Automi e Linguaggi Formali
Soluzioni Appello del 8/7/2022

Gabriel Rovesti

Anno Accademico 2024-2025

Esercizio 1 (12 punti)

Se L è un linguaggio sull'alfabeto $\{0,1\}$, la *rotazione a sinistra* di L è l'insieme delle stringhe

$$\mathrm{ROL}(L) = \{wa \mid aw \in L, w \in \{0,1\}^*, a \in \{0,1\}\}.$$

Per esempio, se $L = \{0, 01, 010, 10100\}$, allora $\mathrm{ROL}(L) = \{0, 10, 100, 01001\}$. Dimostra che se L è regolare allora anche $\mathrm{ROL}(L)$ è regolare.

Soluzione

Dimostriamo che la classe dei linguaggi regolari è chiusa rispetto all'operazione di rotazione a sinistra.

Teorema 1. *Se L è un linguaggio regolare sull'alfabeto $\{0,1\}$, allora $ROL(L)$ è anch'esso un linguaggio regolare.*

Proof. Dato che L è regolare, esiste un DFA $A = (Q, \Sigma, \delta, q_0, F)$ che lo riconosce, dove:

- Q è l'insieme finito degli stati
- $\Sigma = \{0,1\}$ è l'alfabeto
- $\delta : Q \times \Sigma \to Q$ è la funzione di transizione
- $q_0 \in Q$ è lo stato iniziale
- $F \subseteq Q$ è l'insieme degli stati finali

Costruiamo un NFA $A' = (Q', \Sigma, \delta', q_0', F')$ che riconosce $\mathrm{ROL}(L)$ come segue:

- $Q' = Q \times \{0,1\} \cup \{q_0'\}$, dove aggiungiamo un nuovo stato iniziale q_0' e manteniamo traccia dell'ultimo simbolo letto in ogni stato
- $\Sigma = \{0,1\}$ è lo stesso alfabeto
- Lo stato iniziale è il nuovo stato q_0'

- L'insieme degli stati finali è $F' = \{(q, a) \mid \delta(q_0, a) \in Q \text{ e } q \in F\}$
- La funzione di transizione δ' è definita come segue:
  1. $\delta'(q_0', a) = \{(q_0, a)\}$ per ogni $a \in \{0, 1\}$ (transizioni iniziali per memorizzare il primo simbolo)
  2. $\delta'((q, b), a) = \{(\delta(q, a), b)\}$ per ogni $q \in Q$, $a, b \in \{0, 1\}$ (transizioni normali che mantengono traccia del primo simbolo)

Dimostriamo che $L(A') = \text{ROL}(L)$.

Parte 1: Dimostriamo che se $y \in \text{ROL}(L)$, allora $y \in L(A')$.

Se $y \in \text{ROL}(L)$, allora $y = wa$ per qualche $w \in \{0, 1\}^*$ e $a \in \{0, 1\}$ tali che $aw \in L$. Ciò significa che esiste una computazione nell'automa A che accetta aw:

$$q_0 \xrightarrow{a} q_1 \xrightarrow{w} q_f$$

dove $q_f \in F$.

Nell'automa A', possiamo costruire la seguente computazione per $y = wa$:

$$q_0' \xrightarrow{w[1]} (q_0, w[1]) \xrightarrow{w[2]} (q_1, w[1]) \xrightarrow{w[3]} \ldots \xrightarrow{a} (q_f, w[1])$$

Poiché $\delta(q_0, a) = q_1$ e $q_f \in F$, abbiamo $(q_f, w[1]) \in F'$. Quindi, y è accettata da A'.

Parte 2: Dimostriamo che se $y \in L(A')$, allora $y \in \text{ROL}(L)$.

Se $y \in L(A')$, allora esiste una computazione in A' che accetta y:

$$q_0' \xrightarrow{b} (q_0, b) \xrightarrow{y[2]} (q_1, b) \xrightarrow{y[3]} \ldots \xrightarrow{y[n]} (q_f, b)$$

dove $(q_f, b) \in F'$, che significa che $\delta(q_0, b) \in Q$ e $q_f \in F$.

Sia $y = wb$ dove $w = y[2]y[3] \ldots y[n-1]$. Dall'automa A, possiamo costruire la seguente computazione per bw:

$$q_0 \xrightarrow{b} q_1 \xrightarrow{w} q_f$$

Dato che $q_f \in F$, abbiamo $bw \in L$. Pertanto, $y = wb \in \text{ROL}(L)$.

Avendo dimostrato che $L(A') = \text{ROL}(L)$ e che A' è un NFA, possiamo concludere che $\text{ROL}(L)$ è regolare. □

Esercizio 2 (12 punti)

Considera l'alfabeto $\Sigma = \{0, 1\}$, e sia L_2 l'insieme di tutte le stringhe che contengono almeno un 1 nella loro prima metà:

$$L_2 = \{uv \mid u \in \Sigma^* 1 \Sigma^*, v \in \Sigma^* \text{ e } |u| \leq |v|\}.$$

Dimostra che L_2 non è regolare.

Soluzione

Dimostriamo che L_2 non è un linguaggio regolare utilizzando il Pumping Lemma per linguaggi regolari.

Teorema 2. *Il linguaggio* $L_2 = \{uv \mid u \in \Sigma^*1\Sigma^*, v \in \Sigma^* \text{ e } |u| \leq |v|\}$ *non è regolare.*

Proof. Assumiamo per assurdo che L_2 sia regolare. Allora, per il Pumping Lemma, esiste una costante $p > 0$ tale che ogni stringa $s \in L_2$ con $|s| \geq p$ può essere scritta come $s = xyz$ con le seguenti proprietà:

1. $|xy| \leq p$
2. $|y| > 0$
3. Per ogni $i \geq 0$, $xy^iz \in L_2$

Consideriamo la stringa $s = 10^{2p-1} \in L_2$. Questa stringa appartiene a L_2 perché contiene un 1 nella prima posizione (quindi nella prima metà) e la lunghezza della prima metà è p, che è minore o uguale alla lunghezza della seconda metà (p).

Per il Pumping Lemma, s può essere scritta come $s = xyz$ con le proprietà sopra elencate. Dato che $|xy| \leq p$, la sottostringa xy è contenuta interamente nel prefisso 10^{p-1} della stringa s. Abbiamo due casi possibili:

Caso 1: Se y contiene il simbolo 1 (cioè, $y = 10^k$ per qualche $k \geq 0$), allora consideriamo $xy^0z = xz$. In questo caso, la stringa risultante non contiene alcun 1 nella prima metà, quindi $xz \notin L_2$. Questo contraddice il Pumping Lemma.

Caso 2: Se y contiene solo simboli 0 (cioè, $y = 0^k$ per qualche $k > 0$), allora consideriamo xy^2z. In questo caso, la stringa risultante è della forma 10^{2p-1+k}, che ha una lunghezza totale di $2p+k$. La prima metà di questa stringa ha lunghezza $p + \lfloor \frac{k}{2} \rfloor$, mentre la seconda metà ha lunghezza $p + \lceil \frac{k}{2} \rceil$. Se k è dispari, l'unico simbolo 1 si trova all'inizio della stringa, e non è nella prima metà (poiché la prima metà inizia dall'indice 0 e termina all'indice $p + \lfloor \frac{k}{2} \rfloor - 1$). Quindi, $xy^2z \notin L_2$, contraddicendo nuovamente il Pumping Lemma.

In entrambi i casi, otteniamo una contraddizione con il Pumping Lemma. Pertanto, L_2 non può essere regolare. □

Esercizio 3 (12 punti)

Mostra che per ogni PDA P esiste un PDA P_2 con due soli stati tale che $L(P_2) = L(P)$.
Suggerimento: usate la pila per tenere traccia dello stato di P.

Soluzione

Dimostriamo che ogni linguaggio accettato da un PDA può essere accettato da un PDA con soli due stati.

Teorema 3. *Per ogni PDA P, esiste un PDA P_2 con esattamente due stati tale che $L(P_2) = L(P)$.*

Proof. Sia $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ un PDA arbitrario, dove:

- Q è l'insieme finito degli stati
- Σ è l'alfabeto di input

- Γ è l'alfabeto della pila
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma^*)$ è la funzione di transizione
- $q_0 \in Q$ è lo stato iniziale
- $Z_0 \in \Gamma$ è il simbolo iniziale della pila
- $F \subseteq Q$ è l'insieme degli stati finali

Costruiamo un nuovo PDA $P_2 = (Q_2, \Sigma, \Gamma_2, \delta_2, q_0', Z_0', F_2)$ con due soli stati, dove:

- $Q_2 = \{q_0', q_f'\}$ (un stato iniziale e uno stato finale)
- $\Gamma_2 = \Gamma \cup Q$ (usiamo i simboli degli stati originali come simboli di pila)
- $Z_0' = q_0 Z_0$ (il simbolo iniziale della pila include lo stato iniziale di P)
- $F_2 = \{q_f'\}$ (un solo stato finale)

La funzione di transizione δ_2 è definita come segue:
1. Per ogni transizione $(p, \gamma) \in \delta(q, a, X)$ in P, aggiungiamo una transizione in P_2:

$$\delta_2(q_0', a, qX) \ni (q_0', p\gamma)$$

Questa transizione simula la transizione originale di P, aggiornando sia lo stato (memorizzato in cima alla pila) che il contenuto della pila.

2. Per ogni stato $q \in F$ di P, aggiungiamo una transizione che permette a P_2 di passare allo stato finale quando in cima alla pila è presente uno stato finale di P:

$$\delta_2(q_0', \varepsilon, q) \ni (q_f', \varepsilon)$$

3. Per garantire che P_2 accetti le stesse stringhe di P, aggiungiamo anche transizioni che permettono di "scavare" nella pila per trovare il simbolo di stato:

$$\delta_2(q_0', \varepsilon, X) \ni (q_0', \varepsilon) \quad \text{per ogni } X \in \Gamma$$

Queste transizioni permettono di ignorare temporaneamente i simboli di pila di Γ per accedere al simbolo di stato.

Osservazione 1. Questa costruzione assume che P accetti per stato finale. Se P accetta per pila vuota, la costruzione deve essere leggermente modificata.

Dimostriamo ora che $L(P_2) = L(P)$.

Parte 1: Dimostriamo che se $w \in L(P)$, allora $w \in L(P_2)$.

Se $w \in L(P)$, allora esiste una computazione di P che, partendo dalla configurazione iniziale (q_0, w, Z_0), termina in una configurazione $(q_f, \varepsilon, \gamma)$ dove $q_f \in F$ e $\gamma \in \Gamma^*$.

Possiamo costruire una computazione corrispondente in P_2 che simula passo per passo la computazione di P. Inizialmente, P_2 è nella configurazione $(q_0', w, q_0 Z_0)$. Ad ogni passo, P_2 esegue una transizione che corrisponde alla transizione di P, mantenendo in cima alla pila lo stato corrente di P.

Alla fine, quando P raggiunge uno stato finale q_f, P_2 ha q_f in cima alla pila e può eseguire la transizione $\delta_2(q_0', \varepsilon, q_f) \ni (q_f', \varepsilon)$ per passare allo stato finale q_f'. Quindi, $w \in L(P_2)$.

Parte 2: Dimostriamo che se $w \in L(P_2)$, allora $w \in L(P)$.

Se $w \in L(P_2)$, allora esiste una computazione di P_2 che, partendo dalla configurazione iniziale $(q_0', w, q_0 Z_0)$, termina in una configurazione $(q_f', \varepsilon, \gamma')$ dove $\gamma' \in \Gamma_2^*$.

Per raggiungere lo stato q_f', P_2 deve eseguire una transizione $\delta_2(q_0', \varepsilon, q_f) \ni (q_f', \varepsilon)$ dove $q_f \in F$. Questo significa che, prima di questa transizione, P_2 aveva q_f in cima alla pila.

La sequenza di transizioni che ha portato P_2 ad avere q_f in cima alla pila corrisponde a una sequenza valida di transizioni in P che porta P dallo stato iniziale q_0 allo stato finale q_f. Quindi, $w \in L(P)$.

Abbiamo dimostrato che $L(P_2) = L(P)$, completando così la prova. □

Osservazione 2. La costruzione sopra descritta funziona per PDA che accettano per stato finale. Per PDA che accettano per pila vuota, la costruzione è simile, ma invece di tenere traccia degli stati finali, P_2 deve simulare il comportamento di P fino a quando la pila originale di P diventa vuota.