Automi e Linguaggi Formali

Parte 3 – Espressioni Regolari

Università degli Studi di Padova

Davide Bresolin
Ultimo aggiornamento: 16 marzo 2022

Sommario

Operazioni sui linguaggi

- **Unione:**

$$L \cup M = \{w : w \in L \text{ oppure } w \in M\}$$

- **Intersezione:**

$$L \cap M = \{w : w \in L \text{ e } w \in M\}$$

- **Concatenazione:**

$$L.M = \{uv : u \in L \text{ e } v \in M\}$$

- **Complemento:**

$$\overline{L} = \{w : w \notin L\}$$

- **Chiusura (o Star) di Kleene:**

$$L^* = \{w_1 w_2 \dots w_k : k \geq 0 \text{ e ogni } w_i \in L\}$$

Proprietà di chiusura

Se L e M sono linguaggi regolari, allora anche i seguenti linguaggi sono regolari:

- Unione: $L \cup M$
- Intersezione: $L \cap M$
- Concatenazione: $L.M$
- Complemento: $\overline{L}$
- Chiusura di Kleene: L^*

Chiusura per intersezione

Theorem

Se L e M sono regolari, allora anche $L \cap M$ è un linguaggio regolare.

Chiusura per intersezione

Theorem

Se L e M sono regolari, allora anche $L \cap M$ è un linguaggio regolare.

Dimostrazione. Sia L il linguaggio di

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

e M il linguaggio di

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

Possiamo assumere che entrambi gli automi siano deterministici
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.

Dimostrazione (continua).

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 $A_{L\cap M}$ andrà dallo stato (p, q) allo stato (s, t) leggendo a.

Chiusura per intersezione

Dimostrazione (continua).

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 $A_{L \cap M}$ andrà dallo stato (p, q) allo stato (s, t) leggendo a.

$A_{L \cap M}$ accetta una parola solo quando sia A_L che A_M accettano

$$\Leftrightarrow$$

$A_{L \cap M}$ accetta solo quando (p, q) è una coppia di stati finali

Dimostrazione (continua).

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 $A_{L\cap M}$ andrà dallo stato (p, q) allo stato (s, t) leggendo a.

$A_{L\cap M}$ accetta una parola solo quando sia A_L che A_M accettano

$$\Leftrightarrow$$

$A_{L\cap M}$ accetta solo quando (p, q) è una coppia di stati finali

Formalmente

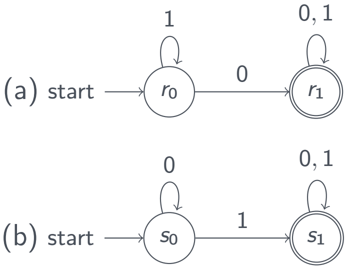
$$A_{L\cap M} = (Q_L \times Q_M, \Sigma, \delta_{L\cap M}, (q_L, q_M), F_L \times F_M),$$

dove

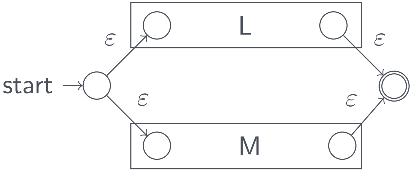
$$\delta_{L\cap M}((p, q), a) = (\delta_L(p, a), \delta_M(q, a))$$

Università degli Studi di Padova

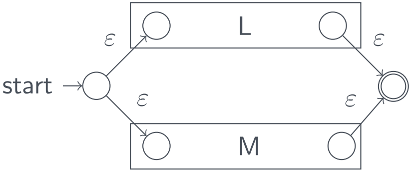
Costruiamo l'automa che rappresenta l'intersezione di (a) e (b)



- $L + M$

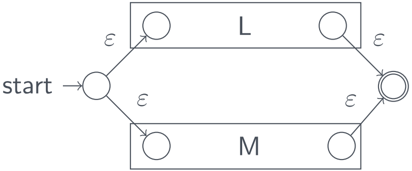


- $L + M$



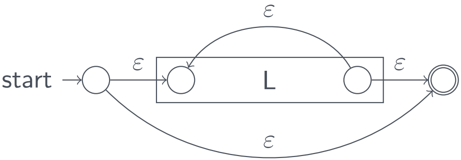
- L.M

- $L + M$



- $L.M$

- L^*



- 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: $01^* + 10^*$
- 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))
  - editor di testo

Espressione regolari: sintassi

Le Espressioni Regolari sono costruite utilizzando

Espressione regolari: sintassi

Le Espressioni Regolari sono costruite utilizzando

- un insieme di costanti di base:
  - ε per la stringa vuota
  - $\emptyset$ per il linguaggio vuoto
  - $\mathtt{a}, \mathtt{b}, \ldots$ per i simboli $a, b, \dots \in \Sigma$

Espressione regolari: sintassi

UNIVERSITÀ DEGLI STUDI DI PADOVA

Le Espressioni Regolari sono costruite utilizzando

- un insieme di costanti di base:
  - ε per la stringa vuota
  - $\emptyset$ per il linguaggio vuoto
  - a, b, . . . per i simboli $a, b, \dots \in \Sigma$
- collegati da operatori:
  - $+$ per l'unione
  - $\cdot$ per la concatenazione
  - $*$ per la chiusura di Kleene
- raggruppati usando le parentesi:
  - ( )

Espressioni regolari: semantica

Se E è un espressione regolare, allora $L(\text{E})$ è il linguaggio rappresentato da E. La definizione di $L(\text{E})$ è induttiva:

- **Caso Base:**
  - $L(\boldsymbol{\varepsilon}) = \{\varepsilon\}$
  - $L(\boldsymbol{\emptyset}) = \emptyset$
  - $L(\mathrm{a}) = \{a\}$

Espressioni regolari: semantica

Se E è un espressione regolare, allora $L(\mathrm{E})$ è il linguaggio rappresentato da E. La definizione di $L(\mathrm{E})$ è induttiva:

- **Caso Base:**
  - $L(\varepsilon) = \{\varepsilon\}$
  - $L(\emptyset) = \emptyset$
  - $L(\mathrm{a}) = \{a\}$
- **Caso induttivo:**
  - $L(\mathrm{E}+\mathrm{F}) = L(\mathrm{E}) \cup L(\mathrm{F})$
  - $L(\mathrm{EF}) = L(\mathrm{E}).L(\mathrm{F})$
  - $L(\mathrm{E}^*) = L(\mathrm{E})^*$
  - $L((\mathrm{E})) = L(\mathrm{E})$

- Scriviamo l'espressione regolare per

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

- Scriviamo l'espressione regolare per

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

$$(01)^* + (10)^* + 1(01)^* + 0(10)^*$$

Espressioni regolari: esempio

- Scriviamo l'espressione regolare per

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

$$(01)^* + (10)^* + 1(01)^* + 0(10)^*$$

oppure

$$(\varepsilon + 1)(01)^*(\varepsilon + 0)$$

Espressioni regolari: precedenza

Come per le espressioni aritmetiche, anche per le espressioni regolari ci sono delle regole di precedenza degli operatori:

1. Chiusura di Kleene
2. Concatenazione (punto)
3. Unione $(+)$

Esempio:

$$01^* + 1 \text{ è raggruppato in } (0(1)^*) + 1$$

e denota un linguaggio diverso da

$$(01)^* + 1$$

Esercizi (1)

Per ognuno dei seguenti linguaggi, costruire una ER sull'alfabeto $\{a, b, c\}$ che li rappresenti:

1. Tutte le stringhe w che contengono un numero pari di a;
2. Tutte le stringhe w che contengono $4k + 1$ occorrenze di b, per ogni $k \geq 0$;
3. Tutte le stringhe la cui lunghezza è un multiplo di 3;

Esercizi (2)

Per ognuno dei seguenti linguaggi, costruire una ER sull'alfabeto $\{0, 1\}$ che li rappresenti:

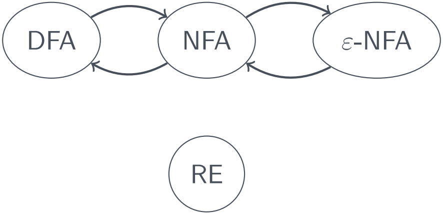
4. Tutte le stringhe w che contengono la sottostringa 101
5. Tutte le stringhe w che **non** contengono la sottostringa 101

Sfida!

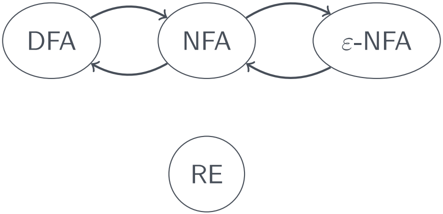
Costruire una ER sull'alfabeto $\{0, 1\}$ per il linguaggio di tutti i numeri binari multipli di 3.

Equivalenza tra FA e RE

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

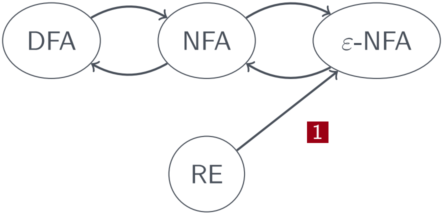


Sappiamo già che DFA, NFA, e ε-NFA sono tutti equivalenti.



Gli FA sono equivalenti alle espressioni regolari:

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

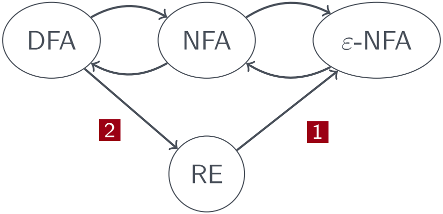


Gli FA sono equivalenti alle espressioni regolari:

1. Per ogni espressione regolare R esiste un ε-NFA A, tale che $L(A) = L(\mathrm{R})$

Equivalenza tra FA e RE

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



Gli FA sono equivalenti alle espressioni regolari:

1. Per ogni espressione regolare R esiste un ε-NFA A, tale che $L(A) = L(\mathrm{R})$
2. Per ogni DFA A possiamo costruire un'espressione regolare R, tale che $L(\mathrm{R}) = L(A)$

Theorem

Per ogni espressione regolare R *possiamo costruire un* ε*-NFA* A *tale che* $L(A) = L(\mathrm{R})$

Theorem

Per ogni espressione regolare R *possiamo costruire un* ε*-NFA* A *tale che* $L(A) = L(\mathrm{R})$

Dimostrazione:

Costruiremo un ε-NFA A con:

- un solo stato finale
- nessuna transizione entrante nello stato iniziale
- nessuna transizione uscente dallo stato finale

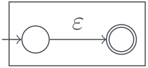
La dimostrazione è per induzione strutturale su R

Caso Base:

Da RE a ε-NFA

Caso Base:

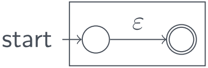
- automa per ε

start 

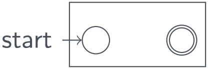
Da RE a ε-NFA

Caso Base:

- automa per ε



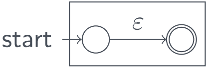
- automa per $\emptyset$



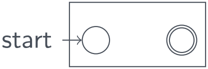
Da RE a ε-NFA

Caso Base:

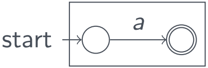
- automa per ε



- automa per $\emptyset$



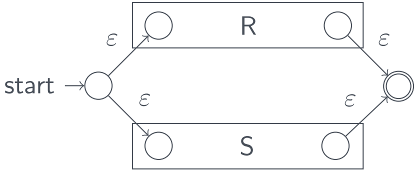
- automa per a



Caso Induttivo:

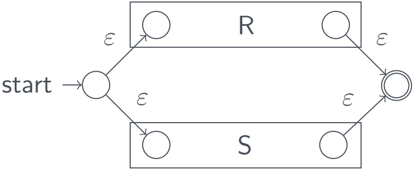
Caso Induttivo:

- automa per $R + S$



Caso Induttivo:

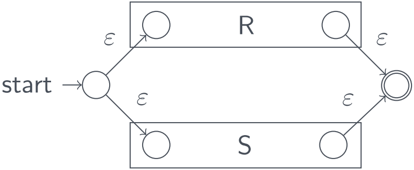
- automa per $R + S$



- automa per RS

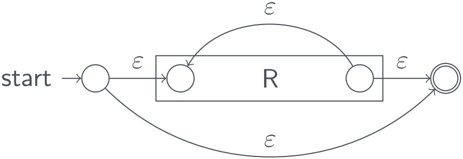
Caso Induttivo:

- automa per $R + S$



- automa per RS

- automa per R^*



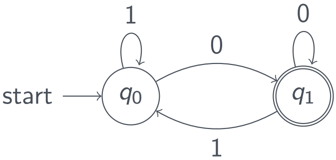
1. Trasformiamo $(0+1)^*1(0+1)$ in ε-NFA
2. Scrivere un'espressione regolare per rappresentare il linguaggio sull'alfabeto $\{a, b, c\}$ che contiene
   - tutte le stringhe che iniziano con a e sono composte solo di a oppure b;
   - la stringa c
3. Trasformare l'espressione regolare dell'esercizio 2 in ε-NFA

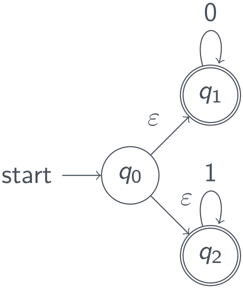
Espressioni regolari: esercizi (2)

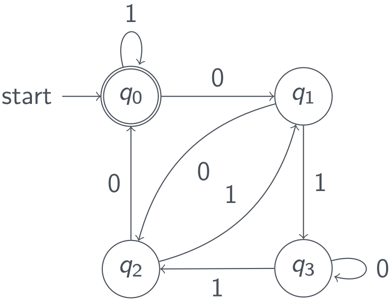
4. Scrivere una espressione regolare per tutte stringhe binarie che cominciano e finiscono per 1
5. Scrivere una espressione regolare per le stringhe binarie che contengono almeno tre 1 consecutivi
6. Scrivere una espressione regolare per le stringhe binarie che contengono almeno tre 1 (anche non consecutivi)
7. Scrivere una espressione regolare per stringhe di testo che descriva le date in formato `GG/MM/AAAA`

Espressioni regolari: esercizi (3)

Costruite una Espressione Regolare equivalente ai seguenti automi:

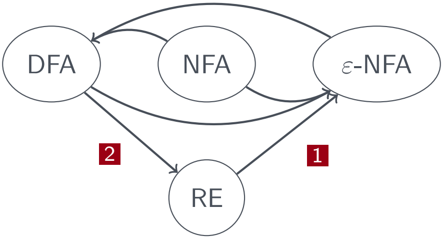






Università degli Studi di Padova

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



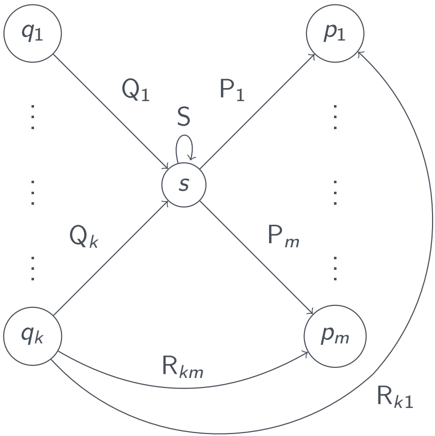
Gli FA sono equivalenti alle espressioni regolari:

1. Per ogni espressione regolare R esiste un ε-NFA A, tale che $L(A) = L(\mathsf{R})$
2. Per ogni FA A possiamo costruire un'espressione regolare R, tale che $L(\mathsf{R}) = L(A)$

Conversione per eliminazione di stati

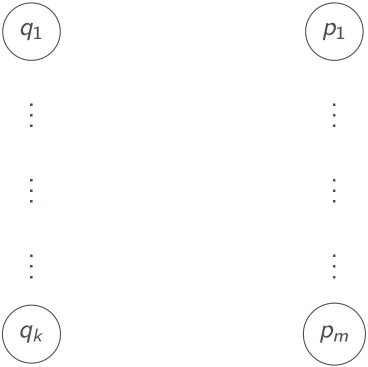
- La procedura che vedremo è in grado di convertire un qualsiasi automa (DFA, NFA, ε-NFA) in una espressione regolare equivalente
- Si procede per eliminazione di stati
- Quando uno stato q viene eliminato, i cammini che passano per q scompaiono
- si aggiungono nuove transizioni etichettate con espressioni regolari che rappresentano i cammini eliminati
- alla fine otteniamo un'espressione regolare che rappresenta tutti i cammini dallo stato iniziale ad uno stato finale
  - $\Rightarrow$ cioè il linguaggio riconosciuto dall'automa

Eliminazione dello stato s



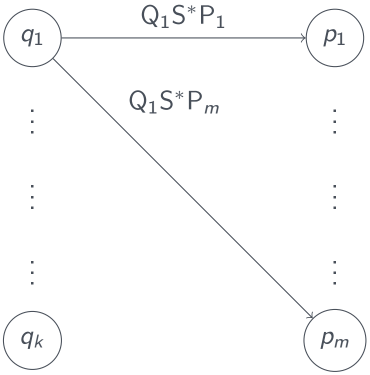
- Lo stato da eliminare può avere un ciclo
- $q_1, \ldots, q_k$ sono i predecessori
- $p_1, \ldots, p_m$ sono i successori
- ci possono essere transizioni dirette tra i predecessori ed i successori

Eliminazione dello stato s



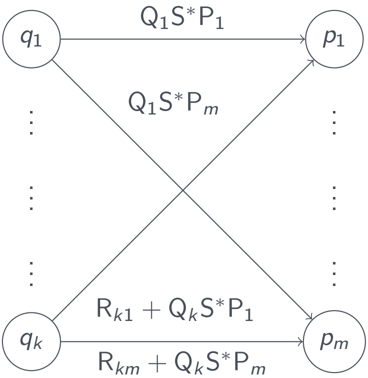
- Dobbiamo **ricreare la transizione** per ogni **coppia predecessore-successore** q_i, p_j

Eliminazione dello stato s



- Dobbiamo ricreare la transizione per ogni coppia predecessore-successore q_i, p_j
- Se non c'è la transizione diretta, l'etichetta è $Q_iS^*P_j$

Eliminazione dello stato s



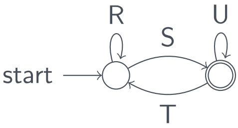
- Dobbiamo ricreare la transizione per ogni coppia predecessore-successore q_i, p_j
- Se non c'è la transizione diretta, l'etichetta è $Q_iS^*P_j$
- Se c'è la transizione diretta, l'etichetta è $R_{ij} + Q_iS^*P_j$

La strategia completa

1. l'automa deve avere un unico stato finale
   - se c'è più di uno stato finale, crea un nuovo stato finale q_f con ε-transizioni provenienti dai vecchi stati finali
2. collassa le transizioni tra la stessa coppia di stati
3. elimina tutti gli stati tranne lo stato iniziale e lo stato finale

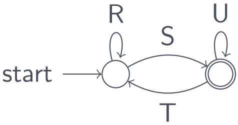
La strategia completa

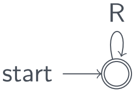
1. l'automa deve avere un unico stato finale
   - se c'è più di uno stato finale, crea un nuovo stato finale q_f con ε-transizioni provenienti dai vecchi stati finali
2. collassa le transizioni tra la stessa coppia di stati
3. elimina tutti gli stati tranne lo stato iniziale e lo stato finale
4. se $q_f \neq q_0$ l'automa finale è



   che è equivalente a $(R + SU^*T)^*SU^*$

La strategia completa

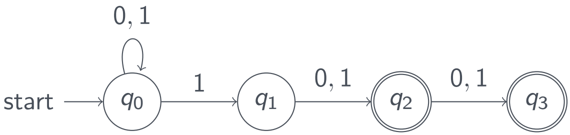
1. l'automa deve avere un unico stato finale
   - se c'è più di uno stato finale, crea un nuovo stato finale q_f con ε-transizioni provenienti dai vecchi stati finali
2. collassa le transizioni tra la stessa coppia di stati
3. elimina tutti gli stati tranne lo stato iniziale e lo stato finale
4. se $q_f \neq q_0$ l'automa finale è 

   che è equivalente a $(R + SU^*T)^*SU^*$
5. se $q_f = q_0$ l'automa finale è 

   che è equivalente a R^*

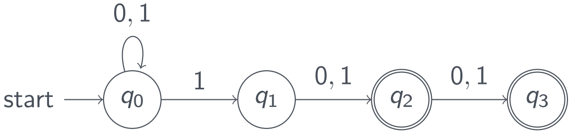
Da FA a RE: esercizi

1. Costruiamo l'espressione regolare equivalente al seguente NFA:

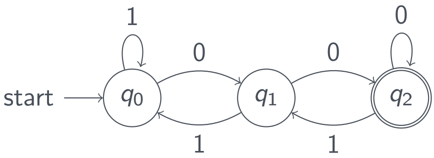


Da FA a RE: esercizi

1. Costruiamo l'espressione regolare equivalente al seguente NFA:



2. Costruiamo l'espressione regolare equivalente al seguente NFA:



3. Costruiamo l'espressione regolare equivalente al seguente NFA: