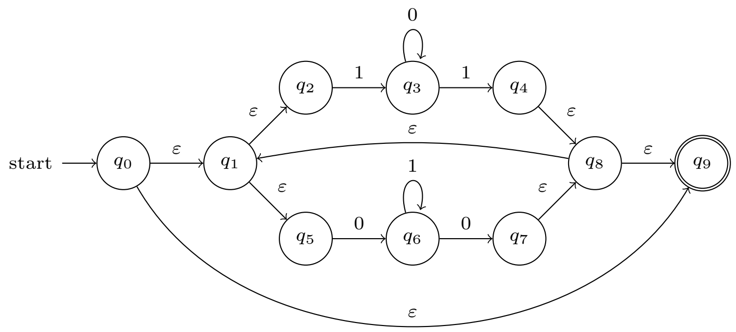
Automi e Linguaggi Formali
Esame del 3 Settembre 2018

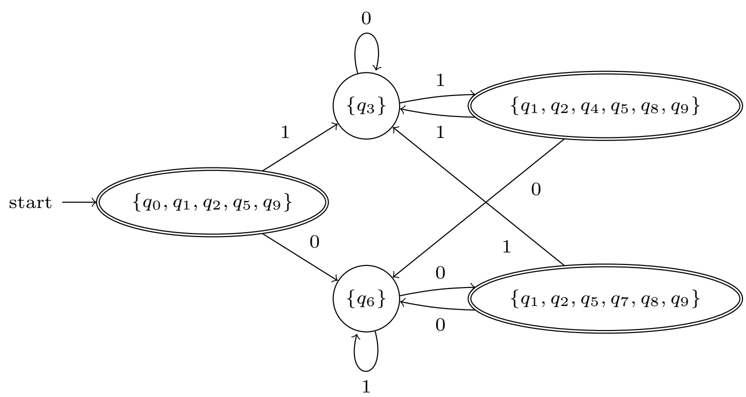
Tempo a disposizione: 2 ore

1. (a) Convertire l'espressione regolare $(\mathbf{10^*1} + \mathbf{01^*0})^*$ in un ε-NFA (si può usare l'algoritmo visto a lezione oppure creare direttamente l'automa). **Importante:** l'automa deve essere nondeterministico e deve sfruttare le ε-transizioni per riconoscere il linguaggio.

L'esercizio ha molte possibili soluzioni, una delle quali è la seguente:



(b) Trasformare l'ε-NFA ottenuto al punto precedente in un DFA.



2. Considerate il linguaggio $L = \{0^{2n}1^m0^n : n, m \geq 0\}$. Questo linguaggio è regolare? Dimostrare formalmente la risposta.

Il linguaggio non è regolare. Supponiamo per assurdo che lo sia:

- *sia h la lunghezza data dal Pumping Lemma;*
- *consideriamo la parola $w = 0^{2h}10^h$, che appartiene ad L ed è di lunghezza maggiore di h;*
- *sia $w = xyz$ una suddivisione di w tale che $y \neq \varepsilon$ e $|xy| \leq h$;*
- *poiché $|xy| \leq h$, allora xy è completamente contenuta nel prefisso 0^{2h} di w, e quindi sia x che y sono composte solo da 0. Inoltre, siccome $y \neq \varepsilon$, possiamo dire che $y = 0^p$ per qualche valore $p > 0$. Allora la parola xy^2z è nella forma $0^{2h+p}10^h$, e quindi non appartiene al linguaggio perché il numero di 0 nella prima parte della parola non è uguale al doppio del numero di 0 nella seconda parte della parola.*

Abbiamo trovato un assurdo quindi L_1 non può essere regolare.

3. Descrivere un automa a pila che accetti per stack vuoto il linguaggio che consiste di stringhe composte di a e b tali che, se n è il numero degli a, allora il numero dei b è tra n e $2n$. Il linguaggio contiene la stringa vuota. Si osservi che nelle stringhe del linguaggio, gli a e i b possono essere mescolati (cioè non necessariamente della forma a^nb^m).

Il PDA P che riconosce L per stack vuoto ha 2 stati q_1 e q_2 e le seguenti transizioni:

$\delta(q_1, a, Z) = \{(q_1, aZ), (q_1, aaZ)\}, \delta(q_1, a, a) = \{(q_1, aa), (q_1, aaa)\}, \delta(q_1, a, b) = \{(q_1, \epsilon), (q_2, \epsilon)\},$

$\delta(q_1, \epsilon, Z) = \{(q_1, \epsilon)\}$,

$\delta(q_1, b, a) = \{(q_1, \epsilon)\}, \delta(q_1, b, Z) = \{(q_1, bZ)\}, \delta(q_1, b, b) = \{(q_1, bb)\}$

$\delta(q_2, \epsilon, b) = \{(q_1, \epsilon)\}, \delta(q_2, \epsilon, Z) = \{(q_1, aZ)\}$

Lo stato q_2 serve quando si consuma l'input a si decide che deve contare doppio e la cima della pila contiene b, allora la transizione da q_1 a q_2 consuma il primo b della pila e lo stato q_2 vede quello che c'era sotto quel b. Se c'era un altro b, lo toglie, mentre se c'era Z, allora aggiunge a allo stack. In entrambi i casi torna in q_1.

4. Dare la definizione precisa del linguaggio L_{ne} e successivamente dimostrare che esso è un linguaggio RE. La prova richiede di esibire una TM che riconosca in tempo finito tutte le stringhe del linguaggio L_{ne}.

$L_{ne} = \{M |\ L(M) \neq \emptyset\}$. Per dimostrare che L_{ne} è RE è necessario definire una TM M che lo riconosce. Gli input di L_{ne} sono tutte le stringhe binarie che vengono interpretate come TM. La TM M sarà come segue: data in input una stringa binaria che rappresenta la TM M', M genera nondeterministicamente tutte le possibili stringhe binarie e per ogni tale stringa w, simula M'' su w (per farlo usa la TM universale), se M' accetta qualcuna delle w prodotte, allora M accetta M', infatti, in questo caso, M' appartiene a L_{ne}. Il punto centrale è la generazione nondeterministica di tutte le possibili stringhe binarie w. Ogni w è finita, e quindi verrà prodotta in un tempo finito e se esiste w che è accettata da M', M accetterà M' in un tempo finito. Se invece M' non accetta alcuna w e quindi $M' \notin L_{ne}$, allora M continuerà il suo calcolo per sempre e questo è compatibile con il fatto che L_{ne} sia RE.

5. Stabilire se il seguente linguaggio $L = \{a^n b^m c^k \mid n = k, n \geq 0, m \geq 0, k \geq 0\}$ è CF oppure no. Se pensate che sia CF, esibite una CFG che lo generi oppure un PDA che lo riconosca (spiegando perché questo è vero). Se pensate che non sia CF, date un argomento che dimostri che effettivamente non lo sia.

L è CF. La seguente grammatica CF genera L:

$S \to aSc \mid B$

$B \to bB \mid \epsilon$

6. Un circuito Hamiltoniano in un grafo G è un ciclo che attraversa ogni vertice di G esattamente una volta. Stabilire se un grafo contiene un circuito Hamiltoniano è un problema NP-completo.

Considerate il seguente problema, che chiameremo HAM375: dato un grafo G con n vertici, trovare un ciclo che attraversa esattamente una volta $n - 375$ vertici del grafo (ossia tutti i vertici di G *tranne* 375).

(a) Dimostrare che il problema HAM375 è in NP fornendo un certificato per il *Sì* che si può verificare in tempo polinomiale.
Sia n il numero di vertici del grafo che è istanza di HAM375. Un certificato per HAM375 è una sequenza ordinata di $n - 375$ vertici distinti. Occorre tempo polinomiale per verificare se ogni nodo è collegato al successivo e l'ultimo al primo.

(b) Mostrare come si può risolvere il problema del circuito Hamiltoniano usando il problema HAM375 come sottoprocedura.
Dato un qualsiasi grafo G che è un'istanza del problema del circuito Hamiltoniano, costruiamo in tempo polinomiale un nuovo grafo G' che è un'istanza di HAM375, aggiungendo a G 375 vertici isolati (senza archi). Chiaramente G' ha un ciclo che attraversa esattamente una volta $n - 375$ vertici del grafo se e solo se G ha un ciclo Hamiltoniano.