Automi e Linguaggi Formali - Q&A

Gabriel Rovesti

Anno Accademico 2024-2025

Esercizio DROP-OUT

Sia A un linguaggio qualsiasi. Definiamo $DROP\text{-}OUT(A)$ come il linguaggio contenente tutte le stringhe che possono essere ottenute rimuovendo un simbolo da una stringa in A. Formalmente, $DROP\text{-}OUT(A) = \{xz \mid xyz \in A \text{ dove } x, z \in \Sigma^*, y \in \Sigma\}$. Dimostra che la classe dei linguaggi regolari è chiusa rispetto all'operazione $DROP\text{-}OUT$.

Soluzione

Teorema 1. *Se A è un linguaggio regolare, allora anche $DROP\text{-}OUT(A)$ è un linguaggio regolare.*

Proof. Dato che A è regolare, esiste un automa a stati finiti deterministico (DFA) $M = (Q, \Sigma, \delta, q_0, F)$ che lo riconosce, dove:

- Q è l'insieme finito degli stati
- Σ 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 automa a stati finiti non deterministico (NFA) $M' = (Q', \Sigma, \delta', q_0', F')$ che riconosce $DROP\text{-}OUT(A)$ come segue:

- $Q' = Q \times Q \times \{0, 1\}$, dove la terza componente è un flag che indica se è stato saltato un simbolo (0 = non saltato, 1 = saltato)
- Σ è lo stesso alfabeto
- $q_0' = (q_0, q_0, 0)$ è lo stato iniziale
- $F' = \{(p, q, 1) \mid q \in F\}$ sono gli stati finali

La funzione di transizione δ' è definita come segue:

1. **Transizioni normali (senza salto):** Per ogni $p \in Q$, $a \in \Sigma$:

$$\delta'((p, q, 0), a) = \{(\delta(p, a), q, 0)\}$$

Questo mantiene il flag a 0 e aggiorna solo il primo componente di stato secondo la transizione originale.

2. **Transizioni con salto:** Per ogni $p, q \in Q$, $a \in \Sigma$:

$$\delta'((p, q, 0), a) \cup = \{(p, \delta(q, a), 1)\}$$

Questa transizione simula il "salto" del simbolo a modificando la seconda componente di stato e impostando il flag a 1.

3. **Transizioni dopo il salto:** Per ogni $p, q \in Q$, $a \in \Sigma$:

$$\delta'((p, q, 1), a) = \{(p, \delta(q, a), 1)\}$$

Dopo che è avvenuto il salto, aggiorniamo solo la seconda componente di stato.

Dimostriamo che $L(M') = DROP\text{-}OUT(A)$.

Parte 1: Dimostriamo che se $w \in DROP\text{-}OUT(A)$, allora $w \in L(M')$.

Se $w \in DROP\text{-}OUT(A)$, allora esiste una stringa $xyz \in A$ con $x, z \in \Sigma^*$, $y \in \Sigma$ tale che $w = xz$. Poiché $xyz \in A$, l'automa M accetta xyz, quindi esiste una sequenza di stati $r_0, r_1, \ldots, r_{|x|+1}, \ldots, r_{|x|+|y|+|z|}$ con $r_0 = q_0$, $r_{|x|+|y|+|z|} \in F$, e per ogni i, $r_{i+1} = \delta(r_i, (xyz)[i+1])$.

Nell'automa M', possiamo costruire una computazione accettante per $w = xz$ come segue:

- Iniziamo nello stato $(q_0, q_0, 0)$
- Leggiamo i primi $|x|$ simboli di w usando le transizioni di tipo 1, raggiungendo lo stato $(r_{|x|}, q_0, 0)$
- Saltiamo il prossimo simbolo (che sarebbe y in xyz) usando una transizione di tipo 2, passando allo stato $(r_{|x|}, r_1, 1)$
- Leggiamo i rimanenti $|z|$ simboli di w usando le transizioni di tipo 3, raggiungendo lo stato $(r_{|x|}, r_{|x|+1+|z|}, 1)$

Poiché $r_{|x|+1+|z|} = r_{|x|+|y|+|z|} \in F$, lo stato finale $(r_{|x|}, r_{|x|+|y|+|z|}, 1) \in F'$. Quindi $w \in L(M')$.

Parte 2: Dimostriamo che se $w \in L(M')$, allora $w \in DROP\text{-}OUT(A)$.

Se $w \in L(M')$, allora esiste una computazione accettante in M' che termina in uno stato $(p, q, 1)$ con $q \in F$. Questa computazione deve includere esattamente una transizione di tipo 2 (l'unico modo per impostare il flag a 1).

Sia i l'indice in cui avviene la transizione di tipo 2. Sia $x = w[1 : i]$ (i primi i simboli di w) e $z = w[i + 1 : |w|]$ (i rimanenti simboli di w). Sia $a \in \Sigma$ il simbolo che viene "saltato" dalla transizione di tipo 2.

Dalla definizione delle transizioni, sappiamo che esiste una computazione in M che accetta la stringa xaz, quindi $xaz \in A$. Per definizione, questo significa che $w = xz \in DROP\text{-}OUT(A)$.

Abbiamo così dimostrato che $L(M') = DROP\text{-}OUT(A)$. Poiché M' è un NFA e ogni NFA può essere convertito in un DFA equivalente, $DROP\text{-}OUT(A)$ è un linguaggio regolare. □

Osservazione 1. Questa costruzione utilizza un NFA con al più $|Q|^2 \cdot 2$ stati, dove $|Q|$ è il numero di stati dell'automa originale. La componente di flag $\{0, 1\}$ è essenziale per garantire che avvenga esattamente un "salto" durante l'elaborazione della stringa.

Dimostrazione per costruzione grafica:

Per illustrare la costruzione, possiamo visualizzare un'idea intuitiva: l'NFA M' mantiene due "tracce" parallele dell'automa originale M:

- La prima traccia (primo componente dello stato) simula M sul prefisso già letto
- La seconda traccia (secondo componente dello stato) simula M sul prefisso letto, dopo aver saltato un simbolo

Inizialmente, entrambe le tracce sono sincronizzate. Quando decidiamo di saltare un simbolo, la prima traccia rimane ferma mentre la seconda procede. Dopo il salto, solo la seconda traccia continua ad avanzare. La stringa è accettata se, dopo aver saltato esattamente un simbolo, la seconda traccia raggiunge uno stato finale dell'automa originale.

Questa costruzione garantisce che $DROP\text{-}OUT(A)$ sia regolare quando A è regolare, dimostrando la chiusura della classe dei linguaggi regolari rispetto all'operazione $DROP\text{-}OUT$.