Espressioni regolari

Espressioni regolari

- Un FA (NFA o DFA) è un metodo per costruire una macchina che riconosce linguaggi regolari.
- Una *espressione regolare* è un modo dichiarativo per descrivere un linguaggio regolare.
- Esempio: $\mathbf{01}^* + \mathbf{10}^*$
- Le espressioni regolari sono usate, ad esempio, in
  - comandi UNIX (`grep`)
  - strumenti per l'analisi lessicale di UNIX (Lex (Lexical analyzer generator) e Flex (Fast Lex)).

Operazioni sui linguaggi

- **Unione:**
  $L \cup M = \{w : w \in L \text{ o } w \in M\}$
- **Concatenazione:**
  $L.M = \{w : w = xy, x \in L, y \in M\}$
- **Potenze:**
  $L^0 = \{\epsilon\},\ L^1 = L,\ L^{k+1} = L.L^k$
- **Chiusura di Kleene:**
  $L^* = \bigcup_{i=0}^{\infty} L^i$

Definizione induttiva di espressioni regolari

- **Base:**
  - ϵ e $\emptyset$ sono espressioni regolari.
    $L(\epsilon) = \{\epsilon\}$ e $L(\emptyset) = \emptyset$.
  - Se $a \in \Sigma$, allora **a** è un'espressione regolare.
    $L(\mathbf{a}) = \{a\}$.
- **Induzione:**
  - Se E è un'espressione regolare, allora (E) è un'espressione regolare. $L((E)) = L(E)$.
  - Se E e F sono espressioni regolari, allora $E + F$ è un'espressione regolare. $L(E + F) = L(E) \cup L(F)$.
  - Se E e F sono espressioni regolari, allora $E.F$ è un'espressione regolare. $L(E.F) = L(E).L(F)$.
  - Se E è un'espressione regolare, allora $E^\star$ è un'espressione regolare. $L(E^\star) = (L(E))^*$.

Esempio

Espressione regolare per
$L = \{w \in \{0,1\}^* : \text{ 0 e 1 alternati in } w\}$

$$(\mathbf{01})^* + (\mathbf{10})^* + \mathbf{0}(\mathbf{10})^* + \mathbf{1}(\mathbf{01})^*$$

o, equivalentemente,

$$(\epsilon + \mathbf{1})(\mathbf{01})^*(\epsilon + \mathbf{0})$$

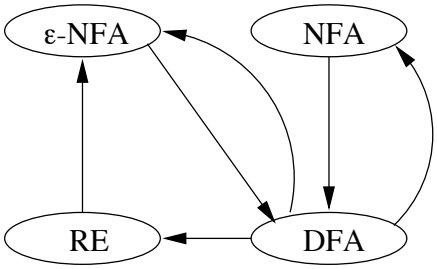
Ordine di precedenza per gli operatori

1. Chiusura
2. Concatenazione (punto)
3. Più ($+$)

Esempio: $\mathbf{01}^* + \mathbf{1}$ è raggruppato in $(\mathbf{0}(\mathbf{1})^*) + \mathbf{1}$

Equivalenza di FA e espressioni regolari

Abbiamo già mostrato che DFA, NFA, e ϵ-NFA sono tutti equivalenti.

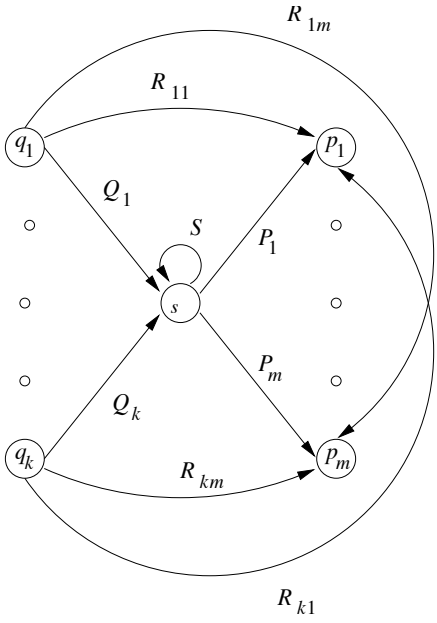


Per mostrare che gli FA sono equivalenti alle espressioni regolari, mostreremo che

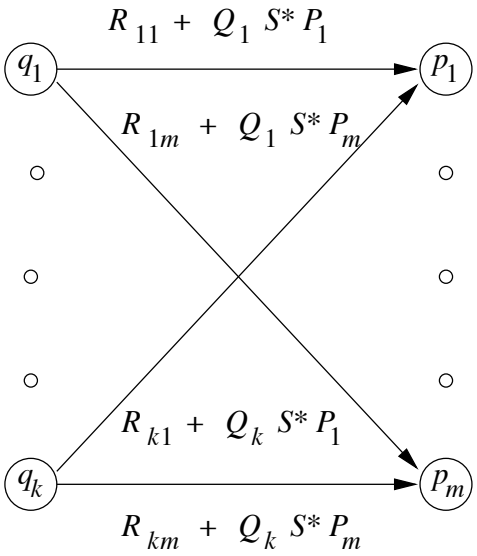
1. Per ogni DFA A possiamo trovare (costruire, in questo caso) un'espressione regolare R, tale che $L(R) = L(A)$.
2. Per ogni espressione regolare R esiste un ϵ-NFA A, tale che $L(A) = L(R)$.

La tecnica di eliminazione di stati

Etichettiamo gli archi con espressioni regolari di simboli

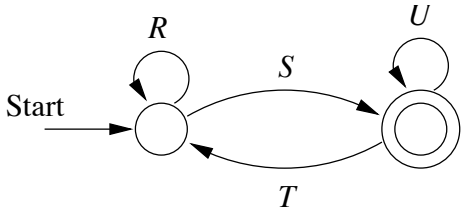


Ora eliminiamo lo stato s.

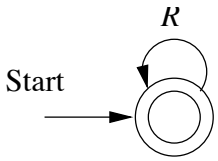


Per lo stato accettante q eliminiamo dall'automa originale tutti gli stati eccetto q_0 e q.

Per ogni $q \in F$ saremo rimasti con A_q della forma



che corrisponde all'espressione regolare $E_q = (R + SU^*T)^*SU^*$, se $q_0 \neq q$, o con A_q della forma



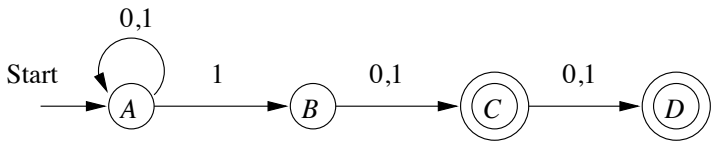
che corrisponde all'espressione regolare $E_q = R^*$, se $q_0 \in F$.

- L'espressione finale è

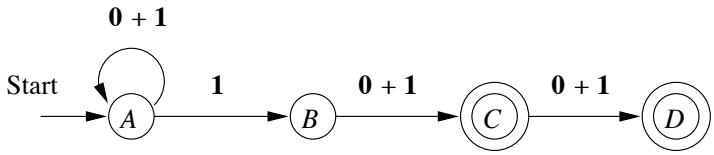
$$\bigoplus_{q \in F} E_q$$

Esempio

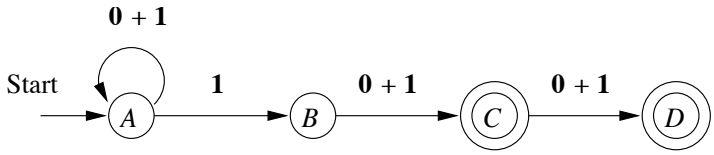
$\mathcal{A}$, dove $L(\mathcal{A}) = \{W : w = x1b,$ o $w = x1bc, \quad x \in \{0,1\}^*, \{b,c\} \subseteq \{0,1\}\}$



La trasformiamo in un automa con espressioni regolari come etichette

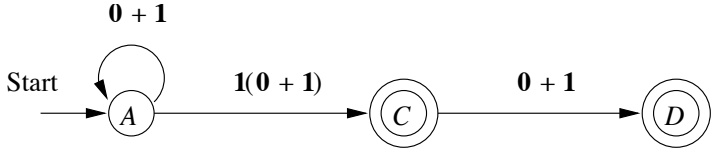


Esempio



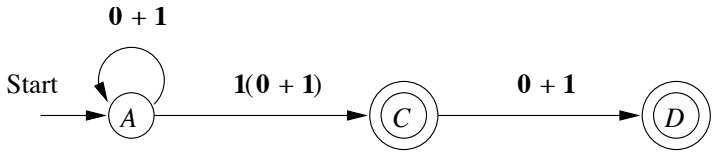
Eliminiamo lo stato B. L'espressione sul nuovo arco che va da A a C è $R_{11} + Q_1 S^* P_1$, dove $R_{11} = \emptyset$, $Q_1 = \mathbf{1}$, S $= \emptyset$, $P_1 = \mathbf{0} + \mathbf{1}$. Considerando che $\emptyset^* = \epsilon$, l'espressione è

$$\emptyset + \mathbf{1}\emptyset^*(\mathbf{0} + \mathbf{1}) = \mathbf{1}\emptyset^*(\mathbf{0} + \mathbf{1}) = \mathbf{1}(\mathbf{0} + \mathbf{1})$$

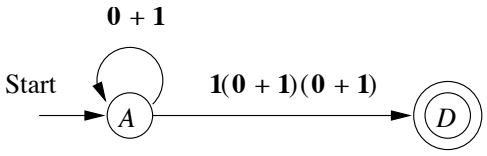


Esempio

Da



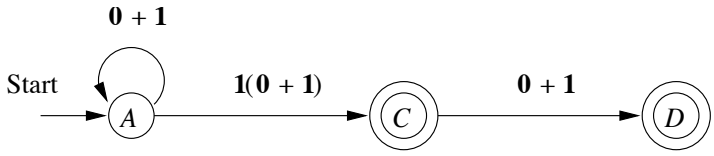
possiamo eliminare lo stato C e ottenere $\mathcal{A}_D$



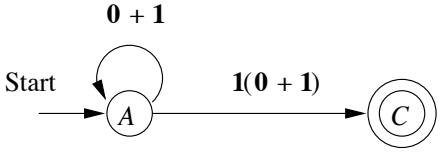
con espressione regolare $(\mathbf{0}+\mathbf{1})^*\mathbf{1}(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})$, dato che $R = \mathbf{0}+\mathbf{1}$, $S = \mathbf{1}(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1})$ e $U = T = \emptyset$ e che quindi $(R + SU^*T)^*SU^* = R^*S$.

Esempio

Da



possiamo eliminare D (e con esso l'arco che va da C a D) e ottenere $\mathcal{A}_C$ con espressione regolare $(\mathbf{0}+\mathbf{1})^*\mathbf{1}(\mathbf{0}+\mathbf{1})$



- L'espressione finale è la somma delle due espressioni regolari precedenti:

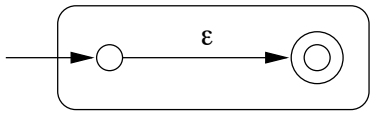
$$(\mathbf{0}+\mathbf{1})^*\mathbf{1}(\mathbf{0}+\mathbf{1})(\mathbf{0}+\mathbf{1}) + (\mathbf{0}+\mathbf{1})^*\mathbf{1}(\mathbf{0}+\mathbf{1})$$

Da espressioni regolari a ϵ-NFA

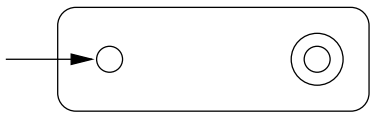
Teorema 3.7: Per ogni espressione regolare R possiamo costruire un ϵ-NFA A, tale che $L(A) = L(R)$.

Dimostrazione: Per induzione strutturale:

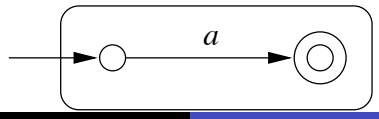
Base: Automa per ϵ, $\emptyset$, e **a**.



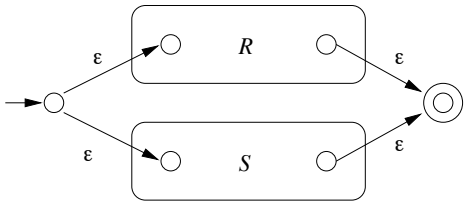
(a)



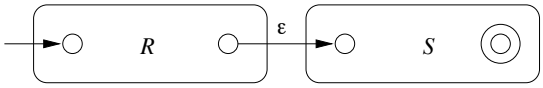
(b)



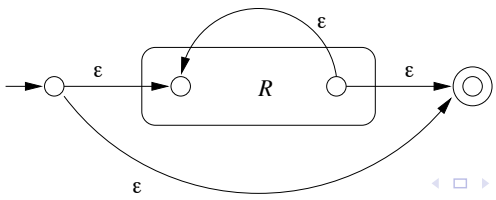
Induzione: Automa per $R + S$, RS, e R^* (quello per (R) coincide con quello per R)



(a)

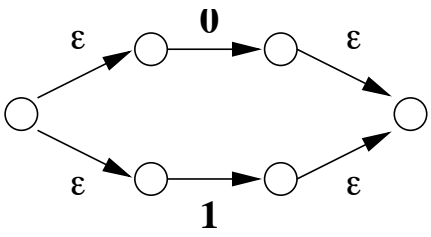


(b)

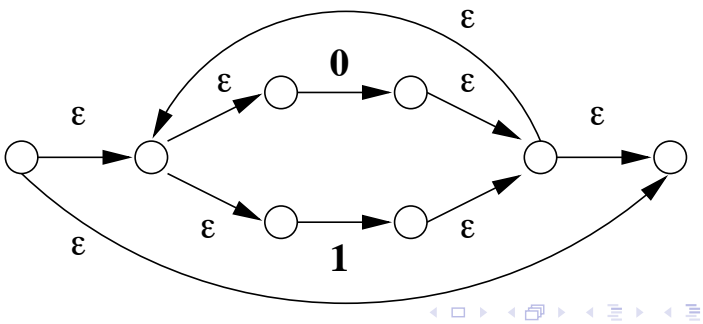


Esempio

Trasformiamo $(\mathbf{0}+\mathbf{1})^*\mathbf{1}(\mathbf{0}+\mathbf{1})$

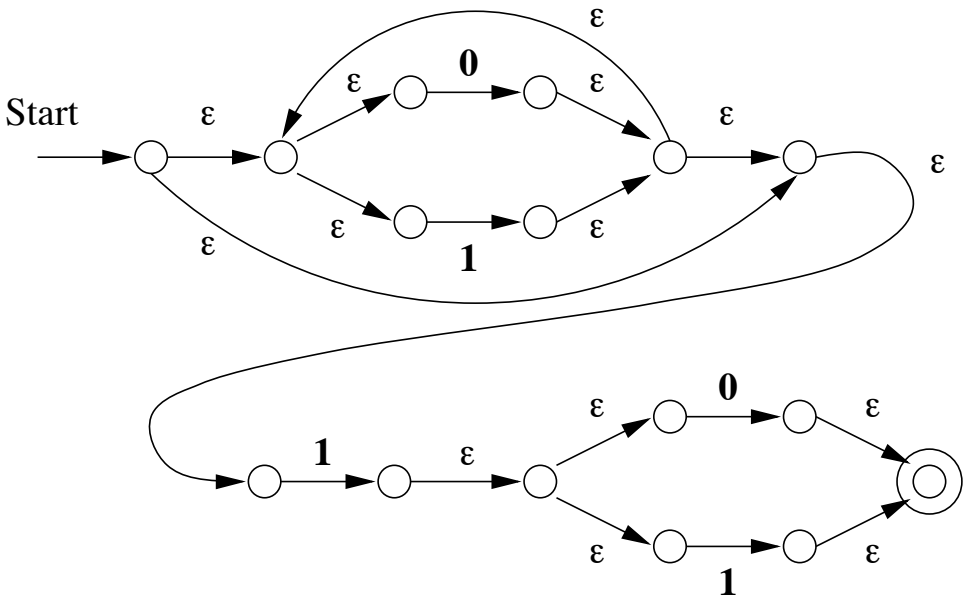


(a)



(b)

Esempio



(c)

Leggi algebriche per i linguaggi

- $L \cup M = M \cup L$.
  L'unione è *commutativa*.
- $(L \cup M) \cup N = L \cup (M \cup N)$.
  L'unione è *associativa*.
- $(LM)N = L(MN)$.
  La concatenazione è *associativa*.

Nota: La concatenazione non è commutativa, *cioè*, esistono L e M tali che $LM \neq ML$.

- $\emptyset \cup L = L \cup \emptyset = L$.
  $\emptyset$ è *l'identità* per l'unione.
- $\{\epsilon\}L = L\{\epsilon\} = L$.
  $\{\epsilon\}$ è l'identità *sinistra* e *destra* per la concatenazione.
- $\emptyset L = L\emptyset = \emptyset$.
  $\emptyset$ è l'annichilatore *sinistro* e *destro* per la concatenazione.

- $L(M \cup N) = LM \cup LN$.
  La concatenazione è *distributiva a sinistra* sull'unione.
- $(M \cup N)L = ML \cup NL$.
  La concatenazione è *distributiva a destra* sull'unione.
- $L \cup L = L$.
  L'unione è *idempotente*.
- $\emptyset^* = \{\epsilon\},\ \{\epsilon\}^* = \{\epsilon\}$.
- $L^+ = LL^* = L^*L,\ L^* = L^+ \cup \{\epsilon\}$

- $(L^*)^* = L^*$. La chiusura è *idempotente*.

Dimostrazione:

$$w \in (L^*)^* \iff w \in \bigcup_{i=0}^{\infty} \left(\bigcup_{j=0}^{\infty} L^j \right)^i$$

$$\iff \exists k, m \in \mathrm{N} : w \in (L^m)^k$$

$$\iff \exists p \in \mathrm{N} : w \in L^p$$

$$\iff w \in \bigcup_{i=0}^{\infty} L^i$$

$$\iff w \in L^*$$

Espressioni Regolari in UNIX

Le espressioni regolari fanno parte del sistema operativo UNIX fin dall'inizio e sono usate in vari comandi che elaborano testi (ad esempio `grep`, `sed`, `lex`, `vi`, `awk`, `ad`, `ex`, `pg`) per:

- elencare/eliminare righe che contengono un'espressione regolare
- sostituire un'espressione regolare (find/replace)
- ...

La sintassi è leggermente diversa da quella vista finora.

Espressioni Regolari in UNIX

- insiemi di caratteri: pattern elementari che specificano la presenza un carattere appartenente ad un certo insieme.
- ancore: legano il pattern a comparire una posizione specifica della riga (es. inizio, fine).
- gruppi: "racchiudono" l'espressione regolare che può quindi essere riferita come una singola entità.
- modificatori: specificano ripetizioni dell'espressione che precede il modificatore stesso.

Espressioni Regolari in UNIX: sintassi

- `[abc]` match any of the characters enclosed
- `[a-d]` match any character in the enclosed range
- `[set]` match any character not in the following set
- `.` match any single character except `<newline>`
- `[:alpha:]` some predefined character sets
- `[:digit:]`
- treat the next char literally. Normally used to escape special characters such as "." and "*"