Sui Linguaggi Regolari: Teorema di Kleene - Pumping Lemma

N.Fanizzi - V.Carofiglio

6 aprile 2016

Teorema di Kleene

Vale la seguente equivalenza:

$$\mathcal{L}_3 \equiv \mathcal{L}_{FSL} \equiv \mathcal{L}_{REG}$$

Dimostrazione.
Lo schema di dimostrazione è il seguente:

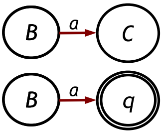
1. $\mathcal{L}_3 \subset \mathcal{L}_{FSL}$ (ed anche $\mathcal{L}_{FSL} \subset \mathcal{L}_3$)
2. $\mathcal{L}_{FSL} \subset \mathcal{L}_{REG}$
3. $\mathcal{L}_{REG} \subset \mathcal{L}_3$

Tesi I. $\mathcal{L}_3 \subset \mathcal{L}_{FSL}$
Sia $L \in \mathcal{L}_3$ cioè $\exists G = (X, V, S, P), G$ di tipo 3: $L(G) = L$
Si deve costruire un FSA $M = (Q, \delta, q_0, F)$ tale che $T(M) = L(G)$

Algoritmo.
Input: $G = (X, V, S, P), G$ di tipo 3
Output: $M = (Q, \delta, q_0, F) \in FSA$

---

1. X alfabeto di ingresso per M
2. $Q = V \cup \{q\}, \quad q \notin V$
3. $q_0 = S$
4. $F = \{q\} \cup \{B \mid B \longrightarrow \lambda \in P\}$
5. $\delta : Q \times X \longrightarrow \wp(Q)$
   1. $\forall B \longrightarrow aC \in P$: $C \in \delta(B, a)$
   2. $\forall B \longrightarrow a \in P$: $q \in \delta(B, a)$



Osservazione:
l'algoritmo può generare NDA (passi 5.a e 5.b)
Occorre dimostrare anche la correttezza dell'automa:
$L(G) = T(M)$

- $L(G) \subseteq T(M)$: sia $w = x_1 x_2 \cdots x_k \in L(G)$
  w può essere generata con una derivazione:
  $S \Longrightarrow x_1 X_2 \Longrightarrow x_1 x_2 X_3 \Longrightarrow \cdots \Longrightarrow x_1 x_2 \dots X_k \Longrightarrow x_1 x_2 \dots x_k$
  Per la sua def. l'automa M, avendo in input w, compie una serie di transizioni che portano da S a $X_1, X_2, \dots, X_k$ fino a q.
- $T(M) \subseteq L(G)$: analogamente

Inoltre, si dimostra in maniera costruttiva

$$\mathcal{L}_{FSL} \subset \mathcal{L}_3$$

Algoritmo.
Input: $M = (Q, \delta, q_0, F) \in FSA$
Output: $G = (X, V, S, P)$ lineare

---

1. X alfabeto di ingresso per M
2. $V = Q$
3. $S = q_0$
4. $P = \{q \longrightarrow xq' \mid q' \in \delta(q, x)\} \cup \{q \longrightarrow x \mid \delta(q, x) \in F\}$

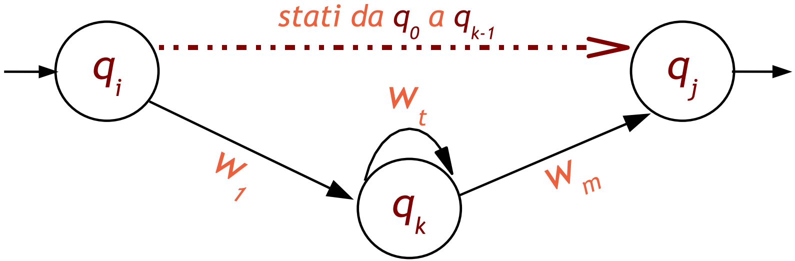
Per esercizio: $L(G) = T(M)$

Tesi II. $\mathcal{L}_{FSL} \subset \mathcal{L}_{REG}$
Sia $L \in \mathcal{L}_{FSL}$ cioè $\exists M = (Q, \delta, q_0, F) \in FSA$ tale che: $T(M) = L$
Supponiamo $Q = \{q_0, q_1, \dots, q_n\}$
Si definisca il linguaggio:

$$R_{ij} = \{w \in X^* \mid \delta^*(q_i, w) = q_j\}$$

contenente le stringhe che fanno transitare M da q_i a q_j.
Per def. di linguaggio accettato da FSA, risulta: $T(M) = \bigcup_{q_j \in F} R_{0j}$
quindi basta dimostrare che ogni linguaggio R_{ij} è regolare
$0 \leq i, j \leq n$

$$R_{ij}^k = \{w \in X^* \mid \delta^*(q_i, w) = q_j \textit{ senza transitare in } q_k, q_{k+1}, \ldots, q_n\}$$



Si

osservi che: $R_{ij}^{n+1} = R_{ij}$

dimostriamo per induzione su k che

$$R_{ij}^k \in \mathcal{L}_{REG} \qquad \forall i,j : 0 \leq i,j \leq n$$

$(k = 0)$ $R_{ij}^0 = \{w \in X^* \mid \delta(q_i, w) = q_j\} \in \mathcal{L}_{REG}$ perchè finito

$(k > 0)$ per ipotesi: $R_{ij}^k \in \mathcal{L}_{REG} \quad \forall i,j :\in \{0,\dots,n\}$
Dimostriamo che $R_{ij}^{k+1} \in \mathcal{L}_{REG}$
Sia $w \in R_{ij}^{k+1}$ per definizione la lettura di w non fa transitare M in nessuno degli stati $q_{k+1}, q_{k+2}, \dots, q_n$.
Si possono avere 2 casi (vedi figura):

1. w non fa transitare M nemmeno in q_k, quindi $w \in R_{ij}^k$ ed R_{ij}^k cioè regolare, per ipotesi.
2. w fa transitare M in q_k. In tal caso riscriviamo w come concatenazione di $m > 1$ sottostringhe:
   $w = w_1 w_2 \cdots w_{m-1} w_m$ con:
   $w_1 \in R_{ik}^k \qquad w_t \in R_{kk}^k \quad 1 < t < m \qquad w_m \in R_{kj}^k$

Data la genericità di w si può scrivere:
$w \in R_{ik}^k \cdot (R_{kk}^k)^{m-2} \cdot R_{kj}^k$ (con $m > 1$) per cui:
$w \in R_{ik}^k \cdot (R_{kk}^k)^* \cdot R_{kj}^k$
Ne consegue allora che: $R_{ij}^{k+1} \subseteq R_{ij}^k \cup R_{ik}^k \cdot (R_{kk}^k)^* \cdot R_{kj}^k$
ma ovviamente: $R_{ij}^k \cup R_{ik}^k \cdot (R_{kk}^k)^* \cdot R_{kj}^k \subseteq R_{ij}^{k+1}$
quindi il linguaggio

$$R_{ij}^{k+1} = R_{ij}^k \cup R_{ik}^k \cdot (R_{kk}^k)^* \cdot R_{kj}^k$$

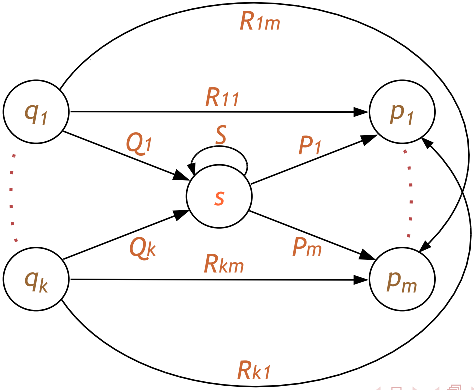
è regolare perchè espresso come unione, concatenazione e iterazione di linguaggi che sono regolari (per ipotesi induttiva)
Risulta dimostrato che $\forall k \in [0, n]$ $R_{ij}^k \in \mathcal{L}_{REG}$ perciò

$$T(M) = \bigcup_{q_j \in F} R_{0j} = \bigcup_{q_j \in F} R_{0j}^{n+1}$$

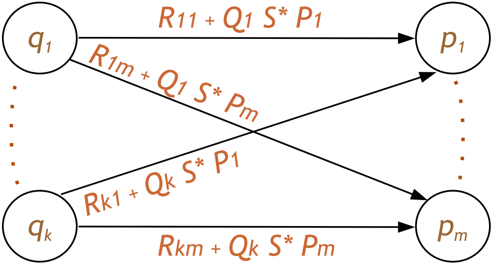
è regolare perchè unione di linguaggi regolari

Algoritmo alternativo [Hopcroft et al.] $\mathcal{L}_{FSL} \subset \mathcal{L}_{REG}$

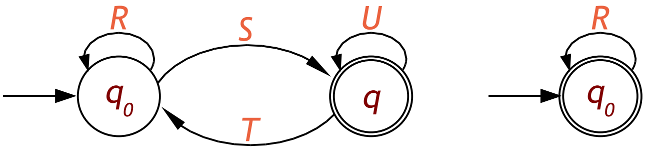
- eliminando uno stato alla volta occorre preservare i cammini che portano dallo stato iniziale a stati finali
- si considerano gli stati predecessori $q_1, \ldots, q_m$ e successori $p_1, \ldots, p_k$ dello stato s da eliminare (ins. non disgiunti)

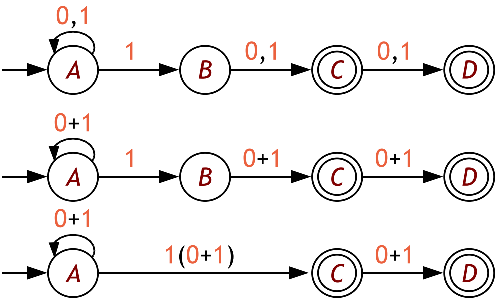


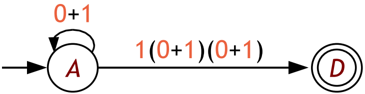
- archi etichettati con espressioni regolari anzichè con simboli: infinite parole possono portare da uno stato ad un altro
- eliminando s, per ogni (q_i, p_j), si etichetta l'arco con l'espressione regolare: $R_{ij} + Q_i S^* P_j$
- un arco assente nell'automa originario si denota con l'espressione $\emptyset$



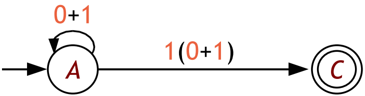
1. **per ogni** stato finale $q \in F$
   si eliminano tutti gli stati tranne q_0 e q:
   applicando l'algoritmo si produce un automa con archi etichettati da espr. regolari
2. **se** $q \neq q_0$ **allora** risulta un automa a due stati
   $\Rightarrow$ soluzione: $(R^* + SU^*T)^*SU^*$
3. **altrimenti** risulta un automa ad un stato
   $\Rightarrow$ soluzione: R^*
4. **output:** somma di tutte le espressioni ottenute al variare di q







$R_{AD} = (0+1)^*1(0+1)(0+1)$



$R_{AC} = (0+1)^*1(0+1)$

$$R = (0+1)^*1(0+1)(0+1) + (0+1)^*1(0+1)$$

Tesi III. $\mathcal{L}_{REG} \subset \mathcal{L}_3$
Sia $L \in \mathcal{L}_{REG}$ quindi vale una delle seguenti condizioni:

- L è finito
- $L = L_1 \cup L_2$ con L_1, L_2 regolari
- $L = L_1 \cdot L_2$ con L_1, L_2 regolari
- $L = (L_1)^*$ con L_1 regolare

Dimostrazione *per induzione sulla costruzione di L*

base L finito $L = \{w_1, w_2, \dots, w_n\}$ allora si può scrivere come unione di linguaggi lineari L_i che generano ognuno una stringa di w_i e la classe $\mathcal{L}_3$ è chiusa rispetto all'unione:

$$L = \bigcup_{i=0}^{n} L_i$$

Si può facilmente dimostrare che $\forall i \in [0, n] : \; L_i \in \mathcal{L}_3$

passo In tutti i tre casi possiamo considerare i linguaggi L_1 e L_2 come lineari per ipotesi di induzione.
Anche la loro unione/concatenazione/iterazione è in $\mathcal{L}_3$ per la chiusura di $\mathcal{L}_3$ rispetto a queste operazioni.
Quindi $L \in \mathcal{L}_3$

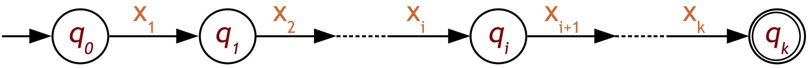
Pumping Lemma per Linguaggi Regolari

Teorema. Sia $M = (Q, \delta, q_0, F) \in$ FSA con $n = |Q|$ e sia $z \in T(M)$, $|z| \geq n$. Allora $z = uvw$ e $uv^*w \subset T(M)$, cioè $\forall t \geq 0\colon\ uv^tw \in T(M)$

Dimostrazione.

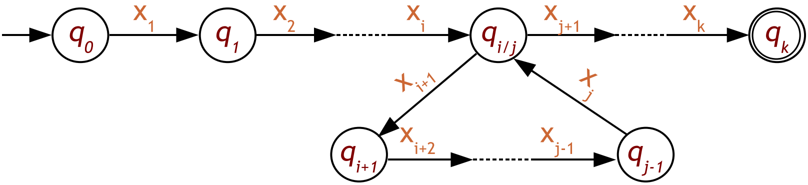
Sia $z = x_1x_2 \cdots x_k \in T(M)$ con $k \geq n$

Si rappresenta il riconoscimento di z con:



Se si ha $|z| \geq n$ si deve passare per almeno $n+1$ stati ma $n = |Q|$ quindi c'è uno stato ripetuto:

$$\exists i, j,\ 0 \leq i < j \leq k\colon\ q_i = q_j$$



Si osservi che z si può scrivere come uvw ove:

- $u = x_1 x_2 \cdots x_i$
- $v = x_{i+1} x_{i+2} \cdots x_j$
- $w = x_{j+1} x_{j+2} \cdots x_k$

Avendo in ingresso $z \in T(M)$, M si porta nello stato: $\delta^*(q_0, z) \in F$ ma questo è lo stesso stato in cui si giunge tramite le stringhe: $uvvw$, $uvvvw$ etc.,...(ciclando t volte tra q_i e q_j)
Dunque: $\forall t \geq 0: \; uv^t w \in T(M)$

Esercizi

1. Determinare la grammatica di tipo 3 che genera il linguaggio descritto da $b^* + (ab)^*$
2. Data la grammatica lineare $G = (X, V, S, P)$ con $X = \{a, b, c\}$, $V = \{S, A, B\}$ e
   $$P = \{S \longrightarrow bA \mid aS \mid b,\ A \longrightarrow aB \mid cS \mid a,$$
   $$B \longrightarrow bA \mid cB \mid c\}$$
   determinare un'espressione regolare per $L(G)$
3. Sia $L = S(R)$ ove $R = (aa + aaa)^*$
   - costruire un automa che riconosce L
   - trasformare l'NDA del punto 1. in FSA
4. Determinare una grammatica G di tipo 3 tale che:
   $$L(G) = \{w \in \{a, b\}^* \mid w \text{ ha un numero pari di } a \text{ e dispari di } b\}$$

5. Sia $L = S(R)$ ove $R = ab(bb)^*c$
   - trovare un automa (NDA) per riconoscere L
   - trasformare l'automa NDA nell'FSA equivalente
6. Data la grammatica lineare $G = (X, V, S, P)$ con
   $X = \{a, b\}$, $V = \{S, B\}$ e
   $P = \{S \longrightarrow aB \quad B \longrightarrow aB \mid bS \mid a\}$
   determinare un automa FSA M tale che: $T(M) = L(G)$
7. Determinare una grammatica lineare G tale che:

$$L(G) = \{w \in \{a, b\}^* \mid w \neq \alpha aa\beta, \ \alpha, \beta \in \{a, b\}^*\}$$

8. Dimostrare che non sono regolari i linguaggi:
   - $L_1 = \{a^k b^k \mid k > 0\}$
   - $L_2 = \{a^n b^m c^k \mid n > k, \ n, m, k > 0\}$
9. Dimostrare che il linguaggio non è regolare:

$$L = \{a^n b^m c^k \mid m > k, \ n, m, k > 0\}$$

Esercizio 1. Determinare la grammatica di tipo 3 che genera il linguaggio descritto da $b^* + (ab)^*$

Vediamo qual é il linguaggio corrispondente:

$S(b^* + (ab)^*) = S(b^*) \cup S((ab)^*) = (S(b))^* \cup (S(ab))^* =$
$\{b\}^* \cup (S(a) \cdot S(b))^* = \{b\}^* \cup (\{a\} \cdot \{b\})^* = \{b\}^* \cup \{ab\}^*$

$G_1 = (X_1, V_1, S_1, P_1)$ con $X_1 = \{b\}, \quad V_1 = \{S_1\},$
$P_1 = \{S_1 \longrightarrow bS_1 \mid \lambda\}$

$G_2 = (X_2, V_2, S_2, P_2)$ con $X_2 = \{a, b\} \quad V_2 = \{S_2, B_2\},$
$P_2 = \{S_2 \longrightarrow aB_2, \; B_2 \longrightarrow b\}$

$G_3 = (X_3, V_3, S_3, P_3)$ con $X_3 = X_2, \quad V_3 = V_2 \cup \{S_3\}$

$$
\begin{aligned}
P_3 = & \{S_3 \longrightarrow \lambda\} \cup (P_2 \setminus \{S_2 \longrightarrow \lambda\}) \cup \\
& \cup \{S_2 \longrightarrow w \in P_2\} \cup \{A \longrightarrow xS_3 \mid A \longrightarrow x \in P_2\} = \\
& = \{S_3 \longrightarrow \lambda\} \cup P_2 \cup \{S_3 \longrightarrow aB_2\} \cup \{B_2 \longrightarrow bS_3\} = \\
& = \{S_3 \longrightarrow \lambda, \; S_2 \longrightarrow aB_2, \; B_2 \longrightarrow b, \; S_3 \longrightarrow aB_2, \; B_2 \longrightarrow bS_3\} = \\
& = \{S_3 \longrightarrow \lambda \mid aB_2, \; S_2 \longrightarrow aB_2, \; B_2 \longrightarrow b \mid bS_3\}
\end{aligned}
$$

Osservazione: S_2 è inutile:
$P_3 = \{S_3 \longrightarrow \lambda \mid aB_2,\ B_2 \longrightarrow b \mid bS_3\}$
Quindi $L(G_3) = \{ab\}^*$

Per chiudere:
$G = (X, V, S, P)$ ove:
$X = X_1 \cup X_3$
$V = V_1 \cup V_3 \cup \{S\} = \{S, S_1, B_2, S_3\}$

$$\begin{aligned} P = & \{S \longrightarrow w \mid S_1 \longrightarrow w \in P_1\} \cup \\ & \{S \longrightarrow w \mid S_3 \longrightarrow w \in P_3\} \cup \\ & P_1 \cup P_3 = \\ = & \{S \longrightarrow bS_1 \mid aB_2 \mid \lambda, \\ & S_1 \longrightarrow bS_1 \mid \lambda,\ S_3 \longrightarrow aB_2 \mid \lambda,\ B_2 \longrightarrow bS_3 \mid b\} \end{aligned}$$

Esercizio 3. Sia $L = S(R)$ ove $R = (aa + aaa)^*$

1. costruire un automa che riconosce L
2. trasformare l'NDA del punto 1. in FSA

---

1. Dato $X = \{a\}$, determiniamo le grammatiche G_1 per $L_1 = \{aa\}$ e G_2 per $L_2 = \{aaa\}$:

$G_1 = (X, V_1, S_1, P_1)$ con $V_1 = \{S_1, A\}$ e

$P_1 = \{S_1 \longrightarrow aA, \quad A \longrightarrow a\}$

$G_2 = (X, V_2, S_2, P_2)$ con $V_2 = \{S_2, B, C\}$ e

$P_2 = \{S_2 \longrightarrow aB, \quad B \longrightarrow aC, \quad C \longrightarrow a\}$

Sia $G_3 = (X, V_3, S_3, P_3)$ la grammatica per $L_3 = L_1 \cup L_2$:

con $V_3 = V_1 \cup V_2 \cup \{S_3\} = \{S_3, S_1, S_2, A, B, C\}$ e

$$\begin{aligned} P_3 = & \{S_3 \longrightarrow w \mid S_1 \longrightarrow w \in P_1\} \cup \\ & \{S_3 \longrightarrow w \mid S_2 \longrightarrow w \in P_2\} \cup P_1 \cup P_2 \\ = & \{S_3 \longrightarrow aA \mid aB\} \cup P_1 \cup P_2 \end{aligned}$$

che contiene produzioni inutili e quindi NT superflui (S_1, S_2)

Per l'iterazione di L_3 costruisco:

$G = (X, V, S, P)$ con

$V = V_3 \cup \{S\} = \{S, S_3, A, B, C\}$

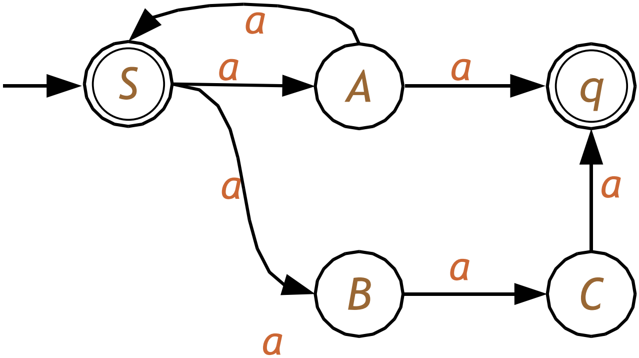
$$
\begin{aligned}
P = & \{S \longrightarrow \lambda\} \cup (P_3 \setminus \{S_3 \longrightarrow \lambda\}) \cup \\
& \cup \{S \longrightarrow w \mid S_3 \longrightarrow w \in P_3\} \cup \\
& \cup \{N \longrightarrow bS \mid N \longrightarrow b \in P_3\} \\
& \cup \{N \longrightarrow bS \mid N \longrightarrow bM, b \neq \lambda, M \longrightarrow \lambda \in P_3\} = \\
= & \{S \longrightarrow \lambda\} \cup \{S_3 \longrightarrow aA \mid aB, A \longrightarrow a, B \longrightarrow aC, C \longrightarrow a\} \cup \\
& \cup \{S \longrightarrow aA \mid aB\} \cup \\
& \cup \{A \longrightarrow aS, C \longrightarrow aS\} \\
= & \{S \longrightarrow aA \mid aB \mid \lambda,\ A \longrightarrow aS \mid a,\ B \longrightarrow aC,\ C \longrightarrow aS \mid a\}
\end{aligned}
$$

2. L'automa a stati finiti si ottiene mediante l'algoritmo dato nella dimostrazione del teorema di Kleene:

- $Q = V \cup \{q\} = \{S, A, B, C, q\}$
- $q_0 = S$
- $F = \{q, S\}$
- δ definita dalla matrice di transizione (trasposta):

δ	S	A	B	C	q
a	$\{A, B\}$	$\{S, q\}$	$\{C\}$	$\{S, q\}$	$\emptyset$

graficamente:



per esercizio: trasformare l'automa in un FSA deterministico

Esercizio 8. Dimostrare che $L_1 = \{a^k b^k \mid k > 0\}$ non è regolare

Supponendo che $L \in \mathcal{L}_{REG}$, per il teorema di Kleene sarà anche $L \in \mathcal{L}_{FSL}$ quindi $\exists M = (Q, \delta, q_0, F)$, tale che $T(M) = L$
Sia $|Q| = n$ e si consideri $w = a^n b^n \in L$.
Leggendo le a di w, M deve passare per almeno $n + 1$ stati, quindi ci devono essere almeno 2 stati coincidenti: q_i e q_j con $i < j$
Quindi nel cammino da q_0 ad uno stato finale c'è un ciclo di lunghezza $j - i$. Allora si può ciclare un numero arbitrario di volte, aggiungendo ogni volta una sottostringa a^{j-i}
Per il Pumping lemma dovrebbe essere

$$a^{n+k(j-i)} b^n \in T(M) \quad \forall k \geq 0$$

Ma queste non sono stringhe di L (assurdo)
pertanto L non è regolare