\}$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $E \rightarrow E + T \mid TT$   
 $T \rightarrow TF \mid F$   
 $F \rightarrow F^* \mid a \mid b$   
 $\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\} = \{+^{**}\}$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

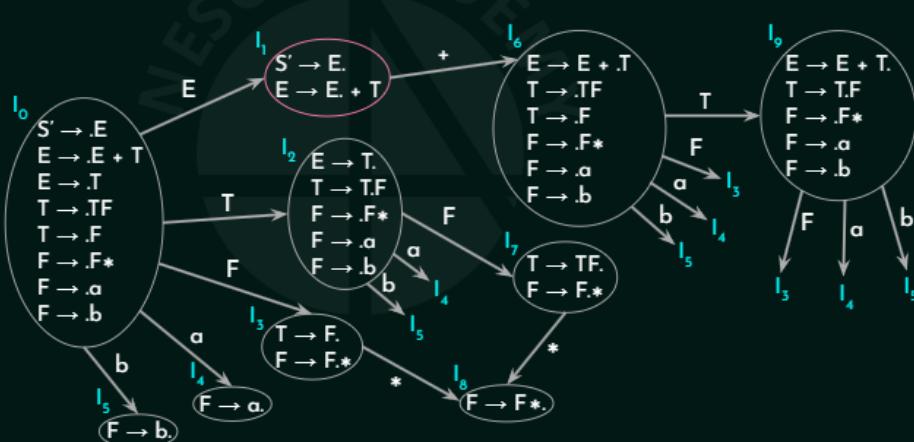


I6I0I9Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow EE \rightarrow E + T E \rightarrow TT \rightarrow TF T \rightarrow FF \rightarrow F+F \rightarrow aF \rightarrow b$   
Sol. \* FOLLOW(E) = {+, \$}  
FOLLOW(T) = {a, b, +, \$}  
FOLLOW(F) = {+, a, b, +, \$}  
\*  $S' \rightarrow .EE \rightarrow .E + T E \rightarrow .TT \rightarrow .TF T \rightarrow .FF \rightarrow .F+F \rightarrow .aF \rightarrow .b$   
 $.E \rightarrow T.T \rightarrow .TF T \rightarrow .FF \rightarrow .F+F \rightarrow .aF \rightarrow .b$   
 $.E \rightarrow E + .TF T \rightarrow .FF \rightarrow .aF \rightarrow .b$   
 $.T \rightarrow T.F T \rightarrow .FF \rightarrow .aF \rightarrow .b$   
 $.F \rightarrow F+F \rightarrow .aF \rightarrow .b$   
 $.E \rightarrow E + T I1 T \rightarrow F.F \rightarrow F+.+ I3 F \rightarrow a. I4 F \rightarrow b. I5 b a F T E + T \rightarrow T F.F \rightarrow F+.+ I7 I2 F a I4 b I5 F \rightarrow F+.+ I8 * T F I3 I4 I5 a b * I3 I5 I4 b a F$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

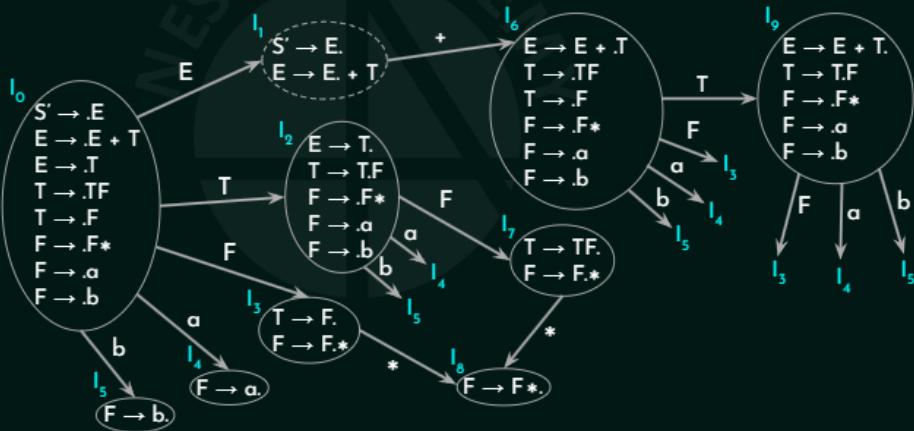


I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow EE \rightarrow E + T$   
 $E \rightarrow TT \rightarrow TF$   
 $T \rightarrow FF \rightarrow F+F \rightarrow aF \rightarrow b$   
**Sol.** \*FOLLOW(E) = {+, \$}  
**FOLLOW(T) = {a, b, +, \$}**  
**FOLLOW(F) = {+, a, b, +, \$}**  
 $*S' \rightarrow .EE \rightarrow .E + T$   
 $E \rightarrow .TT \rightarrow .TF$   
 $T \rightarrow .FF \rightarrow .F+F \rightarrow .aF \rightarrow .b$   
 $.b^*E \rightarrow .T$   
 $.T \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T$   
 $T \rightarrow .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+^*I3F \rightarrow a.I4F \rightarrow b.I5baF$   
 $TE+T \rightarrow .TF.F \rightarrow .aF \rightarrow .b^*I7I2FaI4bI5F \rightarrow F+.I8^*TFI3I4I5ab*I3I5I4baF$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

**Sol.**  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

**FOLLOW(E) = {+, \$}**  
**FOLLOW(T) = {a, b, +, \$}**  
**FOLLOW(F) = {\*, a, b, +, \$}**

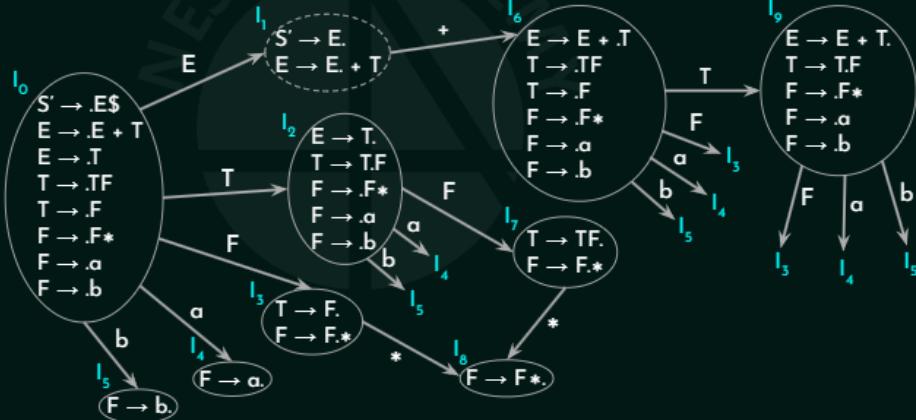


I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow TT$   
 $T \rightarrow FF$   
 $F \rightarrow F+F \rightarrow aF \rightarrow b$   
**Sol.** \*FOLLOW(E) = {+, \$}  
**FOLLOW(T) = {a, b, +, \$}**  
**FOLLOW(F) = {+, a, b, +, \$}**  
 $*S' \rightarrow .EE \rightarrow .E + T$   
 $E \rightarrow .TT \rightarrow .TF$   
 $T \rightarrow .FF \rightarrow .F+F \rightarrow .aF \rightarrow .b$   
 $.b^*E \rightarrow .T$   
 $.T \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T$   
 $T \rightarrow .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+^*I3F \rightarrow a.I4F \rightarrow b.I5baF$   
 $TE+T \rightarrow .TF.F \rightarrow .aF \rightarrow .b^*I7I2FaI4bI5F \rightarrow F+.I8^*TFI3I4I5ab*I3I5I4baF$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$



I6I0I9Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

Sol. \*  
 $\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

$S' \rightarrow .E\$$   
 $E \rightarrow .E + T$   
 $E \rightarrow .T$   
 $T \rightarrow .TF$   
 $T \rightarrow .F$   
 $F \rightarrow .F*$   
 $F \rightarrow .a$   
 $F \rightarrow .b$

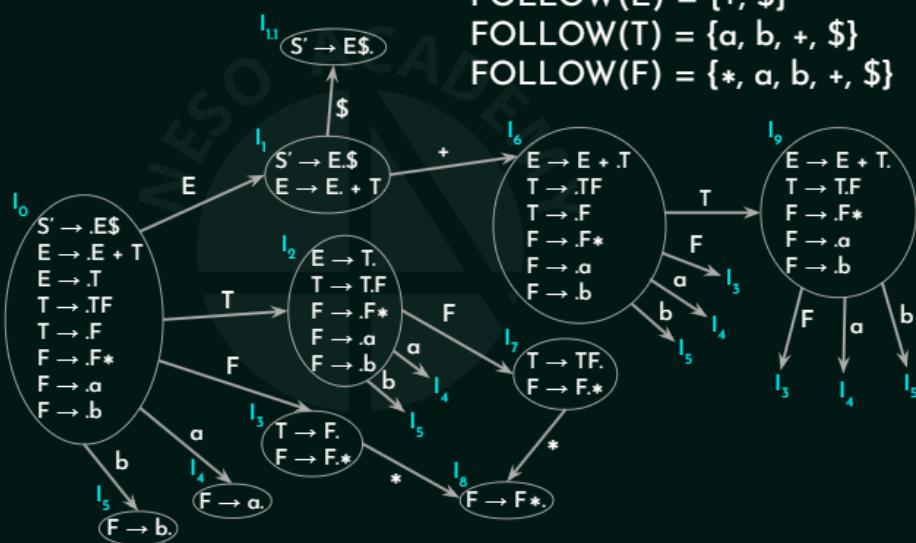
$\text{I}_0: S' \rightarrow E\$$   
 $\text{I}_1: S' \rightarrow E\$$ ,  $E \rightarrow E + T$   
 $\text{I}_2: E \rightarrow T.$ ,  $T \rightarrow T.F$ ,  $F \rightarrow F*$ ,  $F \rightarrow a$ ,  $F \rightarrow b$   
 $\text{I}_3: T \rightarrow F.$ ,  $F \rightarrow F.*$   
 $\text{I}_4: F \rightarrow a$   
 $\text{I}_5: F \rightarrow b$   
 $\text{I}_6: E \rightarrow E + .T$ ,  $T \rightarrow .TF$ ,  $T \rightarrow .F$ ,  $F \rightarrow .F*$ ,  $F \rightarrow .a$ ,  $F \rightarrow .b$   
 $\text{I}_7: T \rightarrow .TF$ ,  $F \rightarrow .F.*$   
 $\text{I}_8: F \rightarrow F.*$   
 $\text{I}_9: E \rightarrow E + .T$ ,  $T \rightarrow .TF$ ,  $F \rightarrow .F*$ ,  $F \rightarrow .a$ ,  $F \rightarrow .b$

Transitions:  
 $I_0 \xrightarrow{E} I_1$ ,  $I_0 \xrightarrow{T} I_2$ ,  $I_1 \xrightarrow{+} I_6$ ,  $I_2 \xrightarrow{+} I_6$ ,  $I_6 \xrightarrow{T} I_3$ ,  $I_3 \xrightarrow{a} I_4$ ,  $I_3 \xrightarrow{b} I_5$ ,  $I_4 \xrightarrow{a} I_5$ ,  $I_4 \xrightarrow{b} I_5$ ,  $I_2 \xrightarrow{F} I_7$ ,  $I_7 \xrightarrow{a} I_4$ ,  $I_7 \xrightarrow{b} I_5$ ,  $I_5 \xrightarrow{*} I_8$ ,  $I_8 \xrightarrow{F} I_3$ ,  $I_3 \xrightarrow{a} I_4$ ,  $I_3 \xrightarrow{b} I_5$ ,  $I_0 \xrightarrow{b} I_5$ ,  $I_4 \xrightarrow{b} I_5$ .

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$*FOLLOW(E) = \{+, \$\}$   
 $FOLLOW(T) = \{a, b, +, \$\}$   
 $FOLLOW(F) = \{+, a, b, +, \$\}$

$S' \rightarrow .E\$E \rightarrow .E + T E \rightarrow .TT \rightarrow .TF T \rightarrow .FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow .T.T \rightarrow .T.FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T. T \rightarrow T.F F \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.\$E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+*I3F \rightarrow a.I4F \rightarrow b.I5baFTE+T \rightarrow TF.F \rightarrow F.+*I7FaI4bI5F \rightarrow F+.I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$I1.1\$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

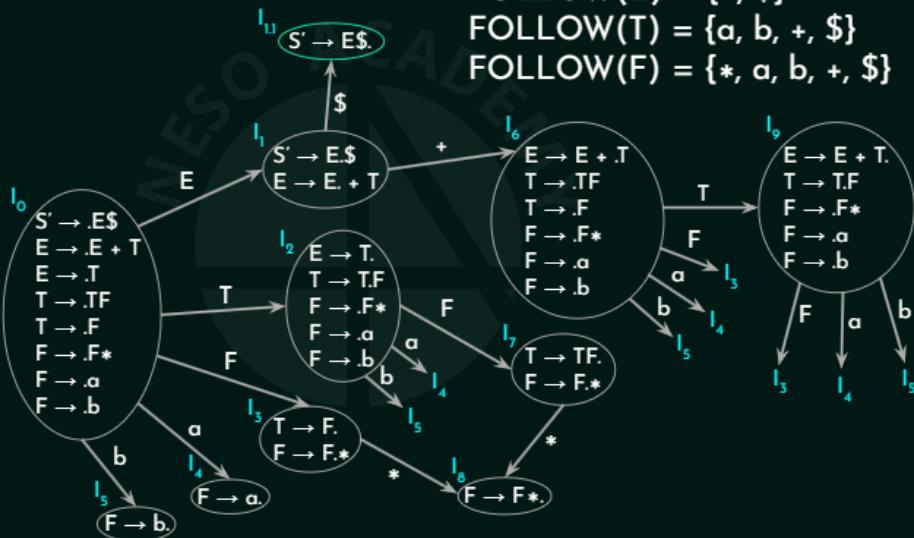
$T \rightarrow TF$

$T \rightarrow F$

$F \rightarrow F^*$

$F \rightarrow a$

$F \rightarrow b$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F^*$   
 $F \rightarrow a$   
 $F \rightarrow b$

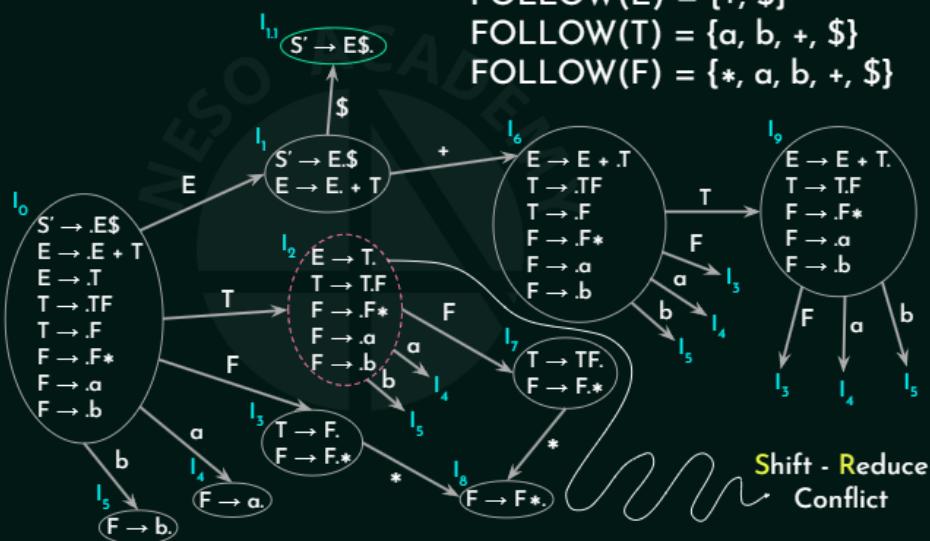
$*FOLLOW(E) = \{+, \$\}$   
 $FOLLOW(T) = \{a, b, +, \$\}$   
 $FOLLOW(F) = \{+, a, b, +, \$\}$

$S' \rightarrow .E\$E \rightarrow .E + T E \rightarrow .TT \rightarrow .TF T \rightarrow .FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow .T.T \rightarrow .T.FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T. T \rightarrow T.F F \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.\$E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+*I3F \rightarrow a.I4F \rightarrow b.I5baFTE+T \rightarrow TF.F \rightarrow F.+*I7FaI4bI5F \rightarrow F+.I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$I1.1\$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$



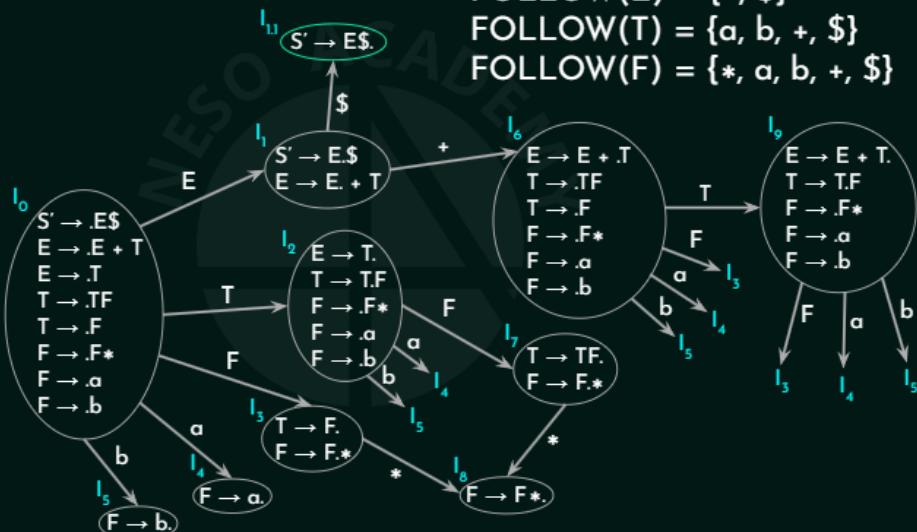
Shift - Reduce Conflict

I2I6I0I9Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$$   
 $E \rightarrow E + T$   $E \rightarrow TT$   $E \rightarrow TF$   $T \rightarrow FF$   $T \rightarrow F+F$   $F \rightarrow aF$   $F \rightarrow bF$   
 $\text{Sol. } * \text{ FOLLOW}(E) = \{+, \$\}$   $\text{FOLLOW}(T) = \{a, b, +, \$\}$   $\text{FOLLOW}(F) = \{+, a, b, +, \$\}$   
 $* S' \rightarrow .E\$$   $E \rightarrow .E + T$   $E \rightarrow .TT$   $E \rightarrow .TF$   $T \rightarrow .FF$   $T \rightarrow .F+F$   $F \rightarrow .aF$   $F \rightarrow .bF$   
 $.aF \rightarrow .b^*E$   $.b^*E \rightarrow E + .TF$   $.TF \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b^*T$   $.b^*T \rightarrow .TFT$   
 $.TFT \rightarrow .FE$   $.FE \rightarrow E + T$   $T \rightarrow .TF$   $F \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b^*S'$   $.b^*S' \rightarrow E.\$$   $E \rightarrow E + .T$   
 $T \rightarrow .TF$   $F \rightarrow .FF$   $.TF \rightarrow .F+F$   $.F+F \rightarrow .aF$   $.aF \rightarrow .b^*F$   $.b^*F \rightarrow F+.+*$   $I3F \rightarrow a.I4F$   
 $\rightarrow b.I5ba$   $FTE + T \rightarrow TF.F$   $F \rightarrow F.+*$   $I7FaI4bI5F \rightarrow F+.+*$   $I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$.I1.1$$   
- Shift - Reduce Conflict

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

$S' \rightarrow E\$E \rightarrow .E\$E \rightarrow .E + T E \rightarrow .TT \rightarrow .TF T \rightarrow .FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow .T.T \rightarrow .T.FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T. T \rightarrow T.F F \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.\$E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+*I3F \rightarrow a.I4F \rightarrow b.I5baFTE+T \rightarrow TF.F \rightarrow F.+*I7FaI4bI5F \rightarrow F+.I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$I1.1\$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

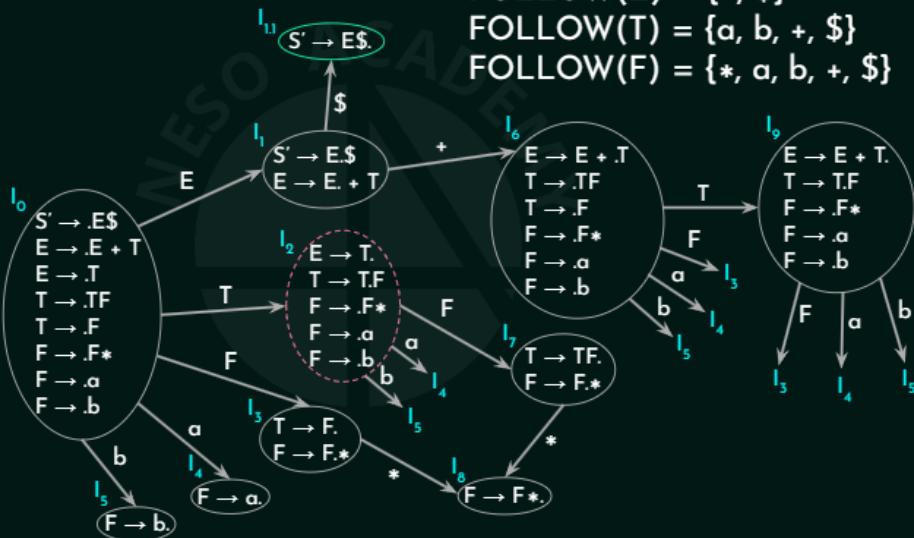
$T \rightarrow TF$

$T \rightarrow F$

$F \rightarrow F^*$

$F \rightarrow a$

$F \rightarrow b$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F^*$   
 $F \rightarrow a$   
 $F \rightarrow b$

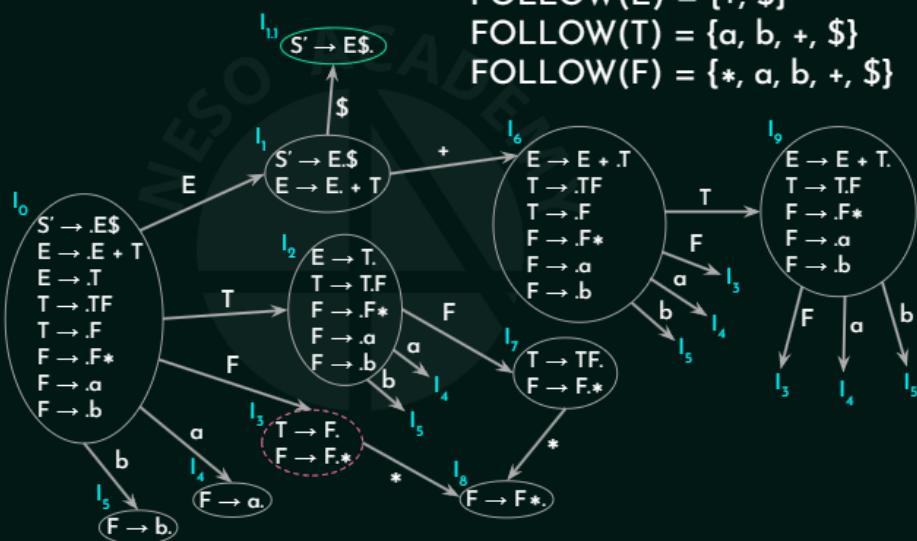
$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

$S' \rightarrow E\$E \rightarrow .E\$E \rightarrow .E + T E \rightarrow .TT \rightarrow .TF T \rightarrow .FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow .T.T \rightarrow .T.FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T. T \rightarrow T.F F \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.\$E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+*I3F \rightarrow a.I4F \rightarrow b.I5baFTE+T \rightarrow TF.F \rightarrow F.+*I7FaI4bI5F \rightarrow F+.I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$I1.1\$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

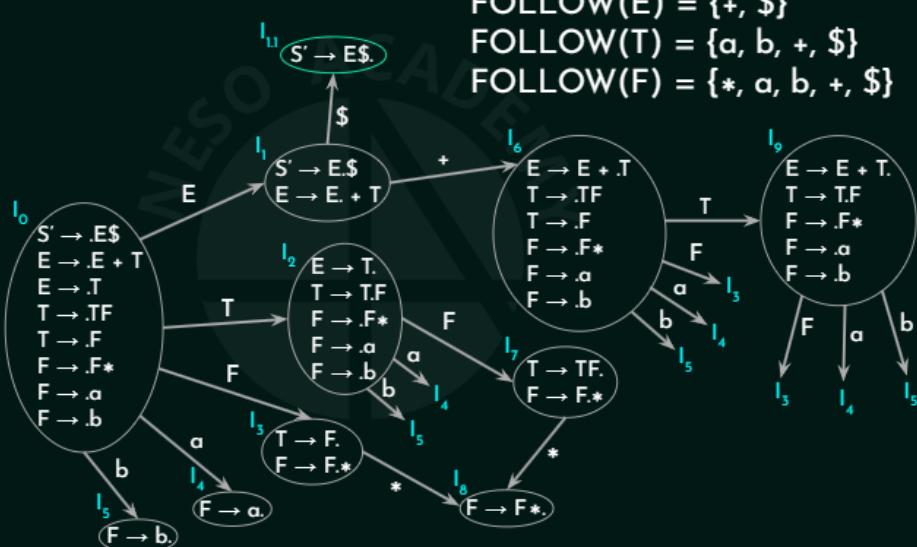


I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$$   
 $E \rightarrow E + T$   $E \rightarrow TT$   $E \rightarrow TF$   $T \rightarrow FF$   $T \rightarrow F+F$   $F \rightarrow aF$   $F \rightarrow bF$   
 $\text{Sol. } * \text{ FOLLOW}(E) = \{+, \$\}$   $\text{FOLLOW}(T) = \{a, b, +, \$\}$   $\text{FOLLOW}(F) = \{+, a, b, +, \$\}$   
 $* \text{ S}' \rightarrow .E\$$   $E \rightarrow .E + T$   $E \rightarrow .TT$   $E \rightarrow .TF$   $T \rightarrow .FF$   $T \rightarrow .F+F$   $.aF \rightarrow .b*E$   $.E \rightarrow .T.T$   $.T \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b*E$   $.E \rightarrow .E + .TF$   $.TF \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b*T$   $.T \rightarrow .TFT$   $.TFT \rightarrow .FE$   $.FE \rightarrow .E + T$   $.T \rightarrow .T.F$   $.F \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b*S'$   $S' \rightarrow E.\$$   $E \rightarrow E + T$   $T \rightarrow F$   $F \rightarrow F*$   $F \rightarrow a$   $F \rightarrow b$   
 $I_1 \xrightarrow{E} I_0$   $I_1 \xrightarrow{T} I_2$   $I_1 \xrightarrow{+} I_6$   $I_2 \xrightarrow{T} I_3$   $I_2 \xrightarrow{F} I_4$   $I_3 \xrightarrow{a} I_4$   $I_3 \xrightarrow{b} I_5$   $I_4 \xrightarrow{a} I_5$   $I_4 \xrightarrow{b} I_6$   $I_5 \xrightarrow{*} I_8$   $I_6 \xrightarrow{T} I_9$   $I_9 \xrightarrow{F} I_3$   $I_9 \xrightarrow{a} I_4$   $I_9 \xrightarrow{b} I_5$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

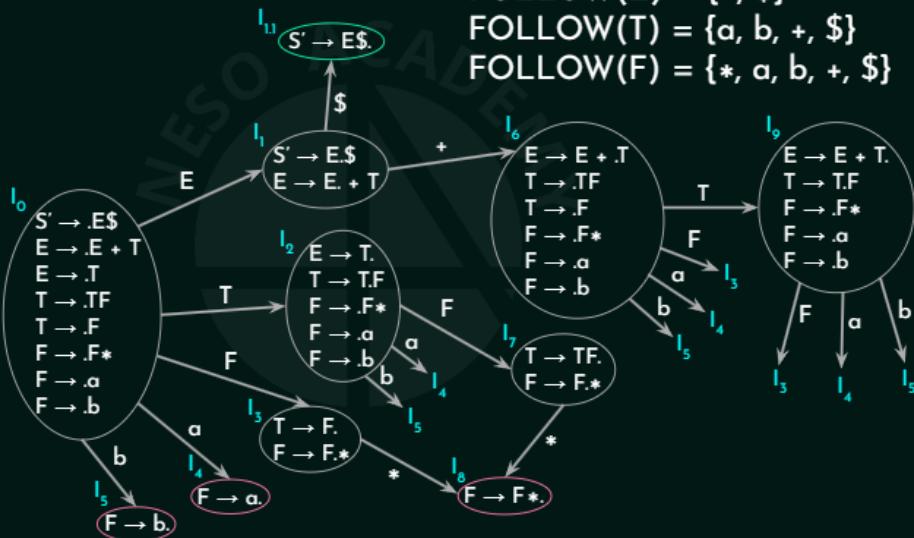
$T \rightarrow TF$

$T \rightarrow F$

$F \rightarrow F*$

$F \rightarrow a$

$F \rightarrow b$



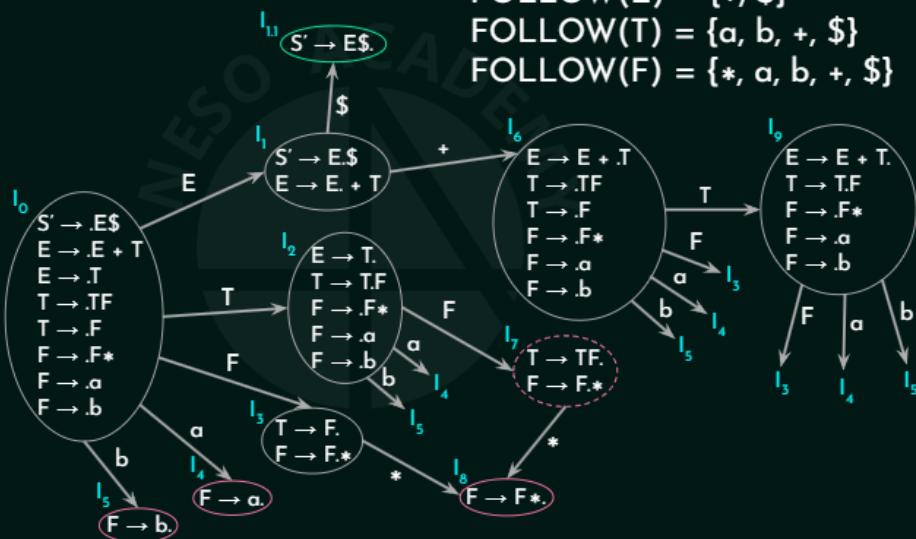
I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

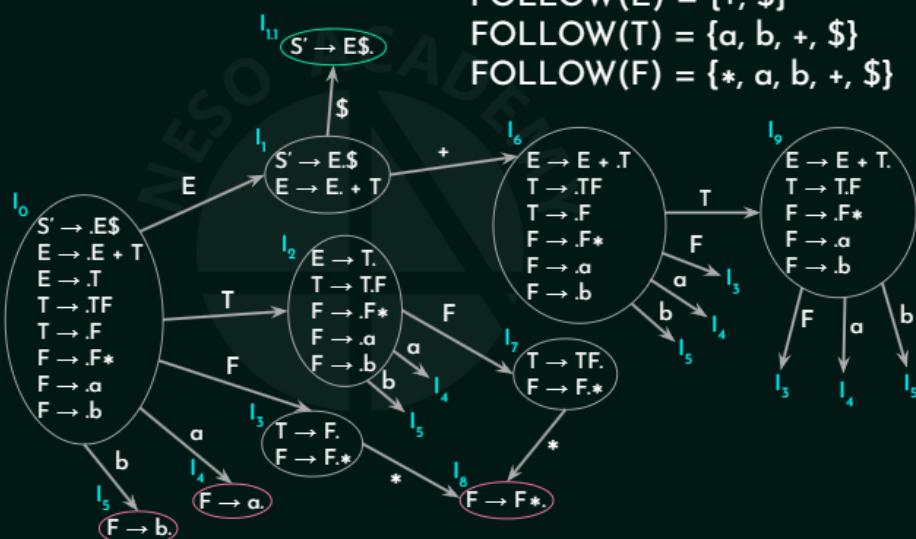


I2I6I0I9Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$$   
 $E \rightarrow E + T$   $E \rightarrow TT$   $E \rightarrow TF$   $T \rightarrow FF$   $T \rightarrow F+F$   $F \rightarrow aF$   $F \rightarrow bF$   
 $\text{SOL. } * \text{ FOLLOW}(E) = \{+, \$\}$   $\text{FOLLOW}(T) = \{a, b, +, \$\}$   $\text{FOLLOW}(F) = \{+, a, b, +, \$\}$   
 $* S' \rightarrow .E\$$   $E \rightarrow .E + T$   $E \rightarrow .TT$   $E \rightarrow .TF$   $T \rightarrow .FF$   $T \rightarrow .F+F$   $F \rightarrow .aF$   $F \rightarrow .bF$   
 $.aF \rightarrow .b^*E$   $.b^*E \rightarrow .E + .TF$   $.TF \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b^*T$   $.b^*T \rightarrow .TFT$   $.TFT \rightarrow .FE$   $.FE \rightarrow .E + .T$   $.T \rightarrow .TF$   $.TF \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b^*S'$   $.b^*S' \rightarrow E.\$$   $E \rightarrow E + T$   $T \rightarrow F$   $F \rightarrow F*$   $F \rightarrow a$   $F \rightarrow b$   
 $I_5 \xrightarrow{ba} I_6$   $I_6 \xrightarrow{T} I_7$   $I_7 \xrightarrow{F} I_8$   $I_8 \xrightarrow{a} I_9$   $I_8 \xrightarrow{b} I_5$   $I_9 \xrightarrow{a} I_4$   $I_9 \xrightarrow{b} I_5$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$*FOLLOW(E) = \{+, \$\}$   
 $FOLLOW(T) = \{a, b, +, \$\}$   
 $FOLLOW(F) = \{+, a, b, +, \$\}$

$S' \rightarrow .E\$E \rightarrow .E + T E \rightarrow .TT \rightarrow .TF T \rightarrow .FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow .T.T \rightarrow .T.FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T. T \rightarrow T.F F \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.\$E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+*I3F \rightarrow a.I4F \rightarrow b.I5baFTE+T \rightarrow TF.F \rightarrow F.+*I7FaI4bI5F \rightarrow F+.I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$I1.1\$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$

$E \rightarrow E + T$

$E \rightarrow T$

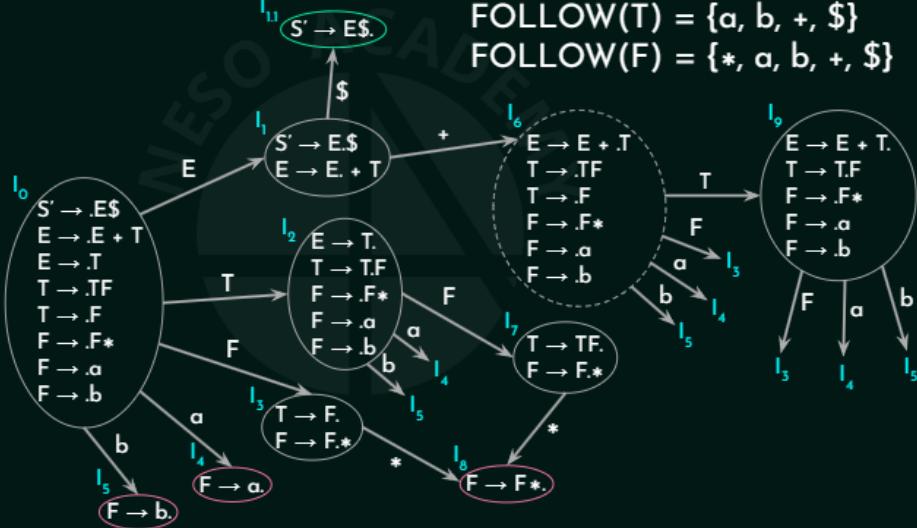
$T \rightarrow TF$

$T \rightarrow F$

$F \rightarrow F^*$

$F \rightarrow a$

$F \rightarrow b$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

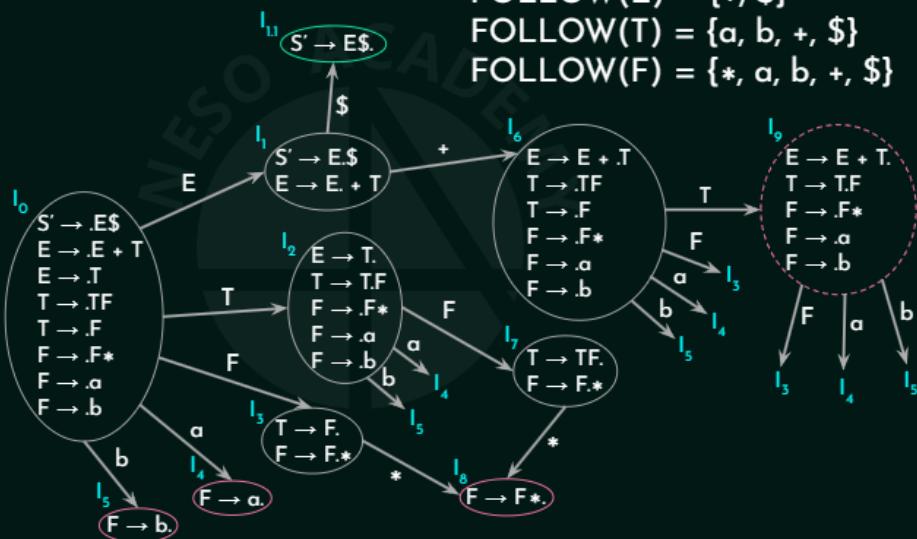
$*FOLLOW(E) = \{+, \$\}$   
 $FOLLOW(T) = \{a, b, +, \$\}$   
 $FOLLOW(F) = \{+, a, b, +, \$\}$

$S' \rightarrow .E\$E \rightarrow .E + T E \rightarrow .TT \rightarrow .TF T \rightarrow .FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow .T.T \rightarrow .T.FF \rightarrow .FF \rightarrow .aF \rightarrow .b^*E \rightarrow E + .TF \rightarrow .FF \rightarrow .aF \rightarrow .b^*T \rightarrow .TFT \rightarrow .FE \rightarrow E + T. T \rightarrow T.F F \rightarrow .FF \rightarrow .aF \rightarrow .b^*S' \rightarrow E.\$E \rightarrow E. + TI1T \rightarrow F.F \rightarrow F.+*I3F \rightarrow a.I4F \rightarrow b.I5baFTE+T \rightarrow TF.F \rightarrow F.+*I7FaI4bI5F \rightarrow F+.I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$I1.1\$$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$

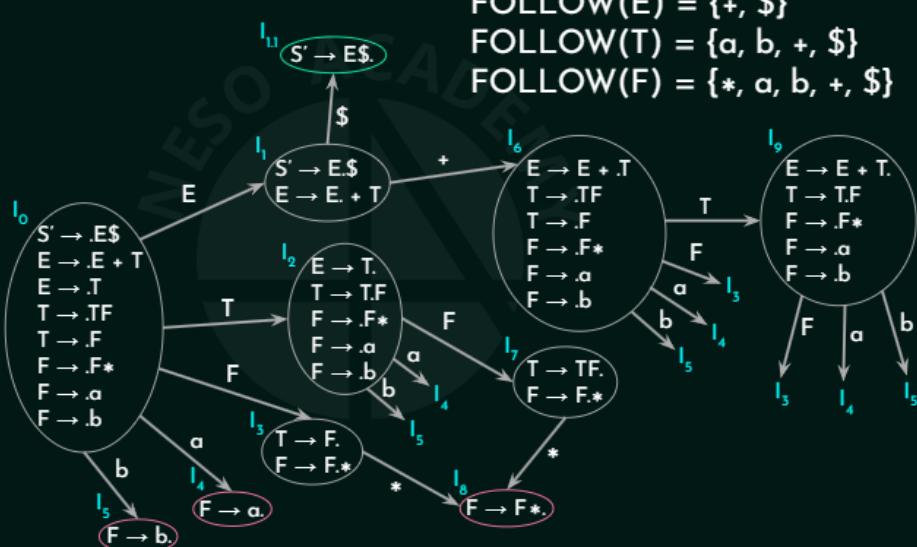


I2I6I0I9Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$$   
 $E \rightarrow E + T$   $E \rightarrow TT$   $E \rightarrow TF$   $T \rightarrow FF$   $T \rightarrow F+F$   $F \rightarrow aF$   $F \rightarrow bF$   
 $S' \rightarrow .E\$$   $E \rightarrow .E + T$   $E \rightarrow .TT$   $E \rightarrow .TF$   $T \rightarrow .FF$   $T \rightarrow .F+F$   $F \rightarrow .aF$   $F \rightarrow .bF$   
 $.aF \rightarrow .b^*E$   $.E \rightarrow .T.T$   $.T \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b^*E$   $.b^*E \rightarrow E + .TF$   $.TF \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b^*T$   $.b^*T \rightarrow .TFT$   $.TFT \rightarrow .FE$   $.FE \rightarrow E + T$   $.T \rightarrow .T.F$   $.F \rightarrow .FF$   $.FF \rightarrow .aF$   $.aF \rightarrow .b^*S'$   $.b^*S' \rightarrow E.\$$   $E \rightarrow E + T$   $T \rightarrow F$   $F \rightarrow F*$   $F \rightarrow a$   $F \rightarrow b$   
 $I1 \rightarrow I2$   $I2 \rightarrow I3$   $I3 \rightarrow I4$   $I4 \rightarrow I5$   $I5 \rightarrow I6$   $I6 \rightarrow I7$   $I7 \rightarrow I8$   $I8 \rightarrow I9$   $I9 \rightarrow I10$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow E\$$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow TF$   
 $T \rightarrow F$   
 $F \rightarrow F*$   
 $F \rightarrow a$   
 $F \rightarrow b$

$\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{*, a, b, +, \$\}$



I2I6I0I9Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow E\$E$   
 $E \rightarrow E + T$   $E \rightarrow TT$   $T \rightarrow TF$   $T \rightarrow FF$   $F \rightarrow F+F$   $F \rightarrow aF$   $F \rightarrow b$   
Sol.  
 $\text{FOLLOW}(E) = \{+, \$\}$   
 $\text{FOLLOW}(T) = \{a, b, +, \$\}$   
 $\text{FOLLOW}(F) = \{+, a, b, +, \$\}$   
 $S' \rightarrow .E\$E$   $\rightarrow .E + T$   $E \rightarrow .TT$   $T \rightarrow .TF$   $T \rightarrow .FF$   $F \rightarrow .FF$   
 $F \rightarrow .aF$   $F \rightarrow .b^*E$   $E \rightarrow .T$   $T \rightarrow .FF$   $F \rightarrow .aF$   $F \rightarrow .b^*E$   $E \rightarrow .+$   $T \rightarrow .TF$   $T \rightarrow .FF$   $F \rightarrow .aF$   $F \rightarrow .b^*T$   $T \rightarrow .TFT$   
 $T \rightarrow .FE$   $E \rightarrow .T$   $T \rightarrow .FF$   $F \rightarrow .aF$   $F \rightarrow .b^*S'$   $S' \rightarrow E.\$E$   $\rightarrow E.+$   $T1T \rightarrow F.F$   $F \rightarrow F.+^*I3F \rightarrow a.I4F$   
 $\rightarrow b.I5baFTE+T \rightarrow TF.F$   $F \rightarrow F.+^*I7FaI4bI5F \rightarrow F+.+^*I8*TFI3I4I5ab*I3I5I4baFS' \rightarrow E.\$.I1.1\$$

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$S \rightarrow AaAb \mid BbBa$$

$$A \rightarrow \epsilon$$

$$B \rightarrow \epsilon$$

Q2:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow AaAb \mid BbBa$   
 $A \rightarrow \epsilon$   
 $B \rightarrow \epsilon$

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.

	FIRST	FOLLOW
$S \rightarrow AaAb \mid BbBa$	{a, b}	{\$}
$A \rightarrow \epsilon$	{ε}	{a, b}
$B \rightarrow \epsilon$	{ε}	{b, a}

	a	b	\$
S	$S \rightarrow AaAb$	$S \rightarrow BbBa$	
A	$A \rightarrow \epsilon$	$A \rightarrow \epsilon$	
B	$B \rightarrow \epsilon$	$B \rightarrow \epsilon$	

LL(1) Parsing Table

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $\{ \$ \} S \rightarrow AaAb \mid BbBa$   
 $A \rightarrow \epsilon$   
 $B \rightarrow \epsilon$

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow S$

$S \rightarrow AaAb \mid BbBa$

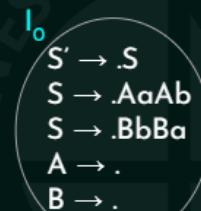
$A \rightarrow \epsilon$

$B \rightarrow \epsilon$

Note:

For productions like,  
 $A \rightarrow \epsilon$ , the LR(0) items:

$A \rightarrow \epsilon \equiv A \rightarrow \epsilon. \equiv A \rightarrow .$



Reduce-Reduce  
Conflict

I0Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow SS \rightarrow AaAb \mid BbBa$   
 $A \rightarrow \epsilon$   
 $B \rightarrow \epsilon$

**Q2:** Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

**Sol.**  $S' \rightarrow S$

$S \rightarrow AaAb \mid BbBa$

$A \rightarrow \epsilon$

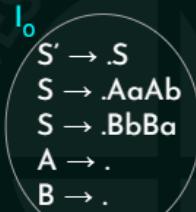
$B \rightarrow \epsilon$

**Note:**

For productions like,

$A \rightarrow \epsilon$ , the LR(0) items:

$A \rightarrow \epsilon \equiv A \rightarrow \epsilon. \equiv A \rightarrow .$



$\boxed{\text{FOLLOW}(A) \cap \text{FOLLOW}(B) \neq \emptyset}$

$\neq \phi$  I0Q2:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow SS \rightarrow AaAb \mid BbBa$   
 $A \rightarrow \epsilon$   
 $B \rightarrow \epsilon$   
**Sol.**  $S' \rightarrow .SS \rightarrow .AaAb$   $S \rightarrow .BbBa$   $A \rightarrow .B \rightarrow .$   
**Note:** For productions like,  $A \rightarrow \epsilon$ , the LR(0) items:  
 $A \rightarrow .\epsilon \equiv A \rightarrow \epsilon. \equiv A \rightarrow .$   
 $\text{FOLLOW}(A) \cap \text{FOLLOW}(B) \neq \emptyset$



# Compiler Design

## Determining the type of Grammar – Set 5



## Outcome

- ☆ Two solved problems on how to determine the type of a given grammar.

Outcome☆Two solved problems on how to determine the type of a given grammar.

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$\begin{aligned} S &\rightarrow AS \mid b \\ A &\rightarrow SA \mid a \end{aligned}$$

Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow AS \mid b$   
 $A \rightarrow SA \mid a$

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.

	FIRST	FOLLOW
$S \rightarrow AS \mid b$	{b, a}	{\$, a, b}
$A \rightarrow SA \mid a$	{a, b}	{b, a}

	a	b	\$
S	$S \rightarrow AS$	$S \rightarrow AS$ $S \rightarrow b$	
A	$A \rightarrow a$	$A \rightarrow SA$	

LL(1) Parsing Table

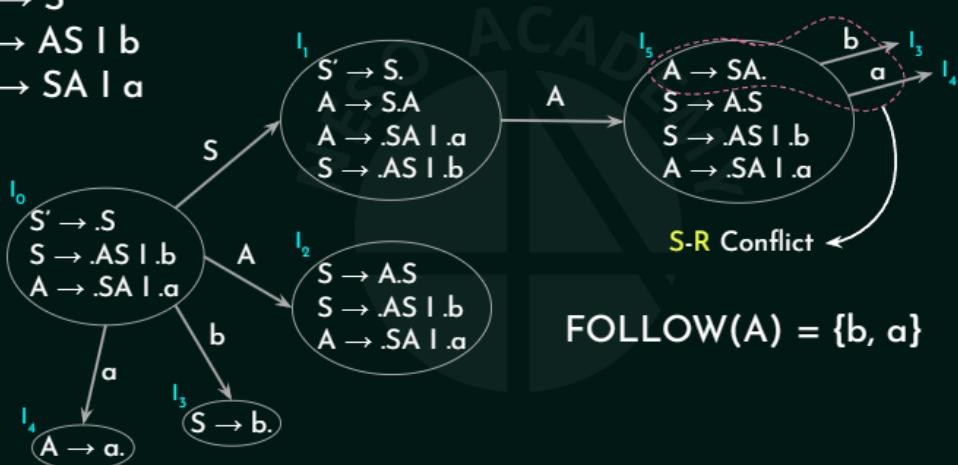
Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow ASS \rightarrow ASS \rightarrow bA \rightarrow SAA \rightarrow a, a, b\}, a\}, b\}S \rightarrow AS \mid bA \rightarrow SA \mid a$   
 FOLLOWFIRST{b, a}\{b\{a\\$\}  
 Sol. LL(1)  
 Parsing Table  
 A\$ b S

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow S$

$S \rightarrow AS \mid b$

$A \rightarrow SA \mid a$



$\text{FOLLOW}(A) = \{b, a\}$

I5I1Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow SS \rightarrow AS \mid bA \rightarrow SA \mid a$   
 Sol.  $S' \rightarrow .SS \rightarrow .AS \mid .bA \rightarrow .SA \mid .a$   
 $S' \rightarrow S.A \rightarrow S.AA \rightarrow .SA \mid .aS \rightarrow .AS \mid .b$   
 $SS \rightarrow A.SS \rightarrow .AS \mid .bA \rightarrow .SA \mid .a$   
 $AS \rightarrow b.I_3 \rightarrow a.I_4$   
 $baA \rightarrow SA.S \rightarrow A.SS \rightarrow .AS \mid .bA$

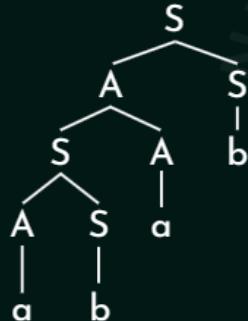
$\rightarrow .SA \mid .aA \mid 3b \mid 4aS - R$  Conflict FOLLOW(A) = {b, a}

Q1: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$\begin{aligned} S &\rightarrow AS \mid b \\ A &\rightarrow SA \mid a \end{aligned}$$

Ambiguous

abab



Q1:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow AS \mid b$   
 $A \rightarrow SA \mid a$   
aababSSAASSAababSASASSAabbaAmbiguous

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

$$\begin{aligned} S &\rightarrow Aa \mid bAc \mid dc \mid bda \\ A &\rightarrow d \end{aligned}$$

Q2:Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S \rightarrow Aa \mid bAc \mid dc \mid bda$   
 $A \rightarrow d$

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.

	FIRST	FOLLOW
$S \rightarrow Aa \mid bAc \mid dc \mid bda$	{d, b}	{\$}
$A \rightarrow d$	{d}	{a, c}

	a	b	c	d	\$
S		$S \rightarrow bAc$ $S \rightarrow bda$		$S \rightarrow dc$ $S \rightarrow Aa$	
A				$A \rightarrow d$	

LL(1) Parsing Table

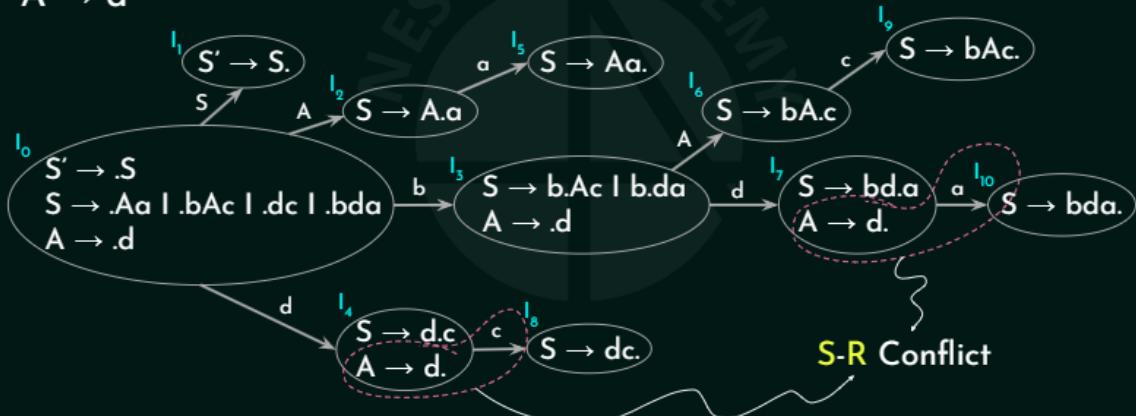
Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $b\} \{d\} S \rightarrow Aa \mid bAc \mid dc \mid bda$   
 $A \rightarrow d$   
FOLLOWFIRST{a, c}{d}{\$}  
Sol. LL(1) Parsing Table  
leadSA\$cbA → dS → bAcS → dcS → AaS → bda

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):

Sol.  $S' \rightarrow S$

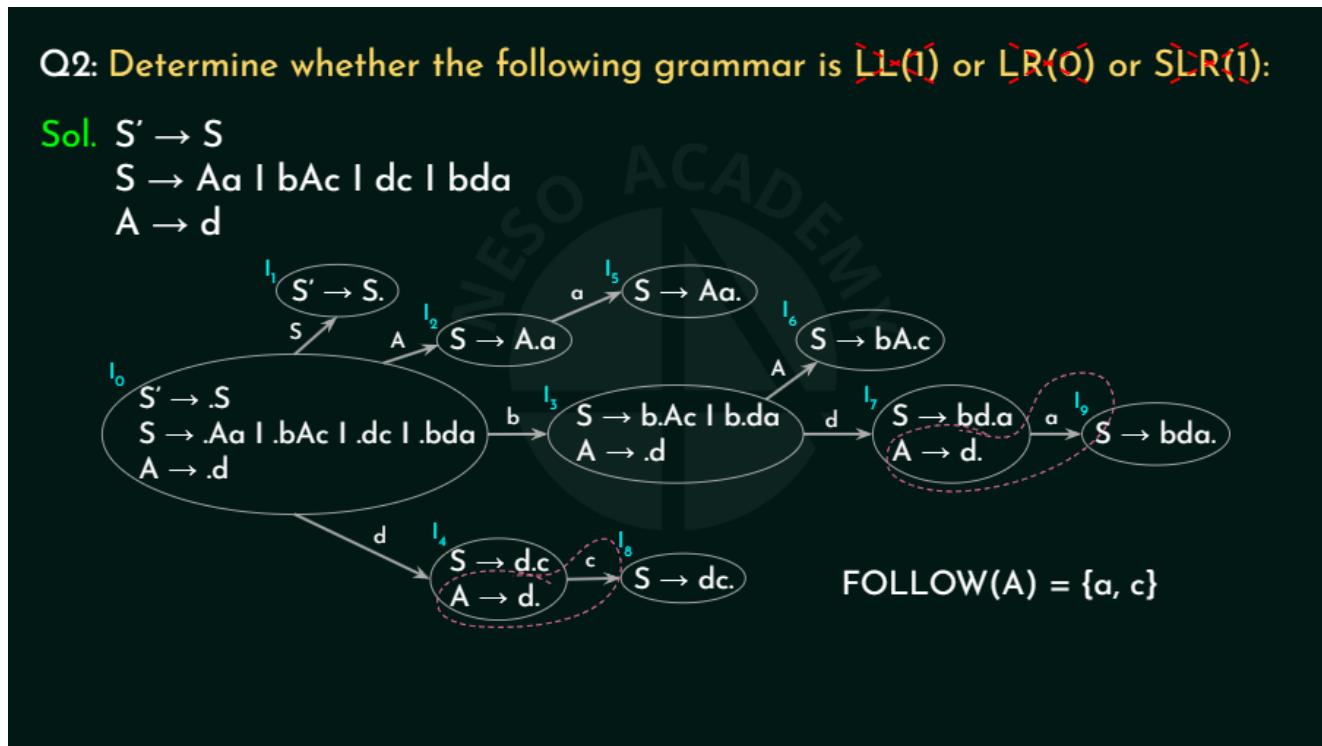
$S \rightarrow Aa \mid bAc \mid dc \mid bda$

$A \rightarrow d$



I4  $S \rightarrow b.Ac \mid b.da$   
 $A \rightarrow .dS' \rightarrow .S$   
 $S \rightarrow .Aa \mid .bAc \mid .dc \mid .bda$   
 $A \rightarrow .d$   
I1  $S \rightarrow Aa \mid bAc \mid dc \mid bda$   
 $A \rightarrow d$   
Sol.  $S \rightarrow d.c$   
 $A \rightarrow d.S \rightarrow bd.a$   
 $A \rightarrow d.S' \rightarrow S$ . I1  $S \rightarrow A.a$  I2  $A \rightarrow bd.S \rightarrow Aa$ . I5  $a \rightarrow dc$ . I8  $c \rightarrow S$  →

bA.cl6AdS → bda.l10aS-R ConflictS → bAc.l9c



$= \{a, c\}S \rightarrow b.Ac \mid b.daA \rightarrow .dS' \rightarrow .S S \rightarrow .Aa \mid .bAc \mid .dc \mid .bdaA \rightarrow .d$

Q2: Determine whether the following grammar is LL(1) or LR(0) or SLR(1):  
 $S' \rightarrow S$   $S \rightarrow Aa \mid bAc \mid dc \mid bda$   $A \rightarrow d$

Sol.  
 $S \rightarrow A.a$   
 $S \rightarrow d.c$   
 $S \rightarrow bAc$   
 $S \rightarrow b.da$   
 $S \rightarrow bda$



Compiler Design Canonical Collection of LR(1) items



## Outcome

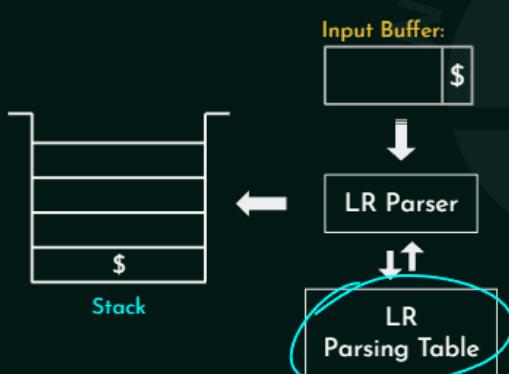
- ☆ Understanding LR(1) items.
- ☆ Derivation of the Canonical Collection of LR(1) items from a given Grammar.

Outcome  
☆ Understanding LR(1) items.  
☆ Derivation of the Canonical Collection of LR(1) items from a given Grammar.

### LR Parsers:

Use Right most derivation - In reverse.

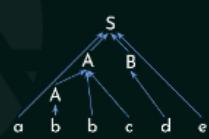
Scan from Left to right.



#### Generation of Parse Tree – Bottom up approach:

$$\begin{array}{l} S \rightarrow aAb \\ A \rightarrow Abc \mid b \\ B \rightarrow d \end{array}$$

Decision:  
When to reduce.

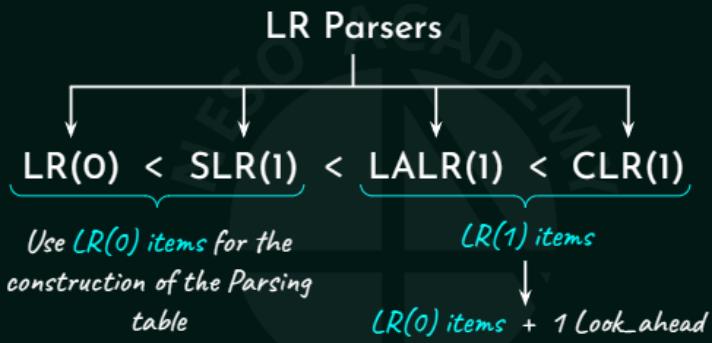


$$\begin{aligned} S &\Rightarrow aAb \\ &\Rightarrow aAde \\ &\Rightarrow aAbcde \\ &\Rightarrow aabcde \end{aligned}$$

(Right most Derivation) - In reverse.

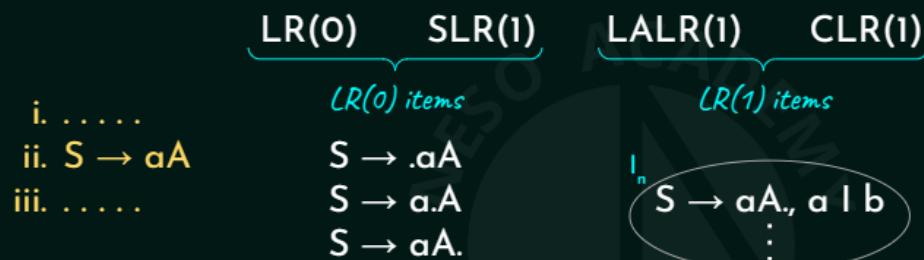
LR Parsers: \$  
LR Parser  
LR Parsing Table  
Input Buffer  
Stack  
Scan from Left to right.  
Use Right most derivation - In reverse.

## LR Parsers:



LR Parsers: LR Parsers  
 LR(0) CLR(1) LALR(1) SLR(1) <<< Use LR(0) items for the construction of the Parsing table  
 LR(1) items LR(0) items + 1 Look\_ahead

## LR Parsers:



States	Action			
	a	b	c	\$
n	$r_{ii}$	$r_{ii}$		

i. .... ii.  $S \rightarrow aA$  iii. .... LR Parsers: LR(0) CLR(1) LALR(1) SLR(1) LR(1) items LR(0) items  $S \rightarrow .aAS \rightarrow a.AS \rightarrow aA.I bS \rightarrow aA., a$  Inriirii States cbn Action a \$

## Canonical collection of LR(1) items:

$S' \rightarrow S$   
 $S \rightarrow AA$   
 $A \rightarrow aA \mid b$

$S' \rightarrow .S, \$$      $\text{FIRST}(\$) = \{\$\}$   
 $S \rightarrow .AA, \$$

### *Rules:*

1. For Start symbol's production, the lookahead is '\$'.
2. In the closure set, the lookahead of the LR(1) items are determined based on the rest of the items' FIRST set.
3. Lookahead will never change during transitions, however, may change within state while determining closure set.

$= \{\$\}$  Canonical collection of LR(1) items:  
 $S' \rightarrow SS \rightarrow AAA \rightarrow aA \mid b$   
 Rules:  
 1. For Start symbol's production, the lookahead is '\$'.  
 2. In the closure set, the lookahead of the LR(1) items are determined based on the rest of the items' FIRST set.  
 3. Lookahead will never change during transitions, however, may change within state while determining closure set.  
 $S' \rightarrow .S, \$$   
 $S \rightarrow .AA, \$$      $\text{FIRST}(A, \$)$   
 $A \rightarrow .aA$   
 $A \rightarrow .b$

## Canonical collection of LR(1) items:

$S' \rightarrow S$   
 $S \rightarrow AA$   
 $A \rightarrow aA \mid b$

$S' \rightarrow .S, \$$   
 $S \rightarrow .AA, \$$      $\text{FIRST}(A, \$)$   
 $A \rightarrow .aA$   
 $A \rightarrow .b$

### *Rules:*

1. For Start symbol's production, the lookahead is '\$'.
2. In the closure set, the lookahead of the LR(1) items are determined based on the rest of the items' FIRST set.
3. Lookahead will never change during transitions, however, may change within state while determining closure set.

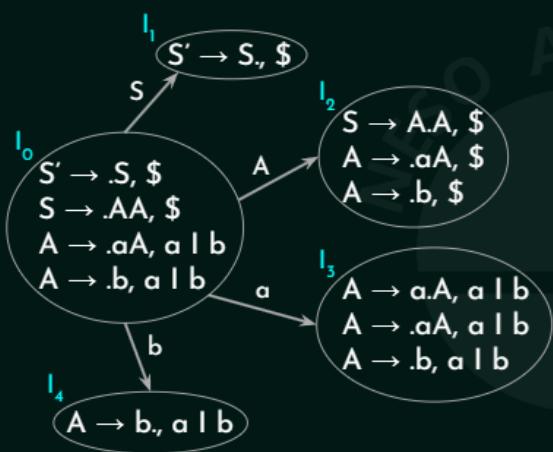
Canonical collection of LR(1) items:  
 $S' \rightarrow SS \rightarrow AAA \rightarrow aA \mid b$   
 Rules:  
 1. For Start symbol's production, the lookahead is '\$'.  
 2. In the closure set, the lookahead heads of the LR(1) items are determined based on the rest of the items' FIRST set.  
 3. Lookahead will never change during transitions, however, may change within state while determining closure set.  
 $S' \rightarrow .S, \$S \rightarrow .AA, \$ \quad FIRST(A) = \{a,b\}$   
 $A \rightarrow .aA, a \mid b \quad A \rightarrow .b, a \mid b$

### Rules:

1. For Start symbol's production, the lookahead is '\$'.
2. In the closure set, the lookahead heads of the LR(1) items are determined based on the rest of the items' FIRST set.
3. Lookahead will never change during transitions, however, may change within state while determining closure set.

Canonical collection of LR(1) items:  
 $S' \rightarrow SS \rightarrow AAA \rightarrow aA \mid b$   
 Rules:  
 1. For Start symbol's production, the lookahead is '\$'.  
 2. In the closure set, the lookahead heads of the LR(1) items are determined based on the rest of the items' FIRST set.  
 3. Lookahead will never change during transitions, however, may change within state while determining closure set.  
 $S' \rightarrow .S, \$S \rightarrow .AA, \$ \quad FIRST(A) = \{a,b\}$   
 $A \rightarrow .aA, a \mid b \quad A \rightarrow .b, a \mid b$

## Canonical collection of LR(1) items:

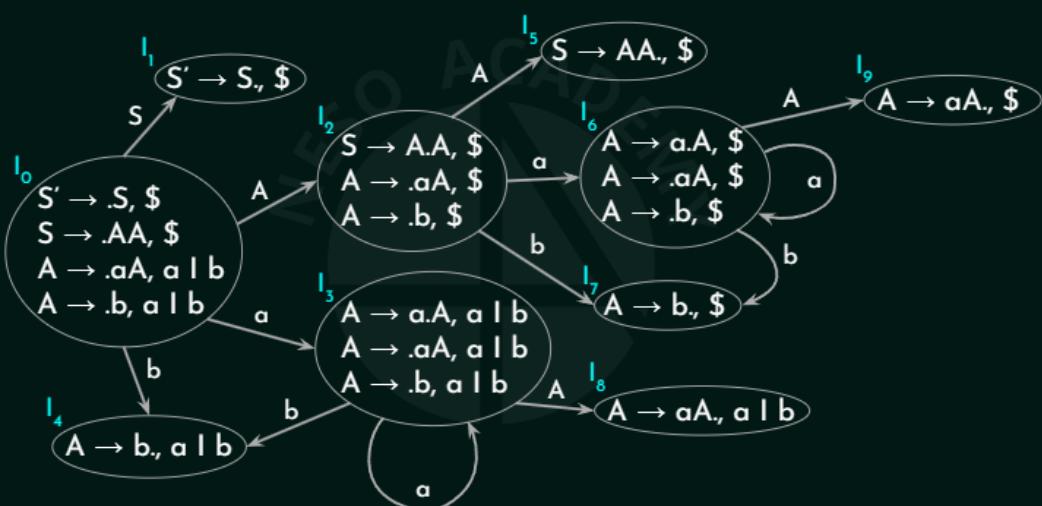


### Rules:

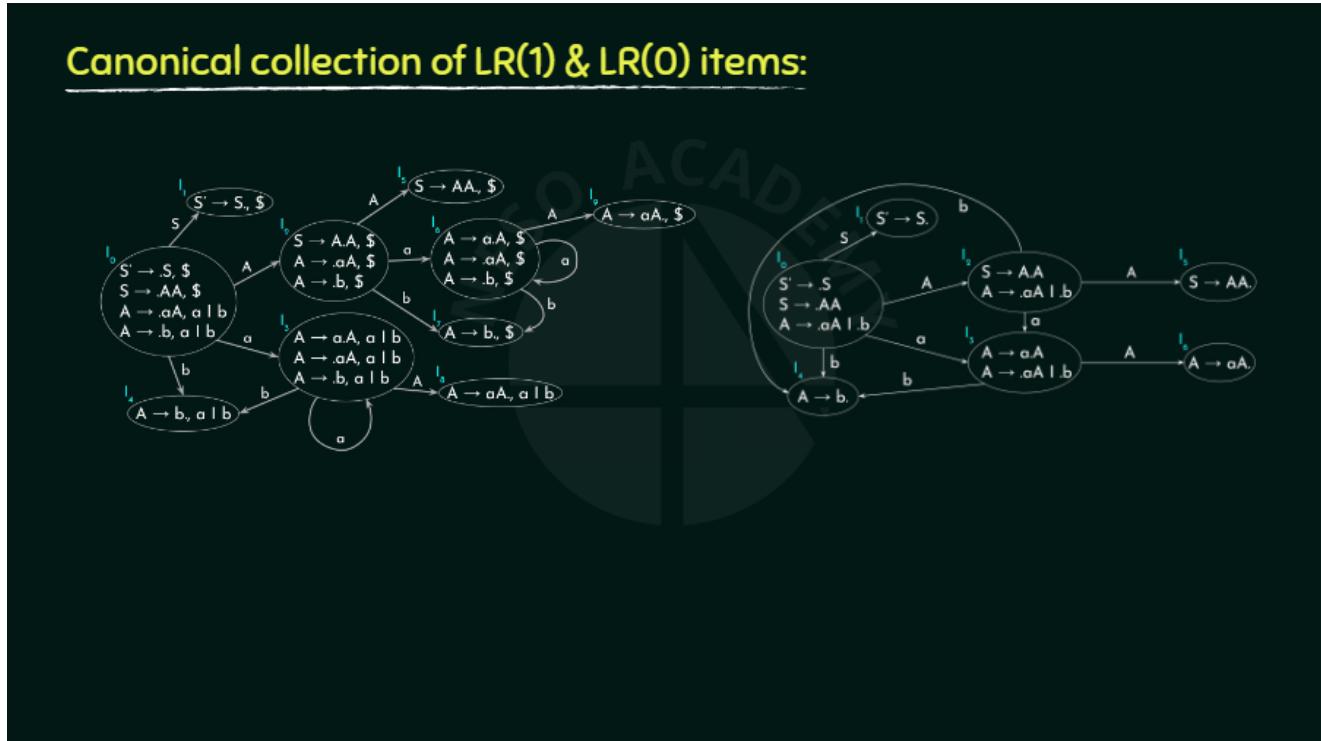
1. For Start symbol's production, the lookahead is '\$'.
2. In the closure set, the lookahead of the LR(1) items are determined based on the rest of the items' FIRST set.
3. Lookahead will never change during transitions, however, may change within state while determining closure set.

Canonical collection of LR(1) items: Rules:  
 1. For Start symbol's production, the lookahead is '\$'.  
 2. In the closure set, the lookahead of the LR(1) items are determined based on the rest of the items' FIRST set.  
 3. Lookahead will never change during transitions, however, may change within state while determining closure set.  
 I2 I3  
 $A \rightarrow .aA \rightarrow .b$   
 $S \rightarrow .S \rightarrow .AA \rightarrow .aA \rightarrow .b$   
 $S \rightarrow .AA \rightarrow .aA \rightarrow .b$   
 $A \rightarrow .aA, a I b \rightarrow .b, a I b$   
 $A \rightarrow .b, a I b \rightarrow .aA, a I b \rightarrow .b, a I b$   
 I0  
 $S' \rightarrow S, \$$   
 $S \rightarrow .AA, \$$   
 $A \rightarrow .aA, a I b$   
 $A \rightarrow .b, a I b$   
 I4  
 $A \rightarrow b., a I b$

## Canonical collection of LR(1) items:



$I_6 I_2 A \rightarrow .aAA \rightarrow .b$   
 Canonical collection of LR(1) items:  
 $S' \rightarrow .S, \$ \rightarrow .AA, \$A \rightarrow .aA, a \mid bA \rightarrow .b, a \mid bS \rightarrow A.A, \$A \rightarrow a.A, a \mid bA \rightarrow b., a \mid b, \$, \$ \rightarrow .b, a \mid bA \rightarrow .aAA \rightarrow .bI_0 S' \rightarrow S., \$I_1 I_4 SAabS \rightarrow AA., \$I_5 AA \rightarrow a.A, \$A \rightarrow .aAA \rightarrow .b, \$, \$aA \rightarrow b., \$I_7 bA \rightarrow aA., a \mid bI_8 I_3 AabA \rightarrow aA., \$I_9 Aab$



Canonical collection of LR(1) & LR(0) items:



Compiler Design CLR(1) Parser



## Outcome

### ★ Construction of CLR(1) Parsing Table.

Outcome★Construction of CLR(1) Parsing Table.

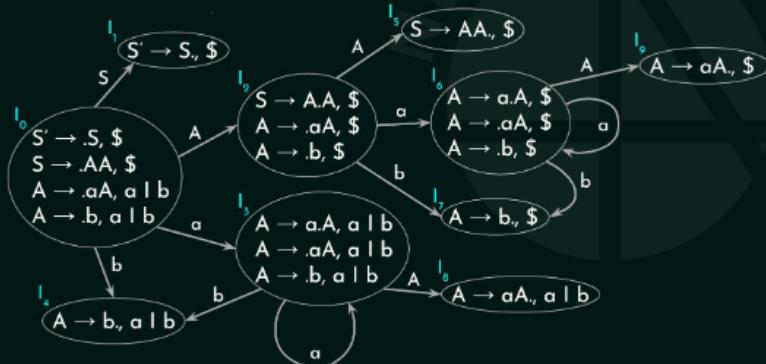
#### Construction of CLR(1) Parsing table:

$$S' \rightarrow S$$

$$\text{i. } S \rightarrow AA$$

$$\text{ii. } A \rightarrow aA$$

$$\text{iii. } A \rightarrow b$$



States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1				Accept	
2	$S_6$	$S_7$		5	
3	$S_3$	$S_4$		8	
4	$r_{iii}$	$r_{iii}$			
5				$r_i$	
6	$S_6$	$S_7$		9	
7				$r_{iii}$	
8	$r_{ii}$	$r_{ii}$			
9				$r_{ii}$	

CLR(1) Parsing Table

Construction of CLR(1) Parsing table:  
 $S' \rightarrow S$ .  
 $i. S \rightarrow AA$   
 $ii. A \rightarrow aA$   
 $iii. A \rightarrow b$   
 Accept  
 1S3S42riiS6S75S3S48riiiriiiriiiriiS6S791092345867riiiAction  
 GO  
 TO States ab\$SACLR(1) Parsing Table

## Comparison:

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

LR(0) Parsing Table

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{iii}$	$r_{iii}$	$r_{iii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

SLR(1) Parsing Table

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_6$	$S_7$		5	
3	$S_3$	$S_4$		8	
4	$r_{iii}$	$r_{iii}$			
5			$r_i$		
6	$S_6$	$S_7$		9	
7			$r_{iii}$		
8	$r_{ii}$	$r_{ii}$			
9			$r_{ii}$		

CLR(1) Parsing Table

Comparison:  
 LR(0) Parsing Table: States 0, 1, 2, 3, 4, 5, 6, Action: a, b, \$, GO TO: 2, 1, 5, 6, r<sub>iii</sub>, r<sub>i</sub>, r<sub>ii</sub>.  
 SLR(1) Parsing Table: States 0, 1, 2, 3, 4, 5, 6, Action: a, b, \$, GO TO: 2, 1, 5, 6, r<sub>iii</sub>, r<sub>i</sub>, r<sub>ii</sub>.  
 CLR(1) Parsing Table: States 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, Action: a, b, \$, GO TO: 2, 1, 5, 8, r<sub>iii</sub>, r<sub>i</sub>, r<sub>ii</sub>.  
 The CLR(1) table includes states 7, 8, 9 which are not present in the LR(0) and SLR(1) tables.  
 The CLR(1) table also includes actions r<sub>iii</sub>, r<sub>i</sub>, r<sub>ii</sub> which are not present in the LR(0) and SLR(1) tables.

# Compiler Design

## LALR(1) Parser

Compiler Design LALR(1) Parser



## Outcome

- ★ Conversion from CLR(1) to LALR(1) Parsing Table.

Outcome★Conversion from CLR(1) to LALR(1) Parsing Table.

### Comparison:

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

LR(0) Parsing Table

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{ii}$		
5	$r_i$	$r_i$	$r_i$		
6	$r_{ii}$	$r_{ii}$	$r_{ii}$		

SLR(1) Parsing Table

Comparison:1ActionGO

TOS States 0 1 2 3 4 5 6 S3 Accept S4 2 S3 S4 S3 S4 rii rii rii rii rii rii rii rii 5 6 ab \$ SALR(0) Parsing

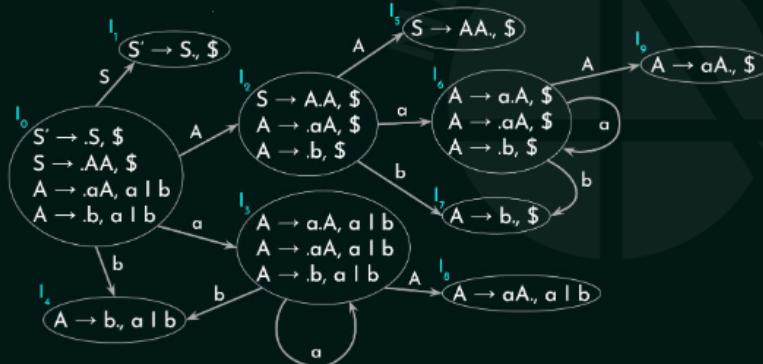
Table 1 Action GO TO States 0 1 2 3 4 5 6 S3 Accept S4 2 S3 S4 S3 S4 rii rii rii 5 6 ab \$ SArii rii rii rii SLR(1)

Parsing Table

## Construction of CLR(1) Parsing table:

$$S' \rightarrow S$$

- i.  $S \rightarrow AA$
- ii.  $A \rightarrow aA$
- iii.  $A \rightarrow b$

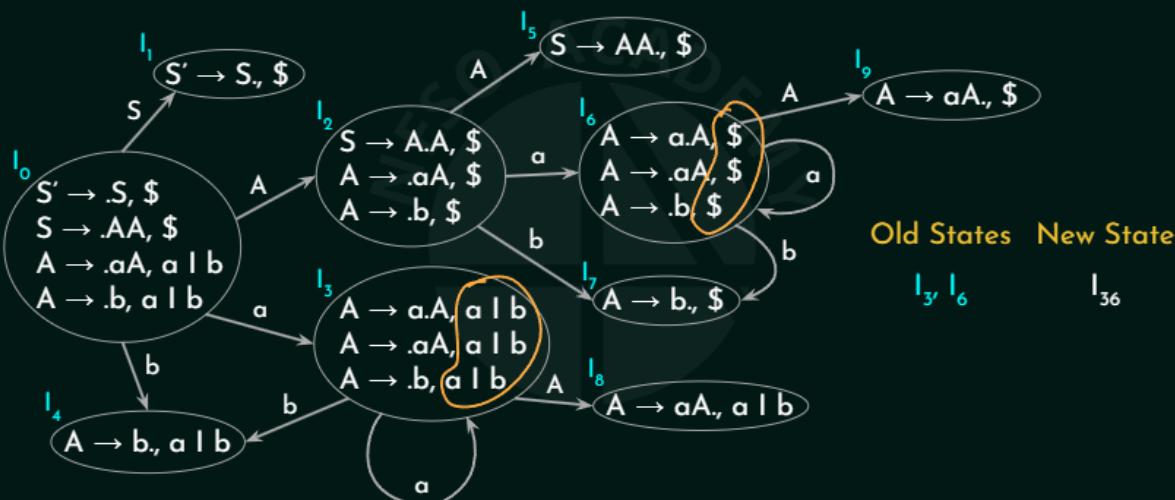


States	Action			GO TO	
	a	b	\$	A	S
I_0	S_3	S_4		2	1
I_1			Accept		
I_2	S_6	S_7		5	
I_3	S_3	S_4		8	
I_4	r_iii	r_iii			
I_5			r_i		
I_6	S_6	S_7		9	
I_7			r_iii		
I_8	r_ii	r_ii			
I_9			r_ii		

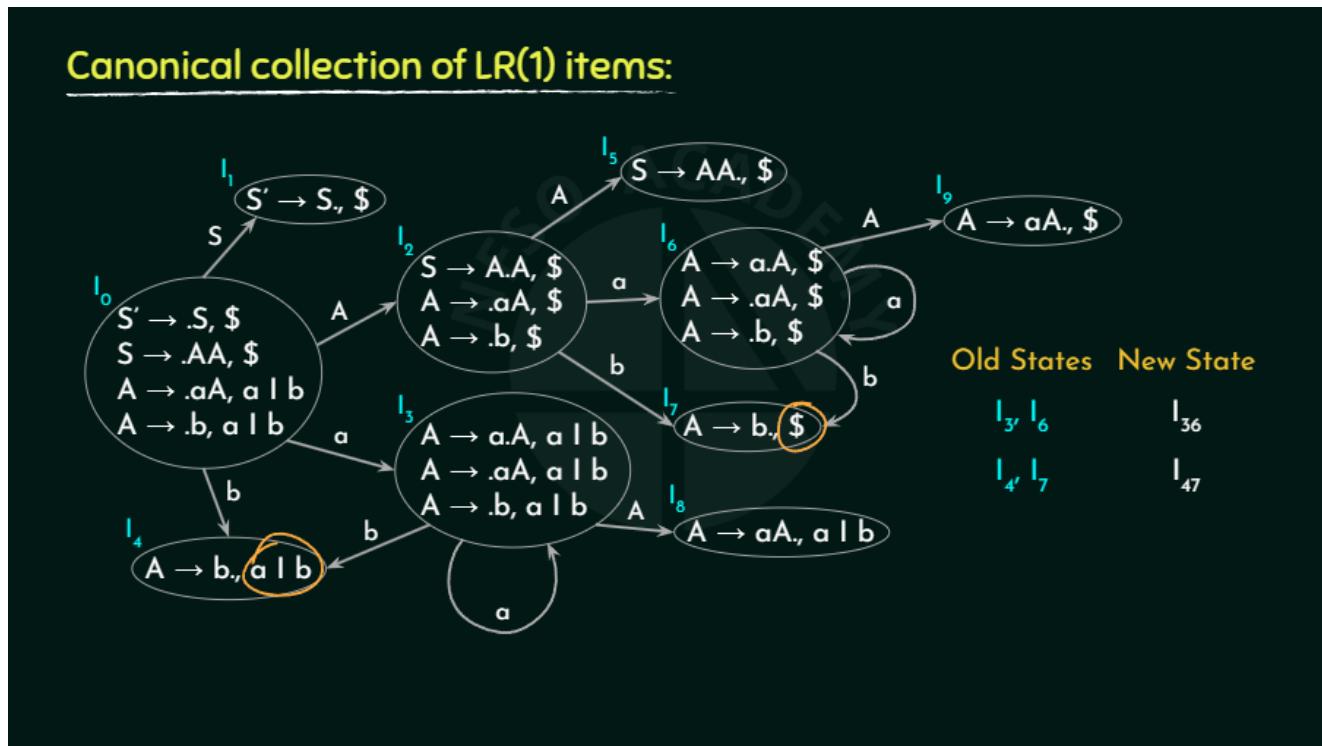
CLR(1) Parsing Table

Construction of CLR(1) Parsing table:  
 $S' \rightarrow S$   
 $i. S \rightarrow AA$   
 $ii. A \rightarrow aA$   
 $iii. A \rightarrow b$   
Accept  
I\_0 S\_3 I\_3 S\_4 I\_4 S\_5 I\_5 S\_6 I\_6 S\_7 I\_7 S\_8 I\_8 S\_9 I\_9 S\_10  
Action GO  
TO States ab\$SACLR(1) Parsing Table

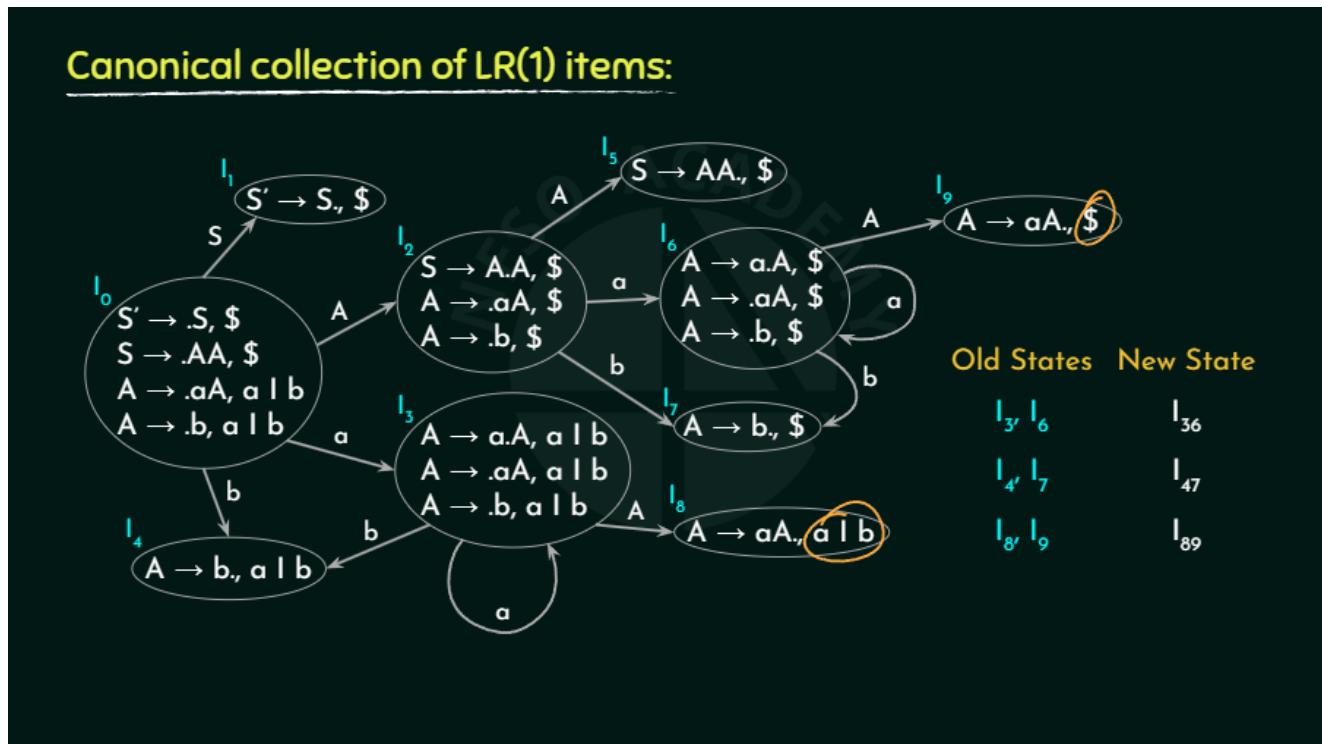
## Canonical collection of LR(1) items:



Canonical collection of LR(1) items:  
I\_0 S' → S, \$\nabla  
I\_1 S → .AA, \$\nabla  
I\_2 S → A.A, \$\nabla  
I\_3 A → .aA, a | b  
I\_4 A → aA, a | b  
I\_5 A → a.a, \$\nabla  
I\_6 A → .aA, a | b  
I\_7 A → aA, a | b  
I\_8 A → aA, \$\nabla  
I\_9 A → b, \$\nabla



Canonical collection of LR(1) items:  
 I6I2A → .aAA → .bS' → .S, \$S → .AA, \$A → .aA, a | bA → .b, a | bS → A.A, \$A → a.A, a | bA → b., a | b, \$, \$, a | b, a | bA → .aAA → .b, \$, \$aA → b., \$I7bA → aA., a | b  
 b18I3AabA → aA., \$ I9AabOld StatesNew StateI3, I6I36I4, I7I47



Canonical collection of LR(1) items:  
 $I_6 I_2 A \rightarrow .aAA \rightarrow .bS' \rightarrow .S$ ,  $\$S \rightarrow .AA$ ,  $\$A \rightarrow .aA$ ,  $a \mid bA \rightarrow .b$ ,  $a \mid bS \rightarrow A.A$ ,  $\$A \rightarrow a.A$ ,  $a \mid bA \rightarrow b.$ ,  $a \mid b, \$ \mid \$, a \mid b, a \mid bA \rightarrow .aAA \rightarrow .bI_0 S' \rightarrow S.$ ,  
 $\$I_1 I_4 S A a b S \rightarrow AA.$ ,  $\$I_5 A A \rightarrow a.A$ ,  $\$A \rightarrow .aAA \rightarrow .b, \$, \$aA \rightarrow b.$ ,  $\$I_7 b A \rightarrow a A.$ ,  $a \mid b I_8 I_3 A a b A \rightarrow a A.$ ,  $\$I_9 A a b O l d \text{ States} N e w \text{ State} I_3, I_6 I_3 I_4, I_7 I_8, I_9 I_4 I_7 I_8 I_9$

### Conversion to LALR(1) Parsing table:

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_6$	$S_7$		5	
3	$S_3$	$S_4$		8	
4	$r_{iii}$	$r_{iii}$			
5			$r_i$		
6	$S_6$	$S_7$		9	
7			$r_{iii}$		
8	$r_{ii}$	$r_{ii}$			
9			$r_{ii}$		

CLR(1) Parsing Table

Old States	New State
$I_3, I_6$	$I_{36}$
$I_4, I_7$	$I_{47}$
$I_8, I_9$	$I_{89}$

Conversion to LALR(1) Parsing  
table:Accept1S3S42riiS6S75S3S48riiiriiiriiriiS6S791092345867riiiActionGO  
TOStatesab\\$ACLR(1) Parsing TableOld StatesNew StateI3, I6I36I4, I7I8, I9I47I89

## Conversion to LALR(1) Parsing table:

States	Action			GO TO	
	a	b	\$	A	S
0	$S_{36}$	$S_{47}$		2	1
1			Accept		
2	$S_{36}$	$S_{47}$		5	
36	$S_{36}$	$S_{47}$		89	
47	r <sub>iii</sub>	r <sub>iii</sub>			
5		r <sub>i</sub>			
36	$S_{36}$	$S_{47}$		89	
47		r <sub>iii</sub>			
89	r <sub>ii</sub>	r <sub>ii</sub>			
89		r <sub>ii</sub>			

States	Action			GO TO	
	a	b	\$	A	S
0	$S_{36}$	$S_{47}$		2	1
1			Accept		
2	$S_{36}$	$S_{47}$		5	
36	$S_{36}$	$S_{47}$		89	
47	r <sub>iii</sub>	r <sub>iii</sub>		r <sub>iii</sub>	
5		r <sub>i</sub>		r <sub>i</sub>	
89	r <sub>ii</sub>	r <sub>ii</sub>		r <sub>ii</sub>	

LALR(1) Parsing Table



# Compiler Design

## Conflicts in CLR(1) and LALR(1) Parsers

Compiler Design Conflicts in CLR(1) and LALR(1) Parsers



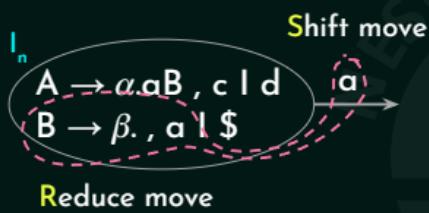
## Outcome

- ☆ S-R and R-R conflicts in CLR(1) and LALR(1) parsing tables.
- ☆ Difference between CLR(1) and LALR(1) parsers w.r.t. R-R conflict.

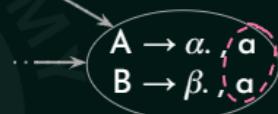
Outcome  
☆ S-R and R-R conflicts in CLR(1) and LALR(1) parsing tables.  
☆ Difference between CLR(1) and LALR(1) parsers w.r.t. R-R conflict.

### Conflicts w.r.t. LR(1) items:

#### Shift - Reduce Conflict:



#### Reduce - Reduce Conflict:

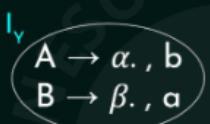
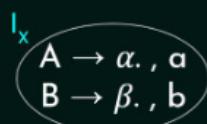


- Not CLR(1).
- Not LALR(1).

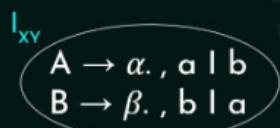
Conflicts w.r.t. LR(1) items:  
Shift - Reduce Conflict:  
 $A \rightarrow \alpha., a$ ,  $aB \rightarrow \beta., a$ ,  $A \rightarrow \alpha.aB, c \mid d$ ,  $B \rightarrow \beta., a \mid \$$   
- Not CLR(1).  
- Not LALR(1).  
Reduce move  
Reduce - Reduce Conflict:  
Shift move

## Conflicts w.r.t. LR(1) items:

❖ Reduce - Reduce Conflict:



} No conflict for CLR(1).



} Conflict for LALR(1).

No S-R Conflict in CLR(1)  $\Rightarrow$  No S-R Conflict in LALR(1)

No R-R Conflict in CLR(1)  $\neq$  No R-R Conflict in LALR(1)



# Compiler Design

## Determining the type of Grammar – Set 6

Compiler Design Determining the type of Grammar - Set 6



## Outcome

- ☆ Two solved problems on how to determine the type of a given grammar.

Outcome☆Two solved problems on how to determine the type of a given grammar.

Q1: Determine whether the following grammar is CLR(1) or LALR(1):

$$\begin{aligned}S &\rightarrow AaAb \mid BbBa \\A &\rightarrow \epsilon \\B &\rightarrow \epsilon\end{aligned}$$

Q1:Determine whether the following grammar is CLR(1) or LALR(1):  
 $S \rightarrow AaAb \mid BbBa$   
 $A \rightarrow \epsilon$   
 $B \rightarrow \epsilon$

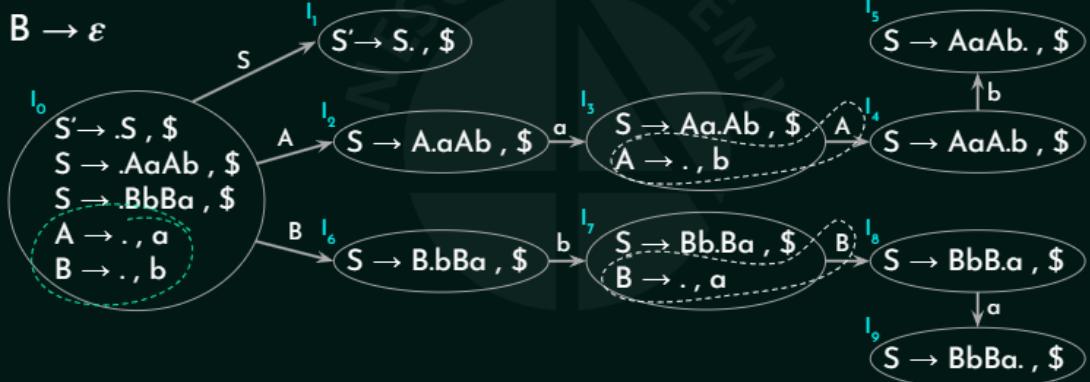
**Q1: Determine whether the following grammar is CLR(1) or LALR(1):**

**Sol.**  $S' \rightarrow S$

$S \rightarrow AaAb \mid BbBa$

$A \rightarrow \epsilon$

$B \rightarrow \epsilon$



**Q1: Determine whether the following grammar is CLR(1) or LALR(1):** Sol.  $S' \rightarrow S$   $S \rightarrow AaAb \mid BbBa$   $A \rightarrow \epsilon$   $B \rightarrow \epsilon$

$I_0: S' \rightarrow .S, \$$   
 $I_1: S \rightarrow .AaAb, \$$   
 $I_2: S \rightarrow .BbBa, \$$   
 $I_3: A \rightarrow .., a$   
 $I_4: B \rightarrow .., b$

$I_5: S \rightarrow AaAb., \$$   
 $I_6: S \rightarrow B.bBa., \$$   
 $I_7: S \rightarrow Bb.Ba., \$$   
 $I_8: S \rightarrow BbB.a., \$$   
 $I_9: S \rightarrow BbBa., \$$

**Q2: Determine whether the following grammar is CLR(1) or LALR(1):**

$S \rightarrow Aa \mid bAc \mid dc \mid bda$

$A \rightarrow d$

**Q2: Determine whether the following grammar is CLR(1) or LALR(1):**  $S \rightarrow Aa \mid bAc \mid dc \mid bda$   $A \rightarrow d$

Q2: Determine whether the following grammar is CLR(1) or LALR(1):

Sol.  $S' \rightarrow S$

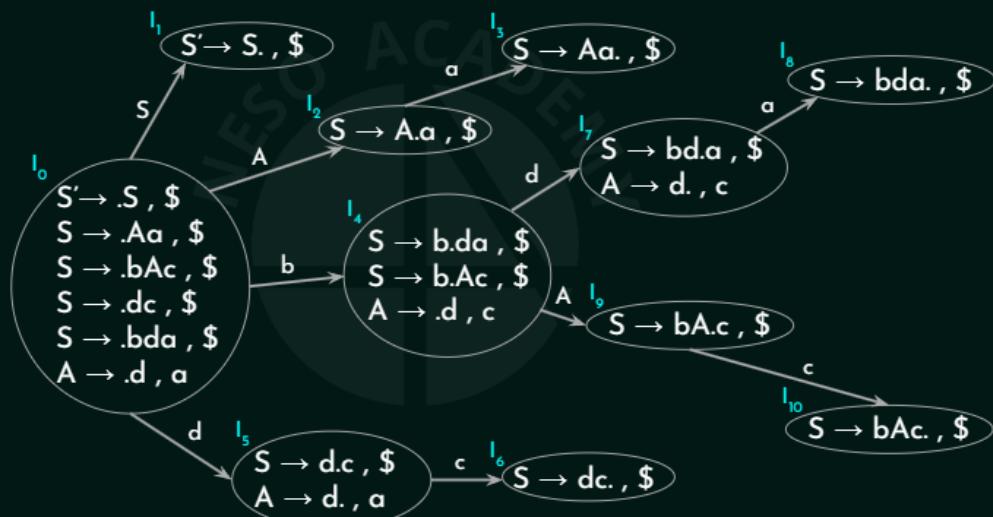
$S \rightarrow Aa$

$S \rightarrow bAc$

$S \rightarrow dc$

$S \rightarrow bda$

$A \rightarrow d$



I4  $S \rightarrow b.da , \$$  S  $\rightarrow b.Ac , \$$  Q2: Determine whether the following grammar is CLR(1) or LALR(1):  
 Sol.  $S' \rightarrow SS \rightarrow Aa S \rightarrow bAc S \rightarrow dc S \rightarrow bda A \rightarrow d$   
 $I0 S' \rightarrow .S , \$$   $S \rightarrow .Aa , \$$   $S \rightarrow .bAc , \$$   $S \rightarrow .dc , \$$   $S \rightarrow .bda , \$$   
 $A \rightarrow .d , a$   
 $I1 S \rightarrow A.a , \$$   $I2 S \rightarrow Aa. , \$$   $I3 A \rightarrow .d , c$   
 $S \rightarrow d.c , \$$   $A \rightarrow d. , a$   
 $I5 S \rightarrow dc. , \$$   $I6 S \rightarrow bd.a , \$$   $A \rightarrow d. , c$   
 $I7 S \rightarrow bda. , \$$   $I8 S \rightarrow bA.c , \$$   $I9 S \rightarrow bAc. , \$$   
 $I10 A \rightarrow dcdca$



# Compiler Design

## Determining the type of Grammar – Set 7



## Outcome

- ☆ Solved problem on how to determine the type of a given grammar.

Outcome☆Solved problem on how to determine the type of a given grammar.

**Q: Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):**

$$\begin{aligned} S &\rightarrow Aa \mid bAc \mid Bc \mid bBa \\ A &\rightarrow d \\ B &\rightarrow d \end{aligned}$$

Q:Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):  
 $S \rightarrow Aa \mid bAc \mid Bc \mid bBa$   
 $A \rightarrow dB \rightarrow d$

Q: Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):

Sol.

	FIRST	FOLLOW
$S \rightarrow Aa \mid bAc \mid Bc \mid bBa$	{d, b}	{\$}
$A \rightarrow d$	{d}	{a, c}
$B \rightarrow d$	{d}	{c, a}

	a	b	c	d	\$
S		$S \rightarrow bAc$ $S \rightarrow bBa$		$S \rightarrow Aa$ $S \rightarrow Bc$	
A				$A \rightarrow d$	
B				$B \rightarrow d$	

LL(1) Parsing Table

, a}, c}, b}Q:Determine whether the following grammar is

LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):S → Aa | bAc | Bc | bBaA → dB →

d FOLLOW{\$}FIRST{d}{d}S → bAcB → dA → dS → AaSol.LL(1) Parsing TableadSAB\$cbS

→ bBaS → Bc{d{a{c

Q: Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):

Sol.  $S' \rightarrow S$

$S \rightarrow Aa$

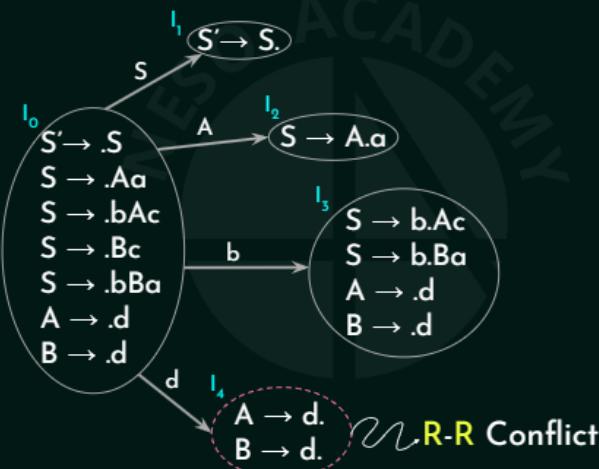
$S \rightarrow bAc$

$S \rightarrow Bc$

$S \rightarrow bBa$

$A \rightarrow d$

$B \rightarrow d$



I0S' → .SQ:Determine whether the following grammar is

LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):Sol.  
S' → SS → Aa S → bAc S → Bc S → bBaA → dB  
→ dS → .Aa S → .bAc S → .Bc S → .bBaA → .dB → .dS' → S.I1S → A.a I2I3S → b.Ac S →  
b.BaA → .dB → .dA → d.B → d.I4SAbdR-R Conflict

Q: Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):

Sol. S' → S

S → Aa

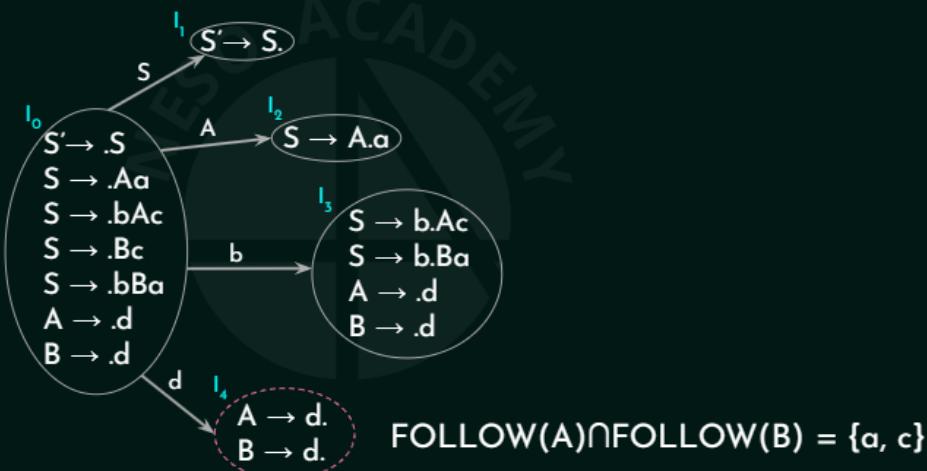
S → bAc

S → Bc

S → bBa

A → d

B → d



I0S' → .SQ:Determine whether the following grammar is

LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):Sol.  
S' → SS → Aa S → bAc S → Bc S → bBaA → dB  
→ dS → .Aa S → .bAc S → .Bc S → .bBaA → .dB → .dS' → S.I1S → A.a I2I3S → b.Ac S →  
b.BaA → .dB → .dA → d.B → d.I4SAbd FOLLOW(A) ∩ FOLLOW(B) = {a, c}

Q: Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):

Sol.  $S' \rightarrow S$

$S \rightarrow Aa$

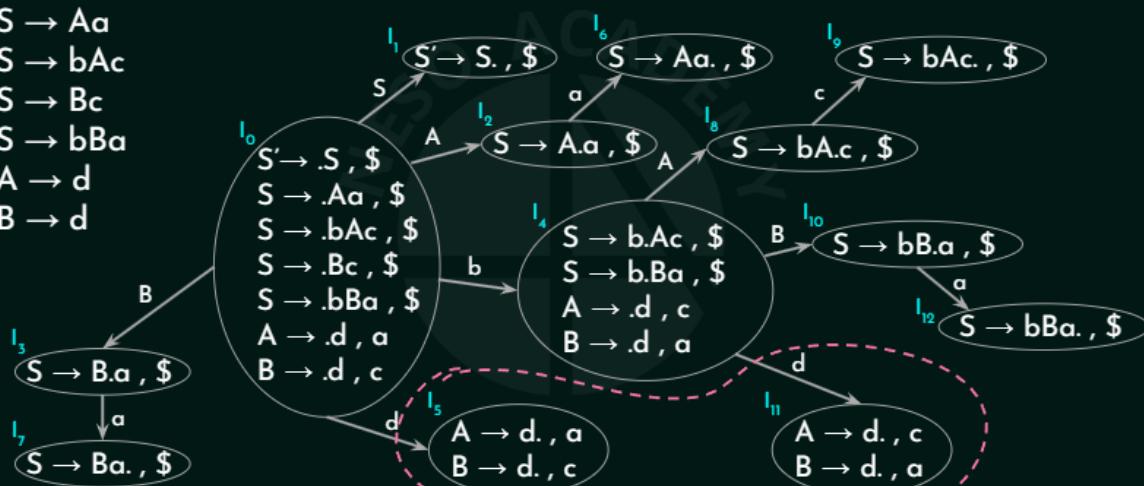
$S \rightarrow bAc$

$S \rightarrow Bc$

$S \rightarrow bBa$

$A \rightarrow d$

$B \rightarrow d$



$S \rightarrow bBa. , \$$  Q: Determine whether the following grammar is

LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1): Sol. I<sub>0</sub>  $S' \rightarrow .S , \$$  I<sub>1</sub>  $S' \rightarrow SS \rightarrow Aa$  I<sub>2</sub>  $S \rightarrow bAc$  I<sub>3</sub>  $S \rightarrow Bc$  I<sub>4</sub>  $S \rightarrow bBaA \rightarrow dB \rightarrow dS \rightarrow .Aa , \$$  I<sub>5</sub>  $S \rightarrow .bAc , \$$  I<sub>6</sub>  $S \rightarrow .Bc , \$$  I<sub>7</sub>  $S \rightarrow .bBa , \$$  I<sub>8</sub>  $A \rightarrow .d , a$  I<sub>9</sub>  $B \rightarrow .d , c$  I<sub>10</sub>  $A \rightarrow d. , a$  I<sub>11</sub>  $B \rightarrow d. , c$  I<sub>12</sub>  $A \rightarrow d. , c$  I<sub>13</sub>  $B \rightarrow d. , a$ , I<sub>14</sub>  $S \rightarrow b.Ba , \$$  I<sub>15</sub>  $A \rightarrow .dB \rightarrow .dA \rightarrow d. , a$  I<sub>16</sub>  $B \rightarrow .dB \rightarrow .dA \rightarrow d. , c$  I<sub>17</sub>  $S \rightarrow Ba. , \$$  I<sub>18</sub>  $A \rightarrow d. , a$  I<sub>19</sub>  $B \rightarrow d. , c$  I<sub>20</sub>  $A \rightarrow d. , c$  I<sub>21</sub>  $B \rightarrow d. , a$ , I<sub>22</sub>  $cAS \rightarrow B.a , \$$  I<sub>23</sub>  $I3S \rightarrow Ba. , \$$  I<sub>24</sub>  $I7Ba$

Q: Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):

Sol.  $S' \rightarrow S$

$S \rightarrow Aa$

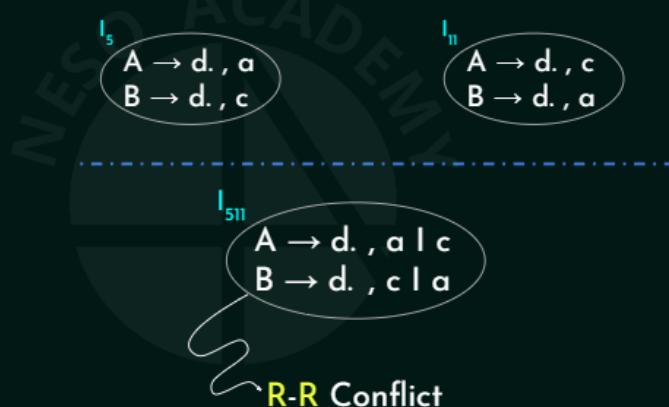
$S \rightarrow bAc$

$S \rightarrow Bc$

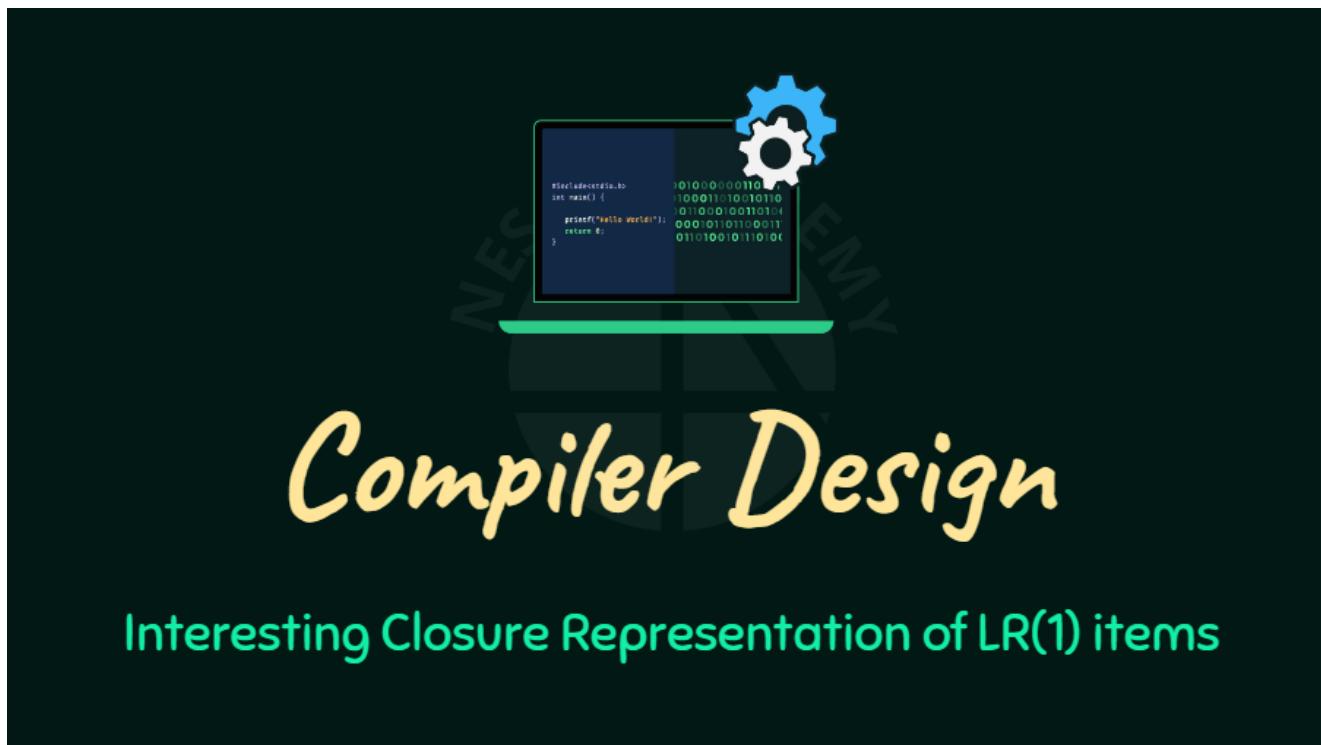
$S \rightarrow bBa$

$A \rightarrow d$

$B \rightarrow d$

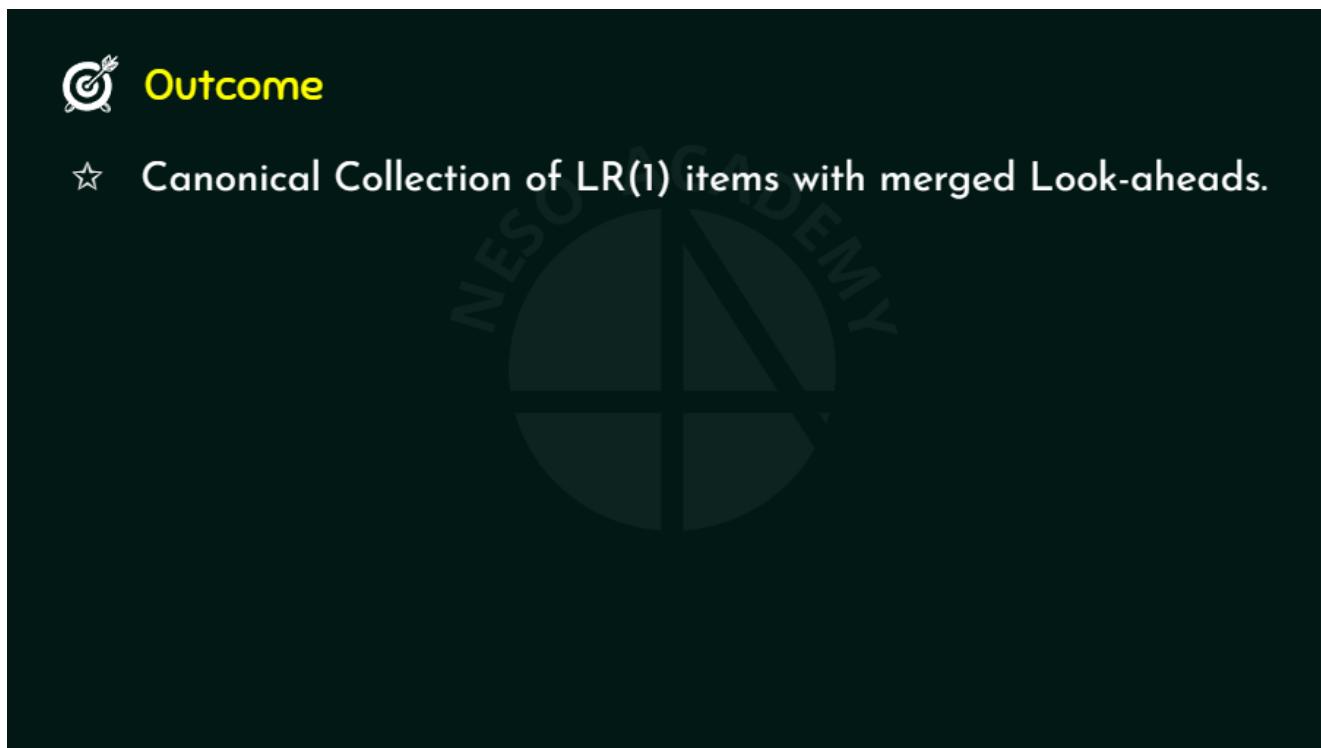


Q:Determine whether the following grammar is LL(1)/LR(0)/SLR(1)/CLR(1)/LALR(1):  
Sol.  
 $S \rightarrow SS$   
 $S \rightarrow Aa$   
 $S \rightarrow bAc$   
 $S \rightarrow Bc$   
 $S \rightarrow bBaA \rightarrow dB \rightarrow dA \rightarrow d.$ ,  
 $aB \rightarrow d.$ ,  
 $cI5A \rightarrow d.$ ,  
 $cB \rightarrow d.$ ,  
 $al11A \rightarrow d.B \rightarrow d.$   
I511, a | c, c | aR-R Conflict



A slide titled "Compiler Design" featuring a blue gear icon and a code editor window showing C-like syntax. Below the title is the subtitle "Interesting Closure Representation of LR(1) items".

Compiler Design Interesting Closure Representation of LR(1) items



A slide titled "Outcome" with the subtitle "Canonical Collection of LR(1) items with merged Look-aheads.".

Outcome☆Canonical Collection of LR(1) items with merged Look-aheads.

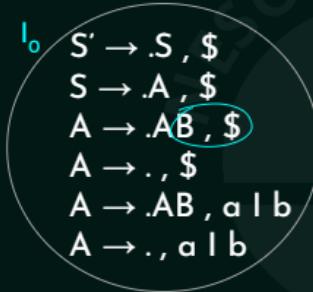
## Closure Representation of LR(1) items:

$S \rightarrow A$   
 $A \rightarrow AB \mid \epsilon$   
 $B \rightarrow aB \mid b$

Closure Representation of LR(1) items:  
 $S \rightarrow AA \rightarrow AB \mid \epsilon B \rightarrow aB \mid b$

## Closure Representation of LR(1) items:

$S' \rightarrow S$   
 $S \rightarrow A$   
 $A \rightarrow AB \mid \epsilon$   
 $B \rightarrow aB \mid b$



$S' \rightarrow SS \rightarrow AA \rightarrow AB \mid \epsilon B \rightarrow aB \mid b \mid 0$   
 $S' \rightarrow .S, \$$   
 $S \rightarrow .AA \rightarrow .ABA \rightarrow .A \rightarrow .ABA \rightarrow ., \$, \$, \$,$   
 $a \mid b, a \mid b$

## Closure Representation of LR(1) items:

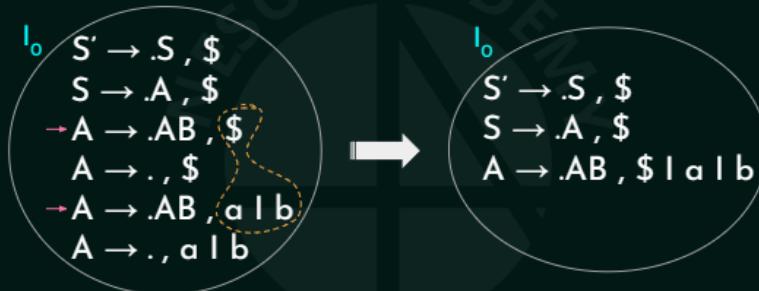
### Closure Representation of LR(1) items:

$S' \rightarrow S$

$S \rightarrow A$

$A \rightarrow AB \mid \epsilon$

$B \rightarrow aB \mid b$



$I_0 S' \rightarrow .S , \$$   
 $S \rightarrow .A , \$$   
 $S' \rightarrow SS \rightarrow AA \rightarrow AB \mid \epsilon B \rightarrow aB \mid b$   
 $I_0 S' \rightarrow .S , \$$   
 $S \rightarrow .A , \$$   
 $A \rightarrow .ABA \rightarrow .. , \$, \$, \$, a | b, a | b$   
 $A \rightarrow .AB , \$ | a | b$   
 $A \rightarrow .. , \$ | a | b$

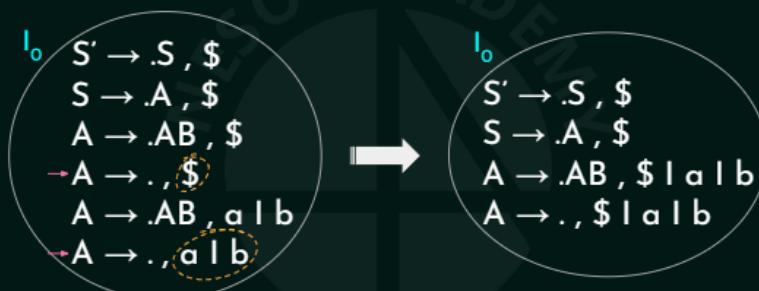
### Closure Representation of LR(1) items:

$S' \rightarrow S$

$S \rightarrow A$

$A \rightarrow AB \mid \epsilon$

$B \rightarrow aB \mid b$



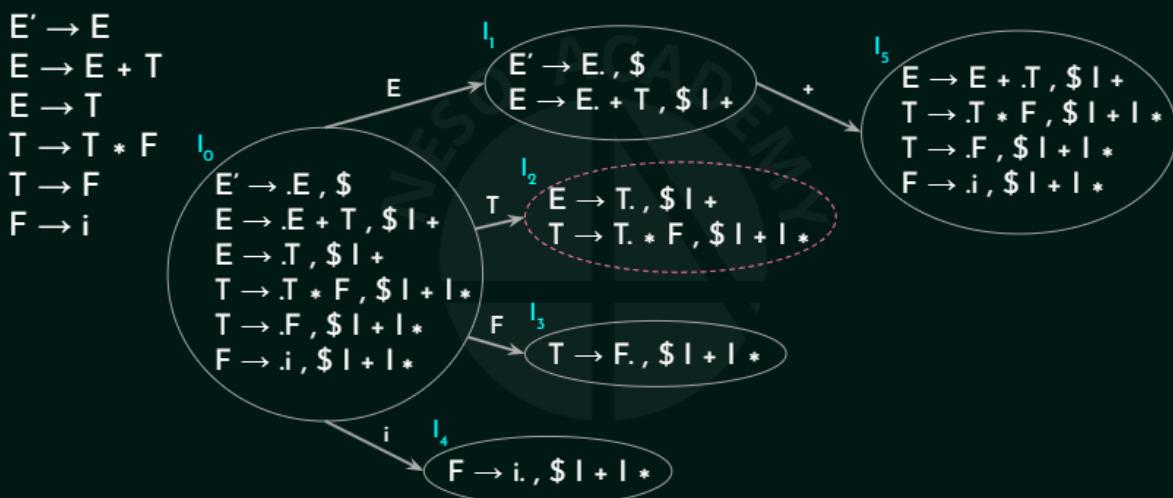
$I_0 S' \rightarrow SS \rightarrow AA \rightarrow AB \mid \epsilon B \rightarrow aB \mid b$   
 $I_0 S' \rightarrow .S , \$$   
 $S \rightarrow .A , \$$   
 $S' \rightarrow .S , \$$   
 $S \rightarrow .A , \$$   
 $A \rightarrow .ABA \rightarrow .. , \$, \$, \$, a | b, a | b$   
 $A \rightarrow .AB , \$ | a | b$   
 $A \rightarrow .. , \$ | a | b$

## Canonical Collection of LR(1) items with merged Look-aheads:

$E' \rightarrow E$	$E' \rightarrow .E, \$$
$E \rightarrow E + T$	$E \rightarrow .E + T, \$I +$
$E \rightarrow T$	$E \rightarrow .T, \$I +$
$T \rightarrow T * F$	$T \rightarrow .T * F, \$I + I *$
$T \rightarrow F$	$T \rightarrow .F, \$I + I *$
$F \rightarrow i$	$T \rightarrow .T * F, *$
	$T \rightarrow .F, *$

$E' \rightarrow EE \rightarrow E + TE \rightarrow TT \rightarrow T + FT \rightarrow FF \rightarrow i$   
 Canonical Collection of LR(1) items with merged Look-aheads:  
 $E' \rightarrow .E, \$$   
 $E \rightarrow .E + TE \rightarrow .T, \$, \$^*I + I + T \rightarrow .T + FT \rightarrow .F^*T \rightarrow .T + FT \rightarrow .F^*, +^*, +^*, \$I +, \$I + I + ^*I + ^*$

## Canonical Collection of LR(1) items with merged Look-aheads:

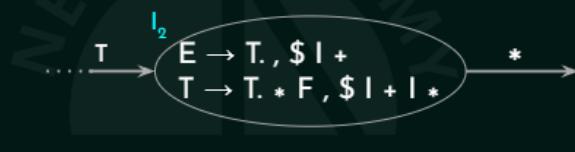


$i \rightarrow .T + F, \$I + T \rightarrow .F, \$I + E \rightarrow EE \rightarrow E + TE \rightarrow TT \rightarrow T + FT \rightarrow FF \rightarrow i$   
 Canonical Collection of LR(1) items with merged Look-aheads:  
 $*I0E' \rightarrow .E, \$$   
 $E \rightarrow .E + T, \$I + E \rightarrow .T, \$I + T \rightarrow .T + F, \$I + I + T \rightarrow .F, \$I + I + F \rightarrow .i, \$I + I + ^*E' \rightarrow E, \$E \rightarrow E. + T, \$I$

$+I_1EE \rightarrow T. , \$ | +T \rightarrow T. + F , \$ | + I + *I_2T | 4F \rightarrow i. , \$ | + I + *I_3T \rightarrow F. , \$ | + I + *Fil + *I + *, \$ | + I + *F \rightarrow .i E \rightarrow E + .T , \$ | + +***$

### Canonical Collection of LR(1) items with merged Look-aheads:

$E' \rightarrow E$   
 $E \rightarrow E + T$   
 $E \rightarrow T$   
 $T \rightarrow T * F$   
 $T \rightarrow F$   
 $F \rightarrow i$



$E' \rightarrow EE \rightarrow E + TE \rightarrow TT \rightarrow T + FT \rightarrow FF \rightarrow i$   
 Canonical Collection of LR(1) items with merged Look-aheads:  
 $*E \rightarrow T. , \$ | +T \rightarrow T. + F , \$ | + I + *I_2T^*$

# Compiler Design

## CFGs – The Big Picture

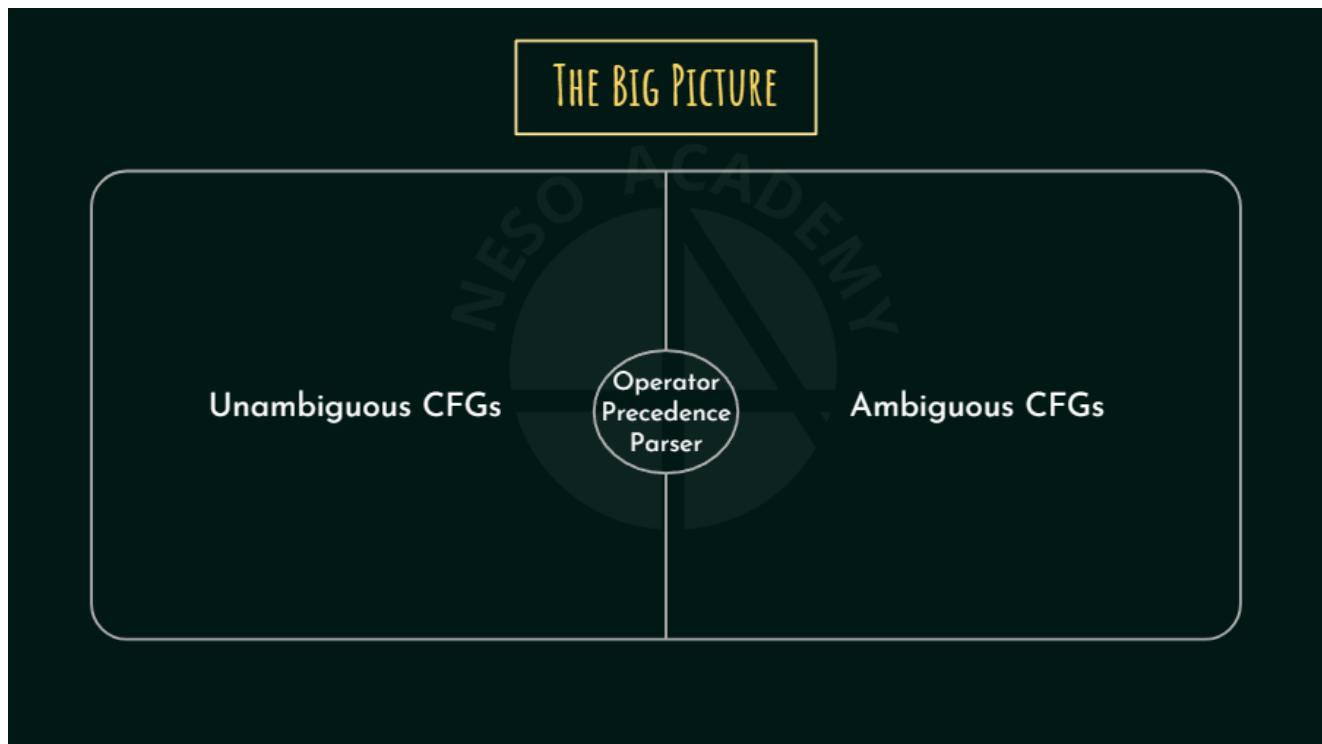
Compiler Design CFGs - The Big Picture



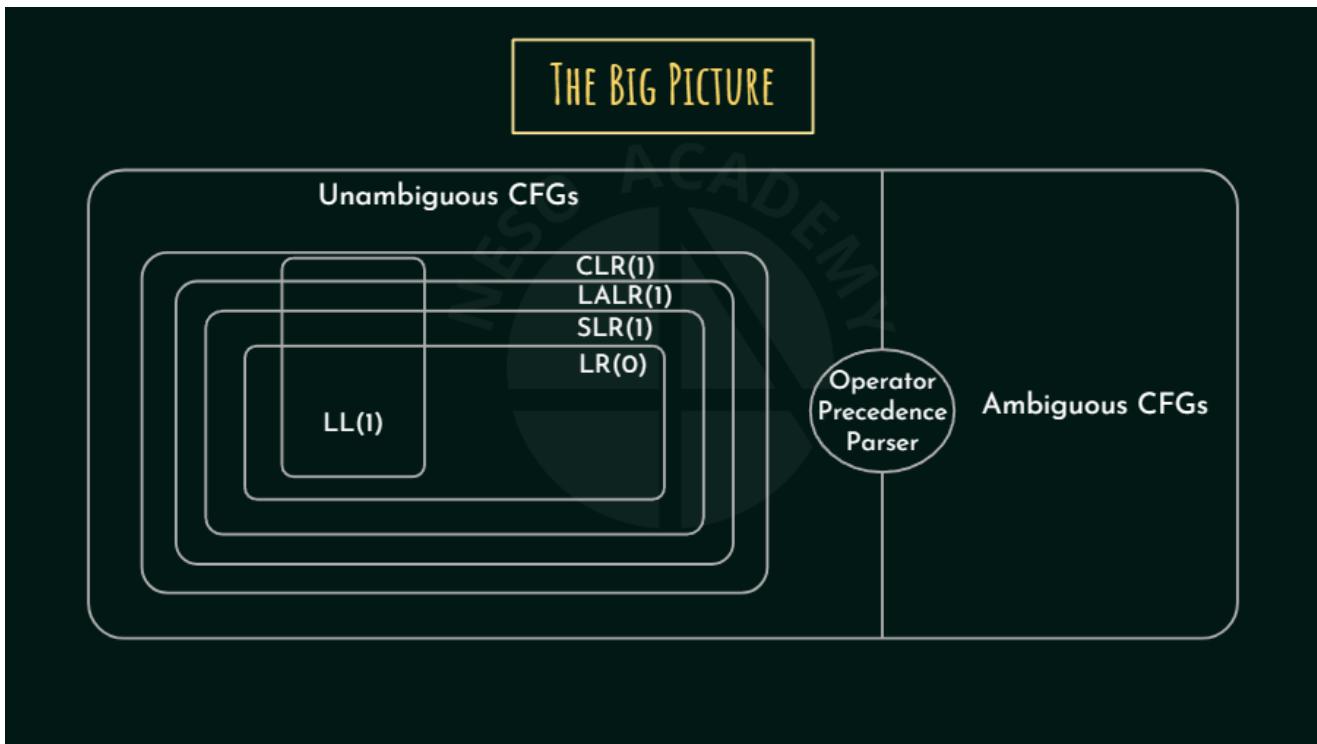
## Outcome

- ★ Relationship of all Context Free Grammars.

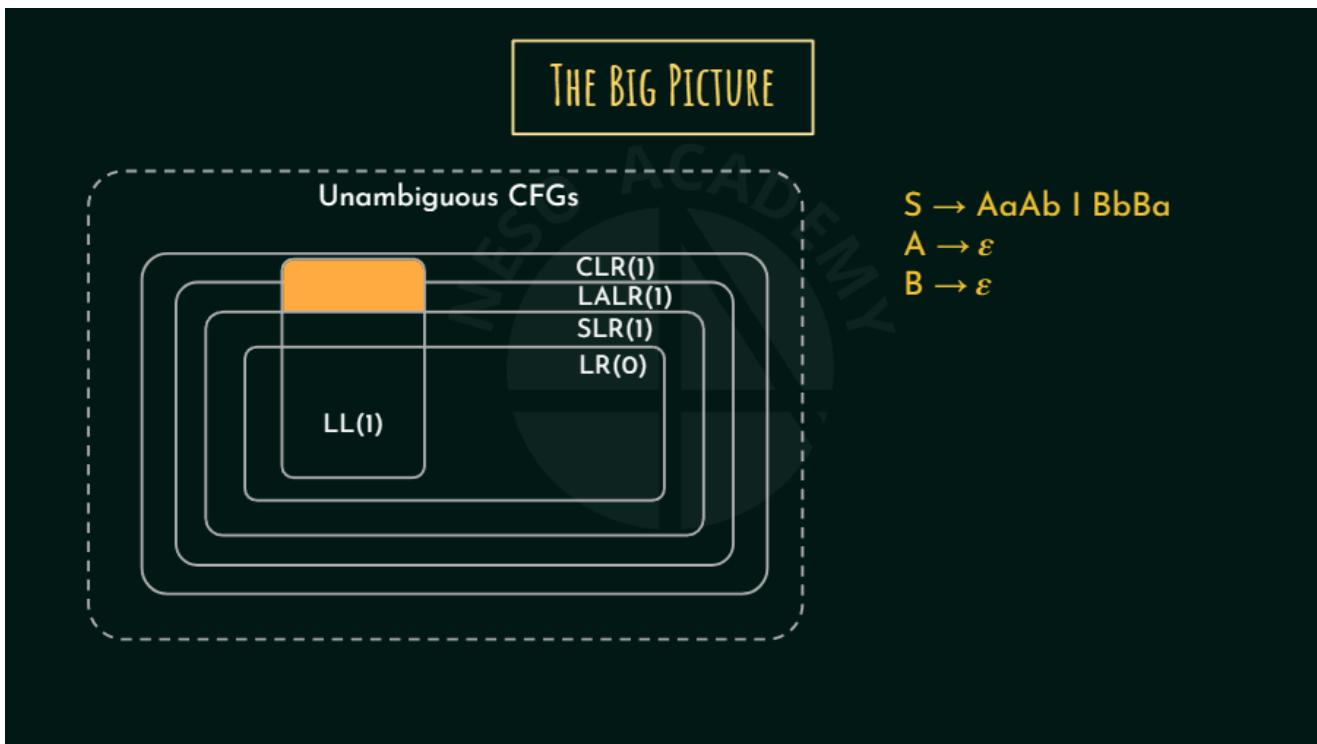
Outcome★Relationship of all Context Free Grammars.



The Big Picture  
Unambiguous CFGs  
Ambiguous CFGs  
Operator Precedence Parser



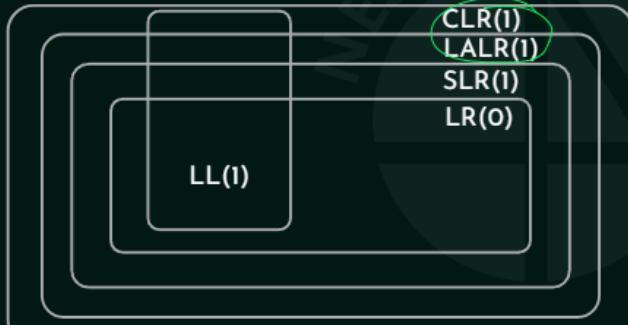
SLR(1) LALR(1) CLR(1) The Big Picture Unambiguous CFGs Ambiguous CFGs Operator Precedence Parser LR(0) LL(1)



SLR(1) LALR(1) CLR(1) The Big Picture Unambiguous CFGs LR(0) LL(1)  $S \rightarrow AaAb \mid BbBa$   $A \rightarrow \epsilon$   $B \rightarrow \epsilon$

## THE BIG PICTURE

Unambiguous CFGs

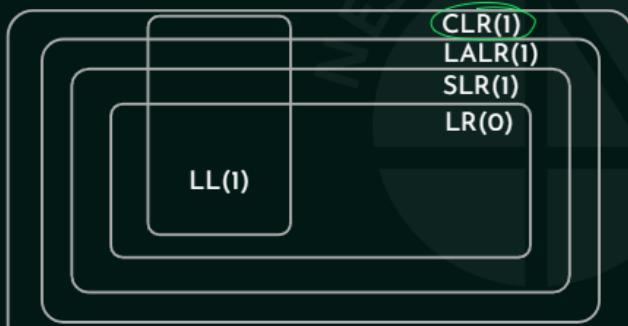


$S \rightarrow Aa \mid bAc \mid dc \mid bda$   
 $A \rightarrow d$

SLR(1)LALR(1)CLR(1)The Big PictureUnambiguous CFGsLR(0)LL(1)  
 $S \rightarrow Aa \mid bAc \mid dc \mid bda$   
 $A \rightarrow d$

## THE BIG PICTURE

Unambiguous CFGs

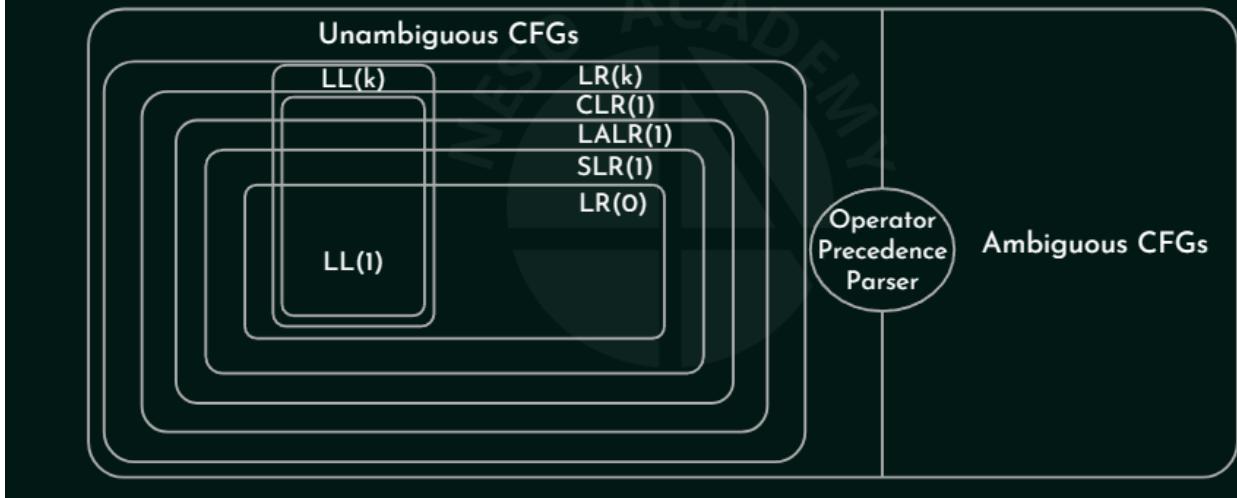


$S \rightarrow Aa \mid bAc \mid dc \mid bda$   
 $A \rightarrow d$

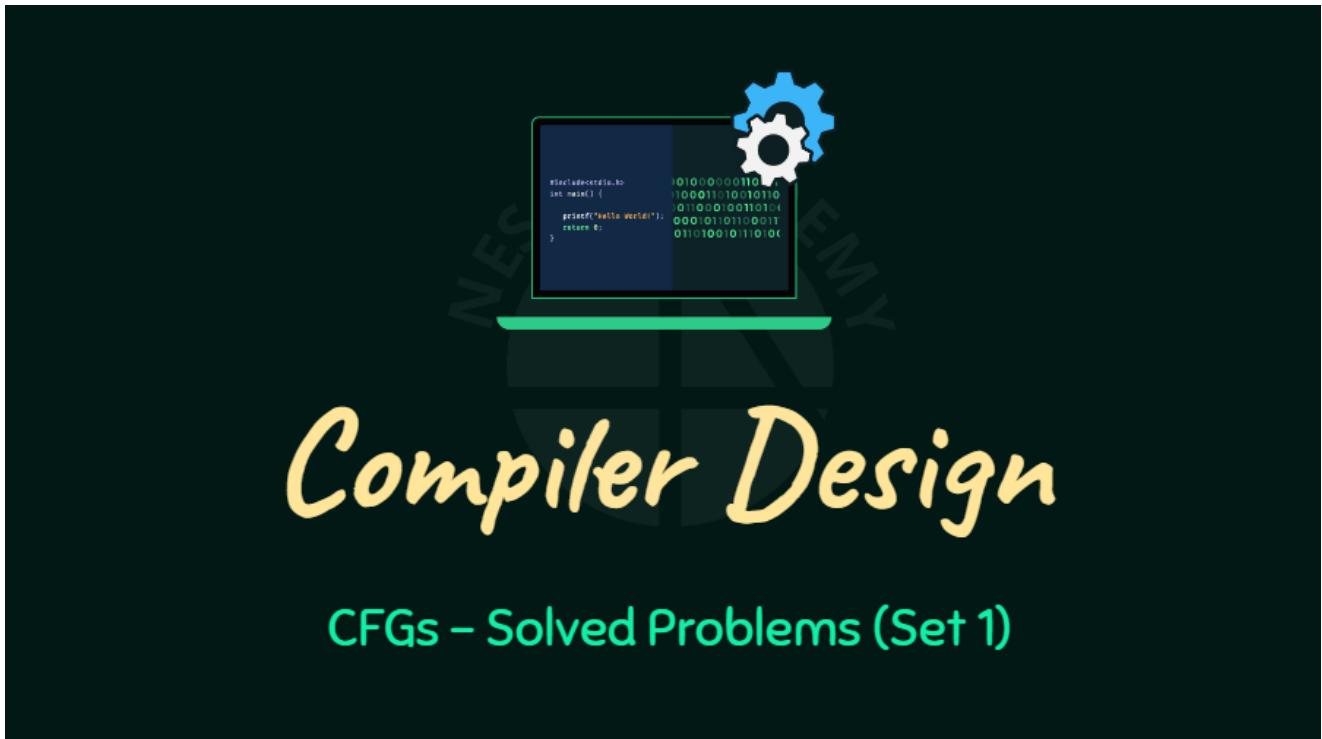
$S \rightarrow Aa \mid bAc \mid Bc \mid bBa$   
 $A \rightarrow d$   
 $B \rightarrow d$

SLR(1)LALR(1)CLR(1)The Big PictureUnambiguous CFGsLR(0)LL(1)  
 $S \rightarrow Aa \mid bAc \mid Bc \mid bBa$   
 $A \rightarrow d$   
 $S \rightarrow Aa \mid bAc \mid Bc \mid bBa$   
 $A \rightarrow d$   
 $B \rightarrow d$

## THE BIG PICTURE



CLR(1)  
LR(k)  
The Big Picture  
Unambiguous CFGs  
Ambiguous CFGs  
Operator Precedence Parser  
LR(0)  
SLR(1)  
LALR(1)  
LL(1)  
LL(k)



Compiler Design CFGs - Solved Problems (Set 1)



## Outcome

- ☆ Four solved problems on CFGs.

Outcome☆Four solved problems on CFGs.

**Q1:** Which of the following statements is true?

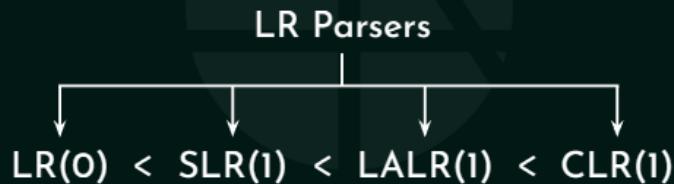
- (A) SLR parser is more powerful than LALR.
- (B) LALR parser is more powerful than Canonical LR parser.
- (C) Canonical LR parser is more powerful than LALR parser.
- (D) The parsers SLR, Canonical LR and LALR have the same power.

GATE 1998

**Q1: Which of the following statements is true?**

- (A) SLR parser is more powerful than LALR.
- (B) LALR parser is more powerful than Canonical LR parser.
- (C) Canonical LR parser is more powerful than LALR parser.**
- (D) The parsers SLR, Canonical LR and LALR have the same power.

GATE 1998



Q1: Which of the following statements is true?  
(A) SLR parser is more powerful than LALR.  
(B) LALR parser is more powerful than Canonical LR parser.  
(C) Canonical LR parser is more powerful than LALR parser.  
(D) The parsers SLR, Canonical LR and LALR have the same power.  
GATE 1998 LR Parsers  
LR(0) CLR(1) LALR(1) SLR(1) <<<

**Q2: Which of the following statements is false?**

- (A) An unambiguous grammar has the same leftmost and rightmost derivation.
- (B) An LL(1) parser is a top-down parser.
- (C) LALR is more powerful than SLR.
- (D) An ambiguous grammar can never be LR( $k$ ) for any  $k$ .

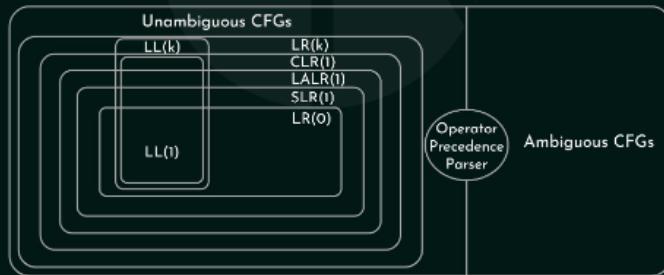
GATE 2001

**Q2: Which of the following statements is false?**

- (A) An unambiguous grammar has the same leftmost and rightmost derivation.
- (B) An LL(1) parser is a top-down parser.
- (C) LALR is more powerful than SLR.
- (D) An ambiguous grammar can never be LR( $k$ ) for any  $k$ .

THE BIG PICTURE

GATE 2001



Q2: Which of the following statements is false?  
(A) An unambiguous grammar has the same leftmost and rightmost derivation.  
(B) An LL(1) parser is a top-down parser.  
(C) LALR is more powerful than SLR.  
(D) An ambiguous grammar can never be LR( $k$ ) for any  $k$ .  
GATE 2001

**Q3: Considering the SLR(1) & LALR(1) parsing tables of a CFG, find out the false statement.**

- (A) GOTO of both tables may be different.
- (B) Shift entries are identical in both tables.
- (C) Reduce entries in tables may be different.
- (D) Error entries in tables may be different.

GATE  
2003

**Q3:** Considering the SLR(1) & LALR(1) parsing tables of a CFG, find out the false statement.

- (A) GOTO of both tables may be different.
- (B) Shift entries are identical in both tables.
- (C) Reduce entries in tables may be different.
- (D) Error entries in tables may be different.

GATE  
2003

Comparison:

$S' \rightarrow S$   
 i.  $S \rightarrow AA$   
 ii.  $A \rightarrow aA$   
 iii.  $A \rightarrow b$

States	Action			GO TO	
	a	b	\$	A	S
0	$S_3$	$S_4$		2	1
1			Accept		
2	$S_3$	$S_4$		5	
3	$S_3$	$S_4$		6	
4	$r_{ii}$	$r_{ii}$	$r_{iii}$		
5			$r_i$		
6	$r_u$	$r_u$	$r_u$		

SLR(1) Parsing Table

States	Action			GO TO	
	a	b	\$	A	S
0	$S_{36}$	$S_{47}$		2	1
1			Accept		
2	$S_{36}$	$S_{47}$		5	
36	$S_{36}$	$S_{47}$		89	
47	$r_{ii}$	$r_{ii}$	$r_{iii}$		
5			$r_i$		
89	$r_u$	$r_u$	$r_u$		

LALR(1) Parsing Table

Q3: Considering the SLR(1) & LALR(1) parsing tables of a CFG, find out the false statement.

(A) GOTO of both tables may be different. (B) Shift entries are identical in both tables.

(C) Reduce entries in tables may be different. (D) Error entries in tables may be

different. GATE 2003

**Q4:** Consider the grammar

GATE 2005

$S \rightarrow (S) \mid a$

Let the number of states in SLR(1), LR(1) and LALR(1) parsers for the grammar be  $n_1$ ,  $n_2$  and  $n_3$ , respectively. The following relationship holds good:

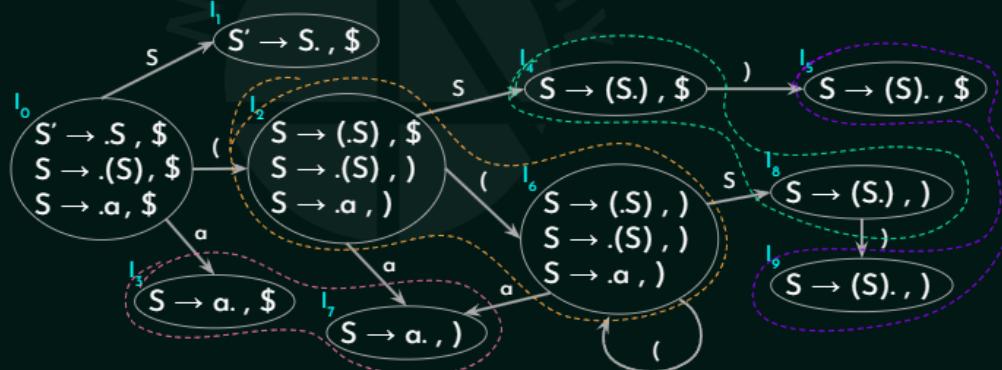
- (A)  $n_1 < n_2 < n_3$
- (B)  $n_1 = n_3 < n_2$
- (C)  $n_1 = n_2 = n_3$
- (D)  $n_1 \geq n_2 \geq n_3$

**Q4:** Consider the grammar

$$S \rightarrow (S) \mid a$$

Let the number of states in SLR(1), LR(1) and LALR(1) parsers for the grammar be  $n_1$ ,  $n_2$  and  $n_3$ , respectively. The following relationship holds good:

Sol.  $S' \rightarrow S$   
 $S \rightarrow (S)$   
 $S \rightarrow a$



I2S → (.S) , \$S' → .S , \$I0, \$, \$Q4:Consider the grammar  
 $S \rightarrow (S) \mid a$ Let the number of states in SLR(1), LR(1) and LALR(1) parsers for the grammar be  $n_1$ ,  $n_2$  and  $n_3$ , respectively. The following relationship holds good:  
 $S' \rightarrow SS \rightarrow (S) S \rightarrow aS \rightarrow .(S) S \rightarrow .aS' \rightarrow S. , \$I1S , ),$   
 $(S \rightarrow .(S) S \rightarrow .a(S \rightarrow a. , \$I3aS \rightarrow (S.) , \$I4S \rightarrow (S). , \$I5S)I6S \rightarrow (.S) , ), ), S \rightarrow .(S) S \rightarrow .a(aS \rightarrow a. , )I7S \rightarrow (S.) , )I8S \rightarrow (S. , )I9S)a(Sol.$

**Q4:** Consider the grammar

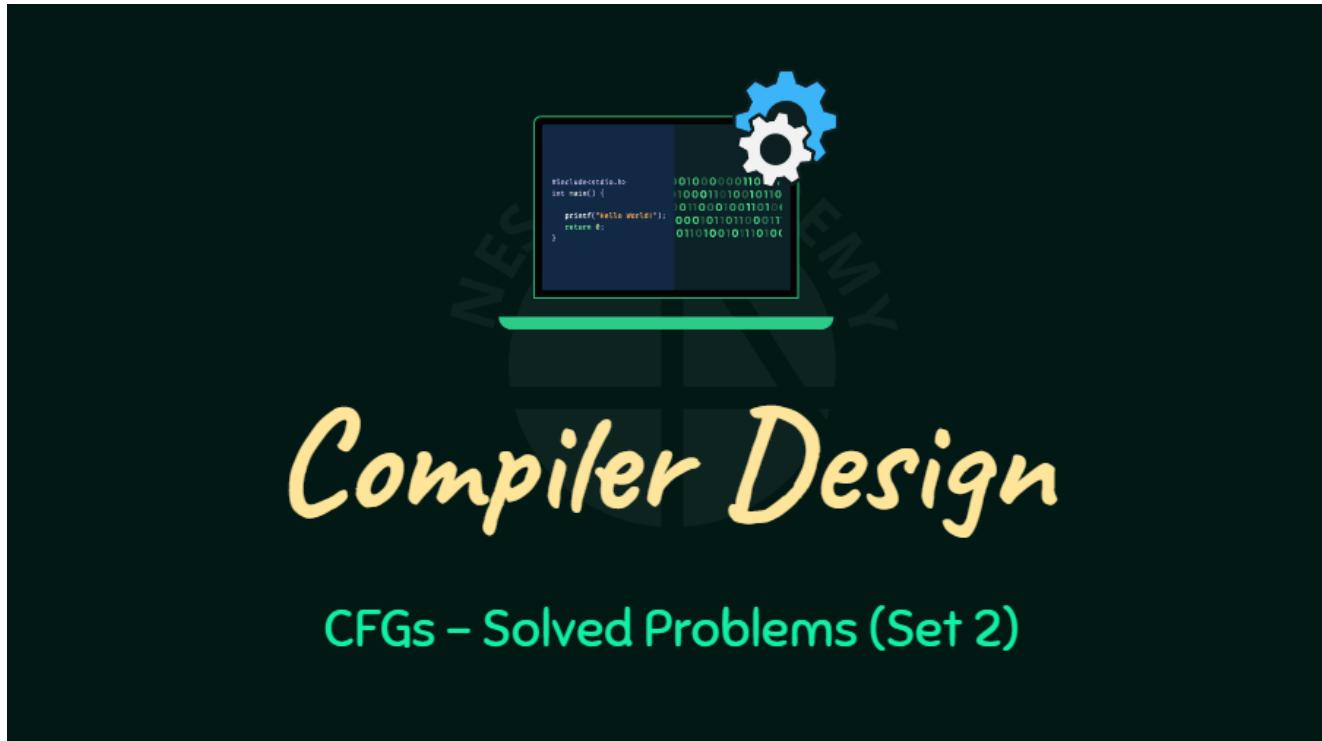
GATE 2005

$$S \rightarrow (S) \mid a$$

Let the number of states in SLR(1), LR(1) and LALR(1) parsers for the grammar be  $n_1$ ,  $n_2$  and  $n_3$ , respectively. The following relationship holds good:

- (A)  $n_1 < n_2 < n_3$
- (B)  $n_1 = n_3 < n_2$**
- (C)  $n_1 = n_2 = n_3$
- (D)  $n_1 \geq n_2 \geq n_3$

Q4: Consider the grammar  $S \rightarrow (S) \mid a$ . Let the number of states in SLR(1), LR(1) and LALR(1) parsers for the grammar be  $n_1$ ,  $n_2$  and  $n_3$ , respectively. The following relationship holds good:  
(A)  $n_1 < n_2 < n_3$   
(B)  $n_1 = n_3 < n_2$   
(C)  $n_1 = n_2 = n_3$   
(D)  $n_1 \geq n_2 \geq n_3$   
GATE 2005



A dark blue slide with a faint watermark of the 'NESC ACADEMY' logo. In the center, there is a white rectangular box containing a C program code snippet and its binary representation. Below the box, the title 'Compiler Design' is written in a large, yellow, cursive font. Underneath the title, the subtitle 'CFGs - Solved Problems (Set 2)' is displayed in a smaller, cyan font.

# Compiler Design

## CFGs - Solved Problems (Set 2)

Compiler Design CFGs - Solved Problems (Set 2)



The slide features a large, semi-transparent watermark of the 'NESC ACADEMY' logo in the background. In the upper left corner, there is a target icon with a bullseye and the word 'Outcome' next to it. Below this, a bullet point contains the text '★ Four solved problems on CFGs.'

★ Four solved problems on CFGs.

Outcome ★ Four solved problems on CFGs.

Q1: Consider the grammar shown below

GATE 2003

$$S \rightarrow CC$$

$$C \rightarrow cC \mid d$$

The grammar is:

- (A) LL(1)
- (B) SLR(1) but not LL(1)
- (C) LALR(1) but not SLR(1)
- (D) LR(1) but not LALR(1)

Q1: Consider the grammar shown below

GATE 2003

$$S \rightarrow CC$$

$$C \rightarrow cC \mid d$$

The grammar is:

- (A) LL(1)
- (B) SLR(1) but not LL(1)
- (C) LALR(1) but not SLR(1)
- (D) LR(1) but not LALR(1)

Q1: Consider the grammar shown below  
 $S \rightarrow CC$   $C \rightarrow cC \mid d$

The grammar is:(A)LL(1)  
(B)SLR(1) but not LL(1)

(C)LALR(1) but not SLR(1)  
(D)LR(1) but not LALR(1)

GATE 2003

**Q1:** Consider the grammar shown below

$$S \rightarrow CC$$

$$C \rightarrow cC \mid d$$

The grammar is:

**Sol.**

	FIRST	FOLLOW
$S \rightarrow CC$	{c, d}	{\$}
$C \rightarrow cC \mid d$	{c, d}	{c, d, \$}

	c	d	\$
S	$S \rightarrow CC$	$S \rightarrow CC$	
C	$C \rightarrow cC$	$C \rightarrow d$	

LL(1) Parsing Table

LL(1) Parsing Table  
SC\$  
**Q1:** Consider the grammar shown below  
 $S \rightarrow CC$   $C \rightarrow cC \mid d$   
The grammar is: {c, d} {\$}  
 $S \rightarrow CCC$   
 $CCC \rightarrow cC \mid d$   
FOLLOWFIRST{c, d}{c, d, \$}  
 $S \rightarrow CCC$   
 $CCC \rightarrow d$   
**Sol.**

**Q2:** Consider the following grammar

GATE 2006

$$S \rightarrow S * E \mid E$$

$$E \rightarrow F + E \mid F$$

$$F \rightarrow id$$

Consider the following LR(0) items corresponding to the grammar:

(i)  $S \rightarrow S * .E$

(ii)  $E \rightarrow F .+ E$

(iii)  $E \rightarrow F + .E$

Which two items will appear in the same state of the Canonical sets of items for the grammar?

(A) (i) and (ii)    (C) (i) and (iii)

(B) (ii) and (iii)    (D) None of the above

**Q2:** Consider the following grammar

GATE 2006

$$S \rightarrow S * E \mid E$$

$$E \rightarrow F + E \mid F$$

$$F \rightarrow id$$

Consider the following LR(0) items corresponding to the grammar:

- (i)  $S \rightarrow S * .E$
- (ii)  $E \rightarrow F .+ E$
- (iii)  $E \rightarrow F + .E$

Which two items will appear in the same state of the Canonical sets of items for the grammar?

- (A) (i) and (ii)
- (C) (i) and (iii)
- (B) (ii) and (iii)
- (D) None of the above**

**Q2:** Consider the following grammar  $S \rightarrow S + E \mid EE \rightarrow F + E \mid FF \rightarrow id$  Consider the following LR(0) items corresponding to the grammar:

- (i)  $S \rightarrow S + .E$
- (ii)  $E \rightarrow F .+ E$
- (iii)  $E \rightarrow F + .E$

Which two items will appear in the same state of the Canonical sets of items for the grammar?

- (A) (i) and (ii)
- (C) (i) and (iii)
- (B) (ii) and (iii)
- (D) None of the above

GATE 2006 \*\*

**Q3:** Consider the grammar

$$E \rightarrow E + n \mid E * n \mid n$$

For a sentence  $n + n * n$ , the handles in the right-sentential form of the reduction are:

- (A)  $n, E + n, E + n * n$
- (B)  $n, E + n, E + E * n$
- (C)  $n, n + n, n + n * n$
- (D)  $n, E + n, E * n$

**Q3:** Consider the grammar

$$E \rightarrow E + n \mid E * n \mid n$$

For a sentence  $n + n * n$ , the handles in the right-sentential form of the reduction are:

- (A)  $n, E + n, E + n * n$
- (B)  $n, E + n, E + E * n$
- (C)  $n, n + n, n + n * n$
- (D)  $n, E + n, E * n$**

GATE 2005



- (A)  $n, E + n, E + n + n$  (B)  $n, E + n, E + E + n$  (C)  $n, n + n, n + n + n$  (D)  $n, E + n, E + n$

**Q3:** Consider the grammar  $E \rightarrow E + n \mid E * n \mid n$ . For a sentence  $n + n + n$ , the handles in the right-sentential form of the reduction are: GATE 2005 \*\*\*\*\*E E n + n + n \* E

**Q4:** Consider the grammar given below

$$S \rightarrow SS \mid a \mid \epsilon$$

The number of inadequate states in the DFA with LR(0) items are:

- (A) 1
- (B) 2
- (C) 3
- (D) 4

**Q4:** Consider the grammar given below

$S \rightarrow SS \mid a \mid \varepsilon$

The number of inadequate states in the DFA with LR(0) items are:

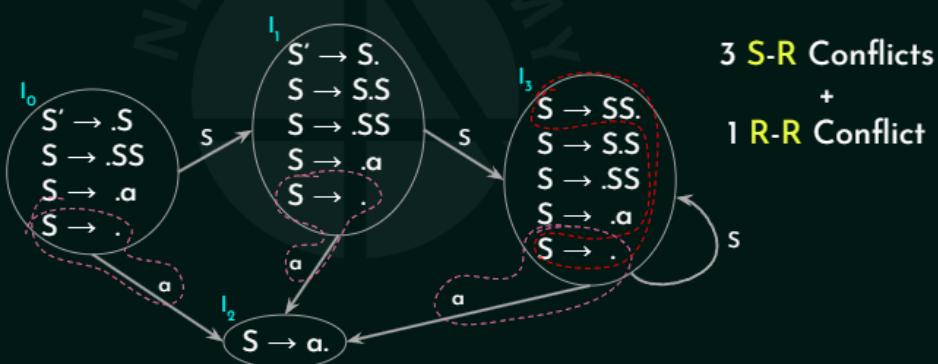
- (A) 1    (B) 2    (C) 3    (D) 4

Sol.  $S' \rightarrow S$

$S \rightarrow SS$

5

S → E



S' → .S10S → SS.S → S.S13(A) 1(B) 2(C) 3(D) 4Q4: Consider the grammar given below

SS | a | ε The number of inadequate states in the DFA with LR(0) items are: Sol.  $S' \rightarrow SS \rightarrow$

$\rightarrow a S \rightarrow S \rightarrow a | 2SaSaa3 S-B$  Conflicts+1 R-B Conflicts



# *Compiler Design*

# Usefulness of Ambiguous Grammars

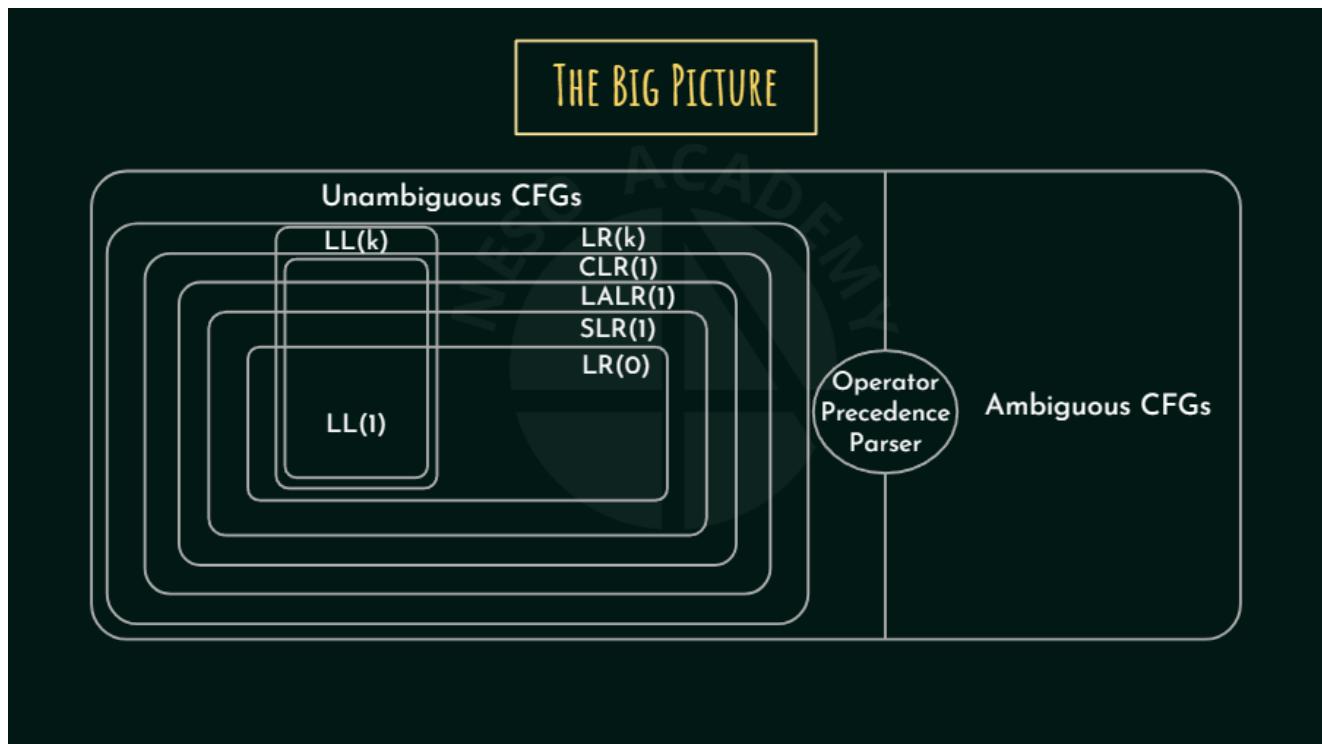
Compiler Design Usefulness of Ambiguous Grammars



## Outcome

- ☆ Advantages of using Ambiguous Grammars.

Outcome ☆ Advantages of using Ambiguous Grammars.



CLR(1)  
LR(k)  
The Big Picture  
Unambiguous CFGs  
Ambiguous CFGs  
Operator Precedence Parser  
LR(0)  
SLR(1)  
LALR(1)  
LL(1)  
LL(k)

## Ambiguous Grammar vs. Unambiguous Grammar:

$X : E \rightarrow E + E \mid E \times E \mid id$

- ↑ #Productions are less.
- ↑ Simple to understand.
- ↑  $X : E \Rightarrow id$  (1 step)
- ↓ Numerous conflicts while parsing.

$Y : E \rightarrow E + T \mid T$

$T \rightarrow T \times F \mid F$

$F \rightarrow id$

- ↓ Associativity is fixed.
- ↓  $Y : E \Rightarrow T \Rightarrow F \Rightarrow id$  (3 steps)

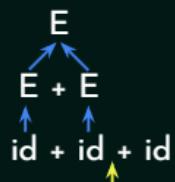
Can be resolved by taking meaningful decisions.

- Associativity is fixed. •  $Y : E \Rightarrow T \Rightarrow F \Rightarrow id$  (3 steps)
- #Productions are less.
- Simple to understand.
- Ambiguous Grammar vs. Unambiguous Grammar:  
 $E \rightarrow E + T \mid T T \rightarrow T \times F \mid FF \rightarrow id$
- $X : E \rightarrow E + E \mid E \times E \mid id$
- Can be resolved by taking meaningful decisions.
- Numerous conflicts while parsing.

## Ambiguous Grammar vs. Unambiguous Grammar:

$X : E \rightarrow E + E \mid E \times E \mid id$

id + id + id



$Y : E \rightarrow E + T \mid T$

$T \rightarrow T \times F \mid F$

$F \rightarrow id$

$E \rightarrow E .+ E$   
 $E \rightarrow E + E.$

S/R ✓

Ambiguous Grammar vs. Unambiguous Grammar:  
 $E \rightarrow E + T \mid T T \rightarrow T \times F \mid FF \rightarrow id$

$X : E \rightarrow E + E \mid E \times E \mid id$

id + id + id  
 id + id + id  
 id + id + id

$E \rightarrow E .+ E$   
 $E \rightarrow E + E.$

S/R ✓

## Ambiguous Grammar vs. Unambiguous Grammar:

X:  $E \rightarrow E + E \mid E \times E \mid id$

id + id x id

id + id x id

Y:  $E \rightarrow E + T \mid T$   
 $T \rightarrow T \times F \mid F$   
 $F \rightarrow id$

E → E + E.  
E → E .x E

✓ S/R

Ambiguous Grammar vs. Unambiguous Grammar:  
 $E \rightarrow E + T \mid T$   $T \rightarrow T \times F \mid F$   $F \rightarrow id$   
X : Y :  
 $E \rightarrow E + E \mid E \times E \mid id$   
 $id + id \times id$   
 $id + id \times id$

## Ambiguous Grammar vs. Unambiguous Grammar:

X:  $E \rightarrow E + E \mid E \times E \mid id$

Yet  
Another  
Compiler  
Compiler



Y:  $E \rightarrow E + T \mid T$   
 $T \rightarrow T \times F \mid F$   
 $F \rightarrow id$

LALR(1)

✓ S/R

✓ R<sub>i</sub>/R<sub>ii</sub>

UNIX



Ambiguous Grammar vs. Unambiguous Grammar:  
 $E \rightarrow E + T \mid T$   $T \rightarrow T \times F \mid F$   $F \rightarrow id$   
X : Y :  
 $E \rightarrow E + E \mid E \times E \mid id$   
UNIX YACC Cet other compiler LALR(1) S/RRi/Rii

## Ambiguous Grammar vs. Unambiguous Grammar:

X : E → E + E | E × E | id

- ↑ #Productions are less.
- ↑ Simple to understand.
- ↑ X : E ⇒ id (1 step)
- ↑ Conflicts can be resolved by taking meaningful decisions.

Y : E → E + T | T

T → T × F | F  
F → id

- ↓ Associativity is fixed.
- ↓ Y : E ⇒ T ⇒ F ⇒ id (3 steps)

- Associativity is fixed. ● Y : E ⇒ T ⇒ F ⇒ id (3 steps)
  - #Productions are less. ● Simple to understand.
  - X : E ⇒ id (1 step)
  - Conflicts can be resolved by taking meaningful decisions.
- Ambiguous Grammar vs. Unambiguous Grammar:  
E → E + T | T T → T × F | F F → id  
X : E → E + E | E × E | id