Fondamenti di Programmazione - 29/11/2012Esame totale sessione straordinaria laureandi e fuoricorso A.A. 11/12I Parziale A.A. 12/13

Co	ognome:	Nome:	Matricola:	
	□ TOTALE:	Svolgere Es.1, Es.2, Es.4	Tempo: 1 ora e 45 min.	
	□ I PARZIALE:	Svolgere Es.1, Es.2, Es.3	Tempo: 1 ora e 45 min.	
Anno di immatricolazione:				

Es. 1) Sia $\Lambda = \{\text{Questo}, \ \dot{\text{E}}, \ \text{Un}, \ \text{Esercizio}, \ \text{Già, Fatto}, \ \text{In, Classe}, \ \text{Ma, Attenti}\} \ \text{e siano}$ $\mathcal{L}_1 = \{\text{Questo}^n \ \dot{\text{E}} \ \text{Un Esercizio}^n | \ n \geqslant 2\}, \ \mathcal{L}_2 = \{\text{Già}^n \ \text{Fatto}^{n+m} \ (\text{In Classe})^m | \ n \geqslant 2, \ m \geqslant 1\} \ \text{e}$ $\mathcal{L}_3 = \{\text{Ma}^n \ \text{Attenti}^m | \ n \geqslant 0, \ m \geqslant 0\} \ \text{linguaggi su} \ \Lambda.$

a) Definire una grammatica che genera il linguaggio:

$$\mathcal{L}_{123} = \{ s_1 s_2 s_3 \mid s_1 \in \mathcal{L}_1, s_2 \in \mathcal{L}_2, s_3 \in \mathcal{L}_3 \}.$$

Per esempio, la stringa

QuestoQuestoÈUnEsercizioEsercizioGiàGiàGiàFattoFattoFattoFattoFattoInClasseInClasseMaMaAttenti appartiene a \mathcal{L}_{123} .

b) La stringa

QuestoQuestoÈUnEsercizioEsercizioGiàFattoInClasseInClasseInClasseMa appartiene a \mathcal{L}_{123} ? Se no, motivare la risposta. Se si, mostrare l'albero di derivazione.

c) Facoltativo - Se la grammatica definita è ambigua, dimostrarlo.

SOLUZIONE:

a)

$$S ::= S_1 S_2 S_3$$

 $S_1 ::= \mathsf{Questo} \ S_1$ Esercizio | $\mathsf{QuestoQuesto\grave{E}UnEsercizioEsercizio}$

$$S_2 ::= S_2' S_2''$$

 $S_2' ::= \operatorname{Già} S_2'$ Fatto | Già $\operatorname{Già}$ FattoFatto

 $S_2'' ::=$ Fatto S_2'' InClasse | FattoInClasse

$$S_3 ::= S_3' S_3''$$

$$S_3' ::= \operatorname{Ma} S_3' \mid \varepsilon$$

$$S_3'' ::=$$
Attenti $S_3'' \mid \varepsilon$

b) La stringa

QuestoQuestoÈUnEsercizioEsercizioGiàFattoInClasseInClasseInClasse

NON appartiene a \mathcal{L}_{123} in quanto devono esserci almeno 2 "Già" e almeno 2 "Fatto". Inoltre, il numero di "Fatto" non è corretto rispetto al numero di "InClasse".

- c) La grammatica definita nel punto a) non è ambigua. Infatti, ... (ragionare sulla struttura delle produzioni e quindi degli alberi di derivazione).
- Es. 2) Dato $A = \{\text{Facile}, \text{Difficile}\}, \text{ sia } A^+ \text{ l'insieme di tutte le stringhe su } A, \text{ ad esclusione della}$

stringa ε . Definire un sistema di transizione per $A_G = \{(\texttt{CompitoCompito} \cdot s)^n | n \ge 1, s \in A\}$ in modo che la semantica di una stringa $s \in A_G$ sia (i) la stringa BENE se il numero di Facile in s è il doppio del numero di Difficile; (ii) la stringa MALE se il numero di Difficile in s è il doppio del numero di Facile; la stringa DIPENDE in ogni altro caso.

Per esempio, la semantica della stringa

 $\hbox{``CompitoCompitoDifficileCompitoCompitoFacileCompitoCompitoDifficileCompitoCompitoDifficile'' \`e DIPENDE,} \\ della \ stringa$

 $\hbox{``CompitoCompitoFacileCompitoCompitoDifficileCompitoCompitoDifficileCompitoCompitoFacile'' \`e BENE, $della stringa$}$

"CompitoCompitoDifficileCompitoCompitoDifficile" è DIPENDE, mentre della stringa

Le configurazioni del sistema contengono, tra le altre, $\{s|s\in A_G\}$. Svolgere l'esercizio specificando anche le restanti configurazioni.

SOLUZIONE:

$$\begin{split} \Gamma &= \{s | s \in A_G\} \ \bigcup \\ &\{ \langle s, n, m \rangle | s \in A_G \cup \{\varepsilon\}, \ n \in \mathbb{N}, \ m \in \mathbb{N} \} \ \bigcup \\ &\{ s | s \in \{ \text{BENE}, \text{MALE}, \text{DIPENDE} \} \} \end{split}$$

$$T = \{s | s \in \{ \text{BENE}, \text{MALE}, \text{DIPENDE} \} \}$$

Assumendo di saper fare la somma di numeri naturali, la moltiplicazione di numeri naturali e il loro confronto (e usando "==" per uguale e "!=" per diverso), la relazione di transizione è così definita:

Es. 3)

Dire se i seguenti comandi COM1 e COM2 sono equivalenti.

COM1: if E then C_1 ; C_3 else C_2 ; C_3 fi COM2: if E then C_1 else C_2 fi; C_3

SOLUZIONE:

[&]quot;CompitoCompitoFacileCompitoCompitoDifficileCompitoCompitoDifficile" è MALE.

N.B.: NON si fanno assunzioni sullo stato iniziale. Si consideri quindi un generico stato σ . la dimostrazione avviene poi ragionando per casi.

CASO I: Consideriamo il caso in cui $\varepsilon \llbracket \mathtt{E} \rrbracket \sigma = tt$.

Per il comando COM1 abbiamo:

$$\frac{\epsilon \llbracket \mathtt{E} \rrbracket \sigma = tt \quad \frac{\left\langle \mathtt{C}_1, \sigma \right\rangle \longrightarrow_{com} \sigma' \quad \left\langle \mathtt{C}_3, \sigma' \right\rangle \longrightarrow_{com} \sigma''}{\left\langle \mathtt{C}_1; \mathtt{C}_3, \sigma \right\rangle \longrightarrow_{com} \sigma''} \quad (com_{,)}}{\left\langle \mathtt{if} \ \mathtt{E} \ \mathtt{then} \ \mathtt{C}_1; \mathtt{C}_3 \ \mathtt{else} \ \mathtt{C}_2; \mathtt{C}_3 \ \mathtt{fi}, \sigma \right\rangle \longrightarrow_{com} \sigma''} \quad (com_{if-tt})}$$

Per il comando COM2 abbiamo:

$$\frac{\epsilon \llbracket \mathtt{E} \rrbracket \sigma = tt \quad \langle \mathtt{C}_1, \sigma \rangle \longrightarrow_{com} \sigma'}{\text{if E then } \mathtt{C}_1 \text{ else } \mathtt{C}_2 \text{ fi}, \sigma \rangle \longrightarrow_{com} \sigma'} \quad (com_{if-tt}) \quad \langle \mathtt{C}_3, \sigma' \rangle \longrightarrow_{com} \sigma''} \quad (com_{;})$$

$$\langle \mathtt{if E then } \mathtt{C}_1 \text{ else } \mathtt{C}_2 \text{ fi}; \mathtt{C}_3, \sigma \rangle \longrightarrow_{com} \sigma''}$$

CASO II: Consideriamo ora il caso in cui $\varepsilon \llbracket \mathtt{E} \rrbracket \sigma = ff$

Per il comando COM1 abbiamo:

$$\frac{\epsilon \llbracket \mathtt{E} \rrbracket \sigma = \mathit{ff} \quad \frac{\left\langle \mathtt{C}_2, \sigma \right\rangle \longrightarrow_{\mathit{com}} \sigma_1' \quad \left\langle \mathtt{C}_3, \sigma_1' \right\rangle \longrightarrow_{\mathit{com}} \sigma_1''}{\left\langle \mathtt{C}_2; \mathtt{C}_3, \sigma \right\rangle \longrightarrow_{\mathit{com}} \sigma''} \quad (\mathit{com}_{;})}{\left\langle \mathtt{if} \ \mathtt{E} \ \mathtt{then} \ \mathtt{C}_1; \mathtt{C}_3 \ \mathtt{else} \ \mathtt{C}_2; \mathtt{C}_3 \ \mathtt{fi}, \sigma \right\rangle \longrightarrow_{\mathit{com}} \sigma_1''} \quad (\mathit{com}_{if-\mathit{ff}})$$

Per il comando COM2 abbiamo:

Non si fatta alcuna ipotesi sullo stato iniziale σ e si sono ottenuti i seguenti finali: nel **CASO** I, σ'' sia per COM1 sia per COM2; nel **CASO** II, σ''_1 sia per COM1 sia per COM2 Quindi, partendo da un generico stato, i due comandi portano in tuti i casi possibili nello stesso stato e quindi nella stessa configurazione terminale. I comandi sono pertanto equivalenti.

- Es. 4) I seguenti programmi P1 e P2 in +/- Java sono identici fino alla linea 29. Considerando che il simbolo "?" indica un generico valore intero che inizializza la variabile j:
- a) si calcoli direttamente lo stato σ_{29} alla linea 29 in tutte le sue componenti ρ_i , μ_j e δ_1 . Non è necessario mostrare i passaggi e le regole utilizzate.
- b) eseguendo le rispettive linee 30-35 dallo stato σ_{29} (si ricorda che $\rho_i(j)$ =? valore generico) dimostrare se i due programmi P1 e P2 sono equivalenti o meno. Per la dimostrazione è sufficiente

<u>indicare le regole da utilizzare</u> nelle linee 30-35 di P1 e P2, <u>mostrare stacks e heaps risultanti</u>, e trarre le conclusioni per i vari casi.

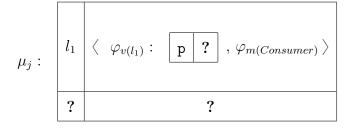
```
P1
                                                                           P2
01: class Producer {
                                                    01: class Producer {
02:
     private int resource;
                                                    02:
                                                         private int resource;
03:
                                                    03:
04:
     public int getResource() {
                                                    04:
                                                         public int getResource() {
05:
       return this.resource;
                                                    05:
                                                           return this.resource;
06:
                                                    06:
     }
07:}
                                                    07:}
                                                    08:
08:
09: class Consumer {
                                                    09: class Consumer {
10:
    private Producer p;
                                                    10:
                                                         private Producer p;
11:
                                                   11:
12:
     public Consumer(int eur) {
                                                    12:
                                                         public Consumer(int eur) {
13:
       if (eur >= 8) this p = new Producer();
                                                    13:
                                                           if(eur >= 8) this.p = new Producer();
14:
                                                   14:
       else ;
                                                           else ;
15:
                                                    15:
16:
                                                    16:
17:
     public Producer producerFactory(int x) {
                                                   17:
                                                         public Producer producerFactory(int x) {
18:
       if(x < 8) this.p = new Producer();
                                                    18:
                                                           if(x < 8) this.p = new Producer();
19:
       return this p;
                                                    19:
                                                           return this p;
20:
                                                    20:
21:}
                                                   21:}
                                                    22:
22:
23: public class Program {
                                                    23: public class Program {
                                                    24:
24:
25:
     public static void main(String[] cLine) {
                                                    25:
                                                         public static void main(String[] cLine) {
       int j = ?;
26:
                                                    26:
                                                           int j = ?;
       Consumer c = new Consumer(j);
                                                           Consumer c = new Consumer(j);
27:
                                                    27:
28:
       Producer p = c.producerFactory(j);
                                                    28:
                                                           Producer p = c.producerFactory(j);
29:
                                                    29:
30:
       if (j >= 8)
                                                    30:
                                                           while (j >= 8) {
31:
        j = p.getResource();
                                                    31:
                                                               j = p.getResource();
       else {
32:
                                                    32:
33:
       p = c.producerFactory(j);
                                                    33:
34:
        j = p.getResource();
                                                    34:
                                                           p = c.producerFactory(j);
35:
                                                    35:
                                                           j = p.getResource();
36:
                                                   36:
37:
     }
                                                    37:
                                                        }
38:}
                                                    38:}
```

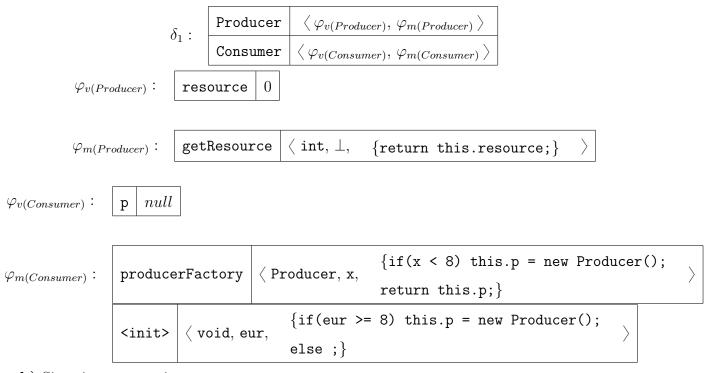
SOLUZIONE:

a) Lo stato alla linea 29 è il seguente:

$$\sigma_{29} = \langle \rho_i, \mu_j, \delta_1 \rangle$$

	cLine	null
0	j	?
$ ho_i$:	С	l_1
	р	?





b) Si ragiona per casi:

CASO I: Supponendo che j $\geqslant 8$ si ha che $\sigma_{29} = \langle \rho_i, \mu_j, \delta_1 \rangle$ è lo stato in cui ρ_i e μ_j sono:

$$ho_i$$
: $egin{array}{|c|c|c|c|c|} { t cLine} & null & & & \\ \hline { t j} & x & (\cos x \geqslant 8) & & & \\ \hline { t c} & & l_1 & & & \\ \hline { t p} & & l_2 & & & \\ \hline \end{array}$

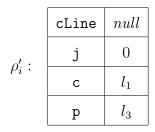
Si osservi che, fino alla linea 29, per $j \ge 8$ l'istanza della classe Producer riferita da l_2 , e quindi da p, è stata creata dal costruttore della classe Consumer.

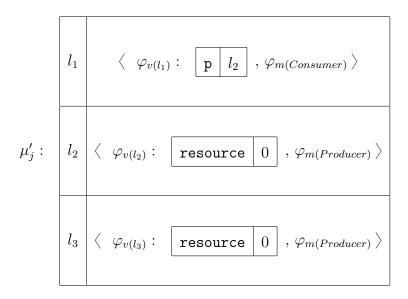
• Programma P1: Applicazione della regola com_{if_true} alle linee 30-35, applicazione della regola com_{assign} per j = p.getResource(); e quindi della regola $exp_{mcallnodecl}$ alla linea 31 per la chiamata p.getResource() che ritorna 0. Quindi, il sistema di transizione porta in una configurazione terminale $\sigma'_{29} = \langle \rho'_i, \mu_j, \delta_1 \rangle$ dove

	cLine	null
o' ·	j	0
$ ho_i'$:	С	l_1
	р	l_2

• Programma P2: Applicazione della regola com_{while_true} alla linea 30-32 (il comando while effettua un solo ciclo), applicazione della regola com_{concat} alle linee 34-35, applicazione della regola com_{assign} per p = c.producerFactory(j); e quindi della regola $exp_{mcallparnodecl}$ alla linea 34 per la chiamata c.producerFactory(j) che ritorna una nuova istanza della classe Producer riferita da l_3 , applicazione della regola com_{assign} per j = p.getResource(); e quindi della regola $exp_{mcallnodecl}$ alla linea 35 per la chiamata p.getResource() che ritorna 0.

Quindi, il sistema di transizione porta in una configurazione terminale $\sigma'_{29} = \langle \rho'_i, \mu'_j, \delta_1 \rangle$ dove

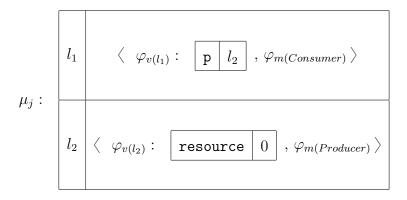




Si osservi che ora si ha una ulteriore istanza della classe Producer, e quindi μ_j del programma **P1** è diverso da μ_j' del programma **P2**.

CASO II: Supponendo che j < 8 si ha che $\sigma_{29} = \langle \rho_i, \mu_j, \delta_1 \rangle$ è lo stato in cui ρ_i e μ_j sono:

	cLine	null
0	j	x (con x < 8)
$ ho_i$:	С	l_1
	р	l_2



Si osservi che, fino alla linea 29, l'istanza della classe Producer riferita da l_2 è stata ora creata dalla chiamata alla funzione c.producerFactory(j).

- Programma P1: Applicazione della regola com_{if_false} alle linee 30-35, applicazione della regola com_{assign} per p = c.producerFactory(j); e quindi della regola $exp_{mcallparnodecl}$ alla linea 33 per la chiamata c.producerFactory(j) che ritorna una nuova istanza della classe Producer riferita da l_3 , applicazione della regola com_{assign} per j = p.getResource(); e quindi della regola $exp_{mcallnodecl}$ alla linea 34 per la chiamata p.getResource() che ritorna 0.
- Programma P2: Applicazione della regola com_{while_false} alle linee 30-32, applicazione della regola com_{concat} alle linee 34-35, applicazione della regola com_{assign} per p = c.producerFactory(j); e quindi della regola $exp_{mcallparnodecl}$ alla linea 34 per la chiamata c.producerFactory(j) che ritorna una nuova istanza della classe Producer riferita da l_3 , applicazione della regola com_{assign} per j = p.getResource(); e quindi della regola $exp_{mcallnodecl}$ alla linea 35 per la chiamata p.getResource() che ritorna 0.

Le tabelle rappresentanti gli stati finali per i due programmi nal CASO II non vengono mostrate. Si può comunque facilmente dedurre che i due programmi portano allo stesso stato finale.

• Conclusione: I CASI I e II esauriscono tutti i casi possibili. Sebbene per il CASO II i due programmi P1 e P2 portano allo stesso stato finale, per il CASO I i due programmi portano a differenti stati finali. Quindi, P1 e P2 NON sono equivalenti.