1

الف) نادرست : $LR(1) \subset LR(2)$

مثال:

$$S \rightarrow A1\ B\ x \mid A2\ B\ y$$
$$A1 \rightarrow a$$
$$A2 \rightarrow a$$
$$B \rightarrow b$$

این گرامر $LR(2)$ است اما $LR(1)$ نیست.

ب) درست است، به ازای هر گرامری $LR(k)$ ، یک گرامر $LR(1)$ وجود دارد که نشان دهنده زبان یکسان است، اما این بدین معنی نیست که برای این دو گرامر، پارسرها یکسان هستند.

ج) درست است، الگوریتم CYK میتواند هر گرامری را در زمان $O(n^3)$ تجزیه کند.

د) نادرست است، به طور مثال گرامر زیر نامبهم است اما $SLR(1)$ نیست

$$S \rightarrow L = R$$
$$S \rightarrow R$$
$$L \rightarrow * R$$
$$L \rightarrow id$$
$$R \rightarrow L$$

2

قدرت:

$$SLR(1) \leq LALR(1) \leq LR(1) \leq LR(k)$$

پیچیدگی پیاده سازی و حافظه:

$$LR(k) >> LR(1) >> LALR(1) = SLR(1)$$

3

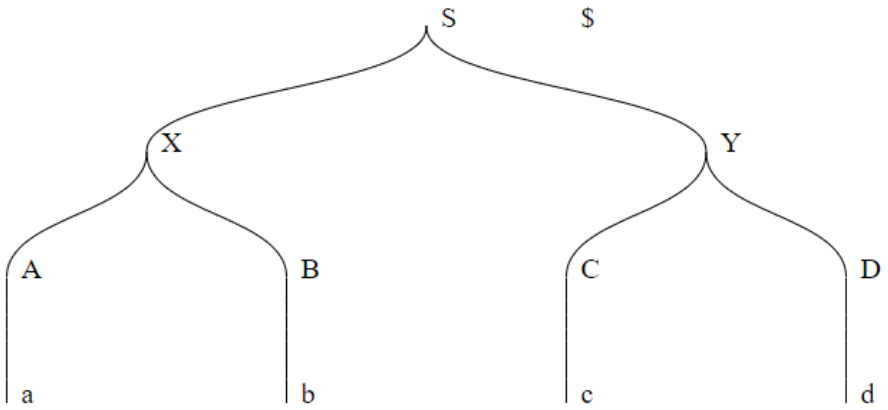
$$n_2 = n_3 < n_1$$

$$2n - 1$$

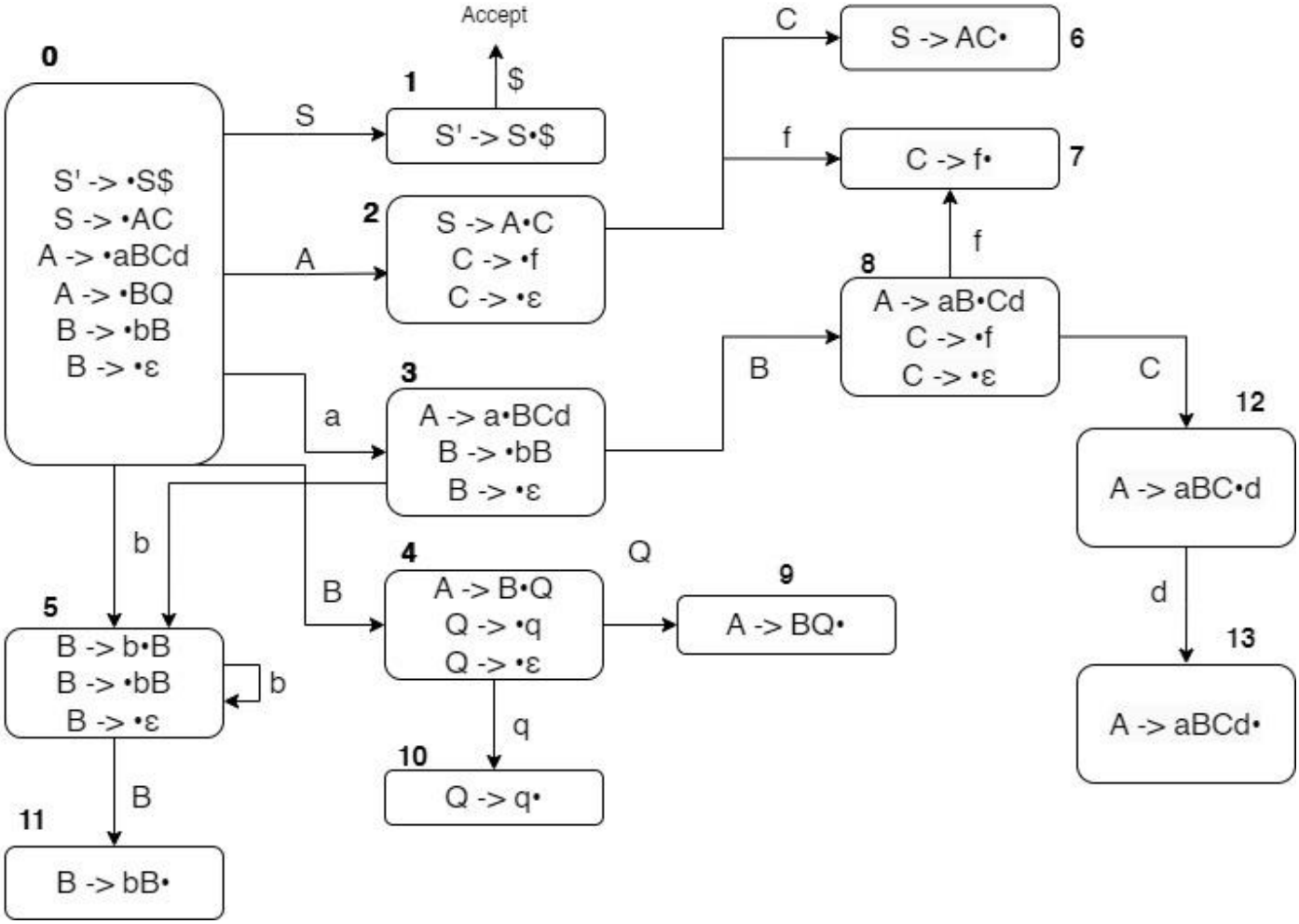
مثال: گرامر زیر را در نظر بگیرید

$$S \to XY$$
$$Y \to CD$$
$$X \to AB$$
$$D \to d$$
$$C \to c$$
$$B \to b$$
$$A \to a$$

برای تجزیه رشته $abcd$ داریم

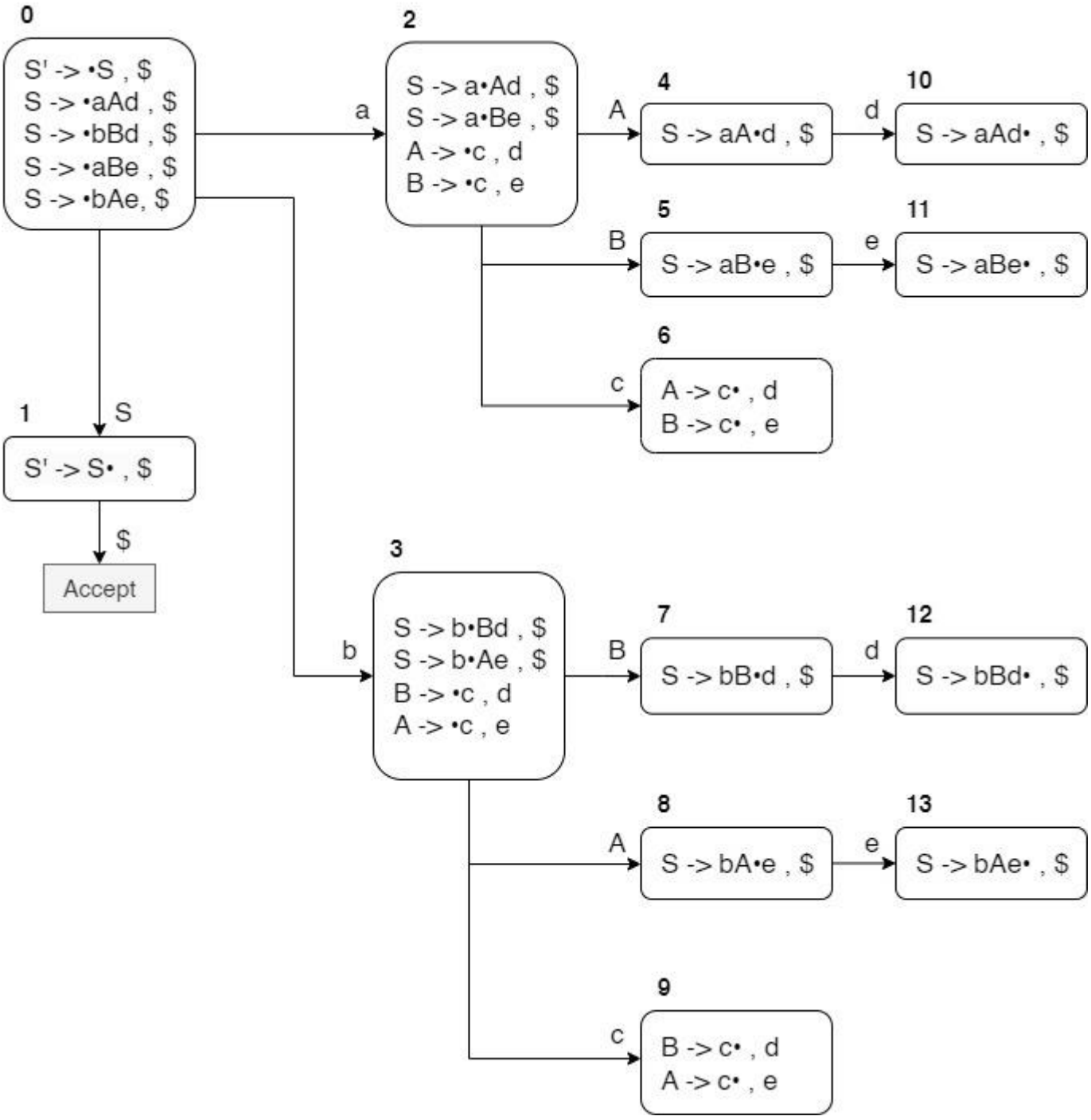


تعداد کاهش ها برابر ۷ است



State	ACTION						GOTO						Productions
	a	**d**	**b**	**f**	**q**	**$**	**S'**	**S**	**A**	**B**	**C**	**Q**	
0	s3	r_5	s5	r_5	r_5	r_5		1	2	4			
1						acc							
2		r_7		s7		r_7					6		
3		r_5	s5	r_5	r_5	r_5				8			**0: S'->S**
4				r_9	s10	r_9						9	**1: S->A C**
5		r_5	s5	r_5	r_5	r_5				11			**2: A->a B C d**
6						r_1							**3: A->B Q**
7		r_6				r_6							**4: B->b B**
8		r_7		s7		r_7					12		**5: B->ε**
9				r_3		r_3							**6: C->f**
10				r_8		r_8							**7: C->ε**
11		r_4		r_4	r_4	r_4							**8: Q->q**
12		s13											**9: Q->ε**
13				r_2		r_2							

Stack	Input	Action
0	a b b d f $	Shift 3
0 3	b b d f $	Shift 5
0 3 5	b d f $	Shift 5
0 3 5 5	d f $	Reduce 5 **B->ε**
0 3 5 5 11	d f $	Reduce 4 **B->b B**
0 3 5 11	d f $	Reduce 4 **B->b B**
0 3 8	d f $	Reduce 7 **C->ε**
0 3 8 12	d f $	Shift 13
0 3 8 12 13	f $	Reduce 2 **A->a B C d**
0 2	f $	Shift 7
0 2 7	$	Reduce 6 **C->f**
0 2 6	$	Reduce 1 **S->A C**
0 1	$	ACCEPT


0
S' -> •S , $
S -> •aAd , $
S -> •bBd , $
S -> •aBe , $
S -> •bAe, $
a
2
S -> a•Ad , $
S -> a•Be , $
A -> •c , d
B -> •c , e
A
4
S -> aA•d , $
d
10
S -> aAd• , $
B
5
S -> aB•e , $
e
11
S -> aBe• , $
c
6
A -> c• , d
B -> c• , e
1
S
S' -> S• , $
$
Accept
b
3
S -> b•Bd , $
S -> b•Ae , $
B -> •c , d
A -> •c , e
B
7
S -> bB•d , $
d
12
S -> bBd• , $
A
8
S -> bA•e , $
e
13
S -> bAe• , $
c
9
B -> c• , d
A -> c• , e

State	ACTION						GOTO				Productions
	a	**d**	**b**	**e**	**c**	**$**	**S'**	**S**	**A**	**B**	
0	s2		s3					1			
1						acc					
2					s6				4	5	
3					s9				8	7	
4		s10									**0: S'->S**
5				s11							**1: S->a A d**
6		r_5		r_6							**2: S->b B d**
7		s12									**3: S->a B e**
8				s13							**4: S->b A e**
9		r_6		r_5							**5: A->c**
10						r_1					**6: B->c**
11						r_3					
12						r_2					
13						r_4					

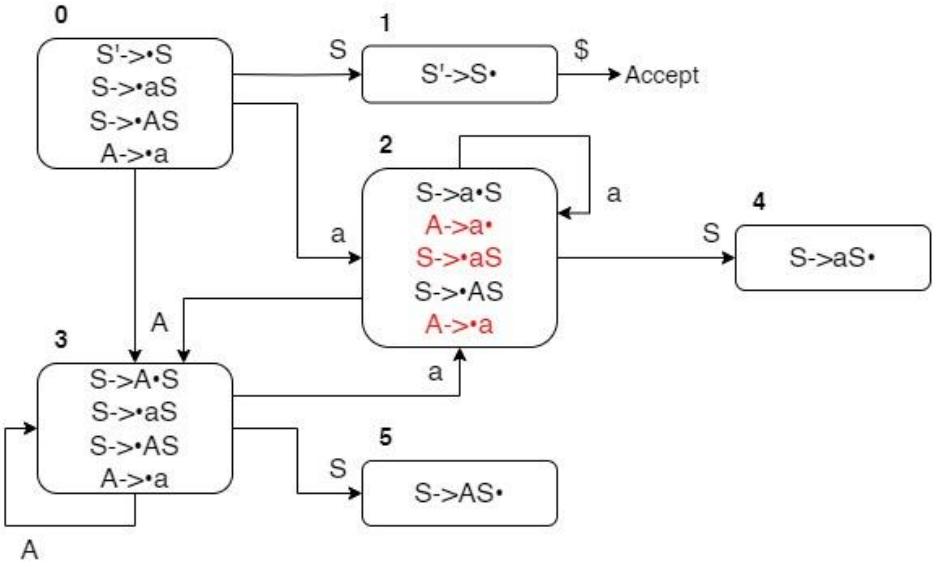
ج) حالات 6 و 9

د) خیر، کانفلیکت بین r5 و r6 برای سمبل‌های d و e

7

الف)

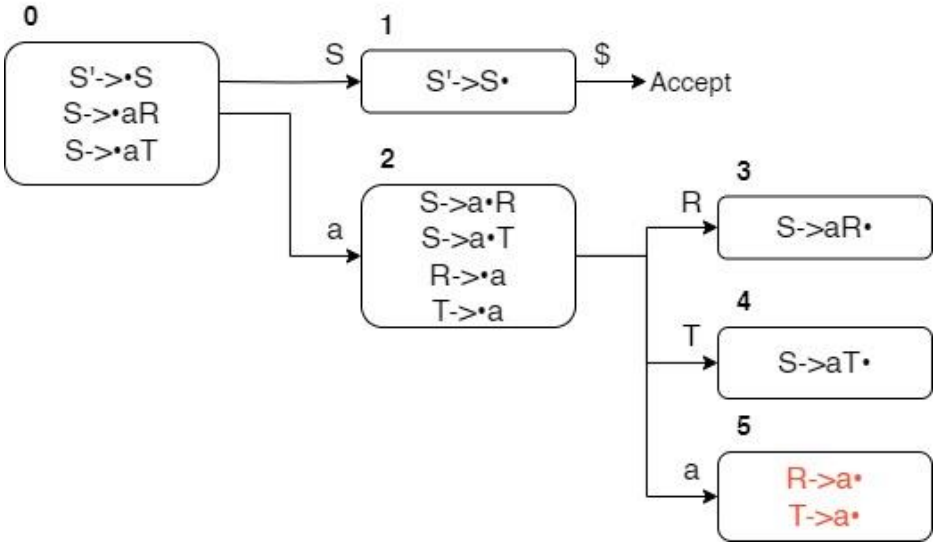
$$S' \rightarrow S\ \$$$
$$S \rightarrow a\ S$$
$$S \rightarrow A\ S$$
$$A \rightarrow a$$



کانفلیکت در استیت 2

ب)

$$S' \rightarrow S\ \$$$
$$S \rightarrow a\ R$$
$$S \rightarrow a\ T$$
$$T \rightarrow a$$
$$R \rightarrow a$$



کانفلیکت در استیت 5

8

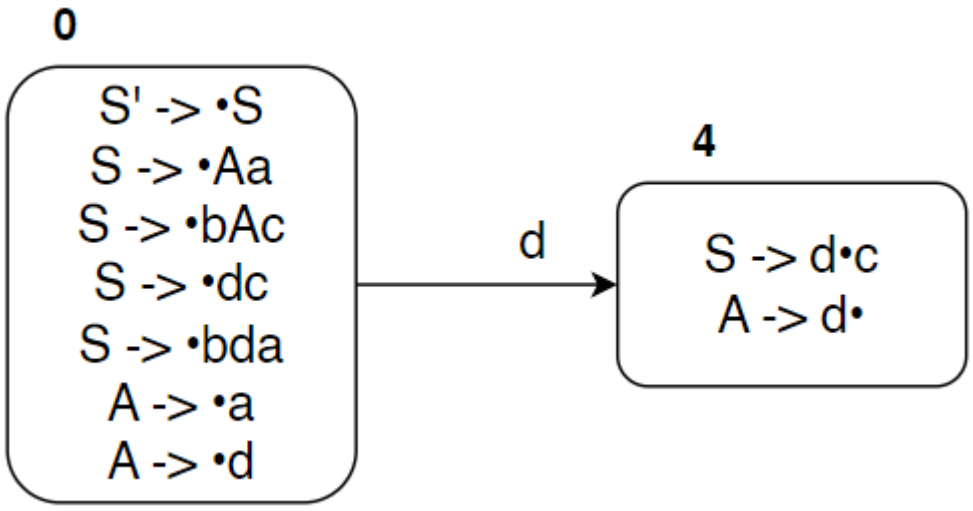
9

الف)

Skipping state diagram, we'll have the following table for $LALR(1)$, which has no conflicts

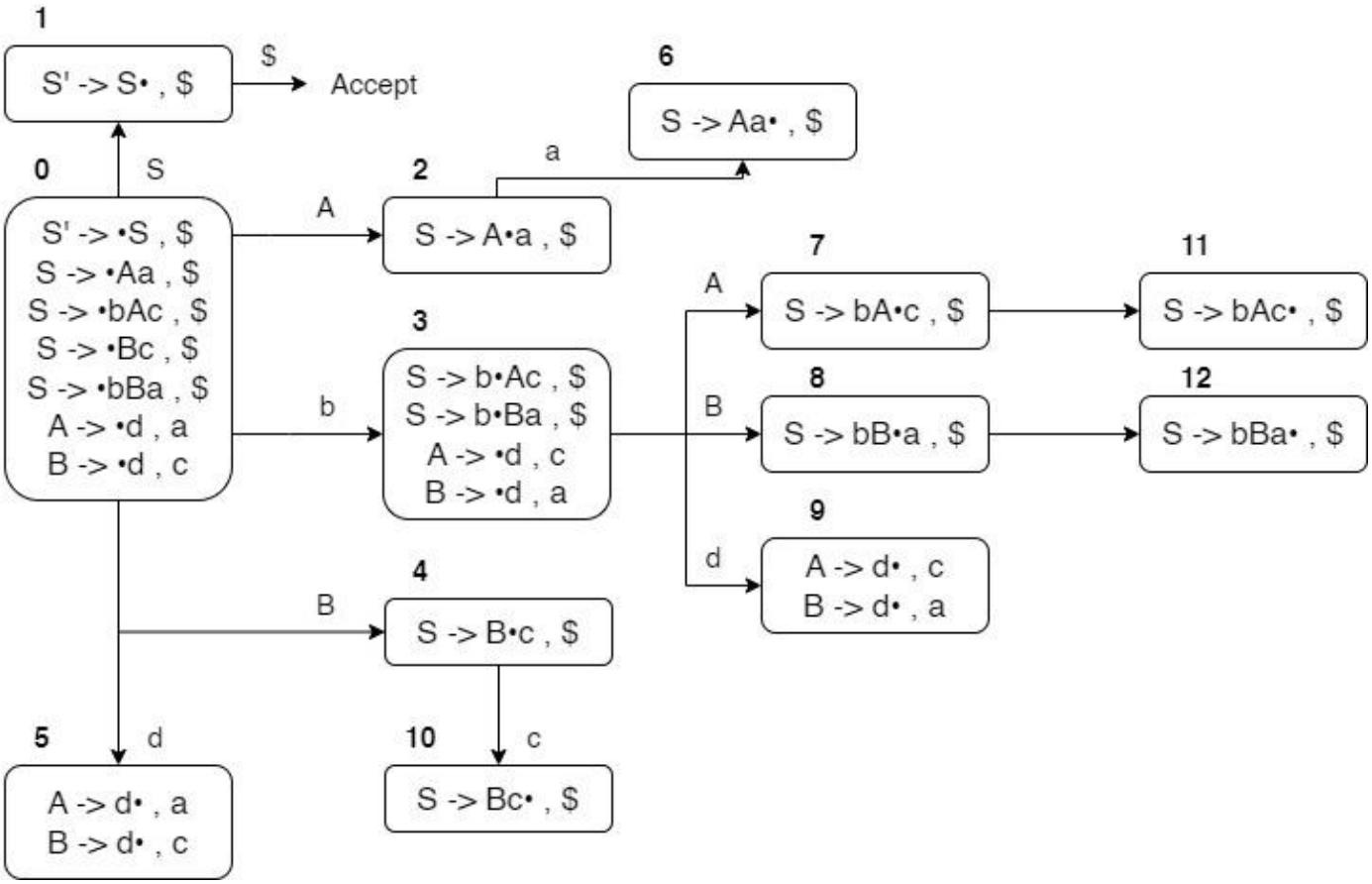
State	ACTION					GOTO			Productions
	a	b	c	d	$	S'	S	A	
0	S5	S3		S4			1	2	
1					acc				
2	S6								
3	S5			S8				7	**0: S'->S**
4	r_6		S9						**1: S->A a**
5	r_5		r_5						**2: S->b A c**
6					r_1				**3: S->d c**
7			S10						**4: S->b d a**
8	S11		r_6						**5: A->a**
9					r_3				**6: A->d**
10					r_2				
11					r_4				

But with a portion of the $SLR(1)$ state diagram and the FOLLOW of A & S equal to $\{b, d, a\}$, we can see that there's a shift-reduce conflict in $state\ 4$, which indicates that this grammar is not $SLR(1)$



ب)

$LR(1)$



همانطور که در تصویر مشخص است، هیچ حالتی دارای کانفلیکت نیست.

اما برای تجزیه $LALR(1)$، باید دو حالت 5 و 6 ادقام شوند که باعث ایجاد کانفلیکت reduce-reduce می‌شود.

الف)

$$terminals: \quad (\quad , \quad) \quad a \quad b \quad \$$$
$$non\ terminals: \quad S' \quad S \quad A$$
$$start\ symbol: S'$$

Starting with an empty set of production rules

$$S' \rightarrow$$
$$S \rightarrow$$
$$S \rightarrow$$
$$A \rightarrow$$
$$A \rightarrow$$

Based on $GOTO(0, S) = 1$ and accept in state 1, we can fill the first production rule

Based on $GOTO(0, A) = 3$, we know that one of the productions start with A and since there is only r2 with length 1 in state 3, the whole production is found

$$S' \rightarrow S$$
$$S \rightarrow$$
$$S \rightarrow A$$
$$A \rightarrow$$
$$A \rightarrow$$

Now, let's find three productions starting from state 0, then we'll assign the productions to non-terminals based on the length

Based on states $\{0, 6, 8, 9, 10\}$ and their GOTO, ACTION, shifts and reductions, we have

(A , S)

Based on states $\{0, 4, 7\}$, we have

a S

based on states $\{0, 5\}$, we have

b

which will give the final grammar

$$S' \rightarrow S$$
$$S \rightarrow (A , S)$$
$$S \rightarrow A$$
$$A \rightarrow a\ S$$
$$A \rightarrow b$$

ب)

Stack	Input	Action
0	(a a b , b) $	Shift 2
0 2	a a b , b) $	Shift 4
0 2 4	a b , b) $	Shift 4
0 2 4 4	b , b) $	Shift 5
0 2 4 4 5	, b) $	Reduce 4 **A->b**
0 2 4 4 3	, b) $	Reduce 2 **S->A**
0 2 4 4 7	, b) $	Reduce 3 **A->aS**
0 2 4 3	, b) $	Reduce 2 **S->A**
0 2 4 7	, b) $	Reduce 3 **A->aS**
0 2 6	, b) $	Shift 8
0 2 6 8	b) $	Shift 5
0 2 6 8 5	) $	Reduce 4 **A->b**
0 2 5 8 3	) $	Reduce 2 **S->A**
0 2 5 8 9	) $	Shift 10
0 2 5 8 9 10	$	Reduce 1 **S->(A,S)**
0 1	$	Accept