Automi e Linguaggi Formali – 28/4/2023
Prima prova intermedia – Soluzione

1. (12 punti) Dimostra che se $L \subseteq \Sigma^*$ è un linguaggio regolare allora anche il linguaggio

$$\mathit{substring}(L) = \{y \in \Sigma^* \mid xyz \in L \text{ con } x, z \in \Sigma^*\}$$

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 ε-NFA A' che accetta il linguaggio $\mathit{substring}(L)$. L'automa A' è costituto dagli stessi stati di A a cui viene aggiunto un nuovo stato iniziale q_0'. La funzione di transizione contiene una ϵ-transizione dal nuovo stato iniziale q_0' verso tutti gli stati di A che sono raggiungibili da q_0, e si comporta come A per gli altri stati. Gli stati finali dell'automa sono tutti gli stati di A a partire dai quali esiste una computazione che raggiunge uno stato finale di A.

Formalmente, $A' = (Q', \Sigma, \delta', q_0', F')$ è definito come segue.

- $Q' = Q \cup \{q_0'\}$, dove q_0' è uno stato nuovo che non appartiene a Q.
- L'alfabeto Σ rimane lo stesso.
- $\delta'(q_0', \varepsilon) = \{q \in Q \mid \text{esiste una computazione da } q_0 \text{ a } q \text{ in } A\}$.
- $\delta'(q, a) = \{\delta(q, a)\}$ per ogni stato $q \neq q_0'$ e $a \in \Sigma$.
- $F' = \{q \in Q \mid \text{esiste una computazione da } q \text{ ad uno stato di } F\}$.

Per dimostrare che A' riconosce il linguaggio $\mathit{substring}(L)$, dobbiamo considerare due casi.

- Se $w \in L$, e data una qualsiasi suddivisione $w = xyz$, esiste una computazione di A che accetta la parola. Definiamo q_1 come lo stato raggiunto da A dopo aver consumato y, e q_2 come lo stato raggiunto da A dopo aver consumato xy. Allora possiamo costruire una computazione di A' che accetta y in questo modo:
$$q_0' \xrightarrow{\varepsilon} q_1 \xrightarrow{y_1} \ldots \xrightarrow{y_n} y_2.$$
Siccome $y_2 \in F'$, la computazione è accettante per A'.
- Se y è accettata dal nuovo automa A', allora esiste una computazione accettante che ha la forma
$$q_0' \xrightarrow{\varepsilon} q_1 \xrightarrow{y_1} \ldots \xrightarrow{y_n} y_{n+1},$$
con $y_{n+1} \in F'$. Per la definzione della funzione di transizione δ', abbiamo che esiste una computazione da q_0 a q_1 di A, che consuma una certa parola x. Per la definizione dell'insieme di stati finali F', esiste una computazione di A che raggiunge uno stato finale di A a partire da y_{n+1}, che consuma una certa parola z. Di conseguenza, la parola xyz è accettata da A.

2. (12 punti) Considera il linguaggio

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

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 $w = 1^k 1^k 0^k$, che è di lunghezza maggiore di k ed appartiene ad L_2 perché il primo terzo della parola è uguale al secondo 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: $z = 1^{k-q-p}1^k0^k$. Consideriamo l'esponente $i = 0$: la parola xy^0z ha la forma

  $$xy^0z = xz = 1^q1^{k-q-p}1^k0^k = 1^{k-p}1^k0^k$$

  Poiché $p > 0$, la sequenza iniziale di 1 è più corta di $2k$, e quindi la parola iterata xy^0z non appartiene ad L_2 perché il primo terzo della parola è fatta solamente da 1 mentre il secondo 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 il linguaggio

$$censor(L) = \{\#^{|w|} \mid w \in L\}$$

è un linguaggio context-free.

Soluzione: Se L è un linguaggio context-free, allora esiste una grammatica $G = (V, \Sigma, R, S)$ che lo genera. Possiamo assumere che questa grammatica sia in forma normale di Chomsky. Per dimostrare che $censor(L)$ 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 contiene solo #: $\Sigma' = \{\#\}$.
- L'insieme di variabili è lo stesso della grammatica G: $V' = V$.
- Il nuovo insieme di regole R' è ottenuto rimpiazzando ogni regola nella forma $A \to b$, con b simbolo terminale, con la regola $A \to \#$, e lasciando invariate le regole nella forma $A \to BC$ e la regola $S \to \varepsilon$ (se presente).
- La variabile iniziale rimane la stessa: $S' = S$.

Data una derivazione $S \Rightarrow^* w$ della grammatica G possiamo costruire una derivazione nella nuova grammatica G' che applica le stesse regole nello stesso ordine, e che deriva una parola dove ogni simbolo terminale di w è rimpiazzato da #. Quindi, G' permette di derivare tutte le parole in $censor(L)$. Viceversa, data una derivazione $S \Rightarrow^* \#^n$ della nuova grammatica G' possiamo costruire una derivazione nella grammatica G che applica le stesse regole nello stesso ordine, e che deriva una parola dove ogni # è rimpiazzato da qualche simbolo terminale in Σ. Quindi, G' permette di derivare solo parole che appartengono a $censor(L)$.