Linguaggi di Programmazione
Docente: Cataldo Musto

Capitolo 6 – Automi a stati finiti (deterministici e non deterministici)

Si ringraziano il Prof. Giovanni Semeraro, il Prof. Marco de Gemmis e il Prof. Pasquale Lops per la concessione del materiale didattico di base

Automi a stati finiti deterministici

- Cosa è un automa a stati finiti?

Automi a stati finiti deterministici

- Cosa è un automa a stati finiti?
  - Una macchina che **riconosce le parole** accettate da un linguaggio

Automi a stati finiti deterministici

- Cosa è un automa a stati finiti?
  - Una macchina che **riconosce le parole** accettate da un linguaggio
  - Concettualmente, è strutturata in modo simile a una macchina di Turing.
    - Prende in input un **alfabeto di simboli** che è in grado di riconoscere (analogo ai simboli di una grammatica). I simboli sono letti a uno a uno dall’automa.
    - Così come per le macchine di Turing, il processo di riconoscimento è guidato da una **funzione di transizione**. Essa determina se il processo di riconoscimento sta procedendo correttamente oppure no.
    - E’ caratterizzato da un **insieme di stati**. Così come per la macchina di Turing, uno «stato» è un passaggio del processo di riconoscimento. Alcuni stati sono detti «finali». Gli stati finali identificano il corretto riconoscimento.

Automi a stati finiti deterministici

- Definizione di automa a stati finiti o accettore a stati finiti o FSA

  Sia X un alfabeto. Un *automa a stati finiti* (*FSA*) è una quadrupla:

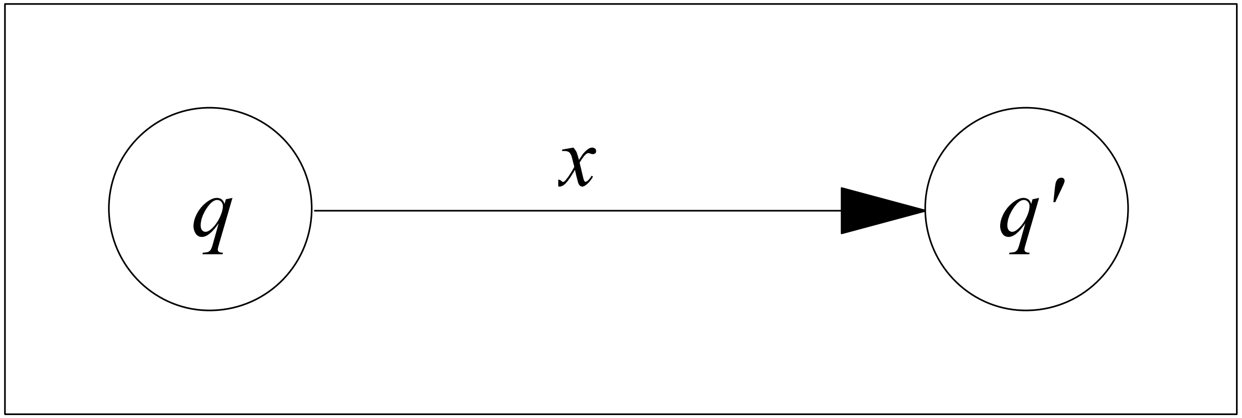
  $$M = (Q, \delta, q_0, F)$$

  ove:
  - X è detto *alfabeto di ingresso*;
  - Q è un insieme finito e non vuoto di *stati*;
  - δ è una funzione da in Q, detta *funzione di transizione*:

    $$\delta : Q \times X \rightarrow Q$$
  - q_0 è lo *stato iniziale*;
  - $F \subseteq Q$ è l'insieme degli *stati di accettazione* o *finali*.

Rappresentazione di FSA

- Un FSA può essere rappresentato mediante:
  - *Grafo degli Stati* o *Diagramma di Transizione* o *Diagramma di Stato*
    - ogni stato $q \in Q$ è rappresentato da un cerchio (nodo) con etichetta q;
    - lo stato iniziale (nodo q_0) ha un arco orientato entrante libero (ossia, che non proviene da nessun altro nodo);
    - per ogni stato $q \in Q$ e per ogni simbolo *x* dell'alfabeto di ingresso, $x \in X$, se $\delta(q, x) = q'$, esiste un arco orientato etichettato con *x* uscente dal nodo *q* ed entrante nel nodo *q'*.



Rappresentazione di FSA

- Un FSA può essere rappresentato mediante:
  - *Tavola di Transizione*

    È una tabella in cui sono riportati gli stati sulle righe e i simboli dell'alfabeto di ingresso sulle colonne.

    Per ogni coppia (stato-ingresso) si legge nella tavola lo stato successivo:

δ	x_1	x_2	$\dots$	$\dots.$	x_n
q_0	q_0^1	q_0^2	$\dots$	$\dots$	q_0^n
q_1	q_1^1	q_1^2	$\dots$	$\dots$	q_1^n
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
q_m	q_m^1	q_m^2	$\dots$	$\dots$	q_m^n

dove l'alfabeto di ingresso e l'insieme degli stati sono rispettivamente:

$$X = \{x_1, x_2, \dots, x_n\} \text{ e } Q = \{q_0, q_1, \dots, q_m\}$$

Quindi si ha che:

$$\delta(q_i, x_j) = q_i^j \text{ con } q_i, q_i^j \in Q,\ x_j \in X$$

Automi a stati finiti deterministici

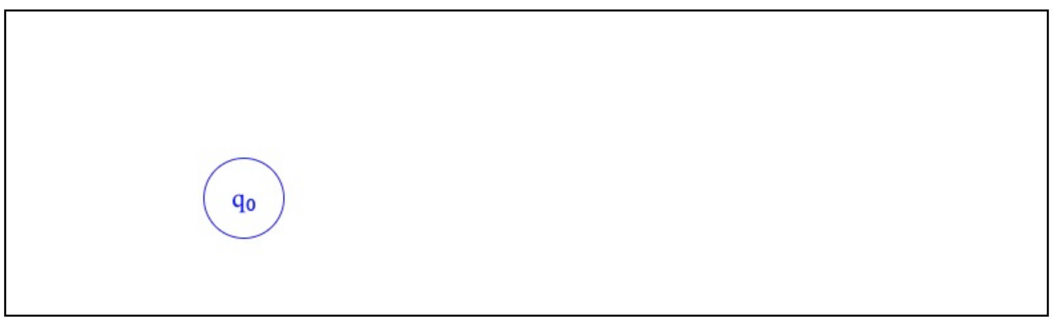
- Talora i valori della funzione di transizione δ non sono definiti per tutte le coppie (stato-simbolo di ingresso) (q, x). In tal caso, si dice che δ è una funzione parziale o definita parzialmente.
- Questo significa che la lettura di x dà luogo in q ad un comportamento dell'automa che non si ritiene utile descrivere ai fini del riconoscimento (nel senso che produrrebbe stringhe non accettate).
- La funzione può anche essere trasformata in funzione totale creando uno stato apposito (detto stato pozza) dove vanno a finire tutte le transizioni che non portano a una parola accettata dal linguaggio.

Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**

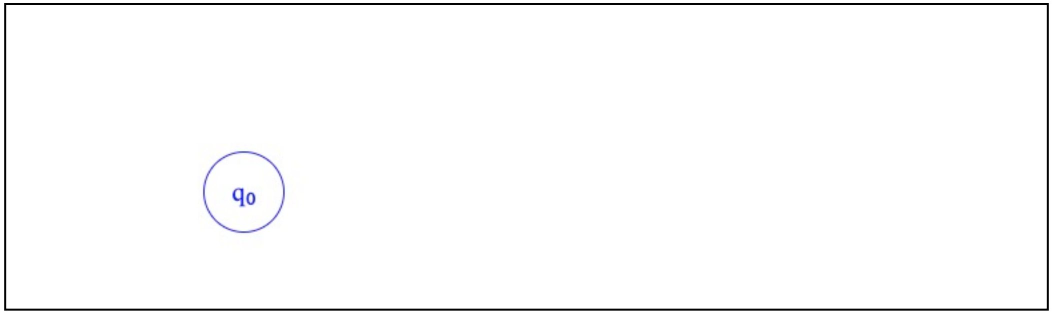
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Il processo di riconoscimento/creazione comincia creando uno stato (detto stato iniziale) tipicamente etichettato con q_0



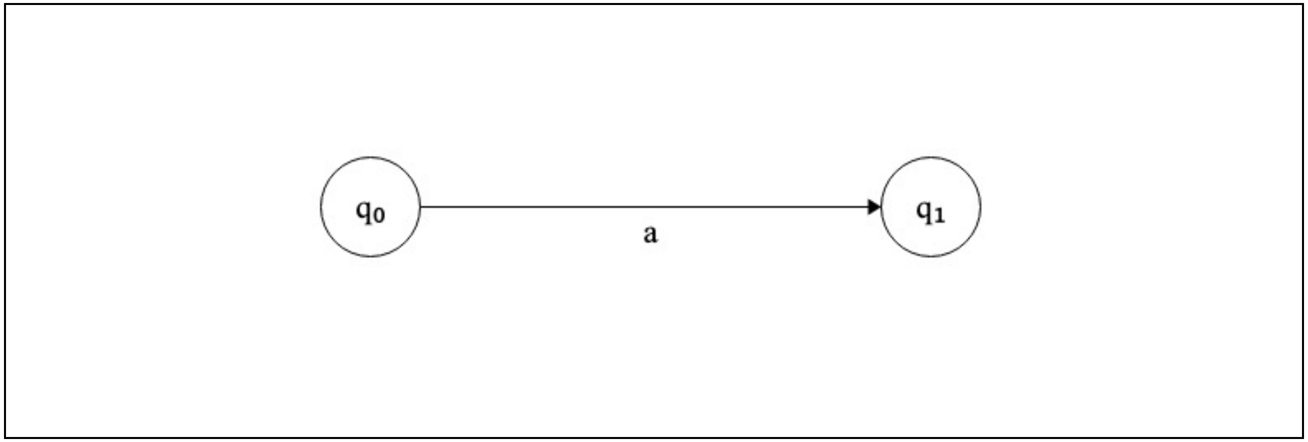
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Quali caratteri possono essere letti?



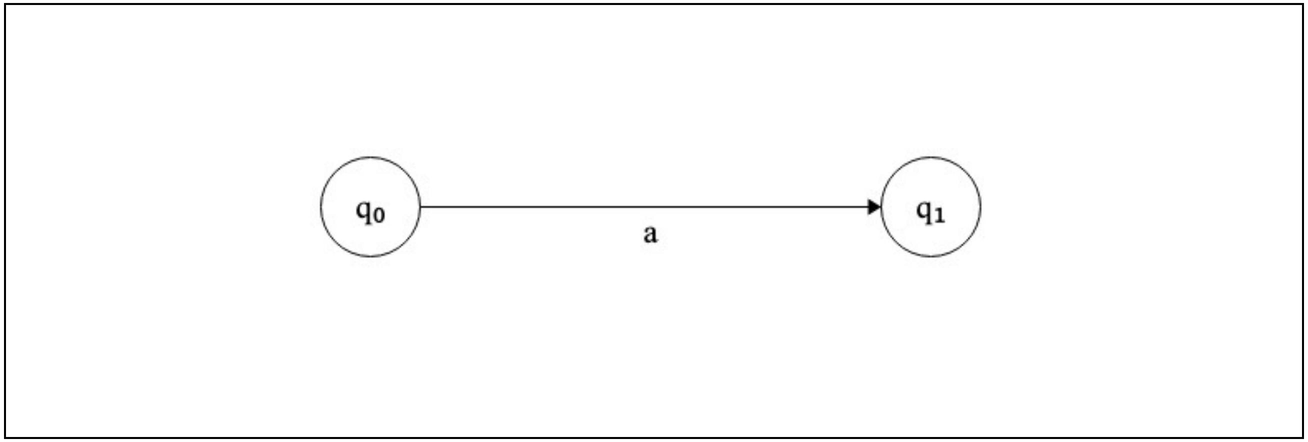
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Le stringhe possono iniziare con a. Quando viene letta una **a**, siccome stiamo «procedendo» nel processo di riconoscimento andiamo a creare un nuovo stato.



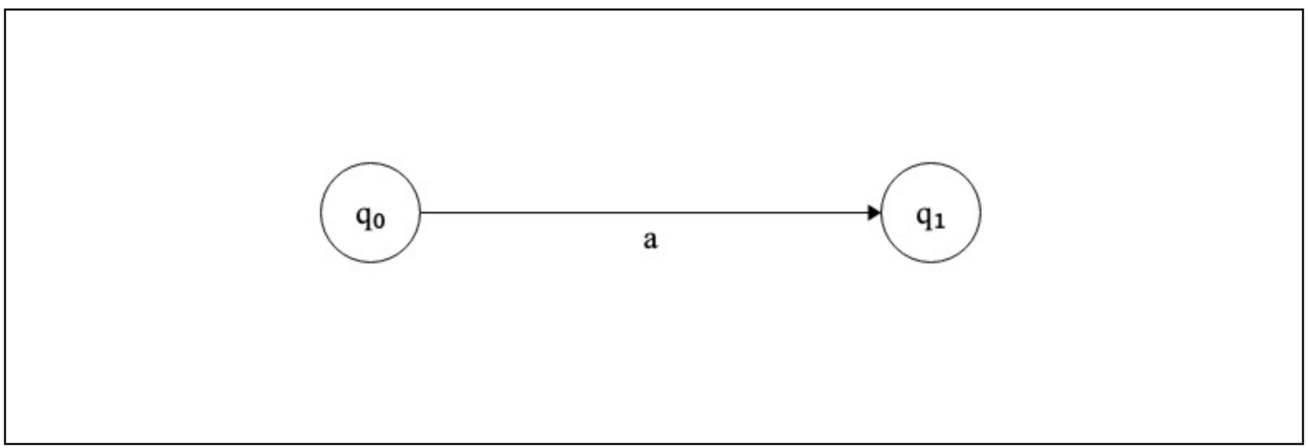
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Importante: così come per le macchine di Turing, non si crea uno stato per ogni carattere letto. Lo stato serve a identificare qualcosa che vogliamo «ricordare»



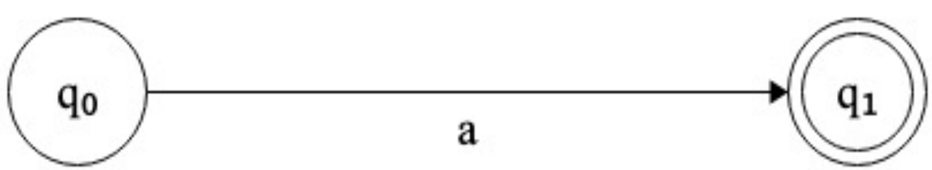
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - La stringa composta da solo ‘a’, è una parola valida del linguaggio?



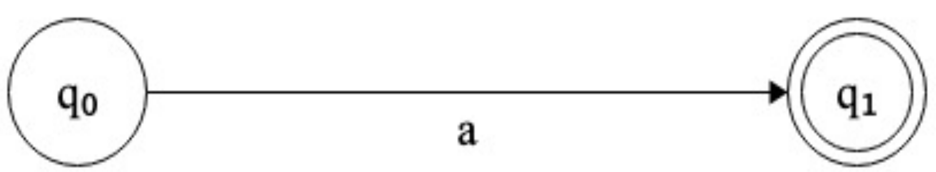
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - La stringa composta da solo ‘a’, è una parola valida del linguaggio? SI. Dunque lo stato viene etichettato come «finale».



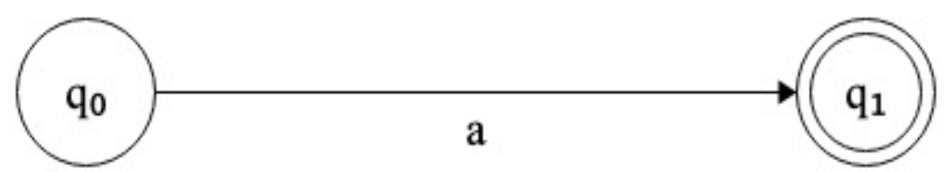
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Cosa possiamo leggere a questo punto?



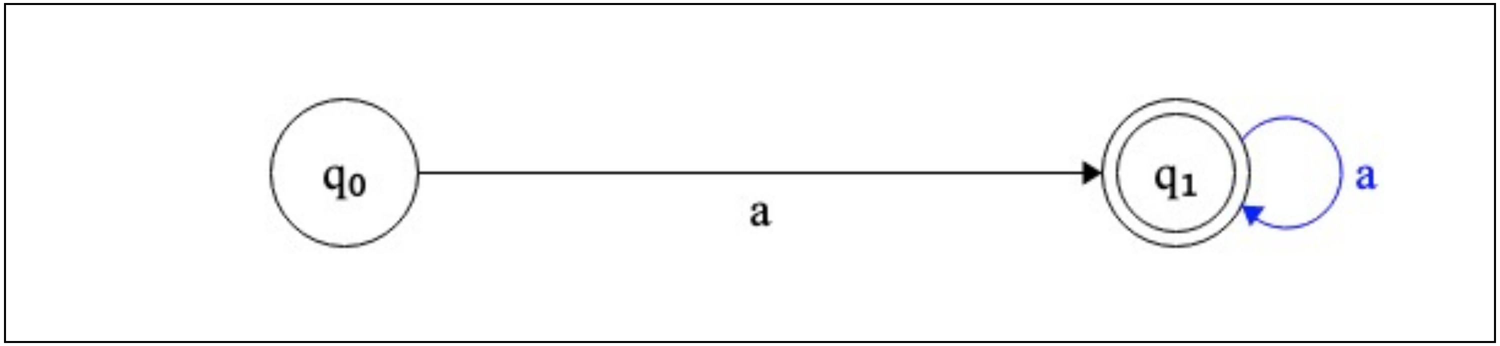
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Cosa possiamo leggere a questo punto? Ad esempio altre ‘a’. La lettura di nuove ‘a’ fa cambiare lo stato o ci mantiene nell’ambito delle parole accettate?



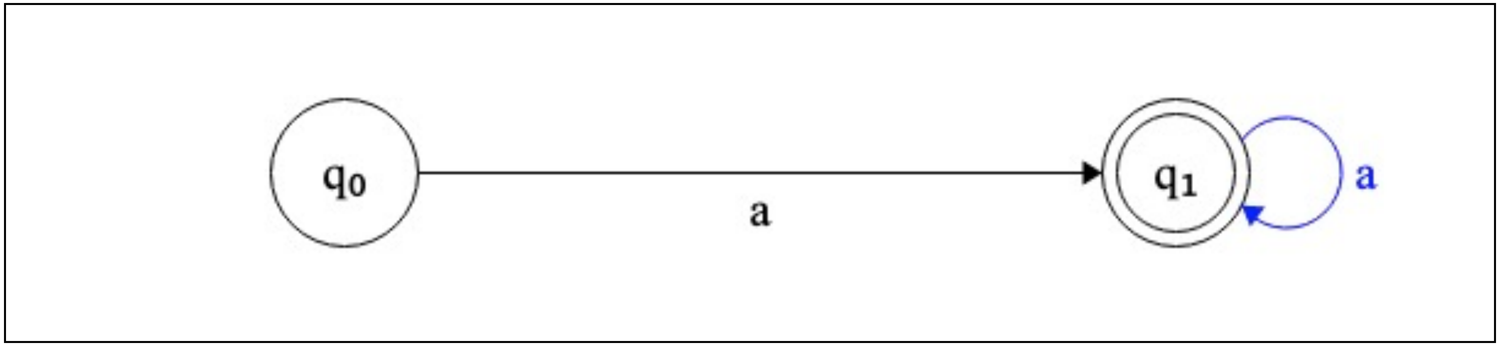
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Aggiungendo altre ‘a’, restiamo sempre nell’ambito delle parole accettate. Questa cosa si codifica in un automa in questo modo.



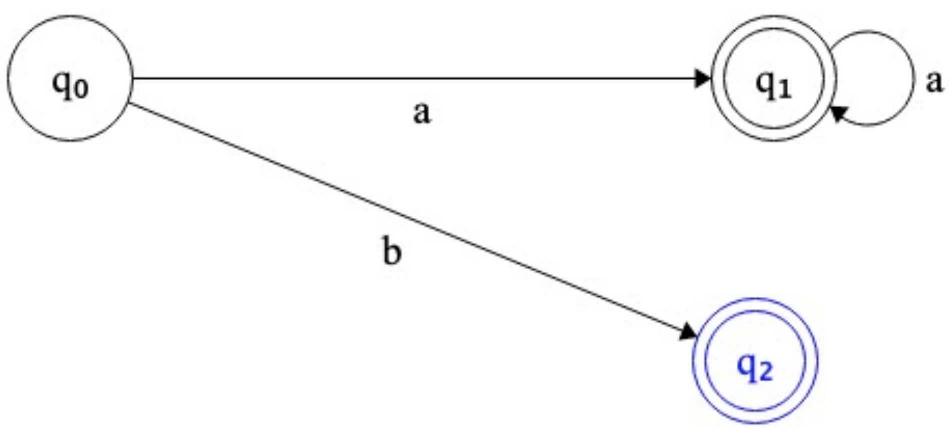
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - A questo punto, quando le 'a' sono terminate, cominciamo con le b. La parola può anche iniziare con b?



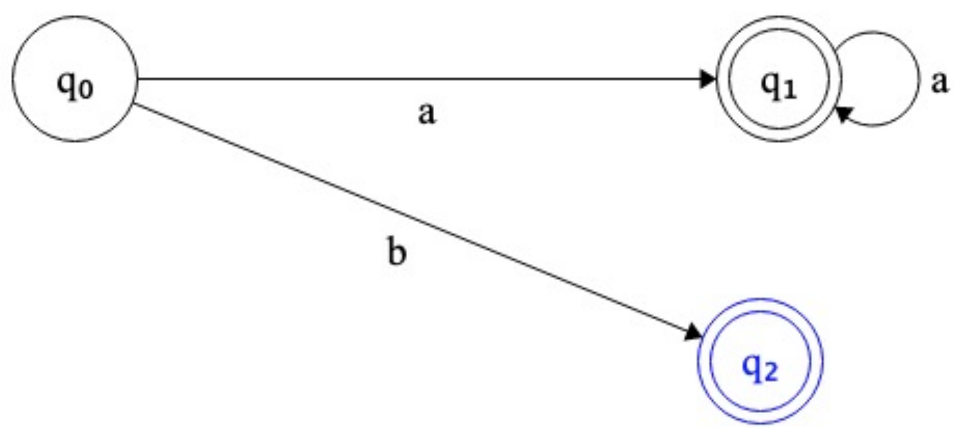
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Si, quindi iniziamo lo stesso processo.



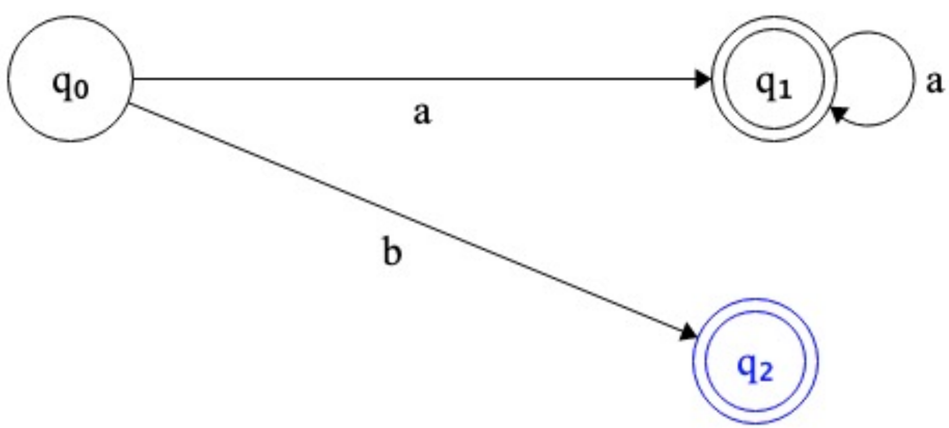
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Cosa manca? Cosa altro può accadere quando sono in q1?



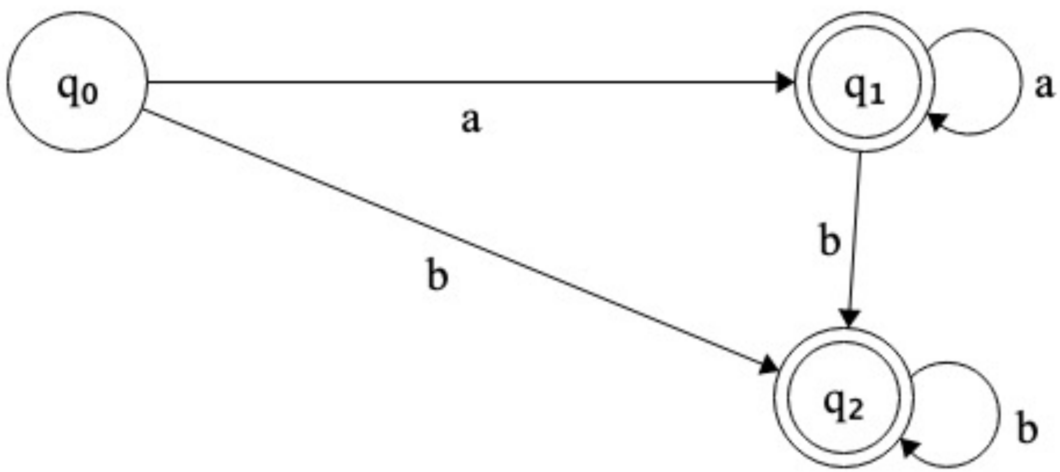
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Cosa manca? Cosa altro può accadere quando sono in q1? Posso leggere delle b!



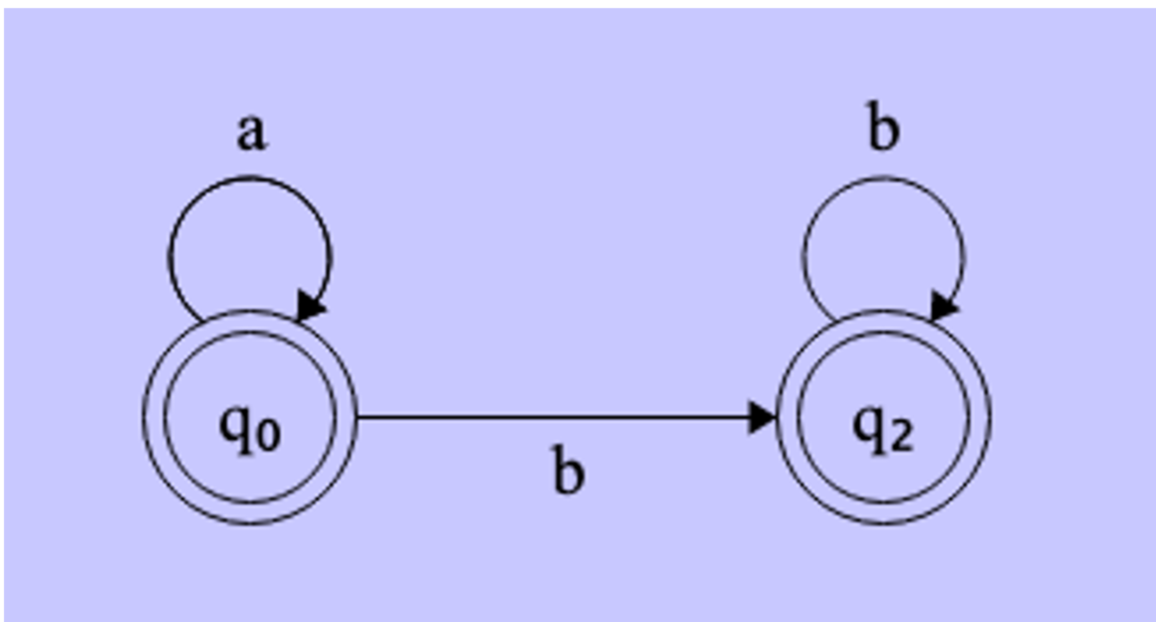
Un esempio semplicissimo

- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Codifichiamo questa informazione



Un esempio semplicissimo

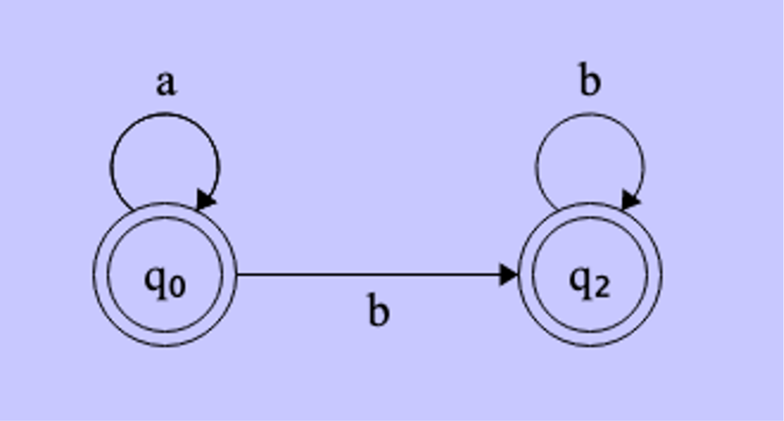
- Immaginiamo di costruire un automa che riconosca il linguaggio **a*b***
  - **Insieme di tutte le sequenze di a oppure di b**
  - Versione definitiva dell’automa (con meno stati)



Un esempio semplicissimo

- Costruiamo la matrice di transizione dell'automa

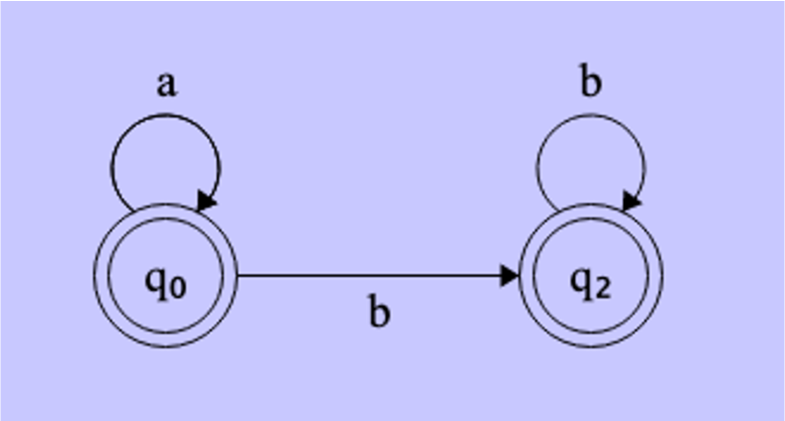
	a	b
q0		
q2		



Un esempio semplicissimo

- Costruiamo la matrice di transizione dell’automa

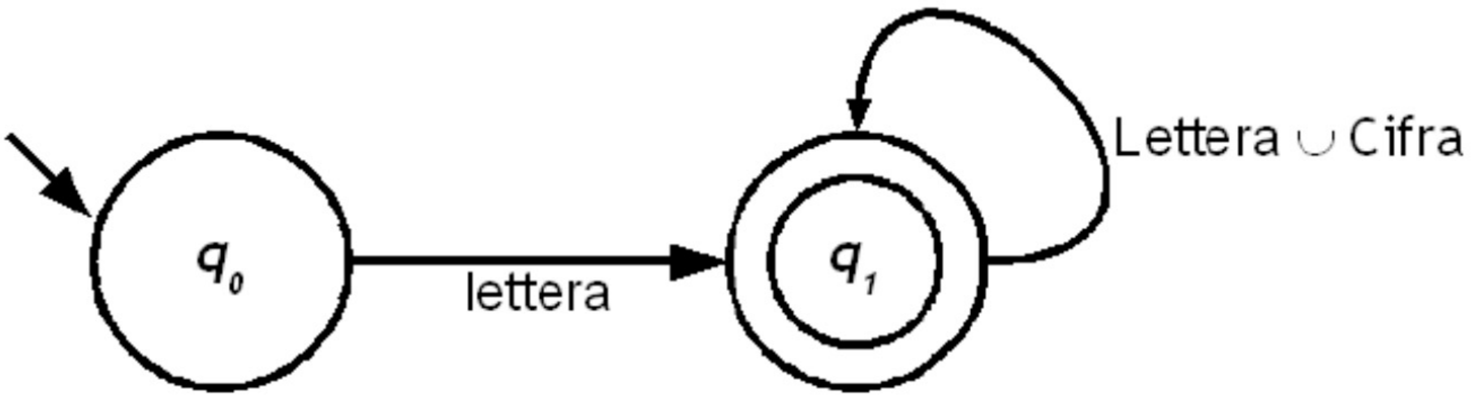
	a	b
q0	q0	q2
q2		q2



Esempi

Gli identificatori C-like

$M = (\{q_0, q_1\}, \delta, q_0, F)$:



Estensione della funzione di transizione

- Nella sua definizione di base, la funzione di transizione prende come input uno stato e un carattere dell'alfabeto, e restituisce un nuovo stato.
- **Normalmente però un automa riceve come input una intera stringa, non un carattere.**
- Estendiamo la funzione di transizione a questo scopo.

Estensione della funzione di transizione

- Si può definire un'estensione della funzione di transizione δ come segue:
  Definizione: *δ^* per FSA*
  Dato un FSA $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X, definiamo per induzione la funzione:
  $$\delta^* : Q \times X^* \to Q$$

- Questa funzione ci dice in **quale stato arriviamo dando in input una stringa** (non un carattere!)

Estensione della funzione di transizione

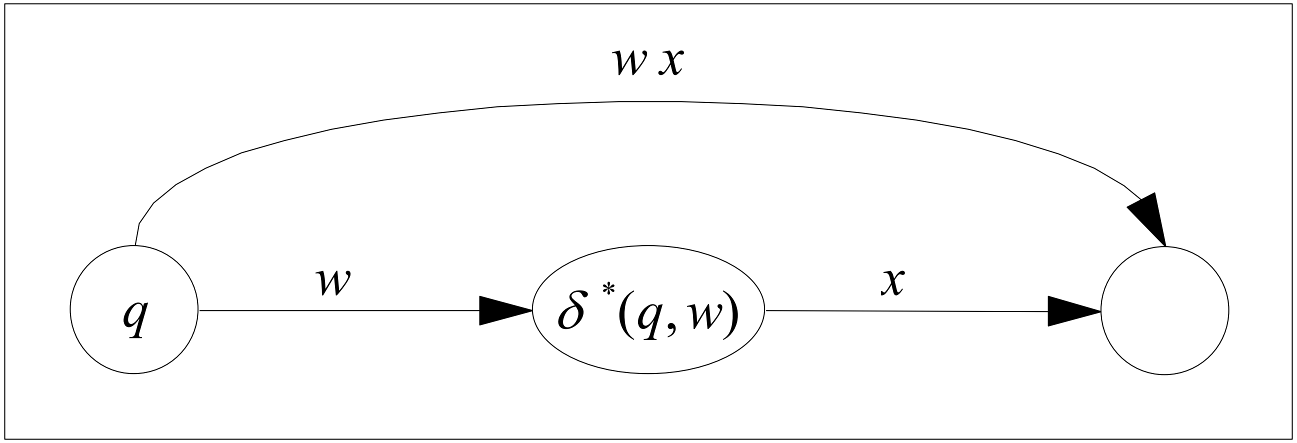
- Si può definire un'estensione della funzione di transizione δ come segue:
  Definizione: δ^* *per FSA*
  Dato un FSA $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X, definiamo per induzione la funzione:
  $$\delta^* : Q \times X^* \to Q$$

- tale che $\delta^*(q, w)$, per $q \in Q$ e $w \in X^*$, sia lo stato in cui *M* si porta avendo in ingresso la parola *w* su *X* e partendo dallo stato *q*.

$$\begin{cases} \delta^*(q, \lambda) = q \\ \delta^*(q, wx) = \delta\left(\delta^*(q, w), x\right) \end{cases} \quad \text{per ogni } q \in Q,\ x \in X,\ w \in X^*$$

Estensione della funzione di transizione

- La figura sottostante riporta una descrizione grafica della definizione precedente.



$$\begin{cases} \delta^*(q,\lambda) = q \\ \delta^*(q,wx) = \delta\left(\delta^*(q,w),x\right) \end{cases} \qquad \text{per ogni } \ q \in Q,\ \ x \in X,\ \ w \in X^*$$

Parola accettata o riconosciuta da un FSA

- Definizione

  Sia $M = (Q, \delta, q_0, F)$ un FSA con alfabeto di ingresso X. Una *parola* $w \in X^*$ è *accettata* (o *riconosciuta*) da M se, partendo dallo stato iniziale q_0, lo stato q in cui l'automa si porta alla fine della sequenza di ingresso w è uno stato finale.

$$w \text{ accettata } \overset{def}{\Leftrightarrow} \delta^*(q_0, w) \in F$$

$$\delta^* : Q \times X^* \rightarrow Q$$

L'output della funzione
è uno stato!

Linguaggio accettato o riconosciuto da un FSA

- Definizione

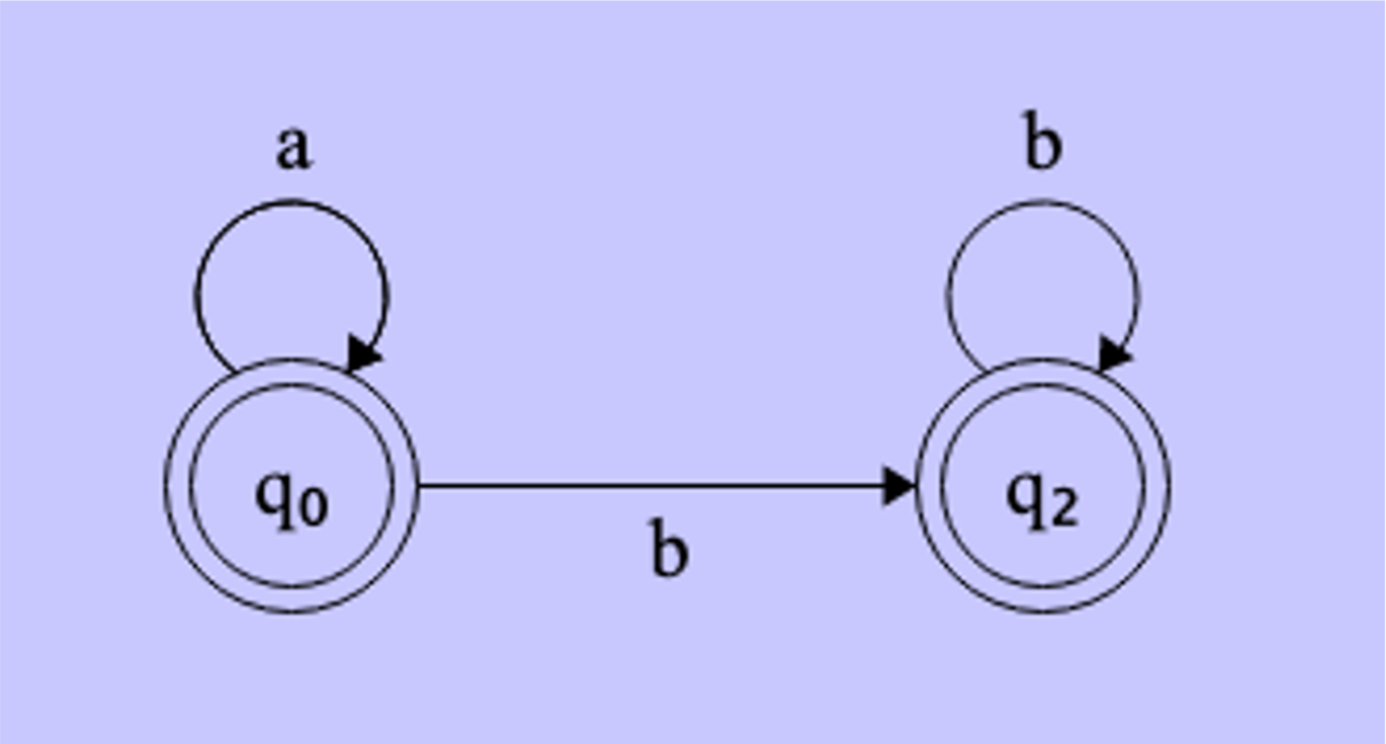
  Sia $M = (Q, \delta, q_0, F)$ un FSA con alfabeto di ingresso X. Il *linguaggio accettato* o *riconosciuto* da M è il seguente sottoinsieme di X^*:

$$T(M) = \{w \in X^* \mid \delta^*(q_0, w) \in F\}$$

  (l'insieme delle parole accettate da M).

Linguaggio accettato o riconosciuto da un FSA

- Esempio



$T(M) = a^* \cdot b^*$

FSA equivalenti

- Definizione

  Sia $M_1 = (Q_1, \delta_1, q_1, F_1)$ ed $M_2 = (Q_2, \delta_2, q_2, F_2)$ due FSA di alfabeto di ingresso X. M_1 ed M_2 si dicono *equivalenti* se:

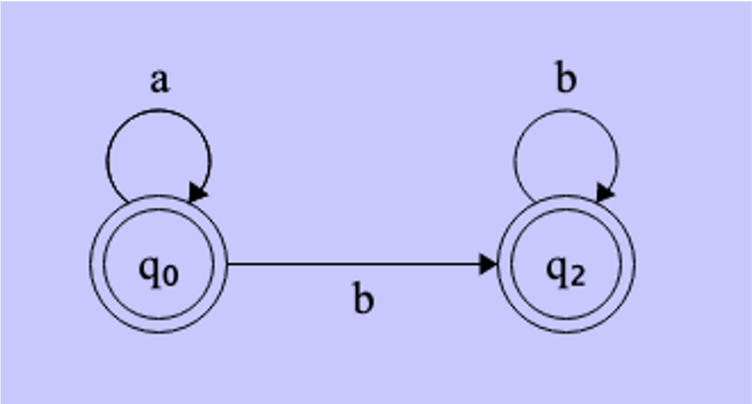
$$T(M_1) = T(M_2)$$

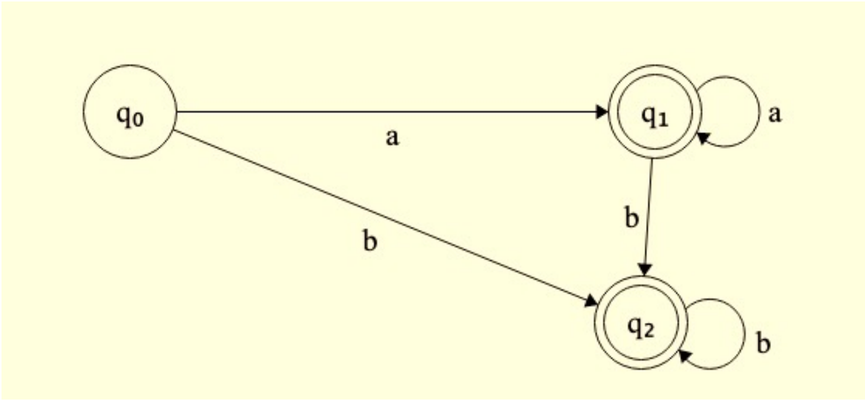
FSA equivalenti

- Definizione

  Sia $M_1 = (Q_1, \delta_1, q_1, F_1)$ ed $M_2 = (Q_2, \delta_2, q_2, F_2)$ due FSA di alfabeto di ingresso X. M_1 ed M_2 si dicono *equivalenti* se:

$$T(M_1) = T(M_2)$$





Linguaggi a stati finiti

- Definizione

  Dato un alfabeto X, un linguaggio L su X è un *linguaggio a stati finiti* (o FSL - Finite State Language) se esiste un FSA M con alfabeto di ingresso X tale che $L = T(M)$.

Classe dei linguaggi a stati finiti

- Definizione

  Di seguito la definizione della *Classe dei Linguaggi a Stati Finiti*

$$\mathcal{L}_{FSL} = \left\{ L \in 2^{X^*} \mid \exists M,\ M \text{ è un } \mathcal{FSA} : L = T(M) \right\}$$

Esercizio

Esercizio 1. Costruire un FSA che accetti questo linguaggio:
$L = \{w \in \{a, b\}^* \mid w \text{ ha un numero pari di } a \text{ e dispari di } b\}$

Esercizio

Esercizio 1. Costruire un FSA che accetti questo linguaggio:
$L = \{w \in \{a, b\}^* \mid w$ ha un numero pari di a e dispari di $b\}$

Soluzione: Sia $M = (Q, \delta, q_0, F) \in$ FSA

- $Q =$

Esercizio

Esercizio 1. Costruire un FSA che accetti questo linguaggio:
$L = \{w \in \{a, b\}^* \mid w$ ha un numero pari di a e dispari di $b\}$

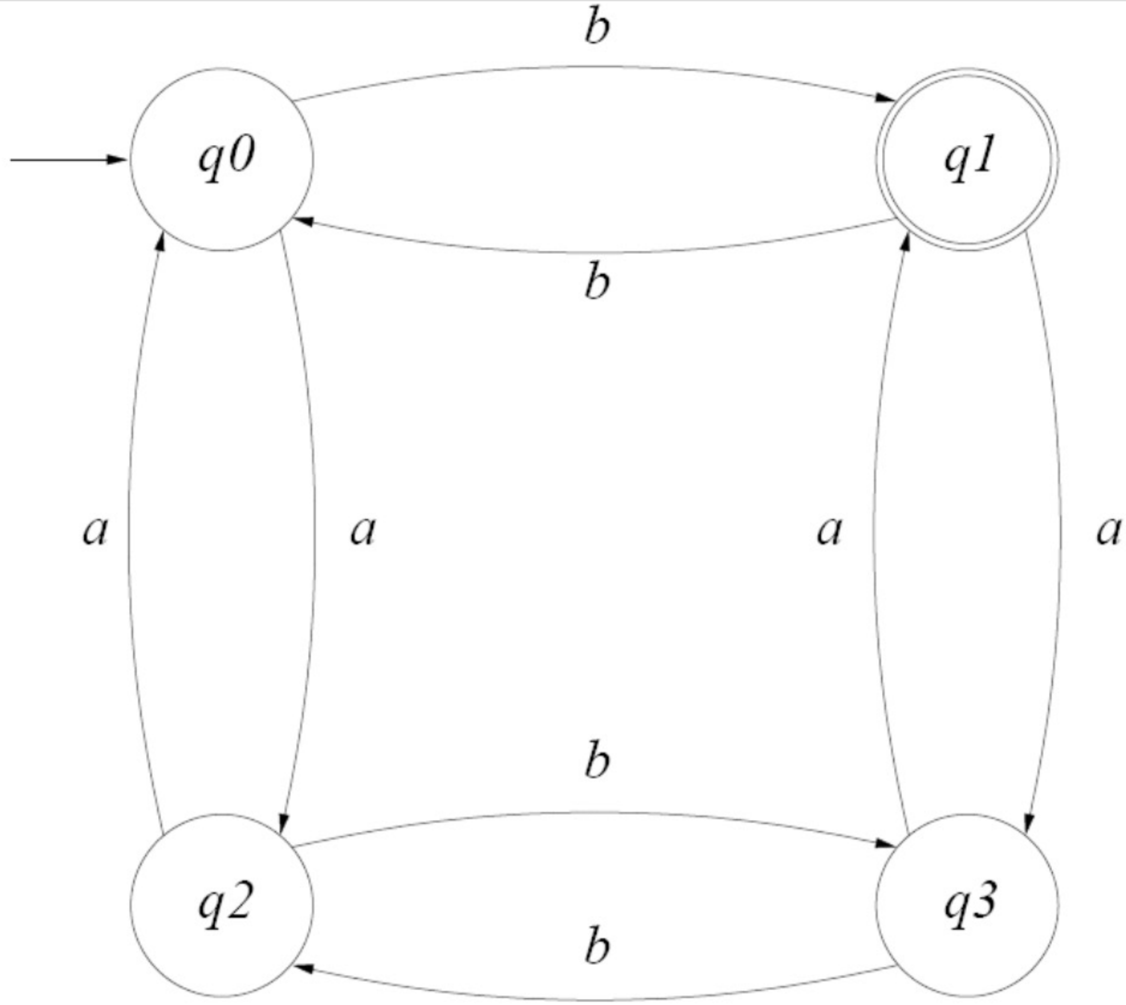
Soluzione: Sia $M = (Q, \delta, q_0, F) \in$ FSA

- $Q = \{q_0, q_1, q_2, q_3\}$ dove
  - q_0 stato per un numero pari di a e di b
  - q_1 stato per un numero pari di a e dispari di b
  - q_2 stato per un numero dispari di a e pari di b
  - q_3 stato per un numero dispari di a e di b

Esercizio

- Costruiamo l’automa

Esercizio – Costruiamo l'Automa



Un altro esempio

Esercizio 2. Costruire un FSA che accetti questo linguaggio:
$L = \{w \in \{a, b\}^* \mid w \neq \alpha aa\beta,\ \alpha, \beta \in \{a, b\}^*\}$

Un altro esempio

Esercizio 2. Costruire un FSA che accetti questo linguaggio:
$L = \{w \in \{a,b\}^* \mid w \neq \alpha aa\beta,\ \alpha,\beta \in \{a,b\}^*\}$

Soluzione: Sia $M = (Q, \delta, q_0, F) \in \mathsf{FSA}$

- $Q = \{q_0, q_1, q_2\}$ dove

Un altro esempio

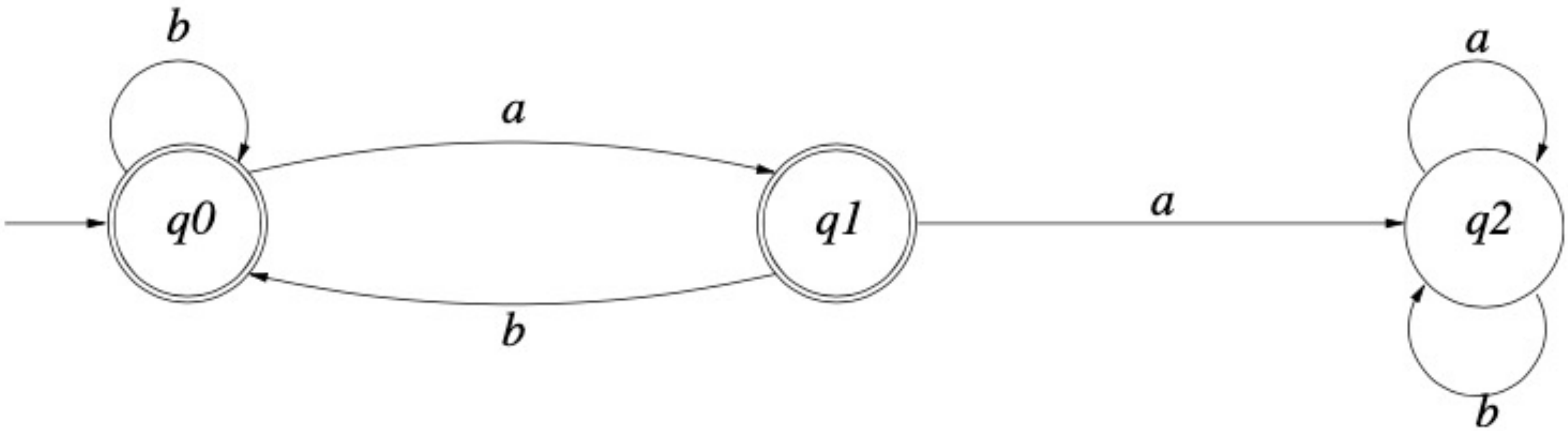
Esercizio 2. Costruire un FSA che accetti questo linguaggio:
$L = \{w \in \{a, b\}^* \mid w \neq \alpha aa\beta,\ \alpha, \beta \in \{a, b\}^*\}$

Soluzione: Sia $M = (Q, \delta, q_0, F) \in$ FSA

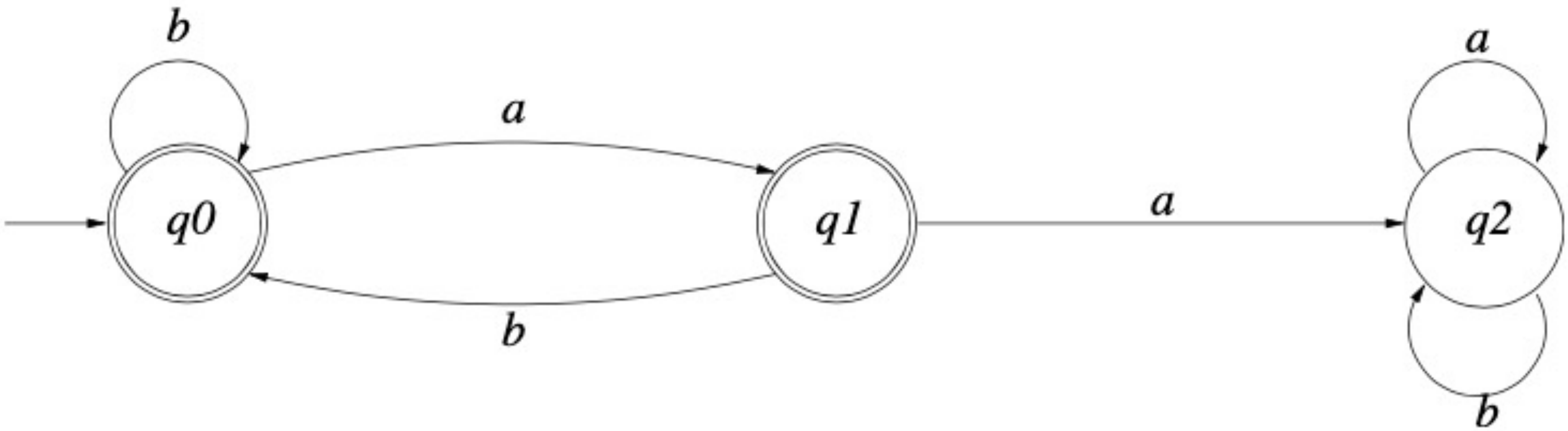
- $Q = \{q_0, q_1, q_2\}$ dove
  - q_0 stato per parole non contenenti due o più a consecutive e terminanti con b
  - q_1 stato per parole non contenenti due o più a consecutive e terminanti con a
  - q_2 stato pozzo per parole contenenti due o più a consecutive

Costruiamo l'automa

Costruiamo l'automa



Costruiamo l'automa



- funzione di transizione δ definita:
  - $\delta(q_0, a) = q_1$
  - $\delta(q_0, b) = \delta(q_1, b) = q_0$
  - $\delta(q_1, a) = \delta(q_2, a) = \delta(q_2, b) = q_2$
- q_0 stato iniziale
- $F = \{q_0, q_1\}$

Proposizione

- I linguaggi a stati finiti sono *chiusi rispetto al complemento*.

  Dimostrazione

  Sia $L \in \mathrm{L}_{FSL}$ un linguaggio a stati finiti sull'alfabeto *X*. Dalla definizione di linguaggio a stati finiti, $L = T(M)$, ove $\boxed{M = (Q, \delta, q_0, F).}$

Consideriamo il complemento di *L*: $\overline{L} = X^* - L$ e l'automa a stati finiti $\overline{M} = (Q, \delta, q_0, Q - F)$. Si ha: $\overline{L} = T(\overline{M})$ (si dimostra per induzione sulla lunghezza di una parola *w*).

Esercizio - Complemento

1 Data $R = \mathtt{a(aa)^*b^*}$ definire un DFA M' tale che $\mathsf{L}(M') = \overline{\mathsf{L}(R)}$

Risoluzione consigliata

1. Costruire prima l'automa che riconosce **a(aa)*b***
2. Capire le caratteristiche delle **parole riconosciute**
3. Definire le caratteristiche delle **parole non riconosciute**
4. Costruire **l'automa complemento**

Esercizio - Complemento

1 Data $\boldsymbol{R} = \mathtt{a(aa)^*b^*}$ definire un DFA $\boldsymbol{M'}$ tale che $\mathsf{L}(\boldsymbol{M'}) = \overline{\mathsf{L}(\boldsymbol{R})}$

Risoluzione consigliata

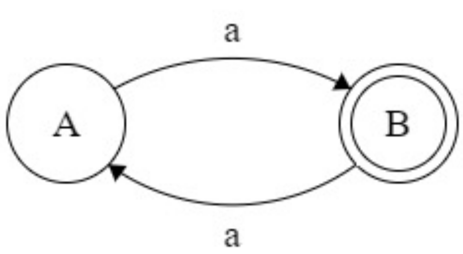
1. Costruire prima l'automa che riconosce **a(aa)*b***
   - **Numero dispari di a**
   - **Seguito eventualmente da una sequenza di b**

Esercizio - Complemento

1 Data $R = \mathsf{a(aa)^*b^*}$ definire un DFA M' tale che $\mathsf{L}(M') = \overline{\mathsf{L}(R)}$

Risoluzione consigliata

1. Costruire prima l'automa che riconosce **a(aa)*b***
   - **Numero dispari di a**

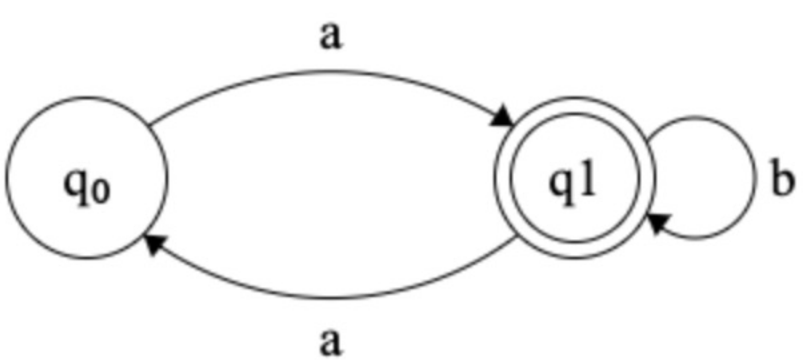


Esercizio - Complemento

1 Data $R = \mathrm{a(aa)^*b^*}$ definire un DFA M' tale che $\mathsf{L}(M') = \overline{\mathsf{L}(R)}$

Risoluzione consigliata

1. Costruire prima l'automa che riconosce **a(aa)*b***
   - **Numero dispari di a**
   - **Seguito eventualmente da una sequenza di b**

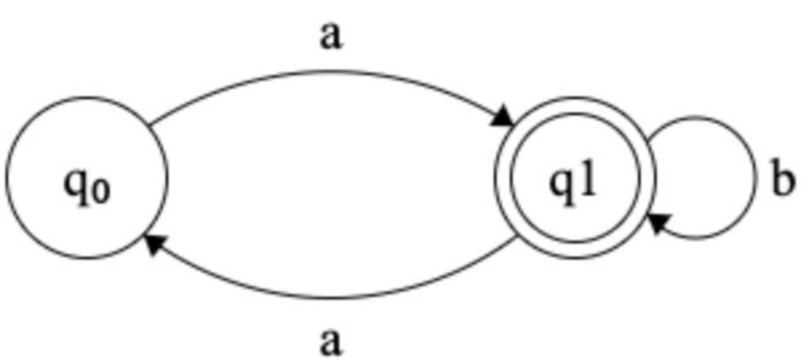


Esercizio - Complemento

1 Data $R = a(aa)^*b^*$ definire un DFA M' tale che $L(M') = \overline{L(R)}$

Risoluzione consigliata

2. Capire le caratteristiche delle parole riconosciute



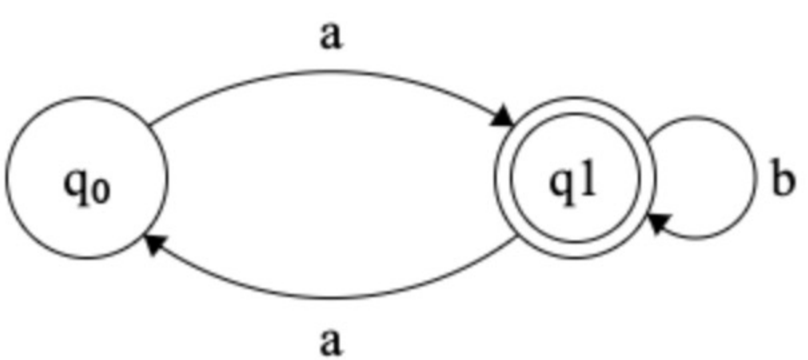
Esercizio - Complemento

1 Data $\boldsymbol{R} = \mathsf{a(aa)^*b^*}$ definire un DFA $\boldsymbol{M'}$ tale che $\mathsf{L}(\boldsymbol{M'}) = \overline{\mathsf{L}(\boldsymbol{R})}$

Risoluzione consigliata

2. Capire le caratteristiche delle parole riconosciute

inizia per a



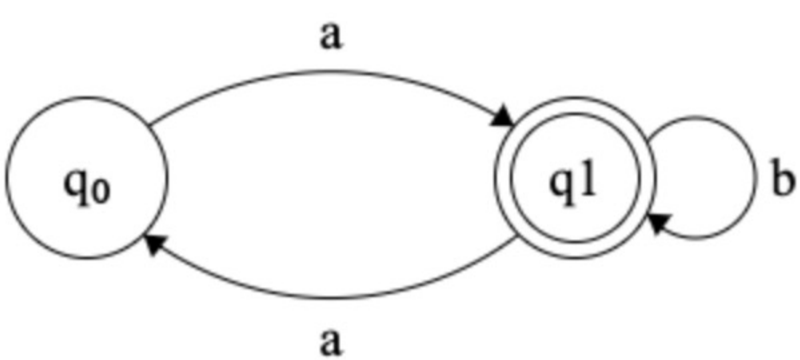
Esercizio - Complemento

1 Data $R = \mathtt{a(aa)^*b^*}$ definire un DFA M' tale che $\mathsf{L}(M') = \overline{\mathsf{L}(R)}$

Risoluzione consigliata

2. Capire le caratteristiche delle parole riconosciute

inizia per a & contiene «a» dispari



Esercizio - Complemento

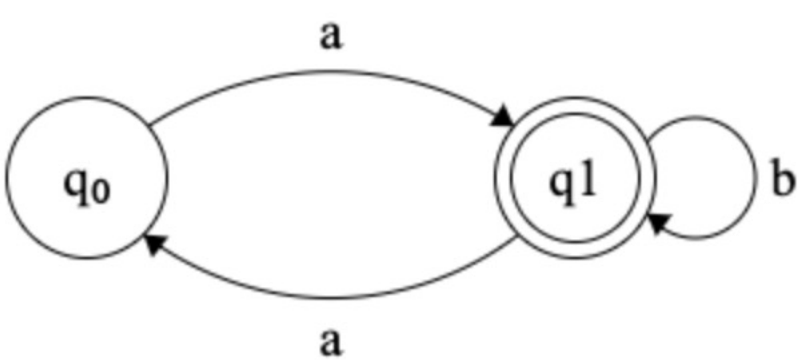
1 Data $R = a(aa)^*b^*$ definire un DFA M' tale che $L(M') = \overline{L(R)}$

Risoluzione consigliata

3. Capire le caratteristiche delle parole **non riconosciute**

NON inizia per a **||** **NON contiene «a» dispari**

A noi interessa costruire il complemento!



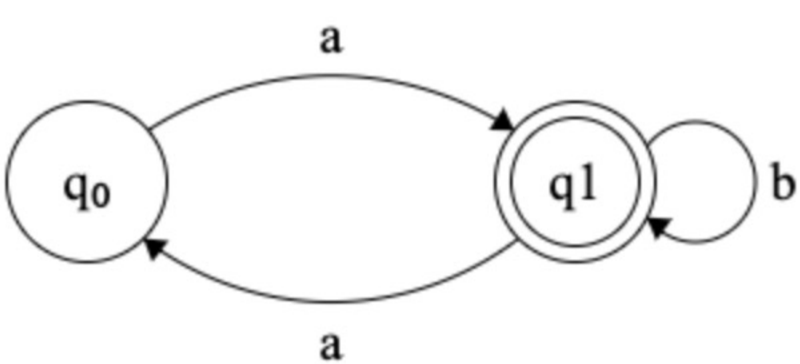
Esercizio - Complemento

1 Data $R = a(aa)^*b^*$ definire un DFA M' tale che $L(M') = \overline{L(R)}$

Risoluzione consigliata

3. Capire le caratteristiche delle parole **non riconosciute**

INIZIA PER B OR CONTIENE A PARI



Esercizio - Complemento

1 Data $R = \mathsf{a(aa)^*b^*}$ definire un DFA M' tale che $\mathsf{L}(M') = \overline{\mathsf{L}(R)}$

Costruiamo l'automa complemento invertendo gli stati

Abbiamo risolto?

Esercizio - Complemento

1 Data $R = \mathtt{a(aa)^*b^*}$ definire un DFA M' tale che $L(M') = \overline{L(R)}$

Costruiamo l'automa complemento invertendo gli stati

Abbiamo risolto?

No, perché non riusciamo a riconoscere le parole che iniziano per b.

Perché?

Esercizio - Complemento

1 Data $R = \mathrm{a}(\mathrm{aa})^*\mathrm{b}^*$ definire un DFA M' tale che $\mathsf{L}(M') = \overline{\mathsf{L}(R)}$

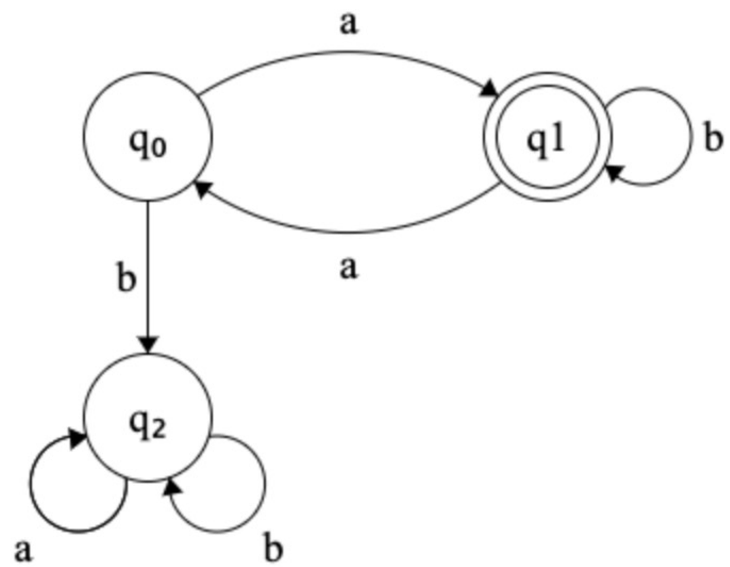
Costruiamo l'automa complemento invertendo gli stati

Abbiamo risolto?
No, perché non riusciamo a riconoscere le parole che iniziano per b.
Perché? Per costruire l'automa complemento, la funzione di transizione deve essere totale. Serve lo stato pozza

Esercizio - Complemento

1 Data $R = a(aa)^*b^*$ definire un DFA M' tale che $L(M') = \overline{L(R)}$

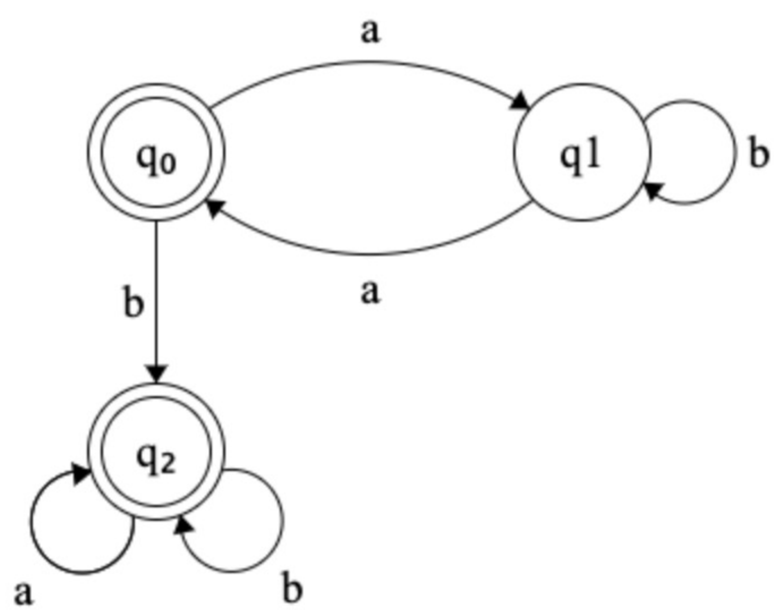
Automa riconoscitore con funzione di transizione totale



Esercizio - Complemento

1 Data $\boldsymbol{R} = \mathsf{a(aa)^*b^*}$ definire un DFA $\boldsymbol{M'}$ tale che $\mathsf{L}(\boldsymbol{M'}) = \overline{\mathsf{L}(\boldsymbol{R})}$

Inversione degli stati per riconoscere il complemento



Automa a stati finiti non deterministico o accettore a stati finiti non deterministico

- Definizione

  Un *automa a stati finiti non deterministico* (*NDA*) con alfabeto di ingresso *X* è una quadrupla:

$$M = (Q, \delta, q_0, F)$$

  ove:

  - per Q, q_0 ed F valgono le definizioni date per gli FSA;
  - $\delta : Q \times X \rightarrow 2^Q$ è la funzione di transizione che assegna ad ogni coppia (stato-simbolo di ingresso) (q, x) un insieme $\delta(q, x) \subseteq Q$ di possibili stati successivi.

Automa a stati finiti non deterministico o accettore a stati finiti non deterministico

- Definizione
  Un *automa a stati finiti non deterministico* (*NDA*) con alfabeto di ingresso X è una quadrupla:

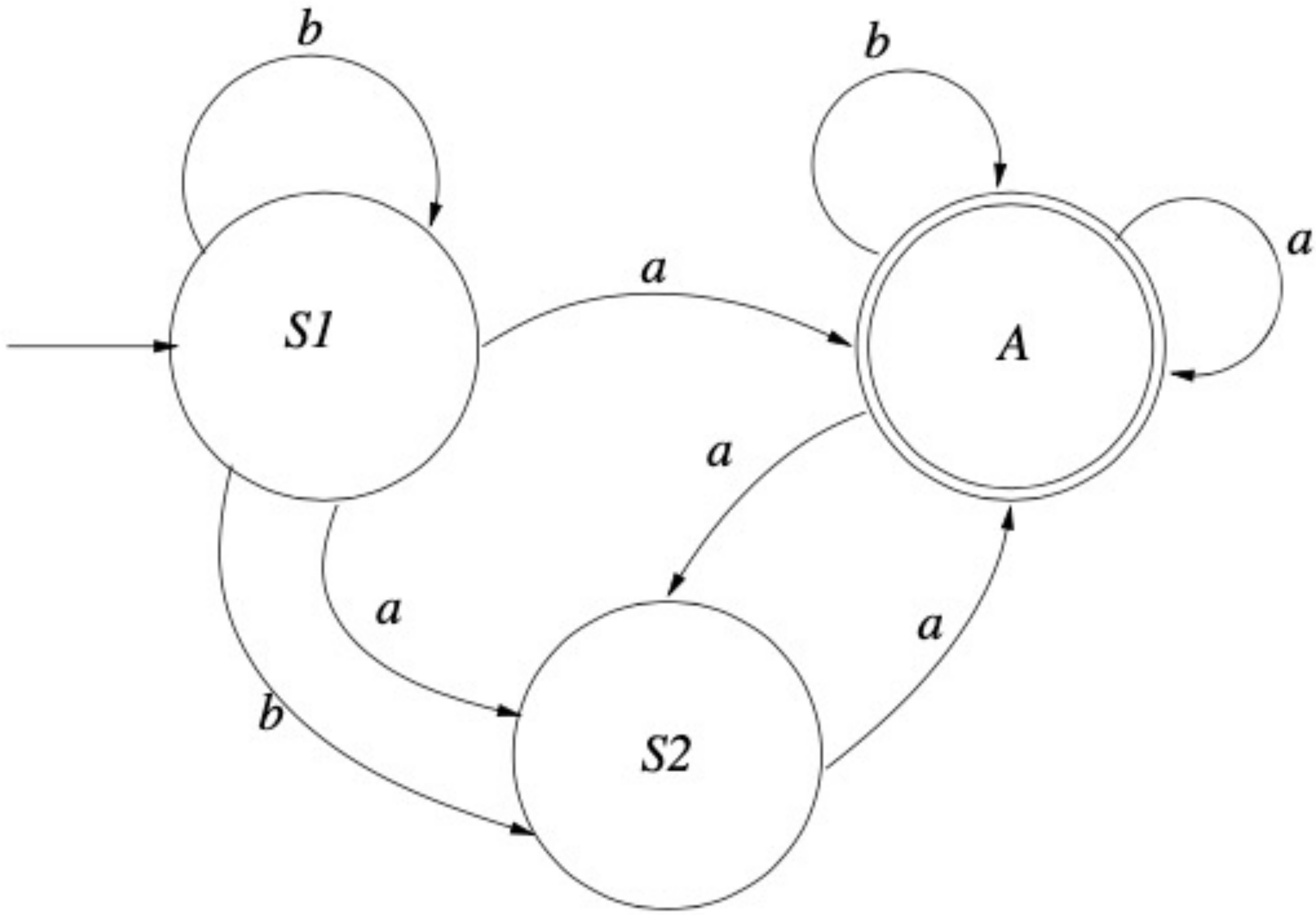
  $$M = (Q, \delta, q_0, F)$$

  ove:
  - per Q, q_0 ed F valgono le definizioni date per gli FSA;
  - $\delta : Q \times X \rightarrow 2^Q$ è la funzione di transizione che assegna ad ogni coppia (stato-simbolo di ingresso) (q, x) un insieme $\delta(q, x) \subseteq Q$ di possibili stati successivi.
  - Il non determinismo risiede nel fatto che l'output di un carattere letto, in relazione ad un dato stato, può essere in questo caso **un insieme di stati.**

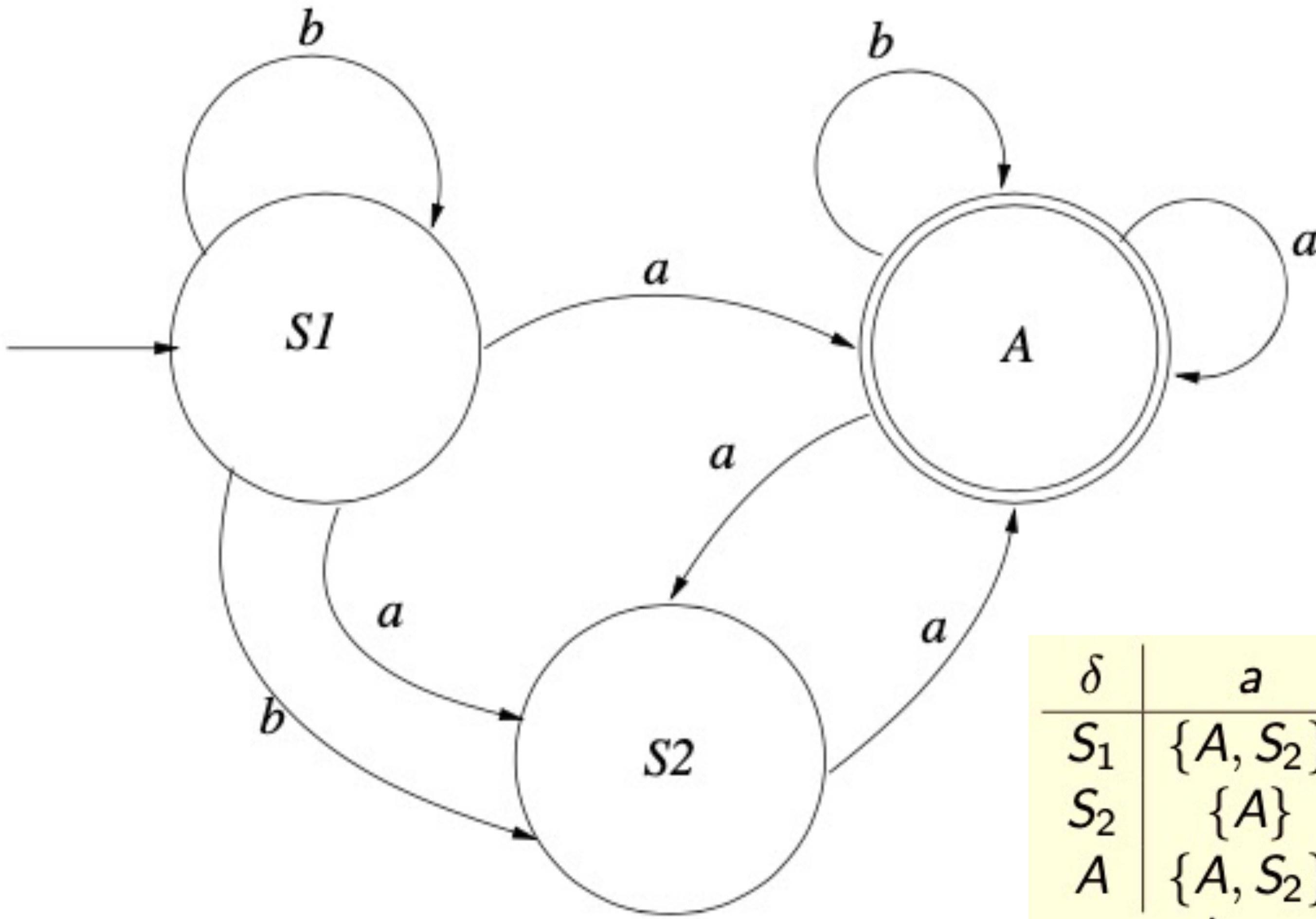
Osservazione

- Un NDA è un FSA con l'unica eccezione che, in corrispondenza di una coppia (stato-simbolo di ingresso) (q, x), vi è un insieme di stati in cui l'automa può transitare (*stati successivi possibili*).

Un esempio di NDA



Un esempio di NDA

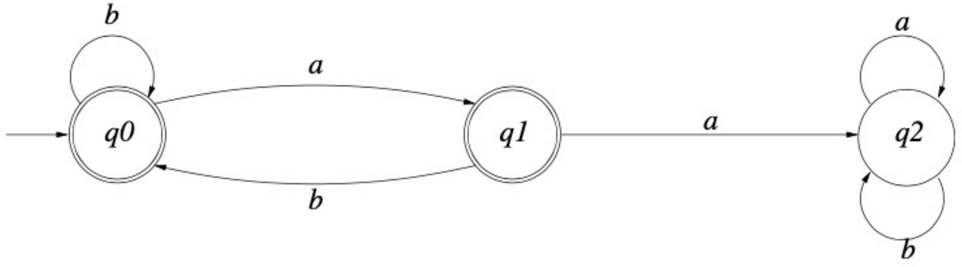


δ	a	b
S_1	$\{A, S_2\}$	$\{S_1, S_2\}$
S_2	$\{A\}$	$\emptyset$
A	$\{A, S_2\}$	$\{A\}$

Recap: Estensione della funz. di transizione

$$\begin{cases} \delta^*(q,\lambda) = q \\ \delta^*(q,wx) = \delta\left(\delta^*(q,w),x\right) \end{cases}$$

per ogni $q \in Q,\ x \in X,\ w \in X^*$

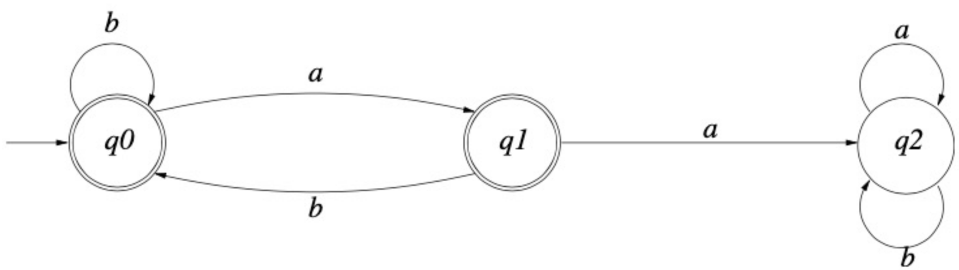


$$\boldsymbol{\delta}^*(q_0, abbb) =$$

Recap: Estensione della funz. di transizione

$$\begin{cases} \delta^*(q,\lambda) = q \\ \delta^*(q,wx) = \delta\left(\delta^*(q,w),x\right) \end{cases}$$

per ogni $q \in Q,\ x \in X,\ w \in X^*$



$$\boldsymbol{\delta}^*(q_0, abbb) =$$

$$\boldsymbol{\delta}(\boldsymbol{\delta}^*(q_0, abb)b) =$$

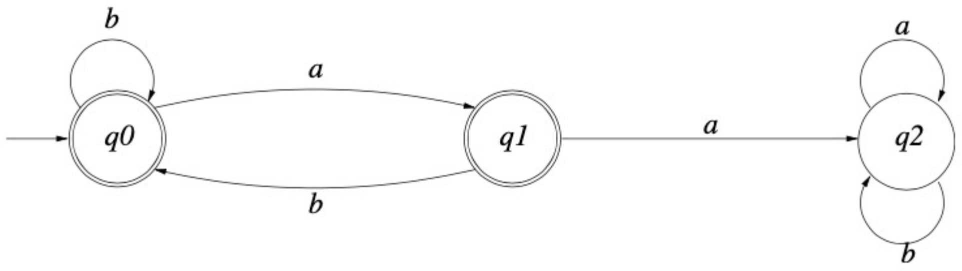
$$\boldsymbol{\delta}(\boldsymbol{\delta}(\boldsymbol{\delta}^*(q_0, ab)b)b) =$$

$$\boldsymbol{\delta}(\boldsymbol{\delta}(\boldsymbol{\delta}(\boldsymbol{\delta}(q_0, a)b)b)b) =$$

Recap: Estensione della funz. di transizione

$$\begin{cases} \delta^*(q,\lambda) = q \\ \delta^*(q,wx) = \delta\big(\delta^*(q,w),x\big) \end{cases}$$

per ogni $q \in Q,\ x \in X,\ w \in X^*$



$$\boldsymbol{\delta}^*(q_0, abbb) =$$

$$\boldsymbol{\delta}(\boldsymbol{\delta}^*(q_0, abb)b) =$$

$$\boldsymbol{\delta}(\boldsymbol{\delta}(\boldsymbol{\delta}^*(q_0, ab)b)b) =$$

$$\boldsymbol{\delta}(\boldsymbol{\delta}(\boldsymbol{\delta}(\boldsymbol{\delta}(q_0, a)b)b)b) = q_0$$

Estensione della funzione di transizione per un NDA

- Come per gli FSA si può definire un'estensione della funzione di transizione δ come segue:
  Definizione: *δ^* per NDA*
  Dato un NDA $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X, definiamo per induzione la funzione:

$$\delta^* : 2^Q \times X^* \to 2^Q$$

$$\begin{cases} \delta^*(p, \lambda) = p \\ \delta^*(p, wx) = \bigcup\limits_{q \in \delta^*(p,w)} \delta(q, x) \end{cases} \quad \text{per ogni } \; p \in 2^Q \; (p \subset Q), \;\; x \in X, \;\; w \in X^*$$

Estensione della funzione di transizione per un NDA

- Come per gli FSA si può definire un'estensione della funzione di transizione δ come segue:
  Definizione: δ^* *per NDA*
  Dato un NDA $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X, definiamo per induzione la funzione:

$$\delta^* : \boxed{2^Q} \times X^* \rightarrow 2^Q$$

$$\begin{cases} \delta^*(p, \lambda) = p \\ \delta^*(p, wx) = \bigcup_{q \in \delta^*(p,w)} \delta(q, x) \end{cases}$$

per ogni $p \in 2^Q \ (p \subset Q), \ x \in X, \ w \in X^*$

Insieme dei possibili sottoinsiemi di Q
(cioè un sottoinsieme degli stati)

Estensione della funzione di transizione per un NDA

- Come per gli FSA si può definire un'estensione della funzione di transizione δ come segue:
  Definizione: δ^* *per NDA*
  Dato un NDA $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X, definiamo per induzione la funzione:

$$\delta^* : 2^Q \times X^* \to 2^Q$$

$$\begin{cases} \delta^*(p, \lambda) = p \\ \delta^*(p, wx) = \bigcup_{q \in \delta^*(p,w)} \delta(q, x) \end{cases} \quad \text{per ogni} \quad p \in 2^Q \ (p \subset Q), \ x \in X, \ w \in X^*$$

Il comportamento della funzione di transizione è l'unione dei comportamenti della funzione di transizione su ogni frammento di stringa

Osservazione

- Analogamente a quanto fatto per gli FSA, si dovrebbero riformulare le definizioni di parola accettata e di linguaggio accettato da un NDA.
- La complicazione, rinveniente dalla computazione non deterministica dello stato successivo in cui un NDA transita, comporta che una stessa parola **può indurre cammini multipli attraverso un NDA, alcuni che terminano in stati di accettazione, altri che terminano in stati di non accettazione.**

Esempio

- Sia dato l'NDA dell'es. precedente. Calcoliamo $\delta^*(\{S_1\}, aba)$

$$\delta^*(\{S_1\}, aba) = \bigcup_{q \in \delta^*(\{S_1\}, ab)} \delta(q, a)$$

:

δ	a	b
S_1	$\{A, S_2\}$	$\{S_1, S_2\}$
S_2	$\{A\}$	$\emptyset$
A	$\{A, S_2\}$	$\{A\}$

Esempio

- Sia dato l'NDA dell'es. precedente. Calcoliamo $\delta^*(\{S_1\},aba)$

$$\delta^*(\{S_1\},aba) = \bigcup_{q \in \delta^*(\{S_1\},ab)} \delta(q,a)$$

$$\delta^*(\{S_1\},ab) = \bigcup_{q' \in \delta^*(\{S_1\},a)} \delta(q',b)$$

:

δ	a	b
S_1	$\{A, S_2\}$	$\{S_1, S_2\}$
S_2	$\{A\}$	$\emptyset$
A	$\{A, S_2\}$	$\{A\}$

Esempio

- Sia dato l'NDA dell'es. precedente. Calcoliamo $\delta^*(\{S_1\}, aba)$

$$\delta^*(\{S_1\}, aba) = \bigcup_{q \in \delta^*(\{S_1\}, ab)} \delta(q, a)$$

$$\delta^*(\{S_1\}, ab) = \bigcup_{q' \in \delta^*(\{S_1\}, a)} \delta(q', b)$$

$$\delta^*(\{S_1\}, a) = \bigcup_{q'' \in \delta^*(\{S_1\}, \lambda)} \delta(q'', a)$$

:

δ	a	b
S_1	$\{A, S_2\}$	$\{S_1, S_2\}$
S_2	$\{A\}$	$\emptyset$
A	$\{A, S_2\}$	$\{A\}$

Esempio

- Sia dato l'NDA dell'es. precedente. Calcoliamo $\delta^*(\{S_1\}, aba)$

$$\delta^*(\{S_1\}, aba) = \bigcup_{q \in \delta^*(\{S_1\}, ab)} \delta(q, a)$$

$$\delta^*(\{S_1\}, ab) = \bigcup_{q' \in \delta^*(\{S_1\}, a)} \delta(q', b)$$

$$\delta^*(\{S_1\}, a) = \bigcup_{q'' \in \delta^*(\{S_1\}, \lambda)} \delta(q'', a)$$

$$\delta^*(\{S_1\}, \lambda) = \{S_1\}$$

:

δ	a	b
S_1	$\{A, S_2\}$	$\{S_1, S_2\}$
S_2	$\{A\}$	$\emptyset$
A	$\{A, S_2\}$	$\{A\}$

Esempio

- Sia dato l'NDA dell'es. precedente. Calcoliamo $\delta^*(\{S_1\}, aba)$

$$\delta^*(\{S_1\}, aba) = \bigcup_{q \in \delta^*(\{S_1\}, ab)} \delta(q, a)$$

:

δ	a	b
S_1	$\{A, S_2\}$	$\{S_1, S_2\}$
S_2	$\{A\}$	$\emptyset$
A	$\{A, S_2\}$	$\{A\}$

$$\delta^*(\{S_1\}, ab) = \bigcup_{q' \in \delta^*(\{S_1\}, a)} \delta(q', b)$$

$$\delta^*(\{S_1\}, a) = \bigcup_{q'' \in \delta^*(\{S_1\}, \lambda)} \delta(q'', a)$$

$$\delta^*(\{S_1\}, \lambda) = \{S_1\}$$

$$\delta^*(\{S_1\}, a) = \delta(S_1, a) = \{A, S_2\}$$

Esempio

- Sia dato l'NDA dell'es. precedente. Calcoliamo $\delta^*(\{S_1\}, aba)$

$$\delta^*(\{S_1\}, aba) = \bigcup_{q \in \delta^*(\{S_1\}, ab)} \delta(q, a)$$

:

δ	a	b
S_1	$\{A, S_2\}$	$\{S_1, S_2\}$
S_2	$\{A\}$	$\emptyset$
A	$\{A, S_2\}$	$\{A\}$

$$\delta^*(\{S_1\}, ab) = \bigcup_{q' \in \delta^*(\{S_1\}, a)} \delta(q', b)$$

$$\delta^*(\{S_1\}, a) = \bigcup_{q'' \in \delta^*(\{S_1\}, \lambda)} \delta(q'', a)$$

$$\delta^*(\{S_1\}, \lambda) = \{S_1\}$$

$$\delta^*(\{S_1\}, a) = \delta(S_1, a) = \{A, S_2\}$$

$$\delta^*(\{S_1\}, ab) = \delta(A, b) \cup \delta(S_2, b) = \{A\} \cup \emptyset = \{A\}$$

$$\delta^*(\{S_1\}, aba) = \delta(A, a) = \{A, S_2\}$$

Parola accettata o riconosciuta da un NDA

- Definizione

  Sia $M = (Q, \delta, q_0, F)$ un NDA con alfabeto di ingresso X. Una *parola* $w \in X^*$ è *accettata* (o *riconosciuta*) da M se, partendo dallo stato iniziale q_0, esiste almeno un modo per M di portarsi in uno stato di accettazione alla fine della sequenza di ingresso w. In formule:

$$w \text{ accettata} \overset{def}{\Leftrightarrow} \exists\, p : p \in \delta^*(\{q_0\}, w) \cap F \;\Leftrightarrow\; \delta^*(\{q_0\}, w) \cap F \neq \varnothing$$

Esempio

- Sia dato l'NDA dell'es. precedente. Calcoliamo $\delta^*(\{S_1\}, aba)$

$$\delta^*(\{S_1\}, aba) = \bigcup_{q \in \delta^*(\{S_1\}, ab)} \delta(q, a)$$

$$\delta^*(\{S_1\}, ab) = \bigcup_{q' \in \delta^*(\{S_1\}, a)} \delta(q', b)$$

$$\delta^*(\{S_1\}, a) = \bigcup_{q'' \in \delta^*(\{S_1\}, \lambda)} \delta(q'', a)$$

$$\delta^*(\{S_1\}, \lambda) = \{S_1\}$$

$$\delta^*(\{S_1\}, a) = \delta(S_1, a) = \{A, S_2\}$$

$$\delta^*(\{S_1\}, ab) = \delta(A, b) \cup \delta(S_2, b) = \{A\} \cup \emptyset = \{A\}$$

$$\delta^*(\{S_1\}, aba) = \delta(A, a) = \{A, S_2\}$$

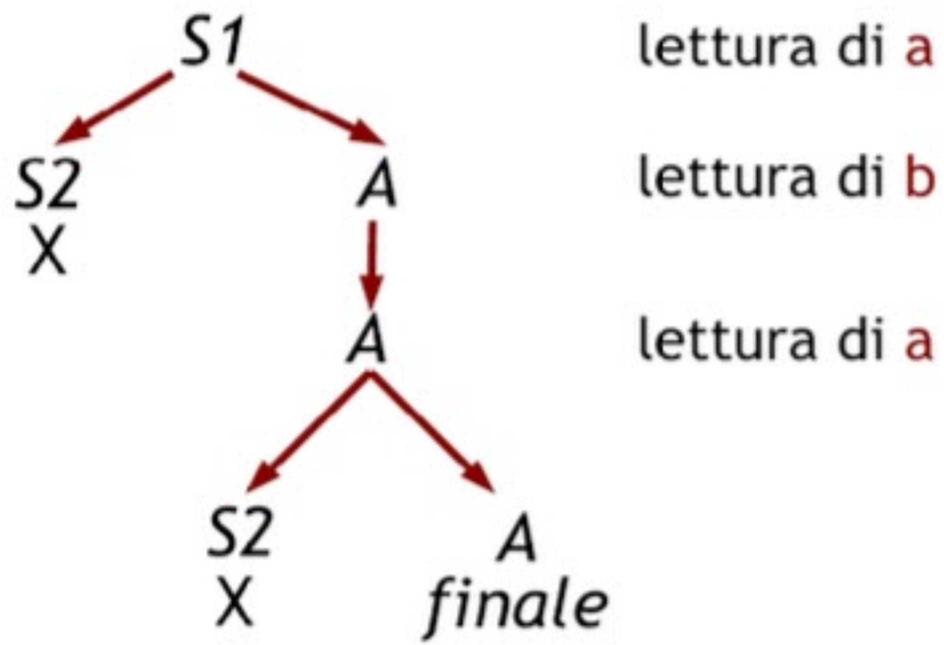
Poiché $F = \{A\}$, si ha:

$\delta^*(\{S_1\}, aba) \cap F = \{A\} \neq \emptyset$

e *w* = *aba* è accettata da M_1.

Esempio

Si consideri la stringa $w = aba$.



Linguaggi accettati o riconosciuto da un NDA

- Definizione

  Sia $M = (Q, \delta, q_0, F)$ un NDA con alfabeto di ingresso X. Il *linguaggio accettato* o *riconosciuto* da M è l'insieme delle parole su X accettate da M:

$$T(M) = \left\{ w \in X^* \mid \delta^*(\{q_0\}, w) \cap F \neq \varnothing \right\}$$

  (è l'insieme delle parole w per le quali esiste almeno un cammino, etichettato con lettere di w nell'ordine da sinistra a destra, attraverso il diagramma degli stati che porta M dallo stato iniziale ad uno degli stati di accettazione).

NDA equivalenti

- Definizione
  Siano $M_1 = (Q_1, \delta_1, q_1, F_1)$ ed $M_2 = (Q_2, \delta_2, q_2, F_2)$ due NDA di alfabeto di ingresso X. M_1 ed M_2 si dicono *equivalenti* se:

$$T(M_1) = T(M_2)$$

Classe dei linguaggi riconosciuti da automi a stati finiti non deterministici

- Definizione

  Di seguito la definizione della *Classe dei Linguaggi riconosciuti da automi a Stati Finiti non deterministici*

$$\mathscr{L}_{NDL} = \left\{ L \in 2^{X^*} \mid \exists M,\ M \text{ è un } \mathscr{NDA} : L = T(M) \right\}$$

Equivalenza delle classi di linguaggi accettati da automi a stati finiti deterministici e non deterministici.

- Teorema
  Le classi dei linguaggi $\mathcal{L}_{FSL}$ e $\mathcal{L}_{NDL}$ sono equivalenti (1ª formulazione).
- Sia L un linguaggio su X. L è un linguaggio a stati finiti se e solo se $L = T(M)$ per qualche NDA M (2ª formulazione).

Dimostrazione (2ª formulazione)

$\Rightarrow$) Sia $L \in 2^{X^*}$ ed $L \in \mathcal{L}_{FSL}$. Dalla definizione di linguaggio a stati finiti, si ha:

$$\exists M_1 : M_1 \text{ è un FSA, } M_1 = (Q_1, \delta_1, q_1, F_1)$$

con alfabeto di ingresso X : $L = T(M_1)$.

Equivalenza delle classi di linguaggi accettati da automi a stati finiti deterministici e non deterministici.

- Dimostrazione (2ª formulazione, continuazione)
  Sulla base di M_1, definiamo il seguente NDA con alfabeto di ingresso X:

$$M_2 = (Q_2, \delta_2, q_2, F_2)$$

  ove:
  - $Q_2 = Q_1$;
  - δ_2 è così definita:

$$\delta_2 : Q_2 \times X \rightarrow 2^{Q_2},$$

$$\delta_2(q,x) = \{\delta_1(q,x)\} \quad \forall q \in Q_2 = Q_1, \;\; x \in X;$$

  - $q_2 = q_1$;
  - $F_2 = F_1$

  M_2 è un NDA ed inoltre accetta lo stesso linguaggio accettato da M_1, ossia si ha: $T(M_2) = T(M_1)$

Equivalenza delle classi di linguaggi accettati da automi a stati finiti deterministici e non deterministici.

- Dimostrazione (2ª formulazione)

$\Leftarrow$) Sia $L \in 2^{X^*}$ ed $L \in \mathcal{L}_{NDL}$. Dalla definizione della classe di linguaggi L_{NDL}, si ha:

$$\exists M, M \text{ è un NDA}, M = (Q, \delta, q_0, F)$$

con alfabeto di ingresso X : $L = T(M)$.
Si può definire allora il seguente algoritmo per la costruzione di un FSA equivalente all'NDA M:

Trasformazione di un automa a stati finiti non deterministico in un automa deterministico equivalente

- Sia $M = (Q, \delta, q_0, F)$ un automa accettore a stati finiti non deterministico di alfabeto di ingresso X. <u>M può essere trasformato in un automa deterministico M'</u> di alfabeto di ingresso X come segue: $M' = (Q', \delta', q'_0, F')$

  ove:
  - $Q' = 2^Q$
  - L'insieme degli stati diventa l'insieme dei sottoinsiemi. **Ogni sottoinsieme diventa uno stato!**

Trasformazione di un automa a stati finiti non deterministico in un automa deterministico equivalente

- Sia $M = (Q, \delta, q_0, F)$ un automa accettore a stati finiti non deterministico di alfabeto di ingresso X. <u>M può essere trasformato in un automa deterministico M'</u> di alfabeto di ingresso X come segue: $M' = (Q', \delta', q'_0, F')$

  ove:
  - $Q' = 2^Q$
  - $q'_0 = \{q_0\}$
  - $F' = \{p \subseteq Q \mid p \cap F \neq \emptyset\}$

Trasformazione di un automa a stati finiti non deterministico in un automa deterministico equivalente

- Sia $M = (Q, \delta, q_0, F)$ un automa accettore a stati finiti non deterministico di alfabeto di ingresso X. <u>M può essere trasformato in un automa deterministico M'</u> di alfabeto di ingresso X come segue: $M' = (Q', \delta', q_0', F')$
  ove:

$$\delta' : Q' \times X \to Q' \quad \ni' \quad \forall q' = \{q_1, q_2, \ldots, q_i\} \in Q',\ \forall x \in X :$$

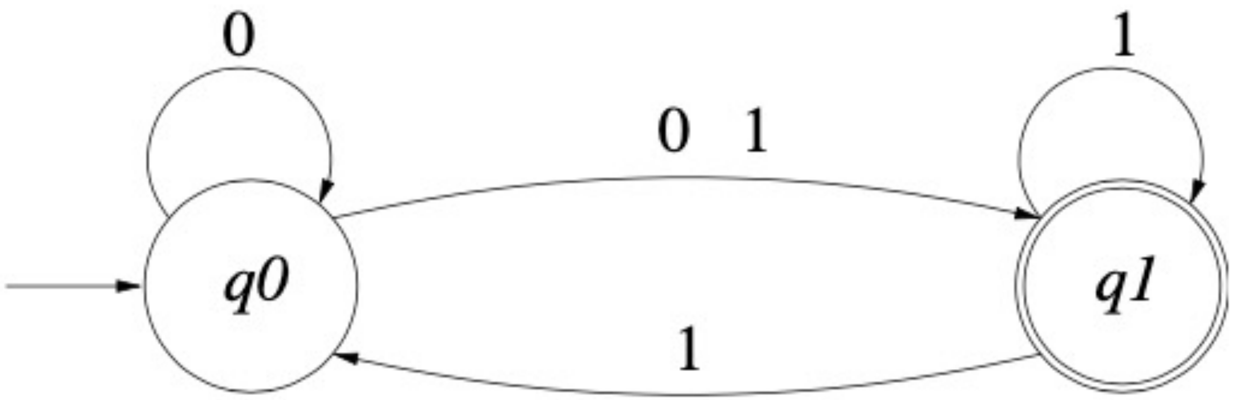
$$\delta'(q', x) = \delta'(\{q_1, q_2, \ldots, q_i\}, x) = \bigcup_{j=1}^{i} \delta(q_j, x) = \bigcup_{q \in q'} \delta(q, x)\,.$$

Si può dimostrare che M' è equivalente a M, ossia che

$$T(M') = T(M)$$

Esercizio

Esercizio Trasformare in FSA questo NDA:



Soluzione: Sia $M' = (Q', \delta', q_0', F') \in$ FSA

- $Q' = \{\emptyset, \{q_0\}, \{q_1\}, Q\}$
- $\{q_0\}$ stato iniziale
- $F' = \{\{q_1\}, Q\}$
- funzione di transizione δ definita:

Esercizio

- Applichiamo l’algoritmo per trasformare l’automa

Esercizio

- Applichiamo l'algoritmo per trasformare l'automa

la funzione di transizione $\delta : Q \times X \to 2^Q$ è definita dalla seguente tavola di transizione:

δ	q_0	q_1
0	$\{q_0, q_1\}$	$-$
1	$\{q_1\}$	$\{q_0, q_1\}$

Esercizio

- Applichiamo l'algoritmo per trasformare l'automa

la funzione di transizione $\delta : Q \times X \rightarrow 2^Q$ è definita dalla seguente tavola di transizione:

δ	q_0	q_1
0	$\{q_0, q_1\}$	$-$
1	$\{q_1\}$	$\{q_0, q_1\}$

Definiamo M' come segue:

$$M' = (Q', \delta', q_0', F') \text{ con } X = \{0,1\} \text{ come alfabeto di ingresso}$$

ove:

i) $Q' = 2^Q = 2^{\{q_0, q_1\}}$;

ii) $q_0' = \{q_0\}$;

iii) $F' = \{\{q_1\}, \{q_0, q_1\}\}$;

Esercizio

- Applichiamo l'algoritmo per trasformare l'automa

iv) la funzione di transizione δ' è definita come segue:

$$\delta' : Q' \times X \to Q'$$

Esercizio

- Applichiamo l'algoritmo per trasformare l'automa

iv) la funzione di transizione δ' è definita come segue:

$$\delta' : Q' \times X \to Q'$$

- $\delta'(\{q_0\},0) = \delta(q_0,0) = \{q_0, q_1\}$;
- $\delta'(\{q_0\},1) = \delta(q_0,1) = \{q_1\}$;
- $\delta'(\{q_1\},0) = \delta(q_1,0) =$ non è definita;
- $\delta'(\{q_1\},1) = \delta(q_1,1) = \{q_0, q_1\}$;
- $\delta'(\{q_0, q_1\},0) = \delta(q_0,0) \cup \delta(q_1,0) = \{q_0, q_1\}$;
- $\delta'(\{q_0, q_1\},1) = \delta(q_0,1) \cup \delta(q_1,1) = \{q_0, q_1\}$;

Esercizio

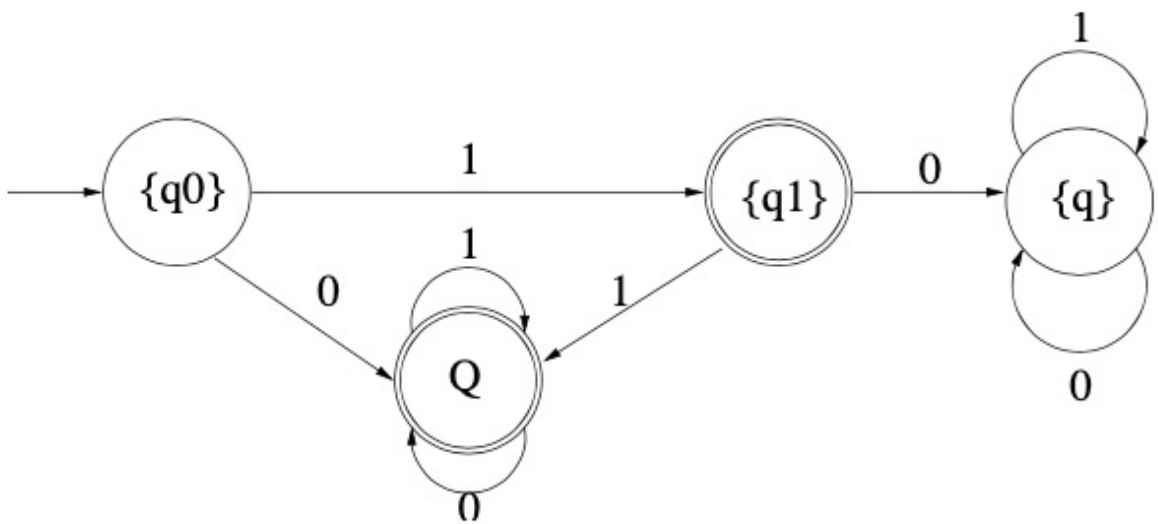
- Applichiamo l'algoritmo per trasformare l'automa

La tavola di transizione che riassume la definizione della funzione δ' è:

δ'	$\{q_0\}$	$\{q_1\}$	$\{q_0, q_1\}$
0	$\{q_0, q_1\}$	$-$	$\{q_0, q_1\}$
1	$\{q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1\}$

Esercizio

- Applichiamo l’algoritmo per trasformare l’automa



Con $\{q\}$ stato pozzo aggiunto per definire una funzione di transizione totale

Domande?

-It gets easier.
-Huh?
Every day, it gets a little easier.
-Yeah?
-But you gotta do it every day.
That's the hard part.
But it does get easier.