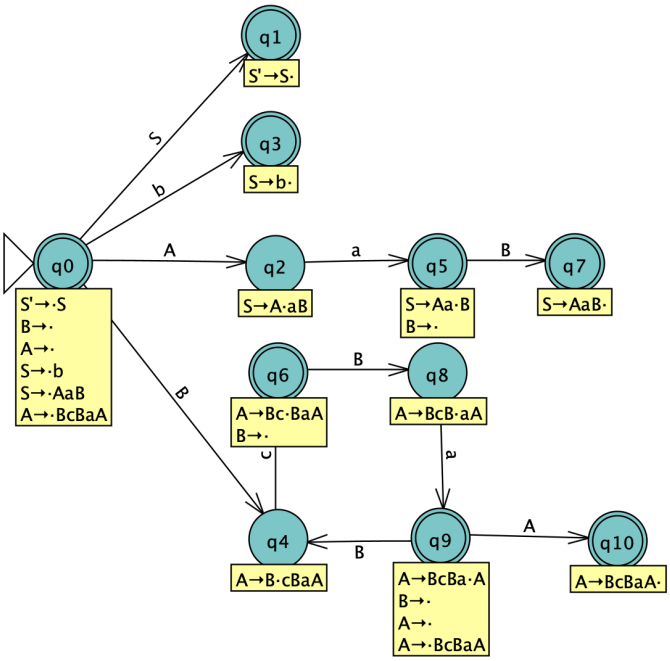
Esercizio 1

Gli items di $J[Aa]$ sono:

$$S \to Aa \cdot B$$

$$B \to \cdot$$

Per lo svolgimento guardare la soluzione al primo esercizio degli "Esercizi Tipo - Parte II". Riporto l'automa.



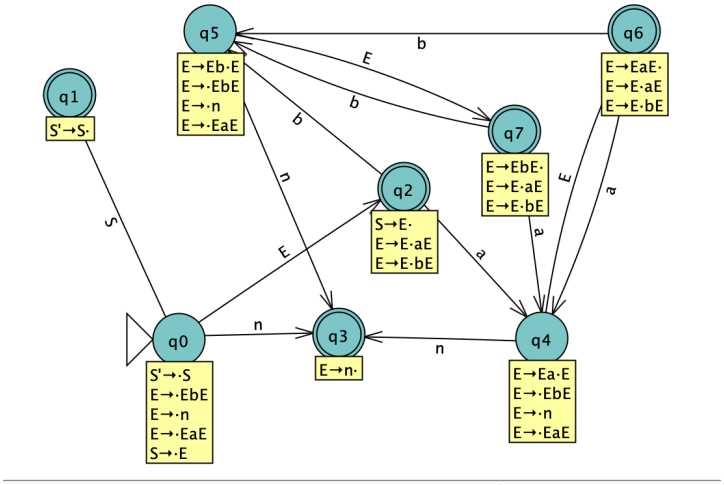
Esercizio 2

Seguendo le transazioni dallo stato iniziale dell'automa $\mathcal{A}$ si arriva allo stato 6 dell'automa, che non presenta alcun conflitto. Rimando sempre al primo esercizio delgli "Esercizi Tipo - Parte II" per lo svolgimento. Riporto la tabella SRL(1).

Stato	**a**	**b**	**c**	**$**	**A**	**B**	**S**
0	R4 / R5	S3	R5	R5	2	4	1
1				acc			
2	S5						
3				R4			
4			S6				
5	R5		R5	R5		7	
6	R5		R5	R5		8	
7				R1			
8	S9						
9	R4 / R5		R5	R5	10	4	
10	R3						

Esercizio 3

Automa caratteristico



First & Follow

	First	Follow
S	$\{n\}$	$\{\$\}$
E	$\{n\}$	$\{a, b, \$\}$

Tabella di parsing

Stato	a	b	n	$	E	S
0			S3		2	1
1				acc		
2	S4	S5		R1		
3	R2	R2		R2		
4			S3		6	
5			S3		7	
6	R3/~~S4~~	R3/~~S5~~		R3		
7	R4/~~S4~~	R4/~~S5~~		R4		

Parsing

Effettuiamo il parsing della parola *nbnan*:

Parola	State Stack	Symbol Stack	Azioni	Regole Semantiche
nbnan	0			
nbnan	03	*n*	S3	
nbnan	0 ~~3~~ 2	~~*n*~~ *E*	R2 G2	$E.v = n.lexval = 4$
nbnan	025	*Eb*	S5	
nbnan	0253	*Ebn*	S3	
nbnan	025 ~~3~~ 7	*Eb* ~~*n*~~ *E*	R2 G7	$E.v = n.lexval = 3$
nbnan	0 ~~257~~ 2	~~*EbE*~~ *E*	R4 G2	$E.v = E.v + E.v = 4 + 3 = 7$
nbnan	024	*Ea*	S4	
nbnan	0243	*Ean*	S3	
nbnan	024 ~~3~~ 6	*Ea* ~~*n*~~ *E*	R2 G6	$E.v = n.lexval = 3$
nbnan	0 ~~246~~ 2	~~*EaE*~~ *E*	R3 G6	$E.v = E.v + E.v = 7 \cdot 3 = 21$
nbnan	0 ~~2~~ 1	~~*E*~~ *S*	R1 G1	$S.v = E.v = 21$
nbnan	01	*S*	acc	

Risultato

La valutazione della parola $4b3a3$ secondo la SDD $\mathcal{S}$ è 21.

Esercizio 4

Funzioni ed attributi

- *node*: attributo sintetizzato che contiene informazioni sul tipo ('+', '*' o 'id'), eventuali riferimenti alla tabella dei simboli e i riferimenti ai figli del nodo, se presenti
- $newLeaf(label,\ value)$: funzione ausiliaria che va a creare un nodo foglia, quindi per questa grammatica con label 'id' e value un riferimento alla tabella dei simboli
- $newNode(label,\ child_1,\ child_2)$: funzione ausiliaria che va a creare un nuovo nodo con label l'operatore (+ o *) e il riferimento ai nodi figli

SDD

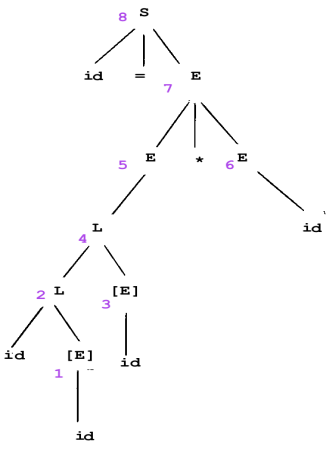
$E \to E_1 + E_2$	$\{E.node = newNode('+',\ E_1.node,\ E_2.node);\}$
$E \to E_1 * E_2$	$\{E.node = newNode('*',\ E_1.node,\ E_2.node);\}$
$E \to (E_1)$	$\{E.node = E_1.node;\}$
$E \to id$	$\{E.node = newLeaf(id,\ id.entry);\}$

Esercizio 5

Riporto per comodità la SDT:

$S \to id = E$	$\{gen(table.get(id) = E.addr);\}$
$S \to L = E$	$\{gen(L.array.base[L.addr] = E.addr);\}$
$E \to E_1 * E_2$	$\{E.addr = newTemp();\quad gen(E.addr = E_1.addr \cdot E_2.addr);\}$
$E \to E_1 + E_2$	$\{E.addr = newTemp();\quad gen(E.addr = E_1.addr + E_2.addr);\}$
$E \to L$	$\{E.addr = newTemp()\quad gen(E.addr = L.array.base[L.addr]);\}$
$E \to id$	$\{gen(E.addr = table.get(id));\}$
$L \to id[E]$	$\{L.array = table.get(id);\quad L.type = arg2(table.getType(id));$ $L.width = width(L.type);$ $L.addr = newTemp();\qquad gen(L.addr = E.addr * L.width);\}$
$L \to L_1[E]$	$\{L.array = L_1.array;\qquad L.type = arg2(L_1.type);$ $L.width = width(L.type);$ $t = newTemp();\qquad gen(t = E.addr \cdot L.width);$ $L.addr = newTemp();\qquad gen(L.addr = t + L_1.addr);\}$

A questo punto dobbiamo trovare l'albero di derivazione ed applicare le regole semantiche:

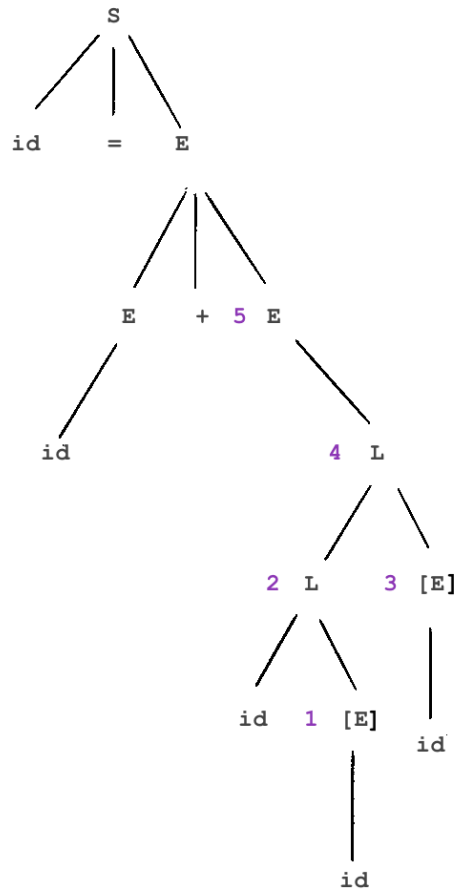


1. $E.addr = i$
2. $L.array = a$
   $L.type = array(3, integer)$
   $L.width = 12$ # [3 * 4]
   $L.addr = t_0$
   $t_0 = i \cdot 12$
3. $E.addr = j$
4. $L.array = a$
   $L.type = int$
   $L.width = 4$
   $t_1 = newTemp()$
   $t_1 = 4 \cdot j$
   $L.addr = t_2$
   $t_2 = t_0 + t_1$
5. $E.addr = t_3$
   $t_3 = a[t_2]$
6. $E.addr = c$
7. $E.addr = t_4$
   $t_4 = t_3 \cdot c$
8. $b = t_4$

Esercizio 6

Per risolvere l'esercizio possiamo utilizzare la stessa SDT dell'esercizio precedente. Per prima cosa costruiamo l'albero di derivazione per la stringa:

$$b = c + a[i][j]$$



La quinta riduzione è:

$$E \to L$$

Le sue regole semantiche sono:

- $E.addr = newTemp()$;
- $gen\big(E.addr = L.array.base[L.addr]\big)$;

La seconda riduzione genererà un nuovo temporaneo t_0, la quarta riduzione ne genererà altri due t_1 e t_2.
Fatte queste assunzioni possiamo dire che il codice generato per questa riduzione sarà:

$$t_3 = a[t_2]$$

Esercizio 7

Le regole semantiche per

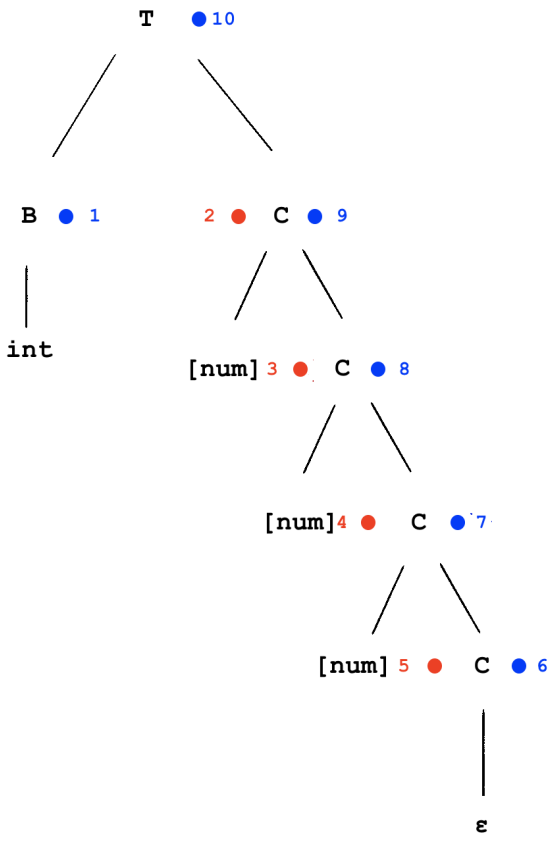
$$L \to L_1[E]$$

sono:

- $L.array = L_1.array$
- $L.type = arg2(L_1.type)$
- $L.width = width(L.type)$
- $t = newTemp()$
- $gen(t = L.width * E.addr)$
- $L.addr = newTemp()$
- $gen(L.addr = L_1.addr + t)$

Esercizio 8

Costruiamo l'albero di derivazione e diamo un ordine per la valutazione delle regole semantiche. Come nelle slide i nodi **blu** sono sintetizzati, mentre quelli **rossi** sono ereditatii.



1. $B.t = integer \quad \{B \to int\}$
2. $C.b = B.t \quad \{T \to BC\}$
3. $C_1.b = C.b \quad \{C \to [num]C_1\}$
4. $C_1.b = C.b \quad \{C \to [num]C_1\}$
5. $C_1.b = C.b \quad \{C \to [num]C_1\}$
6. $C.t = C.b \quad \{C \to \epsilon\}$
7. $C.t = array(num.lexval, C_1.t)$
   $\{C \to [num]C_1\}$
8. $C.t = array(num.lexval, C_1.t)$
   $\{C \to [num]C_1\}$
9. $C.t = array(num.lexval, C_1.t)$
   $\{C \to [num]C_1\}$
10. $T.t = C.t \quad \{T \to BC\}$