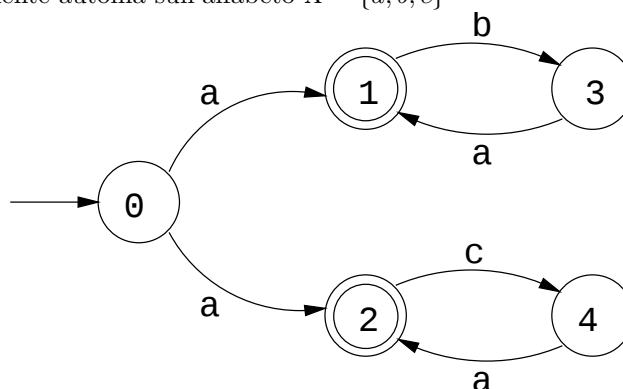


PROGRAMMAZIONE – I Appello del 3/12/2002

SOLUZIONE

ESERCIZIO 1 (6 punti)

Dato il seguente automa sull'alfabeto $\Lambda = \{a, b, c\}$

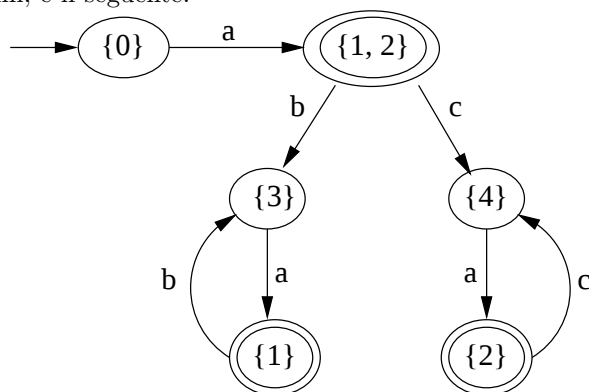


- Descrivere formalmente il linguaggio riconosciuto dall'automa.
- Trasformare l'automa dato in un automa equivalente deterministico usando l'algoritmo della costruzione dei sottoinsiemi. Spiegare almeno un passo dell'algoritmo.

Soluzione

Il linguaggio accettato è l'insieme $\{a(ba)^n \mid n \geq 0\} \cup \{a(ca)^n \mid n \geq 0\}$.

L'automa deterministico equivalente, costruito con l'algoritmo della costruzione dei sottoinsiemi, è il seguente:



Lo stato iniziale è, per definizione, $\{0\}$. Per costruire le transizioni uscenti da questo stato si calcola $T_{\{0\}}^a = \{1, 2\}$ cioè l'insieme degli stati che si raggiungono, nell'automa di partenza, a partire da uno stato qualsiasi dell'insieme $\{0\}$ e leggendo il simbolo a . Si costruisce allora una transizione dallo stato $\{0\}$ allo

stato $\{1, 2\}$ dell'automa risultato, etichettata con il simbolo a . Si calcola poi lo stesso insieme per gli altri simboli dell'alfabeto Λ . In questo caso $T_{\{0\}}^b = T_{\{0\}}^c = \{\}$. Quindi non ci sono transizioni uscenti dallo stato $\{0\}$ etichettate con b o con c , dato che l'automa di partenza, nello stato 0, non ha transizioni uscenti etichettate con questi simboli.

ESERCIZIO 2 (6 punti)

Siano date le seguenti categorie sintattiche con relative produzioni:

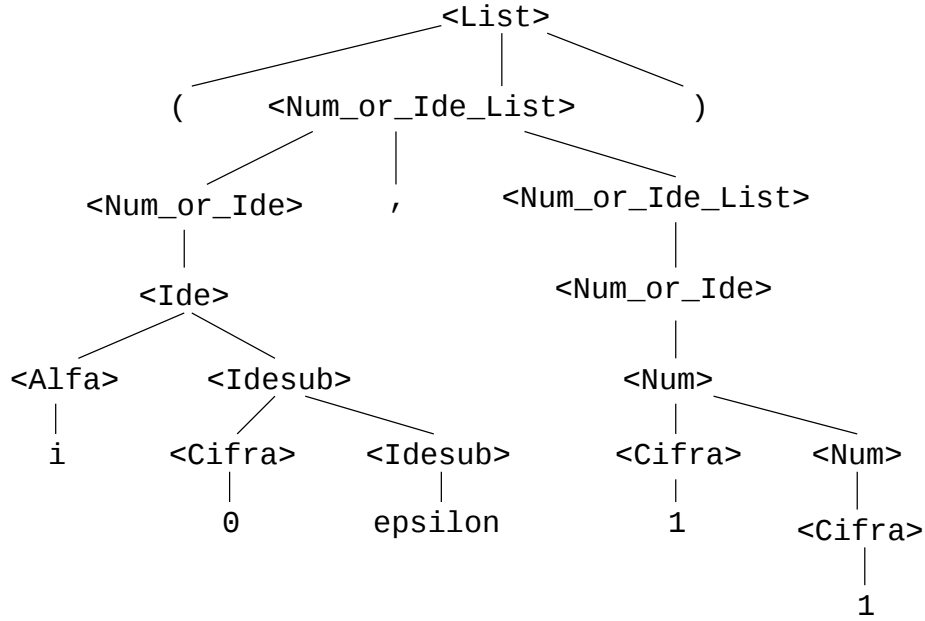
```
<Cifra> ::= 0|1|...|9
<Num> ::= <Cifra> | <Cifra> <Num>
<Alfa> ::= a|b|c|...|z|A|B|C|...|Z
<Ide> ::= <Alfa> <Idesub>
<Idesub> ::= <Alfa> <Idesub> | <Cifra> <Idesub> | epsilon
```

- Usando le categorie sintattiche date come sottocategorie, dare le produzioni di una categoria sintattica $\langle \text{List} \rangle$ (ed eventuali sottocategorie di comodo) che genera le liste di numeri (elementi della categoria sintattica $\langle \text{Num} \rangle$) e/o di identificatori (elementi della categoria sintattica $\langle \text{Ide} \rangle$). Ogni lista si apre con una parentesi tonda aperta, gli elementi sono separati da virgole, può contenere un numero qualsiasi di elementi (anche zero) e si chiude con una parentesi tonda chiusa. La lista vuota è la stringa $\langle () \rangle$. Altri esempi sono $\langle (1) \rangle$, $\langle (23, a) \rangle$ e $\langle (12, a34, pippo, 0) \rangle$.
- Disegnare l'albero di derivazione della lista $\langle (i0, 11) \rangle$ usando la grammatica che si è data al punto (a).

Una possibile soluzione

```
<List> ::= () | ( <Num_or_Ide_List> )
<Num_or_Ide_List> ::= <Num_or_Ide> | <Num_or_Ide> , <Num_or_Ide_List>
<Num_or_Ide> ::= Num | Ide
```

L'albero di derivazione è il seguente.



ESERCIZIO 3 (6 punti)

Si supponga di estendere la categoria sintattica dei comandi con la seguente produzione

$\langle \text{Com} \rangle ::= \text{ifzerorepeat } (\langle \text{Exp} \rangle, \langle \text{Com} \rangle, \langle \text{Com} \rangle) \text{ until } \langle \text{Exp} \rangle;$

Un comando **ifzerorepeat** (E_1 , C_1 , C_2) **until** E è un ciclo in cui il corpo viene eseguito in ogni caso alla prima iterazione. Se la guardia E ha valore **falso** nello stato risultante dall'esecuzione del corpo allora il ciclo viene ripetuto, altrimenti l'esecuzione del comando termina. L'esecuzione del corpo (E_1 , C_1 , C_2) consiste nell'esecuzione del comando C_1 se l'espressione E_1 , nello stato corrente, ha valore zero o nell'esecuzione del comando C_2 se l'espressione E_1 , nello stato corrente, ha valore diverso da zero.

Definire le regole di semantica operativa per il nuovo comando con riferimento allo stato composto solo da una pila di frame.

Soluzione

$$com_{ifrep-0-tt} \quad \frac{\langle E_1, \sigma \rangle \rightarrow_{exp} 0 \quad \langle C_1, \sigma \rangle \rightarrow_{com} \sigma' \quad \langle E, \sigma' \rangle \rightarrow_{exp} tt}{\langle \text{ifzerorepeat } (E_1, C_1, C_2) \text{ until } E; \sigma \rangle \rightarrow_{com} \sigma'}$$

$$com_{ifrep-non0-tt} \quad \frac{\langle E_1, \sigma \rangle \rightarrow_{exp} n \quad n \neq 0 \quad \langle C_2, \sigma \rangle \rightarrow_{com} \sigma' \quad \langle E, \sigma' \rangle \rightarrow_{exp} tt}{\langle \text{ifzerorepeat } (E_1, C_1, C_2) \text{ until } E; \sigma \rangle \rightarrow_{com} \sigma'}$$

$$com_{ifrep-0-ff} \quad \frac{\langle E_1, \sigma \rangle \rightarrow_{exp} 0 \quad \langle C_1, \sigma \rangle \rightarrow_{com} \sigma' \quad \langle E, \sigma' \rangle \rightarrow_{exp} ff \quad \langle \text{ifzerorepeat } (E_1, C_1, C_2) \text{ until } E; \sigma' \rangle \rightarrow_{com} \sigma''}{\langle \text{ifzerorepeat } (E_1, C_1, C_2) \text{ until } E; \sigma \rangle \rightarrow_{com} \sigma''}$$

$$com_{ifrep-non0-ff} \quad \frac{\langle E_1, \sigma \rangle \rightarrow_{exp} n \quad n \neq 0 \quad \langle C_2, \sigma \rangle \rightarrow_{com} \sigma' \quad \langle E, \sigma' \rangle \rightarrow_{exp} ff \quad \langle \text{ifzerorepeat } (E_1, C_1, C_2) \text{ until } E; \sigma' \rangle \rightarrow_{com} \sigma''}{\langle \text{ifzerorepeat } (E_1, C_1, C_2) \text{ until } E; \sigma \rangle \rightarrow_{com} \sigma''}$$

ESERCIZIO 4 (6 punti)

Completare la definizione del seguente metodo.

```
public int metodo(int[] a)
/** Pone a zero gli elementi strettamente negativi dell'array puntato da a.
 * Restituisce la somma degli elementi positivi dell'array puntato da a.
 */
```

Una possibile soluzione

```
public int metodo(int[] a) {
int somma = 0;
for (int i = 0; i < a.length; i++)
    if (a[i] < 0)
        a[i] = 0;
    else
        somma = somma + a[i];
return somma;
}
```

ESERCIZIO 5 (6 punti)

Si consideri il seguente programma.

```
prog { class Foo {
    public int x;
    public void m(int x, int y) {
        x = x + y;
        this.x = this.x + x;
    }
} (1)
{
    Foo o1 = new Foo;
    o1.x = 30;
    Foo o2 = new Foo;
    o2.x = 20; (2)
    o2.m(o1.x, o2.x); (3)
}
}
```

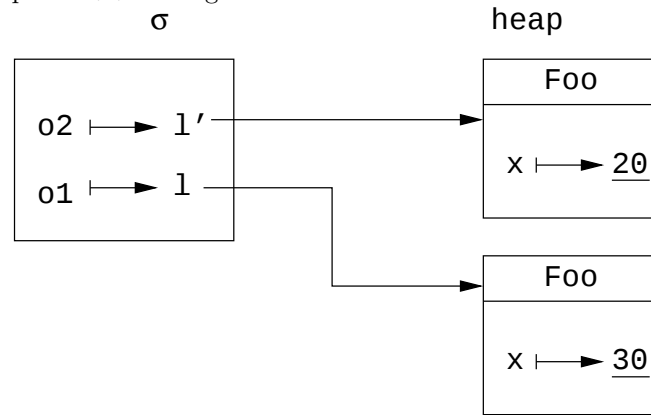
- (a) Disegnare l'ambiente delle classi al punto (1) del programma.
- (b) Disegnare la pila di frame e l'heap ai punti (2) e (3) del programma.

Soluzione

L'ambiente delle classi ρ_c al punto (1) è il seguente.

Foo	$\{x \mapsto \bar{\omega}\}$	<table><tr><td>m</td><td>(x, y)</td><td>$\{x = x + y; \text{ this.x} = \text{this.x} + x;\}$</td></tr></table>	m	(x, y)	$\{x = x + y; \text{ this.x} = \text{this.x} + x;\}$
m	(x, y)	$\{x = x + y; \text{ this.x} = \text{this.x} + x;\}$			

Lo stato al punto (2) è il seguente.



Lo stato al punto (3) è il seguente.

