

**Informatica Teorica I (Informatica Teorica primo modulo)**  
**Esame del 13 novembre 2003**

**Tempo a disposizione: 120 minuti**

**Regole del gioco:** Libri e quaderni chiusi, vietato scambiare informazioni con altri; indicare su tutti i fogli, con chiarezza, nome e numero di matricola; consegnare solo i fogli con le domande (questi).

**Esercizio 1 (20%)** Determina le espressioni regolari che descrivono i seguenti linguaggi su  $\Sigma = \{a, b\}$ :

**1.1** Stringhe lunghe esattamente tre caratteri.

$(a+b)(a+b)(a+b)$   
*oppure*  $aaa+bbb+abb+bab+bba+baa+aba+aab$

**1.2** Stringhe la cui lunghezza è un multiplo di tre (anche zero).

$((a+b)(a+b)(a+b))^*$   
*oppure*  $(aaa+bbb+abb+bab+bba+baa+aba+aab)^*$

**1.3** Stringhe la cui lunghezza è pari (anche zero).

$((a+b)(a+b))^*$   
*oppure*  $(aa+ab+ba+bb)^*$

**1.4** Stringhe per le quali la lunghezza di ogni sottosequenza di sole **a** è pari (esempio:  $\epsilon$ , **aa**, **aaaabaa**, **bbaabbbaaaa**, ...)

$(aa+b)^*$   
*oppure*  $((aa)^*b^*)^*$

**1.5** Stringhe per le quali la lunghezza di ogni sottosequenza di sole **a** è pari e non esistono due **b** consecutive (esempio:  $\epsilon$ , **aa**, **aabaabaaaab**, **baaaabaa**, ...)

$(b+\epsilon)(aa+aab)^*$   
*oppure*  $(aa+baa)^*(b+\epsilon)$   
*oppure*  $b(aa+aab)^* + (aa+aab)^*$

**Esercizio 2 (20%)** Determina un'espressione regolare che descriva il linguaggio generato dalla seguente grammatica regolare.

$S \rightarrow aS \mid bA \mid bB \mid bC \mid \varepsilon$

$A \rightarrow aS$

$B \rightarrow bA$

$C \rightarrow bD$

$D \rightarrow bD \mid aS$

$S = aS + bA + bB + bC + \varepsilon$

$A = aS$

$B = bA$

$C = bD$

$D = bD + aS$

$D = b^*aS$

$C = bb^*aS$

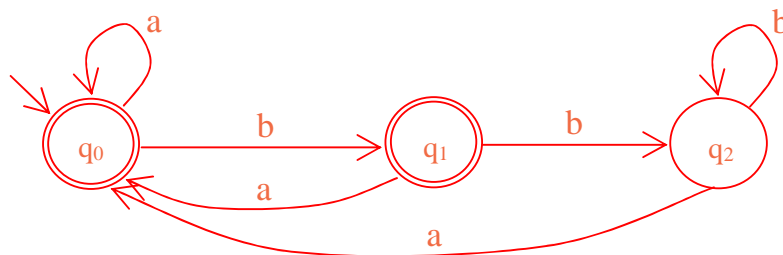
$B = baS$


$S = aS + baS + bbaS + bbb^*aS + \varepsilon$

$S = (a + ba + bba + bbb^*a)^* = (b^*a)^*$

**Esercizio 3 (20%)**

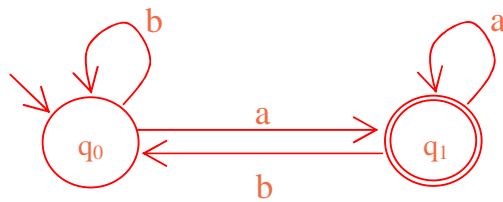
3.1) Costruisci un ASF deterministico che riconosca il linguaggio delle stringhe di  $(a+b)^*$  tali che ogni sequenza di due o più **b** è seguita da almeno una **a**. Esempi di stringhe del linguaggio sono:  $\varepsilon$ , **b**, **aaa**, **bbba**, **bbaab**, **bbabbbbaa**.



3.2)  Costruisci un ASF (deterministico o non deterministico) che riconosca l'intersezione del linguaggio  $L_1$  descritto al punto precedente con il linguaggio  $L_2$  delle stringhe di  $(a+b)^*$  lunghe almeno un carattere e terminanti con **a**.

il linguaggio  $L_2$  delle stringhe su  $(a+b)^*$  terminanti con a è un sottoinsieme del linguaggio  $L_1$ , in quanto in una stringa di  $L_2$  ogni sequenza di due o più b è sempre seguita da una a. Dunque l'intersezione di  $L_1$  e  $L_2$  coincide con  $L_2$

Un automa riconoscatore è il seguente:



**Esercizio 4 (20%)** Come dimostreresti che qualsiasi espressione regolare definisce un linguaggio regolare? (Puoi supporre nota l'identità tra grammatiche regolari e automi a stati finiti).

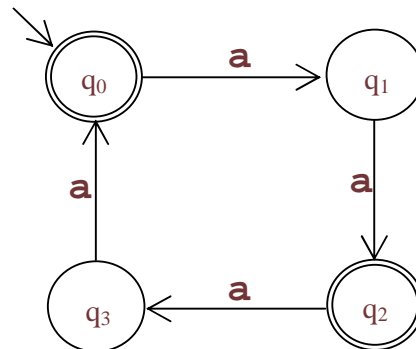
Presa un'espressione regolare generica essa può essere ricondotta, considerando la sua definizione ricorsiva, all'unione, alla concatenazione e alla chiusura stella dei linguaggi elementari  $\{\epsilon\}$  e  $\{a\}$ , per ogni  $a \in \Sigma$ .

Tali linguaggi sono regolari (perché generabili tramite grammatiche regolari o perché riconoscibili da opportuni automi a stati finiti elementari).

Le proprietà di chiusura ci assicurano così che l'espressione regolare definisce effettivamente un linguaggio regolare.

N.B.: il fatto che da una grammatica regolare è possibile generare un'espressione regolare, non è sufficiente a dimostrare che OGNI espressione regolare definisce un linguaggio regolare. Occorrerebbe dimostrare che dalle grammatiche regolari si possono ottenere TUTTE le espressioni regolari.

**Esercizio 5 (20%)** Mostra le classi di equivalenza di Myhill-Nerode per il linguaggio su  $\Sigma=\{a\}$  riconosciuto dal seguente ASF



Costruisco le classi di equivalenza di  $R_M$

$$\begin{aligned}
 C_0 &= \{(aaaa)^*\} & \rightarrow (aaaa)^* + aa(aaaa)^* &= (aa)^* \\
 C_1 &= \{a(aaaa)^*\} & \rightarrow a(aaaa)^* + aaa(aaaa)^* &= a(aa)^* \\
 C_2 &= \{aa(aaaa)^*\} & \rightarrow (aaaa)^* + aa(aaaa)^* &= (aa)^* \\
 C_3 &= \{aaa(aaaa)^*\} & \rightarrow a(aaaa)^* + aaa(aaaa)^* &= a(aa)^*
 \end{aligned}$$

(‘ $\rightarrow$ ’ = “che mi danno una stringa di L se concatenate con il suffisso...”)

ora noto che le stringhe di  $C_0$  e  $C_2$  sono equivalenti in  $R_L$  (“ammettono gli stessi completamenti”). Lo stesso dicasi per le stringhe di  $C_1$  e  $C_3$ .

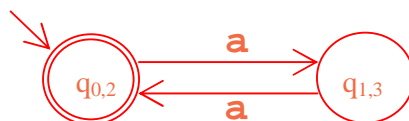
Ne segue che le classi di equivalenza di Myhill-Nerode sono:

$$\begin{aligned}
 C_{0,2} &= \{(aaaa)^* + aa(aaaa)^*\} \\
 C_{1,3} &= \{a(aaaa)^* + aaa(aaaa)^*\}
 \end{aligned}$$

Che si possono riscrivere anche

$$\begin{aligned}
 C_{0,2} &= \{(aa)^*\} \\
 C_{1,3} &= \{a(aa)^*\}
 \end{aligned}$$

Tali classi di equivalenza corrispondono all’AFS con il numero minimo di stati seguente:



N.B.: per questo linguaggio non c’è la classe  $C = \{w \mid \forall z \in \Sigma, wz \notin L\}$  in quanto non esiste una sequenza di ‘a’ che non possa essere completata in modo da generare una stringa del linguaggio