Maria Rita Di Berardini

Linguaggi di Programmazione e Compilatori

Raccolta Esercizi

10 giugno 2009

Università di Camerino
Dipartimenento di Matematica e Informatica

Indice

1

Analisi Lessicale

1.1 Espressioni Regolari

Esercizio 1.1. Scrivere un'espressione regolare per ciascuno dei seguenti linguaggi sull'alfabeto $\{0, 1\}$:

1. Tutte le stringhe
2. Tutte le stringhe eccetto la stringa vuota
3. Tutte le stringhe che iniziano e finiscono con 1
4. Tutte le stringhe che terminano con 00
5. Tutte le stringhe che contengono almeno tre 1
6. Tutte le stringhe il cui numero di 0 è multiplo di 3
7. Tutte le stringhe la cui lunghezza è almeno 1 e al più 3
8. Tutte le stringhe che iniziano e finiscono con lo stesso carattere
9. Tutte le stringhe di lunghezza pari (dispari)
10. Tutte le stringhe che iniziano con 1 ed hanno lunghezza pari
11. L'insieme delle stringhe binarie che rappresentano numeri dispari.
12. L'insieme delle stringhe binarie che rappresentano numeri divisibili per quattro

Soluzione 1.1

1. $(0|1)^*$
2. $(0|1)(0|1)^*$
3. $1(0|1)^*1$
4. $(0|1)^*00$
5. $(0|1)^*1(0|1)^*1(0|1)^*1(0|1)^*$
6. $1^*|(1^*01^*01^*01^*)^*$
7. $(0|1)(\epsilon|0|1)(\epsilon|0|1)$
8. $(0(0|1)^*0)|(1(0|1)^*1)$
9. stringhe di lunghezza pari: $((0|1)(0|1))^*$,
   stringhe di lunghezza dispari: $(0|1)((0|1)(0|1))^*$
10. $1(0|1)((0|1)(0|1))^*$

11. Sono le stringhe la cui ultima cifra è un 1 $(0|1)^*1$
12. Sono le stringhe la cui ultime due cifre sono 00: $(0|1)^*00$

Esercizio 1.2. Scrivere un'espressione regolare per ciascuno dei seguenti linguaggi sull'alfabeto $\Sigma = \{a, b\}$:

1. L'insieme delle stringhe che cominciamo con due a e finiscono con almeno due b
2. L'insieme delle stringhe in cui la prima a precede la prima b
3. Tutte le stringhe che contengono la sottostringa ab
4. Tutte le stringhe che non contengono la sottostringa ab
5. Tutte le stringhe contengono le sottostringhe aa o aba

Soluzione 1.2

1. $aa(a|b)^*bbb^*$
2. $ab(a|b)^*$
3. $(a|b)^*ab(a|b)^*$
4. b^*aa^*
5. $(a|b)^*a(b|\epsilon)a(a|b)^*$

Esercizio 1.3. 1. Fornire tutte le stringhe di lunghezza 4 denotate dall'espressione regolare $a(b|c)^*d$;
2. Fornire tutte le stringhe di lunghezza ≥ 1 e ≤ 4 denotate dall'espressione regolare $a|a^*b$.

Esercizio 1.4. Descrivere (a parole) il linguaggio denotato da ciascuna delle seguenti espressioni regolari:

1. $a(a|b)^*a$
2. $((\epsilon|a)b^*)^*$
3. $(a|b)(\epsilon|a|b)^*bb$
4. $(a|b)^*a(a|b)(a|b)$

Esercizio 1.5. Quale delle seguenti stringhe appartiene al linguaggio denotato dall'espressione regolare $(1|01^*0)^*$:

1. 101100
2. 1001
3. 0000

Esercizio 1.6. Fornire un'espressione regolare che definisce (rispettivamente) l'intersezione e la concatenzione dei due seguenti linguaggi: $A = \{w \in \{0,1\}^* | w \text{ inizia con } 11\}$ e $B = \{w \in \{0,1\}^* | w \text{ inizia con 1 termina con } 0\}$.

Esercizio 1.7. Sia r un espressione regolare su un data alfabeto Σ. Dire se e quando le espressioni regolari $r\epsilon$ ed $r|\epsilon$ denotano lo stesso linguaggio.

1.2 Automi a Stati Finiti

Esercizio 1.8. Disegnare il DFA equivalente alle seguenti espressioni regolari:

(a) $ab(a|b)^*$ (b) $(ab)^*$ (c) a^*b^*

Esercizio 1.9. Fornire un DFA per ognuno dei seguenti linguaggi:

1. Tutte le stringhe sull'alfabeto $\{a, b\}$ che contengono un numero pari di a
2. Tutte le stringhe sull'alfabeto $\{a, b\}$ che contengono un numero pari di a e un numero dispari di b

Esercizio 1.10. Fornire un NFA per ognuno dei seguenti linguaggi:

1. Tutte le stringhe sull'alfabeto $\{a, b\}$ che hanno almeno una a negli ultimi 4 caratteri.
2. Tutte le stringhe sull'alfabeto $\{E, G, L, O, X\}$ che contengono l'espressione regolare $GO * GLE$.

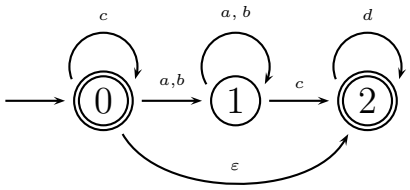
Esercizio 1.11. Fornire un DFA minimo equivalente al NFA dell'Esercizio 1.10-1 e 1.10-2.

Esercizio 1.12. Covertire le seguenti espressioni regolari in automi a stati finiti deterministici minimi:

1. $1^*(01)^*0^*$
2. $0^*(1^*00)^*1^*$
3. $(0^*|1^*)(0^*|1^*)(0^*|1^*)$
4. $1^*(0|10)^*1^*$

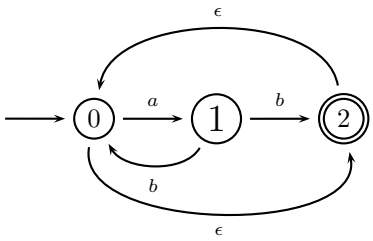
1.3 Miscellanea

Esercizio 1.13. Si condideri il NFA in figura:



1. Si esprima il linguaggio accettato mediante espressione regolare;
2. Si costruisca, uilizzando la costruzione dei sottoinsiemi, un automata deterministico D equivalente;
3. L'automa D del punto precedente é minimo? Si giustifichi formalmente la risposta.

Esercizio 1.14. Si consideri il seguente automa non deterministico



1. Si esprima il linguaggio accettato mediante espressione regolare.
2. Costruire un automa deterministico equivalente a quello dato mediante costruzione dei sottoinsiemi.
3. L'automa deterministico ottenuto é minimo? Si giustifichi formalmente la risposta.

2

Analisi Sintattica

2.1 LL Parsing

Esercizio 2.1. Si consideri la seguente grammatica libera da contesto G:

$$S \to aSc \mid aBc$$
$$B \to bBc \mid @$$

1. Si indichi il linguaggio generato mediante espressione su insiemi;
2. G è una grammatica LL(1)? Se no, fornire una grammatica G' in grado di generare il medesimo linguaggio di G ma LL(1).
3. Esiste un $k \geq 2$ tale che G è LL(k)? Motivare formalmente la risposta

Soluzione 2.1

1. $L(G) = \{a^n b^m @ c^{m+n} \mid m \geq 0, n \geq 1\}$
2. G non è LL(1); infatti, $\mathsf{FIRST}(aSc) = \mathsf{FIRST}(aBc) = \{a\}$ implica $M[S, a] = S \to aSc, S \to aBc$. Possiamo ottenere G' semplicemente fattorizzando a sinistra le produzioni per S ottenendo:

$$S \to aS'$$
$$S' \to Sc \mid Bc$$
$$B \to bBc \mid @$$

	FIRST	FOLLOW
S	$\{a\}$	$\{\$, c\}$
S'	$\{a, b\}$	$\{\$, c\}$
B	$\{b, @\}$	$\{c\}$

	a	b	c	$@$	$\$$
S	$S \to aS'$				
S'	$S' \to Sc$	$S' \to Bc$		$S' \to Bc$	
B		$B \to bBc$		$B \to @$	

Poichè la tabella di parsing non ha entrate multidefinite possiamo concludere che G' è LL(1).

3. G è LL(2). Per dimostrare questa cosa cerchiamo di capire il motivo per cui G non è LL(1). Per quanto visto al punto 1, il problema principale è che non sappiamo come derivare il non terminale S quando in input leggiamo il simbolo a. Infatti, $M[S, a] = S \to aSc, S \to aBc$ ci dice che potremmo usare entrambe le produzioni per S. Possiamo, però, decidere quale delle due scelte è corretta osservando il simbolo che segue immediatamente la a. Se tale simbolo è un'altra a, allora la produzione corretta è $S \to aSc$; se al contrario tale simbolo è una b oppure @ allora dobbiamo ridurre S usando la produzione $S \to aBc$.

Esercizio 2.2. Si verifichi se il seguente linguaggio è LL(k) per qualche k

$$L = \{a^n bc^k \mid n \geq 0, k > 0\} \cup \{b^n ac^k \mid n > 0, k \geq 0\}$$

Soluzione 2.2

Una grammatica in grado di generare il linguaggio L è:

$$\begin{aligned} S &\to A \mid bB \\ A &\to aA \mid bcC \\ B &\to bB \mid aC \\ C &\to cC \mid \epsilon \end{aligned}$$

Tale grammatica non è però LL(1). Infatti, $b \in \mathsf{FIRST}(A) = \{a, b\}$ e $b \in \mathsf{FIRST}(bB)$, implica $M[S, b] = S \to A, S \to bB$. La tabella di parsing completa è descritta in Figura 2.1.

	a	b	c	\$
S	$S \to A$	$S \to A$ $S \to bB$		
A	$A \to aA$	$A \to bcC$		
B	$B \to aC$	$B \to bB$		
C			$C \to cC$	$C \to \epsilon$

Figura 2.1. tabella di parsing per la grammatica G

In questo caso il problema è come ridurre il non terminale S quando in input leggiamo il simbolo b. In realtà possiamo facilmente disambiguare questa scelta, osservando che se tale b è seguita da una c (il che significa che la stringa in input è della forma bc^k per qualche $k > 0$) allora S va riscritto mediante la produzione $S \to A$; se invece la b è seguita da un'altra b oppure da una a (ossia la stringa in input è della forma $b^n a$ per qualche $n > 0$) allora S va riscritto con la produzione $S \to bB$. Questo dimostra che la grammatica G (e quindi il linguaggio L) è LL(2). È tuttavia possibile scrivere una grammatica

G' LL(1) tale che $L(G') = L$. Si consideri ad esempio la seguente grammatica G':

$$
\begin{aligned}
&S \to aA \,|\, bB \\
&A \to aA \,|\, bcC \\
&B \to cC \,|\, B' \\
&B' \to bB' \,|\, aC \\
&C \to cC \,|\, \epsilon
\end{aligned}
$$

Abbiamo che:

- $L(C) = \{c^k \,|\, k \geq 0\}$,
- $L(B') = \{b^n a c^k \,|\, n, k \geq 0\}$,
- $L(B) = cL(C) \cup L(B') = \{c^k \,|\, k > 0\} \cup \{b^n a c^k \,|\, n, k \geq 0\}$,
- $L(A) = \{a^n b c^k \,|\, n \geq 0, k > 0\}$,
- $L(S) = aL(A) \cup bL(B) =$
  $\{a^n b c^k \,|\, n, k > 0\} \cup \{bc^k \,|\, k > 0\} \cup \{b^n a c^k \,|\, n > 0, k \geq 0\} =$
  $\{a^n b c^k \,|\, n \geq 0, k > 0\} \cup \{b^n a c^k \,|\, n > 0, k \geq 0\}$,

Dimostriamo ora che G' è LL(1)

	FIRST	FOLLOW
S	$\{b, a\}$	$\{\$\}$
A	$\{b, a\}$	$= \mathsf{FOLLOW}(S) = \{\$\}$
B	$\{a, b, c\}$	$= \mathsf{FOLLOW}(S) = \{\$\}$
B'	$\{b, a\}$	$= \mathsf{FOLLOW}(B) = \{\$\}$
C	$\{c, \epsilon\}$	$\{\$\}$

	a	b	c	\$
S	$S \to aA$	$S \to bB$		
A	$A \to aA$	$A \to bcC$		
B	$B \to B'$	$B \to B'$	$B \to cC$	
B'	$B' \to aC$	$B' \to bB'$		
C			$C \to cC$	$C \to \epsilon$

Figura 2.2. tabella di parsing per la grammatica G'

Simuliamo ora il comportamento di un parser LL durante il parsing della stringa $w = aabc$

input	stack	azione
$\$S$	$aabc\$$	reduce $S \to aA$
$\$Aa$	$aabc\$$	
$\$A$	$abc\$$	reduce $A \to aA$
$\$Aa$	$abc\$$	
$\$A$	$bc\$$	reduce $A \to bcC$
$\$Ccb$	$bc\$$	
$\$Cc$	$c\$$	
$\$C$	$\$$	reduce $C \to \epsilon$
$\$$	$\$$	accept

Esercizio 2.3. Si consideri la seguente grammatica libera da contesto G:

$$E \to E + T \mid T$$
$$T \to TF \mid F$$
$$F \to F^* \mid a \mid b$$

1. Dire se G è LL(1). Se no, fornire una grammatica G' tale che $L(G') = L(G)$ e G' è LL(1).
2. Si simuli il comportamento di un parser LL per G' in corrispondenza della stringa $a^* + b + a$.

Esercizio 2.4. Si consideri la seguente grammatica libera da contesto:

$$S \to \mathbf{s} \mid \mathbf{w}(B)S$$
$$B \to B \vee B \mid \mathbf{t} \mid \mathbf{f}$$

1. Si dimostri che G è ambigua. Suggerimento: mostrare due alberi di derivazioni distinti per la stringa $\mathbf{w}(\mathbf{t} \vee \mathbf{t} \vee \mathbf{f})\mathbf{s}$.
2. Fornire una grammatica G' non ambigua e LL(1) tale che $L(G') = L(G)$. Suggerimento: modificare le produzioni di G in modo che l'operatore $\vee$ sia associativo a sinistra.

Esercizio 2.5. La seguente grammatica è LL(k) per qualche k? Se sì, trovate il più piccolo k tale che la grammatica è LL(k).

$$S \to AC \mid aaB$$
$$A \to aA \mid \epsilon$$
$$B \to bB \mid d$$
$$C \to b \mid cC$$

2.2 LR Parsing

Esercizio 2.6. Si consideri la seguente grammatica libera da contesto G:

$$S \to aAa \quad A \to bSc \quad B \to aBc$$
$$S \to Ba \quad A \to \epsilon \quad B \to Sa$$

1. Si dimostri che G è SLR(1) ma non LL(1)
2. Simulare il comportamento del parser LR in corrispondenza della stringa *abacaca*.
3. Sia $C' = J_0, \cdots, J_n$ la collezione canonica di items LR(1) per G. Dire se C' può produrre dei conflitti shift/reduce o reduce/reduce. Giustificare formalmente la risposta.

Esercizio 2.7. Si consideri la seguente grammatica libera da contesto G:

$$\begin{array}{lll} S \to aAbB & A \to cA & B \to aBc \\ & A \to b & B \to ca \end{array}$$

1. Si costruisca la collezione canonica C di items LR(1) per G e si dimostri che G è LALR.
2. Simulare il comportamento del parser LR in corrispondenza della stringa *abacac*.
3. Si dimostri se $L = \{ab^n a \,|\, n \geq 0\} \cup \{ab^n A \,|\, n \geq 1\}$ contiene solo viable prefixes per G.

Esercizio 2.8. Si consideri la seguente grammatica:

$$\begin{array}{l} S \to B \,|\, Caa \\ B \to bC \\ C \to bbCa \,|\, \epsilon \end{array}$$

Si indichi il linguaggio generato mediante espressioni su insiemi. La grammatica è LR(1)? Se sì, si dia la tabella di un parser shift-reduce e si mostri il parsing della stringa *bbba*.

Esercizio 2.9. Si dimostri che la seguente grammatica libera da contesto è SLR(1) ma non LL(1).

$$\begin{array}{lll} S \to bA & B \to bB & B \to a \\ A \to bB & B \to aS & \end{array}$$

Soluzione 2.9

- Poichè $\mathsf{FIRST}(aS) = \mathsf{FIRST}(a) = \{a\}$ la tabella di parsing LL ha un entrata multidefinita in corrispondenza del non terminale S e del simbolo a: $M[S, a] := B \to aS, B \to a$. Questo dimostra che la grammatica in input non è LL(1).
  Homework: dimostrare che la grammatica è LL(2) tenendo presente che il linguaggio generato è $L(S) = \{(b^{n+2}a)^m \,|\, n \geq 0, m > 0\}$.
- Collezione canonica di items LR(0):

$$I_0 = closure(\{S' \to \bullet S\}) : \begin{array}{l} S' \to \bullet S \\ S \to \bullet bA \end{array}$$

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

$$goto(I_0, b) = I_2 : S \to b \bullet A$$
$$A \to \bullet bB$$

$$goto(I_2, A) = I_3 : S \to bA\bullet$$

$$goto(I_2, b) = I_4 : A \to b \bullet B$$
$$B \to \bullet bB$$
$$B \to \bullet aS$$
$$B \to \bullet a$$

$$goto(I_4, B) = I_5 : A \to bB\bullet$$

$$goto(I_4, b) = I_6 : B \to b \bullet B$$
$$B \to \bullet bB$$
$$B \to \bullet aS$$
$$B \to \bullet a$$

$$goto(I_4, a) = I_7 : B \to a \bullet S$$
$$B \to a\bullet$$
$$S \to \bullet bA$$

$$goto(I_6, B) = I_8 : B \to bB\bullet$$

$$goto(I_6, b) = I_6$$

$$goto(I_6, a) = I_7$$

$$goto(I_7, S) = I_9 : B \to aS\bullet$$

$$goto(I_7, b) = I_2$$

- $\mathsf{FOLLOW}(S) = \mathsf{FOLLOW}(A) = \mathsf{FOLLOW}(B) = \{\$\}$. A questo punto possiamo costruire la tabella di parsing SLR(1):

	a	b	\$	S	A	B
s_0		S2		1		
s_1			acc			
s_2		S4			3	
s_3			R1			
s_4	S7	S6	\$			5
s_5			R2	S	A	B
s_6	S7	S6				8
s_7		S2	R5	9		
s_8			R3			
s_9			R4			

Esercizio 2.10. Si dimostri che la seguente grammatica libera da contesto è LL(1) ma non SLR(1).

$$\begin{array}{ll} S \to AaAb & A \to \epsilon \\ S \to BbBa & B \to \epsilon \end{array}$$

Soluzione 2.10 Calcoliamo innanzitutto la FIRST e la FOLLOW per i non terminali della grammatica:

	FIRST	FOLLOW
S	$\{a, b\}$	$\{\$\}$
A	$\{\epsilon\}$	$\{a, b\}$
B	$\{\epsilon\}$	$\{a, b\}$

Allora la tabella LL per la grammatica in input è:

	b	b	$\$$
S	$S \to AaAb$	$S \to BbBa$	
A	$A \to \epsilon$	$A \to \epsilon$	
B	$B \to \epsilon$	$B \to \epsilon$	

Dimostriamo, ora, che G non è SLR(1). Iniziamo con il determinare

$$\begin{array}{rl} I_0 = closure(\{S' \to \bullet S\}) : & S' \to \bullet S \\ & S \to \bullet AaAb \\ & S \to \bullet BbBa \\ & A \to \bullet \\ & B \to \bullet \end{array}$$

Allora, $A \to \bullet, B \to \bullet \in I_0$ e $\mathsf{FOLLOW}(A) = \mathsf{FOLLOW}(B) = \{a, b\}$ implica $M[s_0, a] = M[s_0, b] =$ reduce $A \to \epsilon$/reduce $B \to \epsilon$. La tabella SLR(1) ha un conflitto di tipo reduce/reduce, il che significa che la grammatica non è SLR(1).

Possiamo tuttavia dimostrare che la grammatica in input è LR(1) (in realtà, anche LALR(1)). Osservate come l'insieme iniziale della collezione canonica di items LR(1) è

$$\begin{array}{rl} I_0 = closure(\{[S' \to \bullet S, \$]\}) : & [S' \to \bullet S, \$] \\ & [S \to \bullet AaAb, \$] \\ & [S \to \bullet BbBa, \$] \\ & [A \to \bullet, a] \\ & [B \to \bullet, b] \end{array}$$

In questo caso $[A \to \bullet, a] \in I_0$ significa $M[s_0, a] =$ reduce $A \to \epsilon$, mentre $[B \to \bullet, b] \in I_0$ significa $M[s_0, b] =$ reduce $B \to \epsilon$. Non abbiamo più, quindi, il conflitto reduce/reduce della collezione canonica di items LR(0).

Homework: terminare la costruzione della collezione canonica di items LR(1) e dimostrare che la grammatica è LR(1) ma anche LALR(1).

Esercizio 2.11. Si dimostri che la seguente grammatica è LALR(1) ma non SLR(1).

$$\begin{array}{llll} S \to S;T & T \to V = E & V \to id & E \to V \\ S \to T & T \to id & & E \to num \end{array}$$

Soluzione 2.11 Calcoliamo innanzitutto la FIRST e la FOLLOW per i non terminali della grammatica:

	FIRST	FOLLOW
S	$\{id\}$	$\{;, \$\}$
T	$\{id\}$	$\{;, \$\}$
V	$\{id\}$	$\{=, ;, \$\}$
E	$\{id, num\}$	$\{;, \$\}$

Dimostriamo, innanzitutto, che G non è SLR(1). Iniziamo con il determinare

$$\begin{array}{rl} I_0 = closure(\{S' \to \bullet S\}) : & S' \to \bullet S \\ & S \to \bullet S;T \\ & S \to \bullet T \\ & T \to \bullet V = E \\ & T \to \bullet id \\ & V \to \bullet id \end{array}$$

Ora

$$\begin{array}{rl} I_1 = goto(I_0, id) : & T \to id\bullet \\ & V \to id\bullet \end{array}$$

e poichè $;, \$ \in$ FOLLOW(T) e $;, \$ \in$ FOLLOW(V) abbiamo il seguente conflitto reduce/reduce nella parte action della tabella di parsing: $M[s_1, ;] = M[s_1, \$] =$ reduce $T \to id$/reduce $V \to id$. Costruiamo ora la collezione canonica di items LR(1).

$$\begin{array}{rl} I_0 = closure(\{[S' \to \bullet S, \$]\}) : & [S' \to \bullet S, \$] \\ & [S \to \bullet S;T, \$] \\ & [S \to \bullet T, \$] \\ & [S \to \bullet S;T, ;] \\ & [S \to \bullet T, ;] \\ & [T \to \bullet V = E, ;/\$] \\ & [T \to \bullet id, ;/\$] \\ & [V \to \bullet id, =] \end{array}$$

In forma contratta:

$$\begin{array}{rl} I_0 : & [S' \to \bullet S, \$] \\ & [S \to \bullet S;T, ;/\$] \\ & [S \to \bullet T, ;/\$] \\ & [T \to \bullet V = E, ;/\$] \\ & [T \to \bullet id, ;/\$] \\ & [V \to \bullet id, =] \end{array}$$

$$go(I_0, S) = I_1 : [S' \to S\bullet, \$] \\ [S \to S\bullet; T, ; /\$]$$

$$goto(I_0, T) = I_2 : [S \to T\bullet, ; /\$]$$

$$goto(I_0, V) = I_3 : [T \to V\bullet = E, ; /\$]$$

$$goto(I_0, id) = I_4 : [S' \to \bullet S, \$] \\ [T \to id\bullet, ; /\$] \\ [V \to id\bullet, =]$$

$$goto(I_1, ;) = I_5 : [S \to S; \bullet T, ; /\$] \\ [T \to \bullet V = E, ; /\$] \\ [T \to \bullet id, ; /\$] \\ [V \to \bullet id, =]$$

$$goto(I_3, =) = I_6 : [T \to V = \bullet E, ; /\$] \\ [E \to \bullet V, ; /\$] \\ [E \to \bullet num, ; /\$] \\ [V \to \bullet id, ; /\$]$$

$$goto(I_5, T) = I_7 : [S \to S; T\bullet, ; /\$]$$

$$goto(I_5, V) = I_3$$

$$goto(I_5, id) = I_4$$

$$goto(I_6, E) = I_8 : [T \to V = E\bullet, ; /\$]$$

$$goto(I_6, V) = I_9 : [E \to V\bullet, ; /\$]$$

$$goto(I_6, num) = I_{10} : [E \to num\bullet, ; /\$]$$

$$goto(I_6, id) = I_{11} : [V \to id\bullet, ; /\$]$$

Costruiamo ora la tabella LR(1) per la grammatica in input. Osservate come, poichè non esistono due insiemi della collezione canonica con core comune, tale tabella coincide con la tabella LALR(1).

	id	num	$=$	;	\$	S	T	E	V
s_0	S4					1	2		3
s_1				S5	acc				
s_2				R2	R2				
s_3			S6						
s_4			R5	R4	R4				
s_5		S4					7		3
s_6	S11	S10						8	9
s_7				R1	R1				
s_8				R3	R3				
s_9				R6	R6				
s_{10}				R7	R7				
s_{11}				R5	R5				

Esercizio 2.12. Si dimostri che la seguente grammatica è LR(1) ma non LALR(1).

$$\begin{array}{llll} S \to cS & A \to Ba & B \to d & C \to d \\ S \to A & A \to bBc & & \\ & A \to Cc & & \\ & A \to bCa & & \end{array}$$

Soluzione 2.12

$$\begin{array}{rl} I_0 = closure(\{[S' \to \bullet S, \$]\}) : & [S' \to \bullet S, \$] \\ & [S \to \bullet cS, \$] \\ & [S \to \bullet A, \$] \\ & [A \to \bullet Ba, \$] \\ & [A \to \bullet bBc, \$] \\ & [A \to \bullet Cc, \$] \\ & [A \to \bullet bCa, \$] \\ & [B \to \bullet d, a] \\ & [C \to \bullet d, c] \end{array}$$

$$goto(I_0, S) = I_1 : [S' \to S\bullet, \$]$$

$$\begin{array}{rl} goto(I_0, c) = I_2 : & [S \to c \bullet S, \$] \\ & [S \to \bullet cS, \$] \\ & [S \to \bullet A, \$] \\ & [A \to \bullet Ba, \$] \\ & [A \to \bullet bBc, \$] \\ & [A \to \bullet Cc, \$] \\ & [A \to \bullet bCa, \$] \\ & [B \to \bullet d, a] \\ & [C \to \bullet d, c] \end{array}$$

$goto(I_0, A) = I_3 : [S \rightarrow A\bullet, \$]$

$goto(I_0, B) = I_4 : [A \rightarrow B \bullet a, \$]$

$goto(I_0, b) = I_5 : [A \rightarrow b \bullet Bc, \$]$
$[A \rightarrow b \bullet Ca, \$]$
$[B \rightarrow \bullet d, c]$
$[C \rightarrow \bullet d, a]$

$goto(I_0, C) = I_6 : [A \rightarrow C \bullet c, \$]$

$goto(I_0, d) = I_7 : [B \rightarrow d\bullet, a]$
$[C \rightarrow d\bullet, c]$

$goto(I_2, S) = I_8 : [S \rightarrow cS\bullet, \$]$

$goto(I_2, c) = I_2$

$goto(I_2, A) = I_3$

$goto(I_2, B) = I_4$

$goto(I_2, b) = I_5$

$goto(I_2, C) = I_6$

$goto(I_2, d) = I_7$

$goto(I_4, a) = I_9 : [A \rightarrow Ba\bullet, \$]$

$goto(I_5, B) = I_{10} : [A \rightarrow bB \bullet c, \$]$

$goto(I_5, C) = I_{11} : [A \rightarrow bC \bullet a, \$]$

$goto(I_5, d) = I_{12} : [B \rightarrow d\bullet, c]$
$[C \rightarrow d\bullet, a]$

$goto(I_6, c) = I_{13} : [A \rightarrow Cc\bullet, \$]$

$goto(I_{10}, c) = I_{14} : [A \rightarrow bBc\bullet, \$]$

$goto(I_{11}, a) = I_{15} : [A \rightarrow bCa\bullet, \$]$

Gli unici items con core comune sono I_7 e I_{12} che accorpati producono l'insieme di items:

$$I_{712} : [B \rightarrow d\bullet, a/c]$$
$$[C \rightarrow d\bullet, a/c]$$

che da origine ad un conflitto reduce/reduce della forma: $M[s_{712}, a] = M[s_{712}, c] =$ reduce $B \rightarrow d$/reduce $C \rightarrow d$. Questo dimostra che la grammatica non è LALR(1). Di seguito, invece, riportiamo la tabella LR canonica per la grammatica in input. L'assenza di conflitti prova che tale grammatica è LR canonica – o LR(1).

	a	b	c	d	\$	S	A	B	C
s_0		S5	S2	S7		1	3	4	6
s_1					acc				
s_2		S5	S2	S7		8	3	4	6
s_3					R2				
s_4	S9								
s_5				S12				10	11
s_6			S13						
s_7	R7	R8							
s_8					R1				
s_9					R3				
s_{10}			S14						
s_{11}	S15								
s_{12}	R8	R7							
s_{13}					R5				
s_{14}					R4				
s_{15}					R6				

3

Automi a pila

3.1 Automi a pila

Esercizio 3.1. Fornire un automa a pila in grado di riconoscere i seguenti linguaggi:

- $L_1 = \{a^n b^m a^n \mid n, m \geq 0\}$
- $L_2 = \{a^n b^m c^m \mid n, m \geq 0\}$
- $L_3 = \{a^n b^m \mid n \geq 0, m > n\}$
- $L_4 = \{a^n b^m c^k \mid n, k \geq 0, m > n\}$

Soluzione 3.1

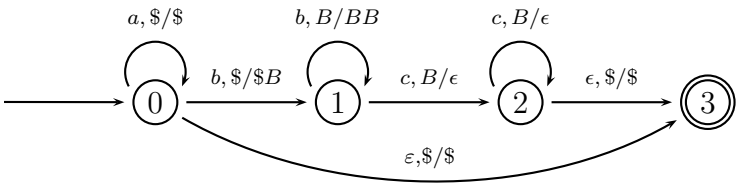


Figura 3.1. PDA per L_2

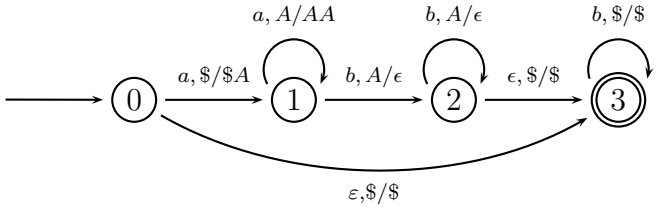


Figura 3.2. PDA per L_3

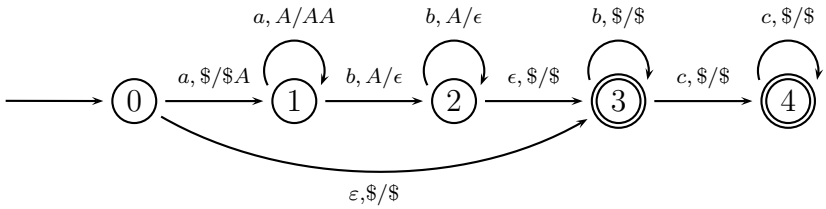


Figura 3.3. PDA per L_4