Proprieta' dei Linguaggi Regolari

- **Pumping Lemma.** Ogni linguaggio regolare soddisfa una proprieta' ben nota, il pumping lemma. Questa tecnica fornisce uno strumento utile per dimostrare che un linguaggio non e' regolare.
- **Proprieta' di chiusura.** E' interessante investigare proprieta' di chiusura dei linguaggi regolari rispetto ad una serie di operazioni, tipo unione, intersezione, complemento. Lo studio di queste proprieta' fornisce tecniche per comporre automi che riconoscono linguaggi piu' semplici rispetto a tali operazioni. In pratica, questi strumenti possono servire per ragionare su automi complessi in modo composizionale.

- **Proprieta' di decisione.** Analisi computazionale di automi, cioe' quando due automi sono equivalenti.
- **Tecniche di minimizzazione.** Utilizzando la capacita' di ragionare sull'equivalenza di automi e' possibile sviluppare algoritmi di miniminizzazione. Questo problema riveste una particolare rilevanza pratica in quanto permette di derivare automi equivalenti con il minimo numero di stati.

Teorema 4.1.
Sia L un linguaggio regolare.
Allora $\exists n, \forall w \in L : |w| \geq n \Rightarrow w = xyz$ tale che:

1. $y \neq \epsilon$
2. $|xy| \leq n$
3. $\forall k \geq 0,\ xy^k z \in L$

Intuitivamente

- Esiste una costante n dipendente dal linguaggio L tale che tutte le stringhe di lunghezza $\geq n$ possono essere scomposte in un dato modo
- E' sempre possibile scegliere una stringa *non vuota* y da replicare, ovvero cancellare o ripetere k volte, pur rimanendo all'interno del linguaggio L

Prova

Se L e' regolare e' riconosciuto da un DFA A tale che $L(A) = L$

$$A = (Q, \Sigma, \delta, q_0, F)$$

Supponiamo che A abbia n stati, quindi

$$Q = \{q_0, \ldots, q_{n-1}\}$$

Prendiamo come costante il valore n, e consideriamo una generica stringa $w \in L$ piu' lunga di n. Quindi avremo $w = a_1 a_2 \ldots a_m$ dove $m >= n$.

Prova

Chiamiamo p_i, per $i \in \{0, \ldots, m\}$, lo stato in cui si trova l'automa A dopo avere esaminato $a_1 a_2 \cdots a_i$ a partire dallo stato iniziale q_0. Formalmente, utilizzando la funzione di transizione estesa

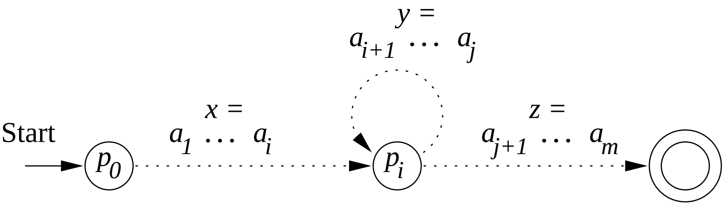
- $p_0 = \hat{\delta}(q_0, \epsilon) = q_0$;
- $p_i = \hat{\delta}(q_0, a_1 a_2 \cdots a_i)$.

Dato che ci sono solo n stati distinti $\Rightarrow \exists i < j : p_i = p_j$

Ora $w = xyz$, dove

1. $x = a_1 a_2 \cdots a_i$
2. $y = a_{i+1} a_{i+2} \cdots a_j$
3. $z = a_{j+1} a_{j+2} \ldots a_m$

L'automa deve essere fatto nel seguente modo,



Notiamo che

1. $y \neq \epsilon$ la stringa y non e' vuota, dato che $i < j$
2. $|xy| \leq n$ dato che gli stati $p_0, \ldots, p_{j-1}$ sono tutti distinti (basta considerare il minimo indice i che si ripete)

Data la forma dell'automa e' chiaro che, eseguendo $k >= 0$ cicli in p_i, l'automa accetta ogni stringa xy^kz. Quindi, per $k >= 0$, abbiamo $xy^kz \in L(A)$.

Esempio: Applicazioni del Pumping Lemma

Sia $L_{01} = \{0^n 1^n : n \geq 1\}$ il linguaggio delle stringhe formate da un certo numero di 0, seguiti dallo stesso numero di 1.
Se L_{01} fosse regolare, allora varrebbe il P.L. Sia $w = 0^n 1^n \in L$ la stringa per n, la costante del P.L.
Per il P.L., $w = xyz$, $|xy| \leq n$, $y \neq \epsilon$ e $xy^k z \in L_{01}$.
Procediamo per casi sulla forma della stringa non vuota y,

- $y = 0^h 1^j$ e' chiaro che ripetendo la stringa k volte, gli 0 e gli 1 vengono mescolati; quindi la stringa ottenuta non sta nel linguaggio
- $y = 0^h$ e' formata solo da 0 (o analogamente $y = 1^h$ e' formata solo da 1),

$$w = \underbrace{000\cdots}_{x}\underbrace{\cdots 0}_{y}\underbrace{0111\cdots 11}_{z}$$

Se consideriamo xz ha meno 0 che 1 e non sta nel linguaggio

Supponiamo che $L_{pr} = \{1^p : p \text{ e' primo } \}$ sia regolare.
Sia n dato dal pumping lemma.
Scegliamo un numero primo $p \geq n + 2$.

$$w = \overbrace{\underbrace{111\cdots\cdots}_{x}\underbrace{\cdots 1}_{\substack{y \\ |y|=m}}\underbrace{1111\cdots 11}_{z}}^{p}$$

Ora $xy^{p-m}z \in L_{pr}$
$|xy^{p-m}z| = |xz| + (p - m)|y| =$
$p - m + (p - m)m = (1 + m)(p - m)$
che non e' primo a meno che uno dei fattori non sia 1.

- $y \neq \epsilon \Rightarrow 1 + m > 1$
- $m = |y| \leq |xy| \leq n, \quad p \geq n + 2$

$\Rightarrow p - m \geq n + 2 - n = 2.$

- pumping lemma: permette di dimostrare in modo formale che un linguaggio non e' regolare
- per trattare classi di linguaggi piu' interessanti (che includono per esempio i linguaggi di programmazione) dovremo considerare meccanismi piu' potenti, come le grammatiche libere dal contesto
- bisogna considerare classi di automi piu' potenti in grado di memorizzare almeno in parte i simboli letti

Siano L e M due linguaggi regolari. Allora i seguenti linguaggi sono regolari:

- *Unione:* $L \cup M$
- *Intersezione:* $L \cap M$
- *Complemento:* $\overline{N}$
- *Differenza:* $L \setminus M$
- *Inversione:* $L^R = \{w^R : w \in L\}$
- *Chiusura:* L^*
- *Concatenazione:* $L.M$

Teorema 4.4. Siano L e M lingaggi regolari. Allora $L \cup M, L.M, L^*$ sono regolari.
Prova. Dato che L ed M sono regolari allora esistono espressioni regolari E ed F, tali che $L = L(E)$ e $M = L(F)$. Allora $L(E + F) = L \cup M$, $L(E.F) = L.M$ e $L(E^*) = L^*$ per definizione.

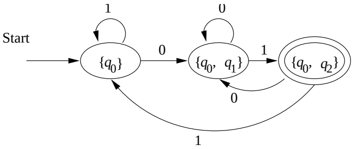
Teorema 4.5. Se L e' un linguaggio regolare su Σ, allora che $\overline{L} = \Sigma^* \setminus L$ e' regolare.
Prova. Sia L riconosciuto da un DFA

$$A = (Q, \Sigma, \delta, q_0, F).$$

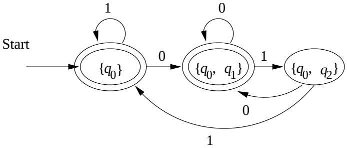
Sia $B = (Q, \Sigma, \delta, q_0, Q \setminus F)$. Allora $L(B) = \overline{L}$.
Infatti, $w \in \hat{\delta}(q_0, w) \in L(B)$ sse $\hat{\delta}(q_0, w) \in Q - F$ sse $w \notin L(A)$.
Nota: $\hat{\delta}(q_0, w)$ e' sempre definita perche' l'automa e' deterministico.

Esempio

Sia L riconosciuto dal DFA qui sotto:



Allora $\overline{L}$ e' riconosciuto da:



L e' il linguaggio delle stringhe che terminano per 01, mentre $\overline{L}$ e' il linguaggio delle stringhe che terminano per 0 o per 11, o sono 1, o sono vuote

Teorema 4.8. Se L e M sono regolari, allora anche $L \cap M$ e' regolare.

Prova. Per la legge di DeMorgan, $L \cap M = \overline{\overline{L} \cup \overline{M}}$. Sappiamo gia' che i linguaggi regolari sono chiusi sotto il complemento e l'unione.

Daremo anche una prova diretta, basata sulla costruzione dell'automa *Prodotto*.

Automa Prodotto

Sia L il linguaggio di

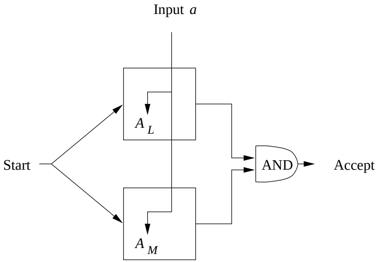
$$A_L = (Q_L, \Sigma, \delta_L, q_L, F_L)$$

e M il linguaggio di

$$A_M = (Q_M, \Sigma, \delta_M, q_M, F_M)$$

Assumiamo senza perdita di generalita' che entrambi gli automi siano deterministici.
Assumiamo senza perdita di generalita' che gli alfabeti siano uguali (basta considerare l'unione)
Costruiremo un automa che simula A_L e A_M in parallelo, e accetta se e solo se sia A_L che A_M accettano.

Se A_L va dallo stato p allo stato s leggendo a, e A_M va dallo stato q allo stato t leggendo a, allora $A_{L \cap M}$ andra' dallo stato (p, q) allo stato (s, t) leggendo a.



$$A_{L\cap M} = (Q_L \times Q_M, \Sigma, \delta_{L\cap M}, (q_L, q_M), F_L \times F_M),$$

dove

$$\delta_{L\cap M}((p, q), a) = (\delta_L(p, a), \delta_M(q, a))$$

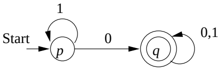
Per induzione su $|w|$ abbiamo

$$\hat{\delta}_{L\cap M}((q_L, q_M), w) = \left(\hat{\delta}_L(q_L, w), \hat{\delta}_M(q_M, w)\right)$$

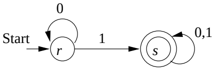
Dato che gli stati finali del prodotto sono coppie di stati finali abbiamo $L(A_{L\cap M}) = L(A_L) \cap L(A_M)$.

Esempio

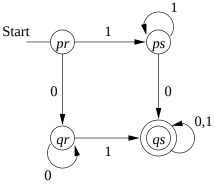
$(c) = (a) \times (b)$



(a)



(b)



(c)

Esempio

1. (a) definisce il linguaggio $1^*0(0+1)^*$, deve esserci almeno uno 0 preceduto da zero o piu' 1
2. (b) definisce il linguaggio $0^*1(0+1)^*$, deve esserci almeno uno 1 preceduto da zero o piu' 0
3. (c) definisce il linguaggio $11^*0(0+1)^* + 00^*1(0+1)^*$, le stringhe iniziano per 1 e contengono 10 o iniziano per 0 e contengono 01

Chisura rispetto alla differenza

Teorema 4.10. Se L e M sono linguaggi regolari, allora anche $L \setminus M$ e' regolare.
Prova. Osserviamo che $L \setminus M = L \cap \overline{M}$. Sappiamo gia' che i linguaggi regolari sono chiusi sotto il complemento e l'intersezione.

Reverse di un linguaggio

- Data una stringa $w = a_1 \dots a_n$ indichiamo con w^R l'inversione di w,

$$w^R = a_n \dots a_1$$

- L'inversione di un linguaggio L e' quindi

$$L^R = \{w^R \mid w \in L\}$$

Esempio: $L = \{\epsilon, 111, 011\}$ allora

$$L^R = \{\epsilon, 111, 110\}$$

Chiusura rispetto al reverse

Teorema 4.11. Se L e' un linguaggio regolare, allora anche L^R e' regolare.

Prova 1: Sia L riconosciuto da un FA A. Modifichiamo A per renderlo un FA per L^R:

1. Giriamo tutti gli archi.
2. Rendiamo lo stato iniziale di A l'unico stato finale.
3. Creiamo un nuovo stato iniziale p_0, con $\delta(p_0, \epsilon) = F$ (gli stati finali di A).

Teorema 4.11. Se L e' un linguaggio regolare, allora anche L^R e' regolare.

Prova 2: Sia L descritto da un'espressione regolare E, $L(E) = L$. Costruiremo un'espressione regolare E^R, tale che $L(E^R) = (L(E))^R$.

Procediamo per induzione strutturale su E.

Base: Se E e' $\boldsymbol{\epsilon}$, $\boldsymbol{\emptyset}$, o $\mathbf{a}$, allora $E^R = E$.

Abbiamo $L((E))^R = L(E)$ in tutti e tre i casi. Infatti, $\{\epsilon\}^R = \{\epsilon\}$, $\emptyset^R = \emptyset$ e $\{a\}^R = \{a\}$.

Abbiamo tre casi:

1. Se $E = F + G$, abbiamo $L(E)^R = L(F)^R + L(G)^R$ (l'inversione dell'unione e' l'unione delle inversioni). Prendiamo quindi $E^R = F^R + G^R$, dove $L(F^R) = L(F)^R$ e $L(G^R) = L(G)^R$ per ipotesi induttiva.
2. Se $E = F.G$, allora $L(E)^R = L(G)^R.L(F)^R$ (ogni stringa che sta nell'inversione di $L(E)$ e' formata da una stringa che sta nell'inversione di G seguita da una stringa che sta nell'inversione di F). Prendiamo quindi $E^R = G^R.F^R$, dove $L(F^R) = L(F)^R$ e $L(G^R) = L(G)^R$ per ipotesi induttiva.

Se $E = F^*$, allora $L(E)^R = (L(F)^R)^*$. Notiamo infatti che una stringa w appartiene ad E sse la sua inversione w^R appartiene a $(L(F)^R)^*$.
Formalmente, $w^R \in L(E)^R$ sse $w \in L(E)$ e $w = x_1. \ldots .x_n$, dove $x_i \in L(F)$. Di conseguenza, $w^R = x_n^R. \ldots .x_1^R$, dove $x_i \in L(F)^R$. Il viceversa e' analogo.

Allora prendiamo $E^R = (F^R)^*$, dove $L(F^R) = L(F)^R$ per ipotesi induttiva