LR(k) PARSER

- The canonical set of items is the parsing technique in which a lookahead symbol is generated while constructing set of items.
- Hence the collection of set of items is referred as LR(1). The value 1 in the bracket indicates that there is one lookahead symbol in the set of items.
- We follow the same steps as discussed in SLR parsing technique and those are

1) Construction of canonical set of items along with the lookahead.
2) Building canonical LR Parsing table.
3) Passing the input string using canonical LR Parsing table.

Construction of canonical set of items along with the lookahead.:

1) For the grammar G initially add $S' \rightarrow \cdot S$ in the set of item C.
2) For each set of items I_i in C and for each grammar symbol X (may be terminal or NT) add closure (I_i, X). This process should be repeated by applying goto (I_i, X) for each X in I_i such that goto (I_i, X) is not empty and not in C. The set of items has to be constructed until no more set of items can be added to C.
3) The closure function can be computed as follows: for each item $A \rightarrow \alpha \cdot X\beta, a$ and rule $X \rightarrow \gamma$ and $b \in FIRST(\beta a)$ such that $X \rightarrow \cdot\gamma$ and b is not in I then add $X \rightarrow \cdot\gamma, b$ to I.
4) Similarly the goto function can be computed as; For each item $[A \rightarrow \alpha \cdot X\beta, a]$ is in I and rule $[A \rightarrow \alpha X \cdot \beta, a]$ is not in goto items then add $[A \rightarrow \alpha X \cdot \beta, a]$ to goto items.

This process is repeated until no more set of items can be added to the collection C.

Eg

$S' \to S$

$S \to CC$

$C \to aC \mid d$

Construct LR(1) set of items for the above grammar.

Soln We will initially add $S' \to \cdot S, \$$ as the first rule in I_0. Now match.

$S' \to \cdot S, \$$ with $[A \to \alpha \cdot X \beta, a]$

Hence $S' \to \cdot S, \$$

$A \to \alpha \cdot X \beta, a$

$A = S', \alpha = \epsilon, X = S, \beta = \epsilon, a = \$$

If there is a production $X \to \gamma, b$ then add $X \to \cdot \gamma, b$

$\because S \to \cdot CC$, $b \in FIRST(\beta a)$

$b \in FIRST(\epsilon \$)$ as $\epsilon \$ = \$$

$b \in FIRST(\$)$

$b = \{\$\}$

$\therefore S \to \cdot CC, \$$ will be added in I_0.

Now $S \to \cdot CC, \$$ is in I_0 we will match it with $A \to \alpha \cdot X\beta, a$

$A = S', \alpha = \epsilon, X = C, \beta = C, a = \$$

If there is a production $X \to \gamma, b$ then add $X \to \cdot \gamma, b$

$\because C \to \cdot aC$ $b \in FIRST(\beta a)$

$C \to \cdot d$ $b \in FIRST(C\$)$

$b \in FIRST(C)$ as $FIRST(C) = \{a, d\}$

$b = \{a, d\}$.

$C \to \cdot aC, a$ or d will be added in I_0.

Hence I_0:

$S' \to \cdot S, \$$

$S \to \cdot CC, \$$

$C \to \cdot aC, a/d$

$C \to \cdot d, a/d$

Hence a/d is used to denote a or d. That means for the production $C \to \cdot aC, a$ and $C \to \cdot aC, d$

Now apply goto on I_0.

$S' \to \cdot S, \$$

$S \to \cdot CC, \$$

$C \to \cdot aC, a/d$

$C \to \cdot d, a/d$

Hence

goto (I_0, S)

I_1 : $S' \to S \cdot, \$$

Now apply goto on C in I_0.

$S \to C \cdot C, \$$ add in I_2. Now as after dot C comes we will add the rules of C.

$X = C, \beta = \epsilon, a = \$$

$X \to \cdot\gamma, b$ where $b \in FIRST(\beta a)$

$C \to \cdot aC$ $\quad b \in FIRST(\epsilon, \$)$

$C \to \cdot d$ $\quad b \in FIRST(\$)$

Hence

$C \to \cdot aC, \$$ and $C \to \cdot d, \$$ will be added to I_2

I_2 : goto (I_0, C)

$S \to C \cdot C, \$$

$C \to \cdot aC, \$$ → add $ for both productions.

$C \to \cdot d, \$$

Now we will apply goto on a of I_0 for the rule $C \to \cdot aC, a/d$ that becomes $C \to \cdot aC, a/d$ will be added in I_3.

$C \to a \cdot C, a/d$

As $A = C, \alpha = a, X = C, \beta = \epsilon, a = a/d$.

Hence $X \to \cdot\gamma, b$

$C \to \cdot aC$ $\quad b \in FIRST(\beta a)$

$C \to \cdot d$ $\quad b \in FIRST(\epsilon a)$ or $FIRST(\epsilon d)$

$b = a/d$

Hence I_3 : goto (I_0, a)

$C \to a \cdot C, a/d$

$C \to \cdot aC, a/d$

$C \to \cdot d, a/d$

Now if we apply goto on d of I_0 in the rule $C \to \cdot d, a/d$ then we get $C \to d \cdot a/d$ hence I_4 becomes

I_4 : goto (I_0, d)

$C \to d \cdot, a/d$

As applying gotos on I_0 is over. we will move to I_1 but we get no new production from I_1, we will apply goto on C in I_2. And there is no closure possible in this state.

I_5 : goto (I_2, C)

$S \to CC \cdot, \$$

We will apply goto on a from I_2 on the rule $C \to \cdot aC, \$$ and we get the state I_6.

$C \to a \cdot C, \$$

$A \to \alpha \cdot X\beta, a$

$A = C, \alpha = a, X = C, \beta = \varepsilon, a = \$$

$X \to \cdot\gamma$

$C \to \cdot aC$ and $C \to \cdot d$

$b \in$ FIRST (βa)

$b \in$ FIRST $(\varepsilon \$)$

$b = \$$

I_6 : goto (I_2, a)

$C \to a \cdot C, \$$

$C \to \cdot aC, \$$

$C \to \cdot d, \$$

You can note one thing that I_3 and I_6 are different because the second component in I_3 and I_6 is different

Apply goto on d of I_2 for the rule $C \to \cdot d, \$$

I_7 : goto (I_2, d)

$C \to d \cdot, \$$

Now if we apply goto on a and d of I_3 we will get I_3 and I_4 respectively and there is no point in repeating the states. So we will apply goto on C of I_3

I_8 : goto (I_3, C)
$C \to aC\cdot, a/d$

For I_4 and I_5 there is no point in applying gotos. Applying gotos on a and d of I_6 gives I_6 and I_7 respectively. Hence we will apply goto on I_6 for C for the rule.

$C \to a\cdot C, \$$

I_9 : goto (I_6, C)
$C \to aC\cdot, \$$

For the remaining states I_7, I_8 and I_9 we cannot apply goto. Hence the process of construction of set of LR(1) items is completed. Thus the set of LR(1) items consists of I_0 to I_9 states.

I_0 :
$S' \to \cdot S, \$$
$S \to \cdot CC, \$$
$C \to \cdot aC, a/d$
$C \to \cdot d, a/d$

I_1 : goto (I_0, S)
$S' \to S\cdot, \$$

I_2 : goto (I_0, C)
$S \to C\cdot C, \$$
$C \to \cdot aC, \$$
$C \to \cdot d, \$$

I_3 : goto (I_0, a)
$C \to a\cdot C, a/d$
$C \to \cdot aC, a/d$
$C \to \cdot d, a/d$

I_4 : goto (I_0, d)
$C \to d\cdot, a/d$

I_5 : goto (I_2, C)
$S \to CC\cdot, \$$

I_6 : goto (I_2, a)
$C \to a\cdot C, \$$
$C \to \cdot aC, \$$
$C \to \cdot d, \$$

I_7 : goto (I_2, d)
$C \to d\cdot, \$$

I_8 : goto (I_3, C)
$C \to aC\cdot, a/d$

I_9 : goto (I_6, C)
$C \to aC\cdot, \$$

Construction of canonical LR parsing table.

Input : An augmented grammar G'

Output : The canonical LR parsing table.

Algorithm :

1) Initially construct set of items $C = \{I_0, I_1, \ldots I_n\}$ where C is a collection of set of LR(1) items for the input grammar G'.

2) The parsing actions are based on each item I_i. The actions are as given below.
   a) If $[A \to \alpha . a\beta, b]$ is in I_i and goto $(I_i, a) = I_j$ then create an entry in the action table action $[I_i, a]$ = shift j

   b) If there is a production $[A \to \alpha ., a]$ in I_i then in the action table set action $[I_i, a]$ = reduce by $A \to \alpha$ Here A should not be S'

   c) If there is a production $S' \to S ., \$$ in I_i then set action $[i, \$]$ = accept

3) The goto part of the LR table can be filled as: The goto transitions for state i is considered for non-terminals only. If goto $(I_i, A) = I_j$ then set goto $[I_i, A] = j$

4) All entries not defined by rule 2 and 3 are considered to be "error"

LR(1) Parsing Table

states	Action a	Action d	Action $	Goto S	Goto C
0	s3	s4		1	2
1			accept		
2	s6	s7			5
3	s3	s4			8
4	r4 (r3)	r4 (r3)			
5			r2 (r1)		
6	s6	s7			9
7			r4 (r3)		
8	r3 (r2)	r3 (r2)			
9			r3 (r2)		

I_0

$A \to \alpha \cdot a \beta, b$

$C \to \cdot aC, a/d$ $\quad$ $C \to \cdot d, a/d$

goto $(I_0, a) = I_3$ $\quad$ goto $(I_0, d) = I_4$

Set action $[0, a] = s3$ $\quad$ Set action $[0, d] = s4$.

I_2

$C \to \cdot aC, \$$ $\quad$ $C \to \cdot d, \$$

goto $(I_2, a) = I_6$ $\quad$ goto $(I_2, d) = I_7$

Set action $[2, a] = s6$ $\quad$ Set action $[2, d] = s7$

I_3

$C \to \cdot aC, a/d$ $\quad$ $C \to \cdot d, a/d$

goto $(I_3, a) = I_3$ $\quad$ goto $(I_3, d) = I_4$

Set action $[3, a] = s3$ $\quad$ Set action $[3, d] = s4$

I_4

$A \to \alpha \cdot, a$

$C \to d \cdot, a/d$

Set action $[4, a] = r4$

" " $[4, d] = r4$

I_5

$S \to CC \cdot, \$$

Set action $[5, \$] = r2$

I_6

$C \to \cdot aC, \$$ $\quad$ $C \to \cdot d, \$$

goto $(I_6, a) = I_6$ $\quad$ goto $(I_6, d) = I_7$

Set action $[6, a] = s6$. $\quad$ Set action $[6, d] = s7$

I_7 $C \to d \cdot, \$$

set action $[7, \$] = r4$

I_8 $C \to aC \cdot, a|d$

set action $[8, a] = r3$

" " $[8, d] = r3$

I_9 $C \to aC \cdot, \$$

set action $[9, \$] = r3$

GOTO Transitions

goto $(I_0, S) = I_1$ goto $(I_0, C) = I_2$

set goto $(0, S) = 1$ set goto $(0, C) = 2$

Parsing the Input using LR(1) parsing table

String is aadd

Stack	Input buffer	Action table	Goto table	Parsing Action
$0	aadd$	action[0,a] = s3		Shift
$0a3	add$	action[3,a] = s3		Shift
$0a3a3	dd$	action[3,d] = s4		Shift
$0a3a3d4	d$	action[4,d] = r4	[3,C] = 8	Reduce by C→d
$0a3a3C8	d$	action[8,d] = r3	[3,C] = 8	Reduce by C→aC
$0a3C8	d$	action[8,d] = r3	[0,C] = 2	Reduce by C→aC
$0C2	d$	action[2,d] = s7		Shift
$0C2d7	$	action[7,$] = r4	[2,C] = 5	Reduce by C→d
$0C2C5	$	action[5,$] = r2	[0,S] = 1	Reduce by S→CC
$0S1	$	action[1,$] = acc		Accept

Eg Show that the following grammar

$S \to Aa \mid bAc \mid Bc \mid bBa$

$A \to d$

$B \to d$

is LR(1)

Construct SLR Parsing table for the grammar:

$S \rightarrow AB \mid gDa$

$A \rightarrow ab \mid c$

$B \rightarrow dC$

$C \rightarrow gC \mid g$

$D \rightarrow fD \mid g$.