Automi e Linguaggi Formali – 20/4/2023
Prima prova intermedia – Soluzione

**1. (12 punti)** Data una parola $w \in \Sigma^*$, definiamo $evens(w)$ come la sottosequenza di $w$ che contiene solo i simboli in posizione pari (iniziando a contare da 1). Per esempio, $evens(INDICEPARI) = NIEAI$. Dimostra che se $L \subseteq \Sigma^*$ è un linguaggio regolare allora anche il linguaggio

$$evens(L) = \{evens(w) \mid w \in L\}$$

è un linguaggio regolare.

**Soluzione:** Se $L$ è un linguaggio regolare, allora sappiamo che esiste un DFA $A = (Q, \Sigma, \delta, q_0, F)$ che riconosce $L$. Costruiamo un $\varepsilon$-NFA $A'$ che accetta il linguaggio $evens(L)$, aggiungendo un flag $0, 1$ agli stati di $A$, dove flag 0 corrisponde a "simbolo in posizione pari" e flag 1 corrisponde a "simbolo in posizione dispari". La funzione di transizione è fatta in modo da alternare i flag 0 e 1 dopo ogni transizione dell'automa. Se lo stato corrente ha flag 0, l'automa consuma il prossimo simbolo della stringa di input e prosegue nello stesso stato che raggiungerebbe $A$ dopo aver consumato lo stesso simbolo. Se lo stato corrente ha flag 1, l'automa prosegue con una $\varepsilon$-transizione verso uno degli stati raggiungibili dall'automa $A$ consumando un simbolo: simulando in questo modo il fatto che i simboli in posizione dispari vanno "saltati".

Formalmente, $A' = (Q', \Sigma, \delta', q'_0, F')$ è definito come segue.

- $Q' = Q \times \{0, 1\}$.
- L'alfabeto $\Sigma$ rimane lo stesso.
- $\delta'((q, 0), a) = \{(\delta(q, a), 1)\}$. Con il flag 0 l'automa consuma un simbolo di input ed ha una sola alternativa possibile: lo stato $\delta(q, a)$ raggiunto da $A$ dopo aver consumato $a$. Il flag diventa 1.
- $\delta'((q, 1), \varepsilon) = \{(q', 0) \mid \text{esiste } a \in \Sigma \text{ tale che } \delta(q, a) = q'\}$. Con il flag 1 l'automa procede nondeterministicamente con una $\varepsilon$-transizione verso uno degli stati raggiungibili da $A$ dopo aver consumato un simbolo arbitrario.
- $q'_0 = (q_0, 1)$. Lo stato iniziale corrisponde allo stato iniziale di $A$ con flag 1 (simbolo in posizione dispari).
- $F' = F \times \{0, 1\}$. L'insieme degli stati finali di $A'$ corrisponde all'insieme degli stati finali di $A$ con qualsiasi flag.

Per dimostrare che $A'$ riconosce il linguaggio $evens(L)$, data una parola $w = w_1 w_2 \dots w_n$, dobbiamo considerare due casi.

- Se $w \in L$, allora esiste una computazione di $A$ che accetta la parola. Di conseguenza, esiste una computazione di $A$ che accetta la parola:

$$s_0 \xrightarrow{w_1} s_1 \xrightarrow{w_2} \dots \xrightarrow{w_n} s_n$$

con $s_0 = q_0$ e $s_n \in F$. Se $w$ è di lunghezza pari, la computazione

$$(s_0, 1) \xrightarrow{\varepsilon} (s_1, 0) \xrightarrow{w_2} (s_2, 1) \xrightarrow{\varepsilon} (s_3, 0) \xrightarrow{w_4} (s_4, 1) \xrightarrow{\varepsilon} \dots \xrightarrow{w_n} (s_n, 1)$$

è una computazione accettante per $A'$ sulla parola $evens(w) = w_2 w_4 w_6 \dots w_n$, perché $(s_n, 1) \in F'$. Se $w$ è di lunghezza dispari, allora la computazione accettante per $A'$ su $evens(w)$ è

$$(s_0, 1) \xrightarrow{\varepsilon} (s_1, 0) \xrightarrow{w_2} (s_2, 1) \xrightarrow{\varepsilon} (s_3, 0) \xrightarrow{w_4} (s_4, 1) \xrightarrow{\varepsilon} \dots \xrightarrow{w_{n-1}} (s_{n-1}, 1) \xrightarrow{\varepsilon} (s_n, 1).$$

- Se $w$ è accettata dal nuovo automa $A'$, allora esiste una computazione accettante che ha la forma

$$(s_0, 1) \xrightarrow{\varepsilon} (s_1, 0) \xrightarrow{w_1} (s_2, 1) \xrightarrow{\varepsilon} (s_3, 0) \xrightarrow{w_2} (s_4, 1) \xrightarrow{\varepsilon} \dots \xrightarrow{w_n} (s_n, 1),$$

se $(s_n, 1)$ è uno stato finale, oppure la forma

$$(s_0, 1) \xrightarrow{\varepsilon} (s_1, 0) \xrightarrow{w_1} (s_2, 1) \xrightarrow{\varepsilon} (s_3, 0) \xrightarrow{w_2} (s_4, 1) \xrightarrow{\varepsilon} \dots \xrightarrow{w_n} (s_n, 1) \xrightarrow{\varepsilon} (s_{n+1}, 0),$$

quando è necessaria una $\varepsilon$-transizione finale per raggiungere uno stato finale. In entrambi i casi possiamo costruire una computazione accettante per $A$ sostituendo le $\epsilon$-transizioni con transizioni che consumano un carattere. Nel primo caso si ottiene una computazione che ha la forma

$$s_0 \xrightarrow{u_1} s_1 \xrightarrow{w_1} s_2 \xrightarrow{u_2} s_3 \xrightarrow{w_2} s_4 \xrightarrow{\varepsilon} \dots \xrightarrow{w_n} s_n,$$

e accetta una parola $u = u_1w_1u_2w_2 \dots u_nw_n$. Nel secondo caso si ottiene una computazione

$$s_0 \xrightarrow{u_1} s_1 \xrightarrow{w_1} s_2 \xrightarrow{u_2} s_3 \xrightarrow{w_2} s_4 \xrightarrow{\varepsilon} \dots \xrightarrow{w_n} s_n \xrightarrow{u_{n+1}} s_{n+1},$$

che accetta la parola $u = u_1w_1u_2w_2 \dots u_nw_nu_{n+1}$. In entrambi i casi $evens(u) = w$, come richiesto.

**2.** **(12 punti)** Considera il linguaggio

$$L_2 = \{uwu \mid u, w \text{ sono stringhe di 0 e 1 tali che } |u| = |w|\}.$$

Dimostra che $L_2$ non è regolare.

**Soluzione:** Usiamo il Pumping Lemma per dimostrare che il linguaggio non è regolare. Supponiamo per assurdo che $L_2$ sia regolare:

- sia $k$ la lunghezza data dal Pumping Lemma;
- consideriamo la parola $w = 1^k0^k1^k$, che è di lunghezza maggiore di $k$ ed appartiene ad $L_2$ perché il primo terzo della parola è uguale all'ultimo terzo;
- sia $w = xyz$ una suddivisione di $w$ tale che $y \neq \varepsilon$ e $|xy| \leq k$;
- poiché $|xy| \leq k$, allora $x$ e $y$ sono entrambe contenute nella sequenza iniziale di 1. Inoltre, siccome $y \neq \varepsilon$, abbiamo che $x = 1^q$ e $y = 1^p$ per qualche $q \geq 0$ e $p > 0$. $z$ contiene la parte rimanente della stringa: $z = 1^{k-q-p}0^k1^k$. Consideriamo l'esponente $i = 3$: la parola $xy^2z$ ha la forma

  $$xy^2z = xz = 1^q1^{2p}1^{k-q-p}0^k1^k = 1^{k+p}0^k1^k$$

  Poiché $p > 0$, la sequenza iniziale di 1 è più lunga della sequenza finale di 1, e quindi la parola iterata $xy^2z$ non appartiene ad $L_2$ perché il primo terzo della parola è fatta solamente da 1 mentre l'ultimo terzo include anche un certo numero di 0.

Abbiamo trovato un assurdo quindi $L_2$ non può essere regolare.

**3.** **(12 punti)** Dimostra che se $L \subseteq \Sigma^*$ è un linguaggio context-free allora anche $L^R$ è un linguaggio context-free, dove $L^R = \{w^R \in \Sigma^* \mid w \in L \text{ e } w^R \text{ è la stringa } w \text{ rovesciata}\}$.

**Soluzione:** Se $L$ è un linguaggio context-free, allora esiste una grammatica $G = (V, \Sigma, R, S)$ che lo genera. Per dimostrare che $L^R$ è context-free, dobbiamo essere in grado di definire una grammatica che possa generarlo. Questa grammatica è una quadrupla $G' = (V', \Sigma', R', S')$ definita come segue.

- L'alfabeto è lo stesso del linguaggio $L$ originale: $\Sigma' = \Sigma$.
- L'insieme di variabili è lo stesso della grammatica $G$: $V' = V$.
- Il nuovo insieme di regole $R'$ è ottenuto "rovesciando" la parte destra delle regole di $R$: $R' = \{A \to u^R \mid A \to u \in R\}$. In questo modo qualsiasi derivazione deve ora seguire le regole rovesciate.
- Si noti che mentre la parte destra delle regole deve rovesciata, l'ordine delle regole nella derivazione non deve essere invertito. Pertanto, la variabile iniziale rimane la stessa: $S' = S$.