



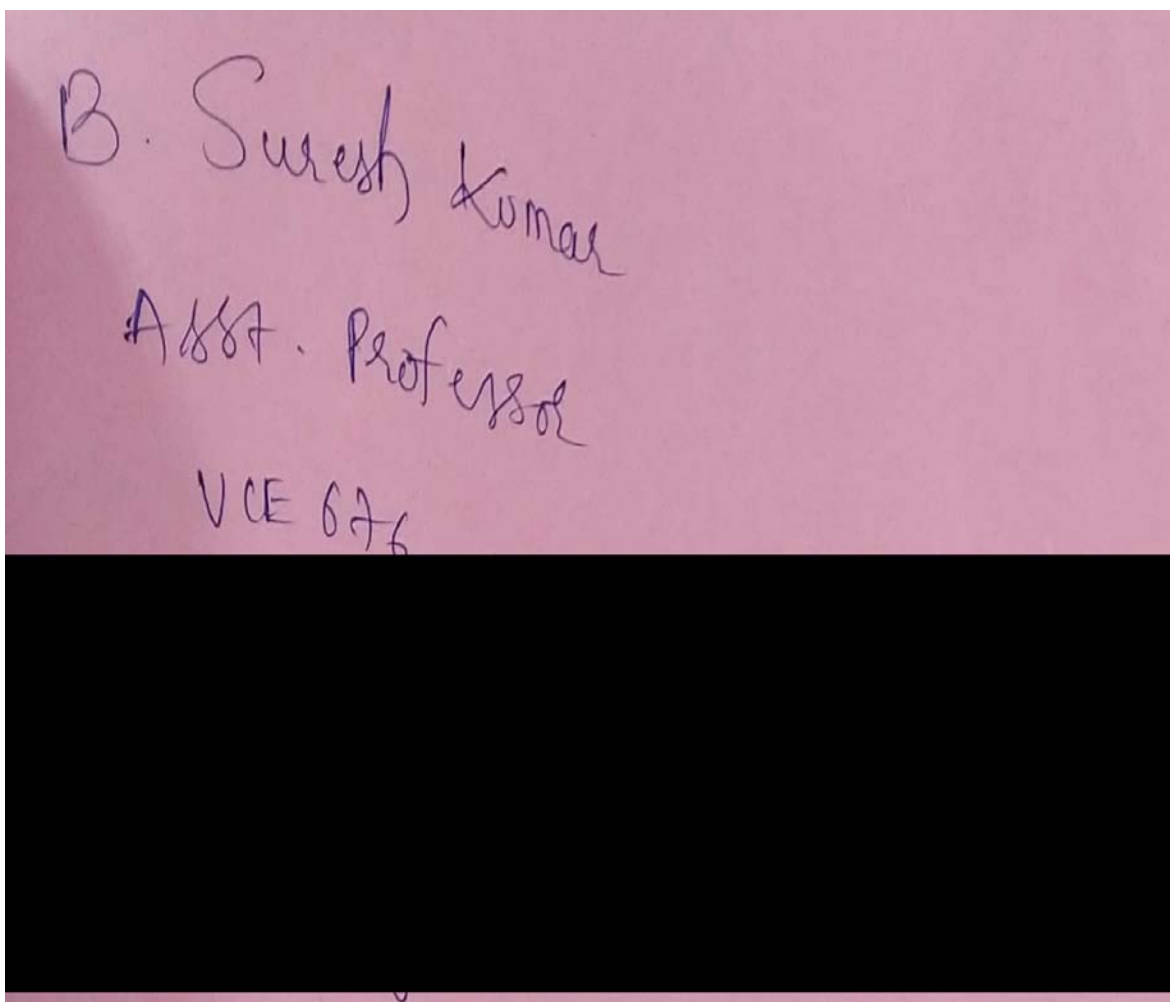
VARDHAMAN COLLEGE OF ENGINEERING (AUTONOMOUS)

Accredited by NAAC with 'A' Grade, NBA, ISO 9001:2008 Certified
Approved by AICTE, New Delhi, Affiliated to JNTUH
Programmes Accredited by NBA



Department of Computer Science & Engineering

Dynamic Activities Conducted by the Faculty Members



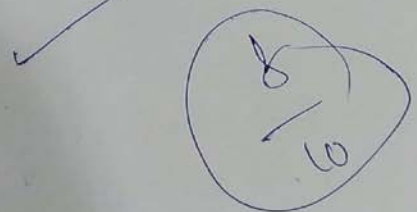
Tue Jan Mon / Tue Jan Tue / ~~Wed~~
 Wed Jan Tue / Wed
 Thu Feb Wed / Thu
 Mar

Tue Jan Tue [reduce Tue → Wed]
 Wed Jan Tue [Tue → Wed]
 Thu Jan Tue [Wed → Thu]
 Mar Jan Tue [Thu → Mar]
 Mar Jan Tue Jan Tue [reduce Tue → Tue Jan Tue]
 Mar Jan Wed Jan Tue [Tue → Wed]
 Mar Jan Thu Jan Tue [Wed → Thu]
 Mar Jan Mar Jan Tue [Thu → Mar]
 Mar Jan Mar Jan Wed [reduce Tue → Wed]
 Mar Jan Mar Jan Thu [Wed → Thu]
 Mar Jan Mar Jan Mar [Thu → Mar]

stack
 Mar
 Thu
 Jan

I/p
 Mar Jan Mar Jan Mar
 Jan Mar Jan Mar
 Jan Mar Jan Mar
 Mar Jan Mar

Action
 shift
 reduce [Thu → Mar]
 shift



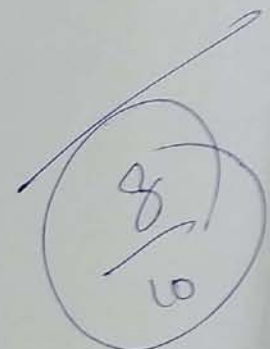
$E \rightarrow T \mid T + E$
 $V \rightarrow T \mid V$
 $T \rightarrow V * T \mid V$
 $V \rightarrow id$
 $id * id + id$

Ellipse \rightarrow Term + Ellipse | Term
 Term \rightarrow Verus * Term | Verus
 Verus \rightarrow id
 JIP: id + id + id

$E \rightarrow T + E \mid T$
 $E \rightarrow T + E \quad (E \rightarrow T)$
 $E \rightarrow V * T + T \quad (V * T), T \rightarrow V * T$
 $E \rightarrow V * T + T \quad (T \rightarrow V * T), T \rightarrow V * T$
 $E \rightarrow V * V + T \quad (T \rightarrow V)$
 $E \rightarrow V * V + V \quad (V \rightarrow id)$
 $E \rightarrow V * V + id \quad (V \rightarrow id)$
 $E \rightarrow V * id + id \quad (V \rightarrow id)$
 $E \rightarrow id * id + id \quad (V \rightarrow id)$

$E \rightarrow T + E \quad (E \rightarrow T)$
 $E \rightarrow T + T \quad (T \rightarrow V * T)$
 $E \rightarrow T + V * T \quad (T \rightarrow V)$
 $E \rightarrow T + V * V \quad (V \rightarrow id)$
 $E \rightarrow T + V * id \quad (V \rightarrow id)$
 $E \rightarrow T + id * id \quad (T \rightarrow V * T)$
 $E \rightarrow V + id * id \quad (V \rightarrow id)$
 $E \rightarrow id + id * id$

task	ilp	Action
\$	id + id * id \$	shift id
id	+ id * id \$	reduce $V \rightarrow id$
v	* id * id \$	shift *
v +	id * id \$	shift id
v + id	x id \$	reduce $V \rightarrow id$
v + v	* id \$	shift +
v + v *	id \$	shift id
v + v * id	\$	reduce $V \rightarrow id$
v + v * v	\$	reduce $T \rightarrow T$
v + v * T	\$	reduce $V \rightarrow T$
v + T * T	\$	reduce $T \rightarrow V * T$
v + T + T	\$	reduce $T \rightarrow E$



$\$ T * E \quad \$$ reduce
 $\$ E \quad \$$ acc

$\rightarrow A+B$
 $\rightarrow aB; / c$
 $\rightarrow bA / b/aA$
 $\rightarrow bba+ac; / b/a/a/a$

$S \rightarrow A+B$ [$S \rightarrow A+B$]
 $\rightarrow A+aB;$ [$B \rightarrow aB;$]
 $\rightarrow A+ac;$ [$B \rightarrow c$]
 $\rightarrow bA+ac;$ [$A \rightarrow bA$]
 $\rightarrow b^2A+ac;$ [$A \rightarrow aA$]
 $\rightarrow b^3A+ac;$ [$A \rightarrow aA$]

Stack	Action
$bba+ac;$	shift
$ba+ac;$	reduce $A \rightarrow b$
$ba+ac;$	shift
$b+ac;$	shift

Stack	Input	Action
$bba+ac; \$$		shift
$ba+ac;$		$A \rightarrow b$ shift
$ba+ac;$		$A \rightarrow b$
$ba+ac;$		shift
$+ac;$		$A \rightarrow A^a$
$+ac;$		$A \rightarrow bA$
$+ac;$		shift
$ac;$		shift
$c;$		shift
$;$		shift $B \rightarrow c$
$;$		shift
$;$		shift $B \rightarrow aB;$
$;$		$S \rightarrow A+B$
$;$		Accept

$\frac{2}{10}$

function table.

B. Suresh Kumar

UCE 67-6

Asst. Prof, CSE

Activity on grammar identification
(operator precedence)

Subject: Compiler Design. V Sem

Function table.
 → ABd
 → alb
 → Lcid/lid } Find precedence function table ?

TA
 HTA/E
 FB
 *FB/E } Find
 (E)/id } First and Follow and precedence function table ?

— ALL THE BEST —

ABd
 alb
 Lcid lid

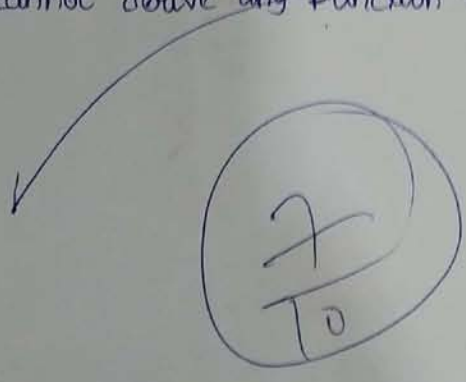
eliminate "L" production rules because it is not useful. so, the production rules follows

→ ABd
 → alb.

replace A by alb and substitute in production "S" in order to get the production grammar format.

→ aBd | bBd
 → alb.

production rules, so we cannot derive any function table.



2. $E \rightarrow TA$
 $A \rightarrow +TA|E$
 $T \rightarrow FB$
 $B \rightarrow *FB|E$
 $F \rightarrow (E)|id.$

$First(E) = First(T) = First(F) = \{c, id\}$

$First(A) =$

	First	Follow
$E \rightarrow TA$	$\{c, id\}$	$\{\$, \}$
$A \rightarrow +TA E$	$\{+, \epsilon\}$	$\{\$, \), \}$
$T \rightarrow FB$	$\{c, id\}$	$\{\$, \), \}$
$B \rightarrow *FB E$	$\{*, \epsilon\}$	$\{\$, \), \}$
$F \rightarrow (E) id$	$\{c, id\}$	$\{\$, \), *, \}$

~~$E \rightarrow TA$~~
 ~~$E \rightarrow +TA|T$~~
 ~~$E \rightarrow +T|E$~~
 $E \rightarrow T|E$

$E \rightarrow TA$
 $E \rightarrow T+TA|T$
 $E \rightarrow T|E$
 $A \rightarrow +E|E$
 $F \rightarrow FB$
 $T \rightarrow idB|(E)B$
 $B \rightarrow *FB|E$
 $B \rightarrow *T|E$

$E \rightarrow T+ET$
 $A \rightarrow +E|E$
 $T \rightarrow idB|(E)B$
 $B \rightarrow *T|E$
 $F \rightarrow (E)|id.$

precedence function

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

precedence function table.

1B5/E

3A/ab

BAB/E/C

LR parsing action table (bintbcharb)

charS/a/b

$$A \rightarrow A\alpha(B \Rightarrow)$$

$$A \rightarrow \beta A'$$

$$A' \rightarrow \alpha A' | \epsilon$$

$$S \rightarrow \epsilon S'$$

$$S' \rightarrow \text{int } L S' | \epsilon$$

$$S \rightarrow \epsilon S''$$

$$S'' \rightarrow L S'' | \epsilon$$

$$L \rightarrow aL' | bL'$$

$$L' \rightarrow \text{char } SL'$$

$$S \rightarrow S'$$

$$S' \rightarrow \text{int } L S' | \epsilon$$

$$S \rightarrow S''$$

$$S'' \rightarrow L S'' | \epsilon$$

$$L \rightarrow aL' | bL'$$

$$L' \rightarrow \text{char } SL'$$

	First	Follow		a	b	int	char
S	int, a, b, ε	\$, a, b, int, char	⇒	S	S → S'	S → S''	S → S'
S'	int, ε	\$, a, b, int, char		S'	S → S'	S → S''	S → S'
S''	a, b, ε	\$, a, b, int, char		S''	S → ε	S → ε	S → ε
L	a, b	\$, a, b, int, char		L			
L'	char	\$, a, b, int, char		L'			

2/10

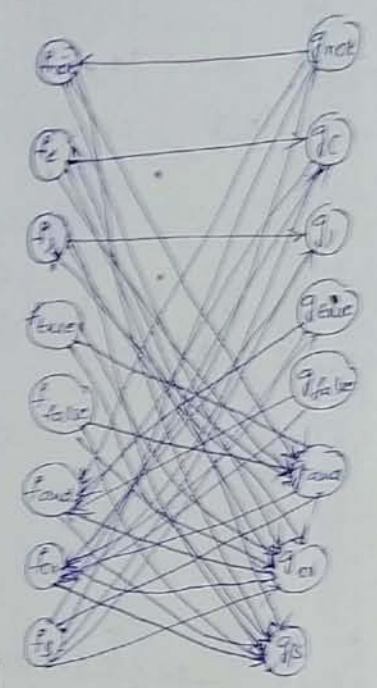
LL(1) parser can't be designed

& given string can't be validated

~~max~~ ^{or} yum | yum
 Yeeha and Yorraara | Yomara
 → not Yomara | (max) | True | false.

Construct a parsing table

- te'
- +te' | ε
- ft'
- ε' → *ft' | ε
- f → (ε) | x | y



Solutions

max → max or yum | yum
 max → ~~ε~~ max' → max → yum or yum max'
 max' → or yum max' | ε max = yum or max | yum
 yum → yeeha and yorraara | yomara
 yomara → not yomara | (max) | true | false

fnot - fnot - ε
 fnot - fnot - ε - ε
 fnot - fnot - ε - ε - ε → LR.

	not	()	true	false	and	or	\$
not	<	=	=	=	=	>	>	>
(=	<	=	=	=	>	>	>
)	=	=	>	=	=	>	>	>
true	=	=	=	-	-	>	>	>
false	=	=	=	-	-	>	>	>
and	<	<	<	<	<	>	>	>
or	<	<	<	<	<	<	>	>
\$	<	<	<	<	<	<	<	-

	not	()	true	false	and	or	\$
f	3	0	0	0	0	2	1	0
g	4	0	0	0	0	2	1	0

7
(0)

2

	First	Follow
$e \rightarrow te'$	$\{ (, x, y \}$	$\{ \$,) \}$
$e' \rightarrow +te' / \epsilon$	$\{ +, \epsilon \}$	$\{ \$,) \}$
$t \rightarrow ft'$	$\{ (, x, y \}$	$\{ \$,), + \}$
$t' \rightarrow *ft' / \epsilon$	$\{ *, \epsilon \}$	$\{ \$,), + \}$
$f \rightarrow (e) / x / y$	$\{ (, x, y \}$	$\{ \$,), +, * \}$

	x	y	()	+	*	\$
e	$e \rightarrow te'$	$e \rightarrow te'$	$e \rightarrow te'$	error	error	error	error
e'				$e' \rightarrow \epsilon$	$e' \rightarrow +te'$	error	$e' \rightarrow \epsilon$
t	$t \rightarrow ft'$	$t \rightarrow ft'$	$t \rightarrow ft'$				
t'				$t' \rightarrow \epsilon$	$t' \rightarrow \epsilon$	$t' \rightarrow *ft'$	$t' \rightarrow \epsilon$
f	$f \rightarrow x$	$f \rightarrow y$	$f \rightarrow (e)$				

$V \rightarrow TS$

$S \rightarrow Sabq/e$

$B \rightarrow * r'f/e/int$

$F \rightarrow id$

Compiler $\rightarrow * Expr \cdot Design$

Design $\rightarrow + Design / Factor ;$

Expr $\rightarrow Expr Factor \cdot / Factor ($

Factor $\rightarrow id / (Factor)$

$/ B$

$= / bB$

B/w find precedence table 2

$* w/M$

→ find A the production of $A \rightarrow BC$, In this B and C both are terminals so we need to change that.

$A \rightarrow BC \cdot B$

$\rightarrow Bb \cdot Bc \mid Bbb \mid B (\because c \rightarrow bBc \mid bB)$

$A \rightarrow BbA \mid Bbb \mid B (\because A \rightarrow BC)$

the productions are

$A \rightarrow BbA \mid Bbb \mid B$

$B \rightarrow wbB \mid w$

$w \rightarrow M * w \mid M$

$M \rightarrow id$

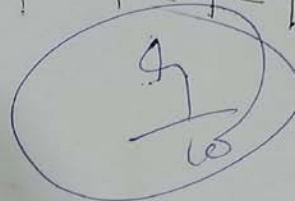
e eliminated c production

because the c production

s not reaching starting symbol

operator precedence Table

operator	id	*	b	\$
id		\rightarrow	\rightarrow	\rightarrow
*	\leftarrow	\leftarrow	\rightarrow	\rightarrow
b	\leftarrow	\leftarrow	\leftarrow	\rightarrow
\$	\leftarrow	\leftarrow	\leftarrow	-



Faculty Name: Mr. B. Suresh Kumar

Activity Name: TAPPS

Class: B.Tech III Year CSE

Subject: Compiler Design

Topic: LR grammars, LR parsers-simple LR