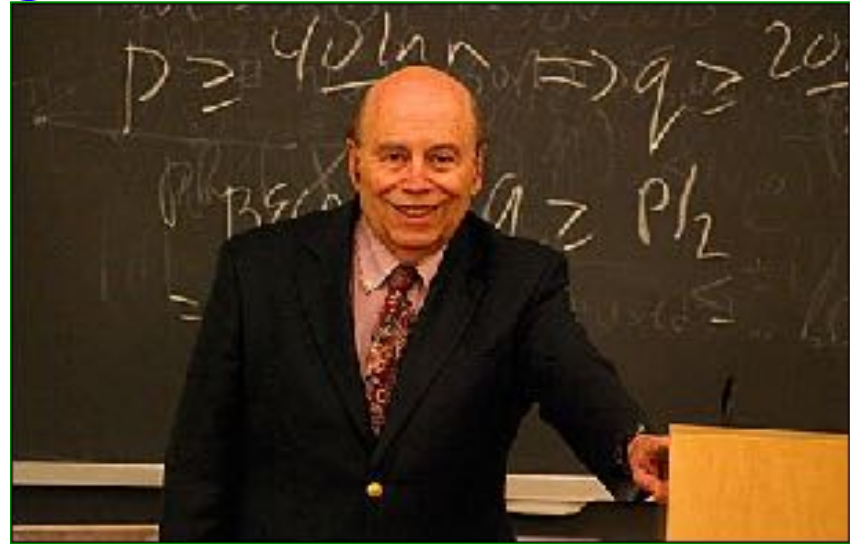


# Sommario

**1. Ancora qualche  
esempio di DFA  
Introduzione,  
definizione ed  
esempi di automa  
a stati finiti  
nondeterministici**



**Michael O. Rabin (1931)**

**2. Dimostrazione  
equivalenza tra  
automi a stati finiti  
nondeterministici e  
deterministici (1959)**

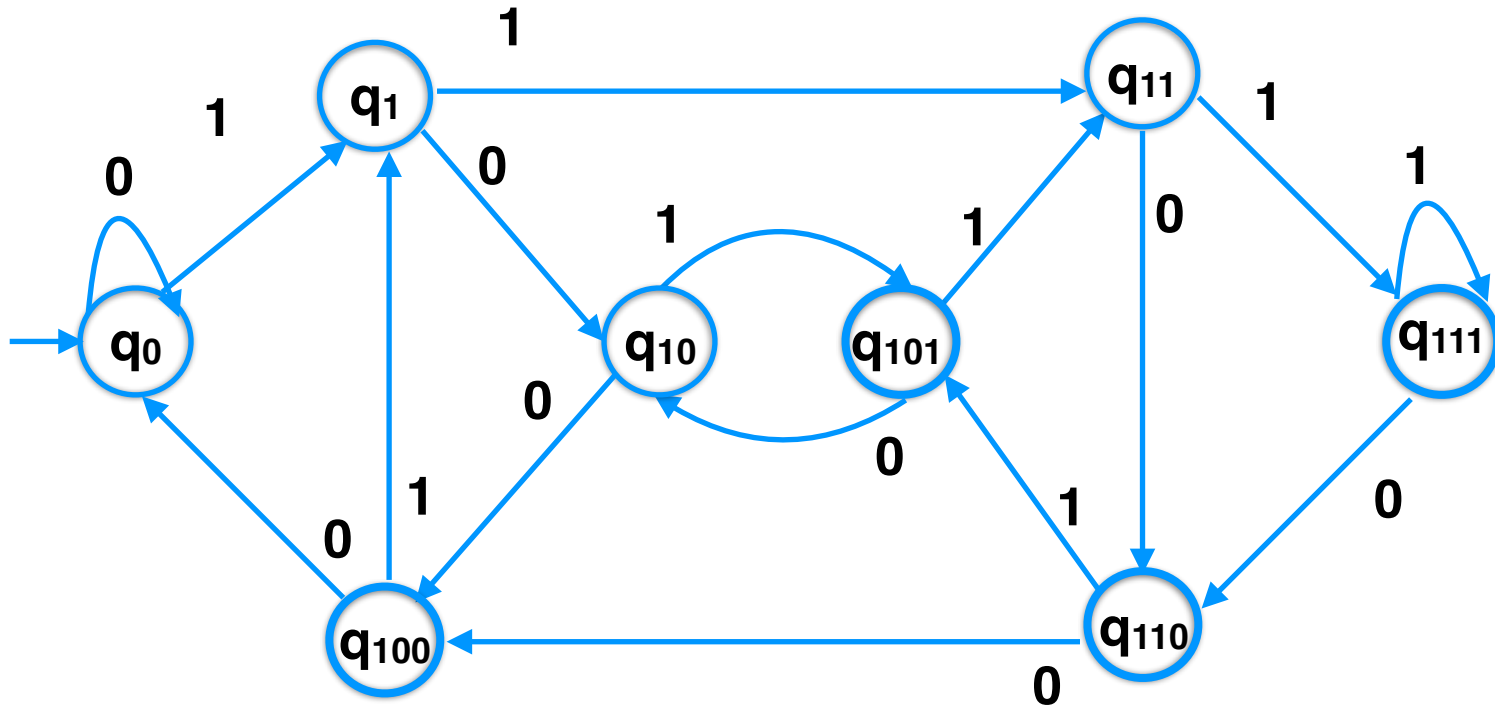


**Dana Scott (1932)**

# ESEMPIO 1

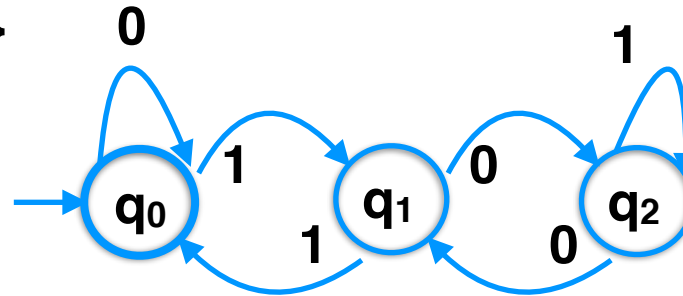
$L = \{c_1c_2\dots c_n1c_{n+1}c_{n+2} \mid c_i \text{ in } \{0,1\}, \text{ per } 1 \leq i \leq n+2\}$  è l'insieme di tutte le parole che hanno un 1 nella terz'ultima posizione.

Qui l'automa disegnato da Emilian Postulache:



## ESEMPIO 2

$L = \{x \mid x \text{ in } \{0,1\}^* \text{ e } x \text{ rappresenta in binario un intero multiplo di } 3\}$



Qui lo stato ricorda il resto della divisione per 3 della stringa binaria letta fino a quel momento.

Con  $x$  in  $\{0,1\}^*$  se  $(q_0, x0) \Rightarrow^* (q_1, 0)$ , vuol dire che  $x$  rappresenta un numero che diviso per 2 dà resto 1, quindi il suo doppio deve dare resto 2, mentre se aggiungo 1 al suo doppio ottengo un numero che è un multiplo di 3. Poiché  $x0$  rappresenta il numero che è il doppio del numero rappresentato da  $x$  e  $x1$  il doppio aumentato di 1, abbiamo la giustificazione delle due transizioni. Ripetendo il ragionamento per ogni stato si ottiene l'automa.

## ESEMPIO 3

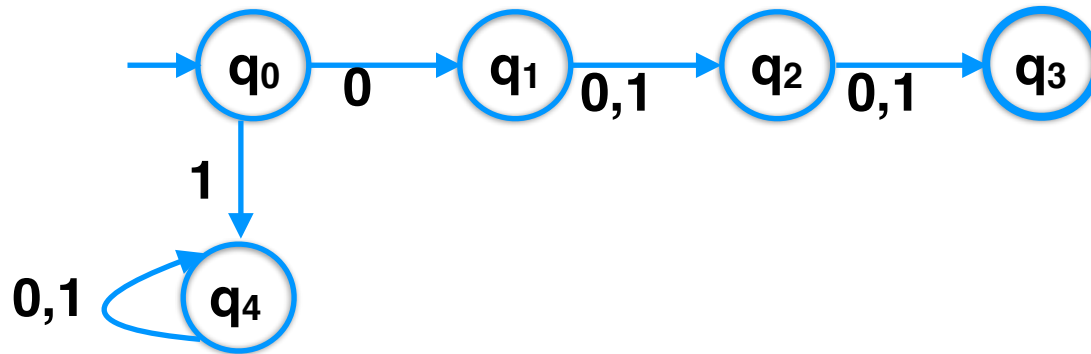
$L = \{ x \mid x \text{ in } \{0,1\}^* \text{ e } |x| = 3 \}$  è l'insieme (**finito**) delle parole binarie di lunghezza 3



Evidentemente il diagramma degli stati di un automa che accetta un insieme finito di parole non può avere cicli su un cammino dallo stato iniziale ad uno finale.

# ESEMPIO 4

$L = \{ x \mid x \text{ in } \{0,1\}^* \text{ e } |x| = 3 \}$  è l'insieme (**finito**) delle parole binarie di lunghezza 3, che cominciano per 0.



Evidentemente il diagramma degli stati di un automa che accetta un insieme finito di parole non può avere cicli su un cammino dallo stato iniziale ad uno finale.

# Operazioni sui linguaggi

Abbiamo dimostrato la chiusura della classe dei linguaggi accettati da un DFA,  $\mathcal{L}(\text{DFA})$ , rispetto alle operazioni insiemistiche:

$L, L' \subseteq \Sigma^*$

- **unione:**  $L \cup L'$
- **intersezione:**  $L \cap L'$
- **complemento:**  $\neg L = \Sigma^* - L$

Ricordiamo che queste operazioni **non sono indipendenti:**

$$\neg(X \cup X') = \neg X \cap \neg X'$$

$$\neg(X \cap X') = \neg X \cup \neg X'$$

Inoltre altre operazioni sugli insiemi si possono definire in funzioni di queste, per esempio la differenza  $L - L' = \{x \mid x \text{ è in } L \text{ ma non in } L'\}$ .  $L - L' = L \cap \neg L'$

# MICHAEL O. RABIN

Michael O. Rabin è nato nel 1931 a Breslau, Germany (oggi Wrocław, in Polonia). In 1935 è emigrato in Palestina, dove si è laureato. Ha poi perfezionato la sua preparazione con un Ph.D. alla Princeton nel 1956. Poco dopo viene invitato dall'IBM, nel centro di ricerca di New York, dove insieme a Dana Scott ha scritto l'articolo dal titolo "Finite Automata and Their Decision Problems", nel quale sono introdotti gli automi a stati finiti non deterministici e che valse loro, nel 1976, il **premio Turing**. Tra l'altro, ha anche introdotto gli automi a stati finiti probabilistici, il test di primalità noto come Miller-Rabin test. Questo test è un algoritmo randomizzato molto veloce. Più tardi ha introdotto un criptosistema, noto come Rabin cryptosystem, un sistema asimmetrico la cui sicurezza si basa sull'intrattabilità della fattorizzazione intera.

Ha anche inventato con Karp un ben noto algoritmo per il problema della ricerca di un pattern in un testo.

# Dana Scott

**E' nato nel 1932 a Berkeley in California, è professore emerito di Computer Science, Philosophy, and Mathematical Logic at Carnegie Mellon University, ed è ora in pensione.**

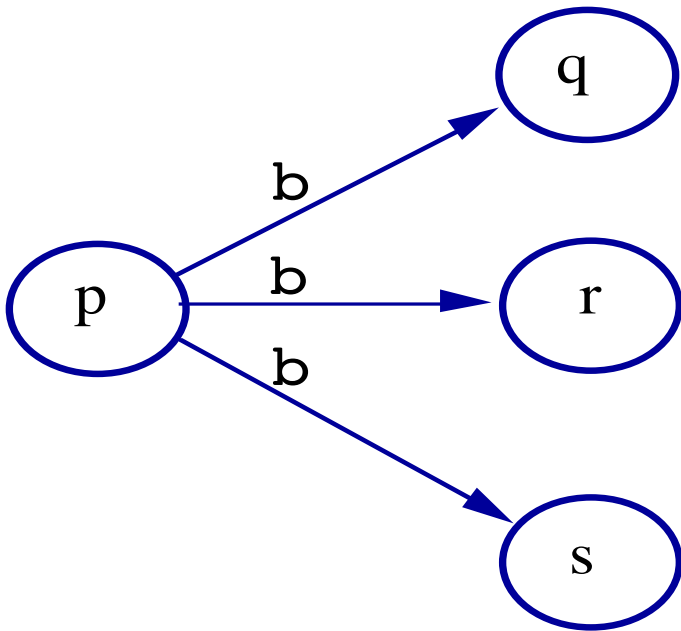
**Si è occupato di logica, in particolare di logica modale e di semantica dei linguaggi di programmazione, cercando di dotarla di una solida base matematica. Questo è sfociato nel cosiddetto approccio alla Scott-Strachey per la semantica denotazionale.**

**Ha anche scritto una prova alternativa a quella di Paul Cohen dell'indipendenza dell'ipotesi del continuo: non ci sono insiemi la cui cardinalità è strettamente compresa tra quella dei numeri naturali e quella dei reali.**

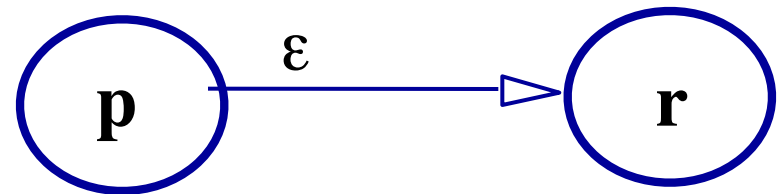


# AUTOMI A STATI FINITI NON DETERMINISTICI

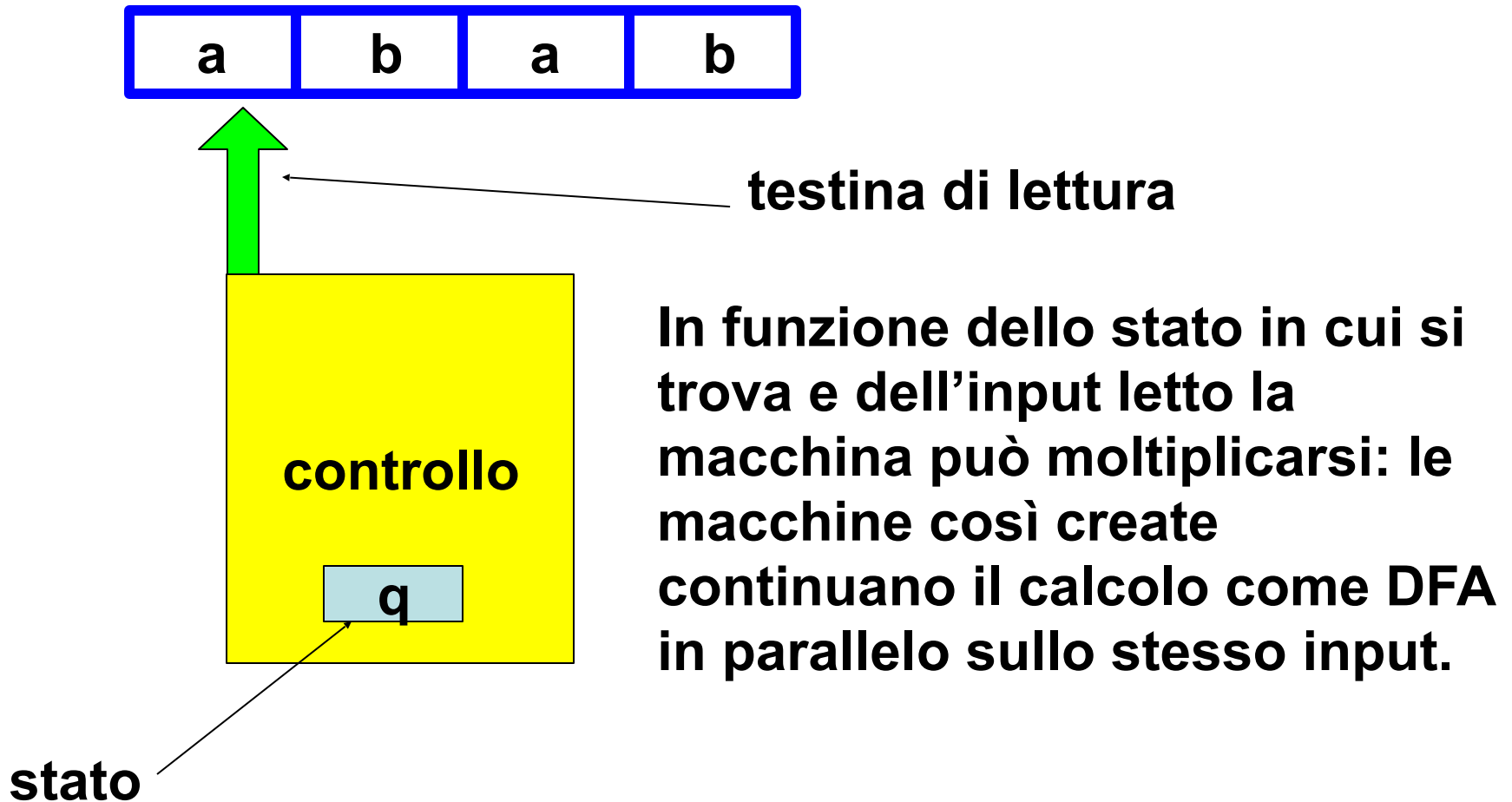
Un automa finito non deterministico può avere a disposizione più stati nei quali passare a seguito della lettura di un simbolo



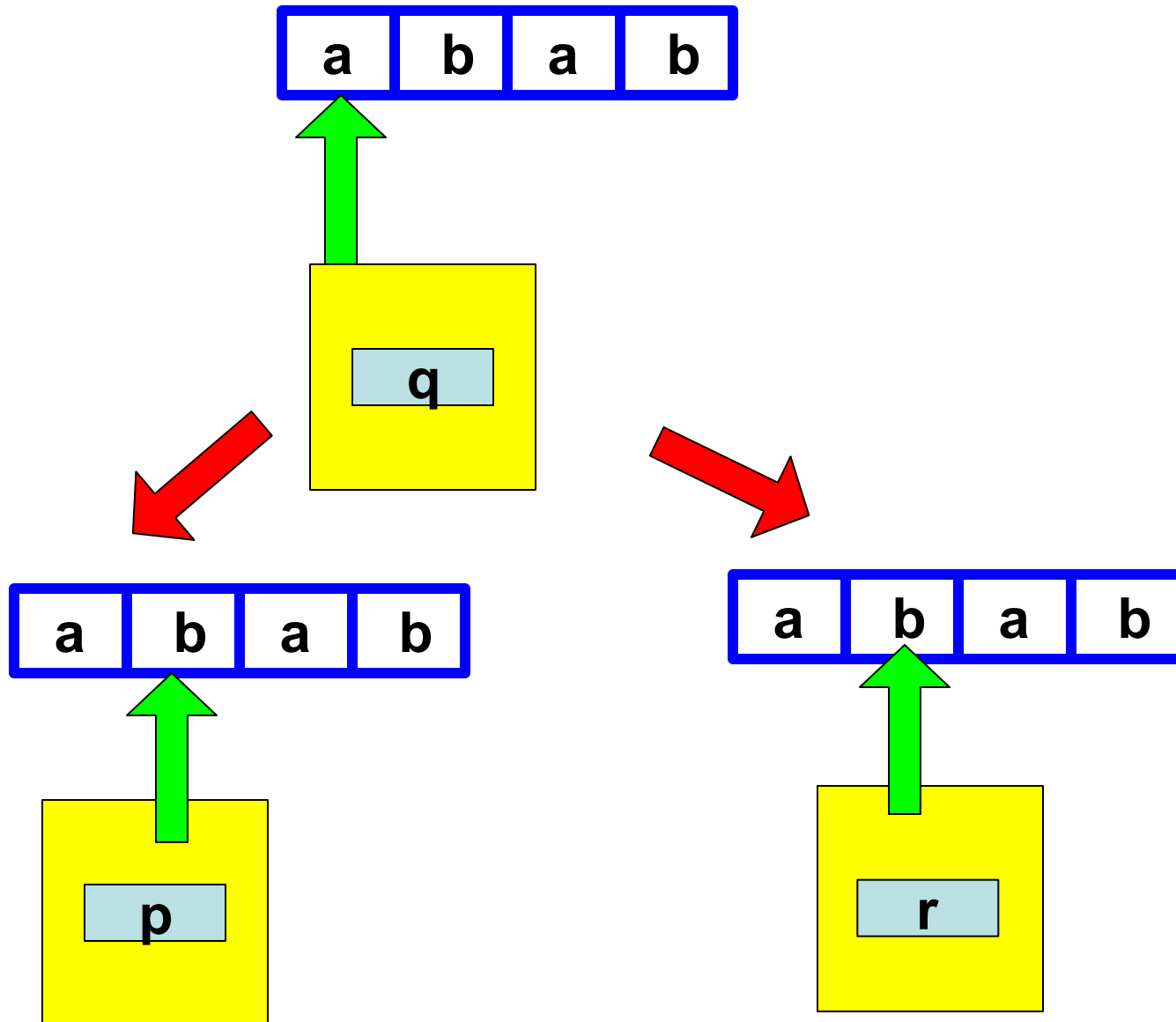
oppure può cambiare stato senza leggere e consumare input.



# Possibile (non unico) modello mentale



# Un esempio parziale di esecuzione



# Modello formale

Un **automa a stati finiti non deterministico**, in breve NFA (Nondeterministic Finite Automaton), è una quintupla  $M = (Q, \Sigma, \delta, q_0, F)$  dove

- $Q$  è l'insieme finito degli stati;
- $\Sigma$  è l'insieme finito dei simboli di input;
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow P(Q)$  è la funzione (totale) di transizione;
- $q_0$  è lo stato iniziale;
- $F \subseteq Q$  è l'insieme degli stati finali o di accettazione

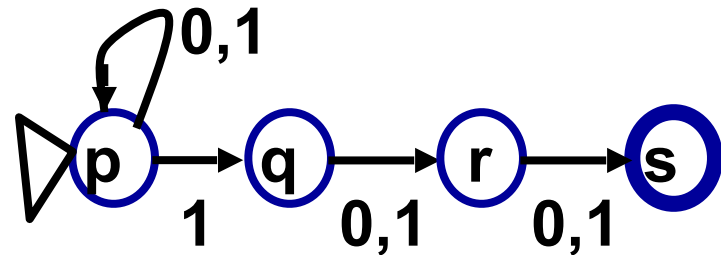
# Esempio

Consideriamo il linguaggio  $L = \{x \mid x \text{ è in } \{0,1\}^* \text{ e } x \text{ ha } 1 \text{ come terzultima lettera}\}$ .

Sia  $M = (Q, \Sigma, \delta, p, F)$  il DFA dato da:  $Q = \{p, q, r, s\}$ ,  
 $\Sigma = \{0,1\}$ ,  $F = \{s\}$

La funzione di transizione  $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow P(Q)$  è descritta nella tabella:

$\delta$	1	0	$\varepsilon$
p	{p,q}	{p}	$\emptyset$
q	{r}	{r}	$\emptyset$
r	{s}	{s}	$\emptyset$
s	$\emptyset$	$\emptyset$	$\emptyset$



# CONFIGURAZIONI E PASSI

Una **configurazione** di un NFA  $M$  è un elemento di  $Q \times \Sigma^*$ .

Per un input  $x \in \Sigma^*$ ,

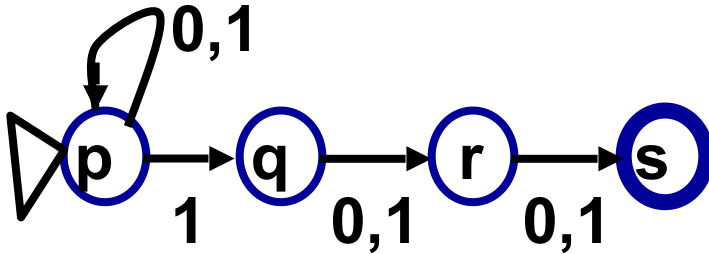
la **configurazione iniziale** o **di partenza** è  $(q_0, x)$

La relazione "**porta a**" (in un passo)  $\Rightarrow_M$  è una relazione binaria su  $Q \times \Sigma^*$ , così definita:

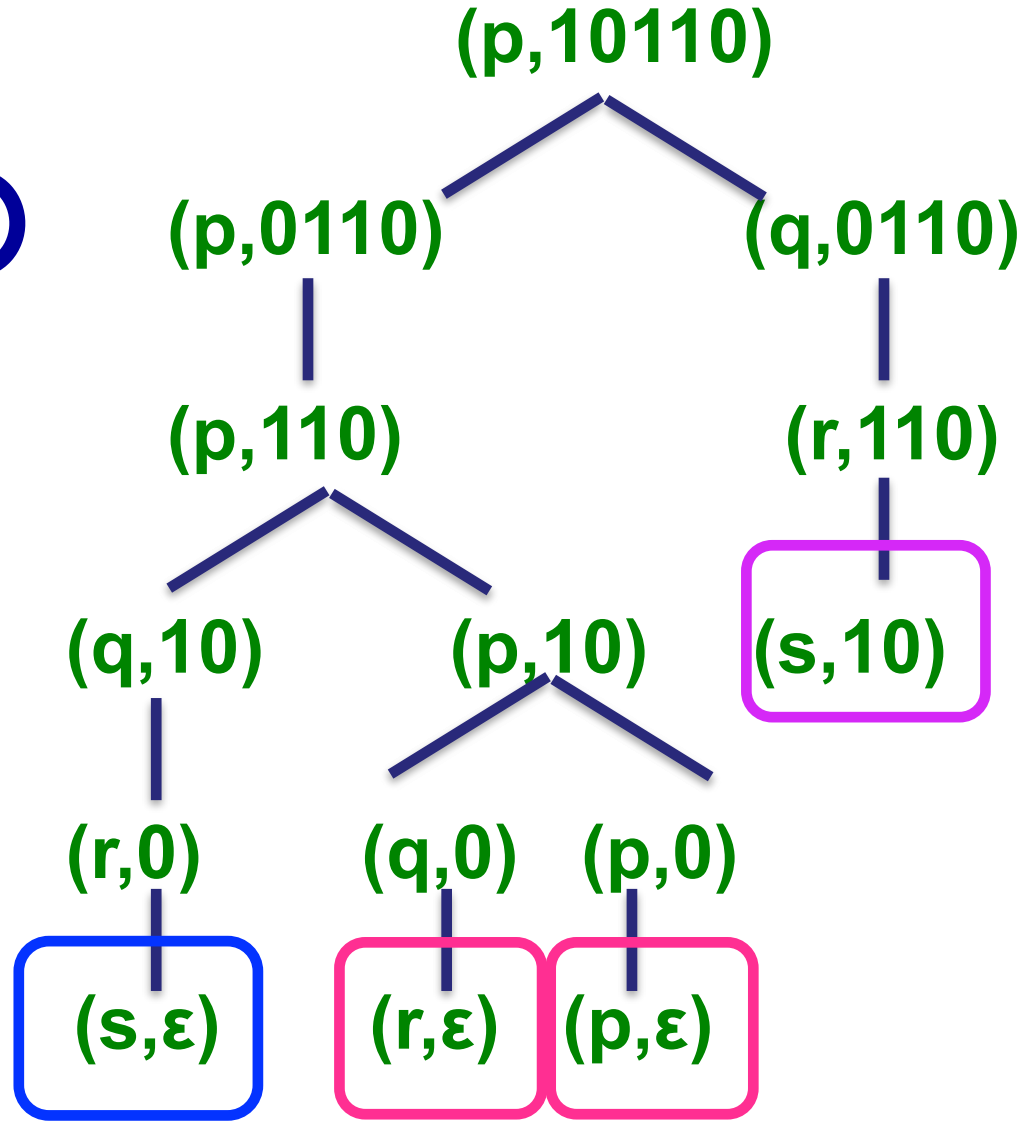
se  $p, q \in Q$ ,  $a \in \Sigma$  e  $x \in \Sigma^*$ ,

allora  $(p, ax) \Rightarrow_M (q, x)$  se e solo se  $q \in \delta(p, a)$

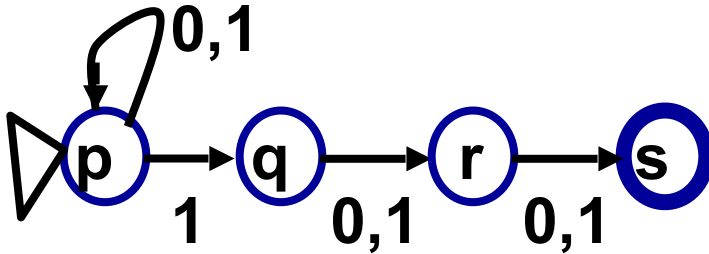
# Esempio di parola accettata



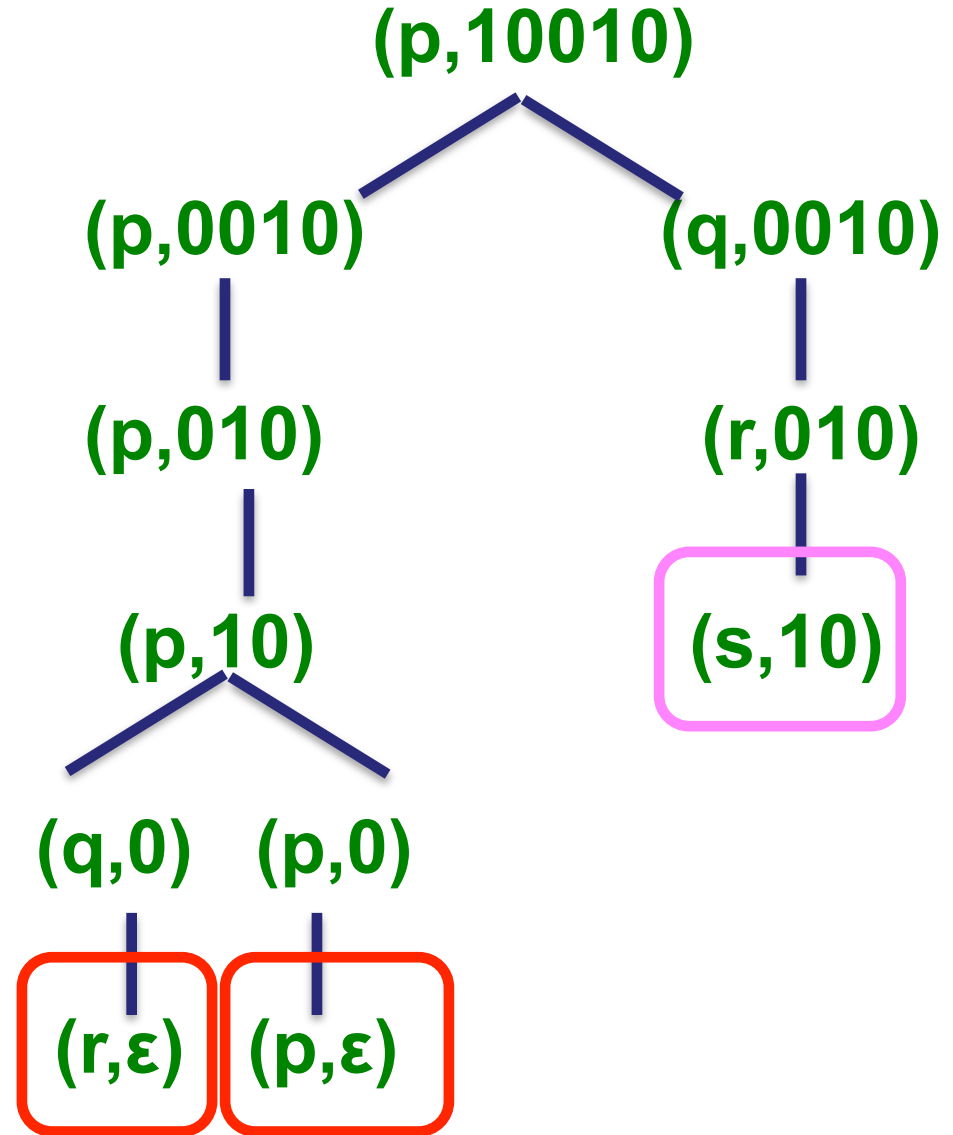
Per questo automa e l'input 10110 la configurazione iniziale è  $(p, 10110)$ . Si può usare un albero per rappresentare tutte le configurazioni raggiunte a partire da quella iniziale.



# Esempio di parola non accettata



Per questo automa e l'input 10010 la configurazione iniziale è  $(p, 10010)$ .  
Usiamo ancora un albero per rappresentare tutte le configurazioni raggiunte a partire da quella iniziale.





# Accettazione

Dato un NFA  $M = (Q, \Sigma, \delta, q_0, F)$ , una **parola**  $x \in \Sigma^*$  **è accettata** da  $M$  se  $\exists q \in F$  tale che

$$(q_0, x) \Rightarrow_M^* (q, \varepsilon)$$

dove  $\Rightarrow_M^*$  la chiusura riflessiva e transitiva di  $\Rightarrow_M$

e il **linguaggio accettato** da  $M$  è

$$L(M) = \{x \mid x \in \Sigma^* \text{ ed è accettata da } M\}$$

**La classe dei linguaggi accettati** da un NFA è

$$\mathcal{L}(\text{NFA}) = \{L \mid \exists M \in \text{NFA e } L(M) = L\}$$

# Rifiuto

Dato un NFA  $M = (Q, \Sigma, \delta, q_0, F)$ , una **parola**  $x \in \Sigma^*$  **è rifiutata** da  $M$  se  $\forall q$  tale che  $(q_0, x) \Rightarrow_M^* (q, \varepsilon)$  q non è in  $F$ .

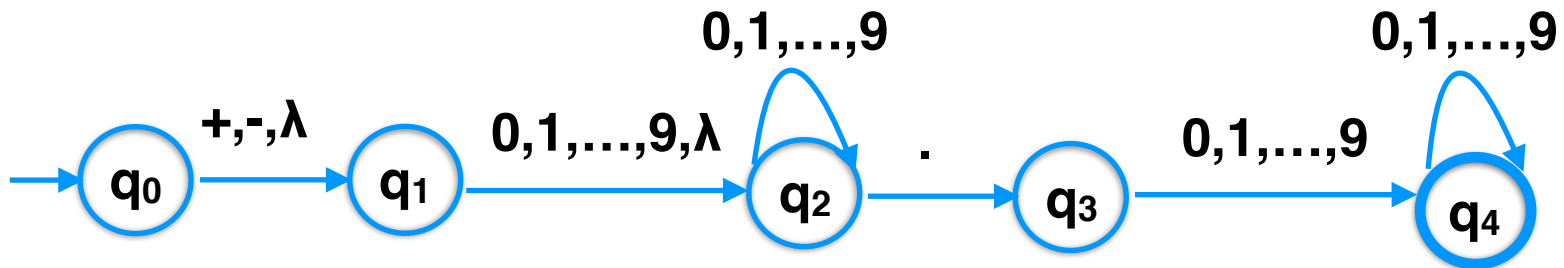
Può capitare che  $(q_0, x) \Rightarrow_M^* (q, ay)$ , anche con  $q$  in  $F$ ,  $y \in \Sigma^*$  e

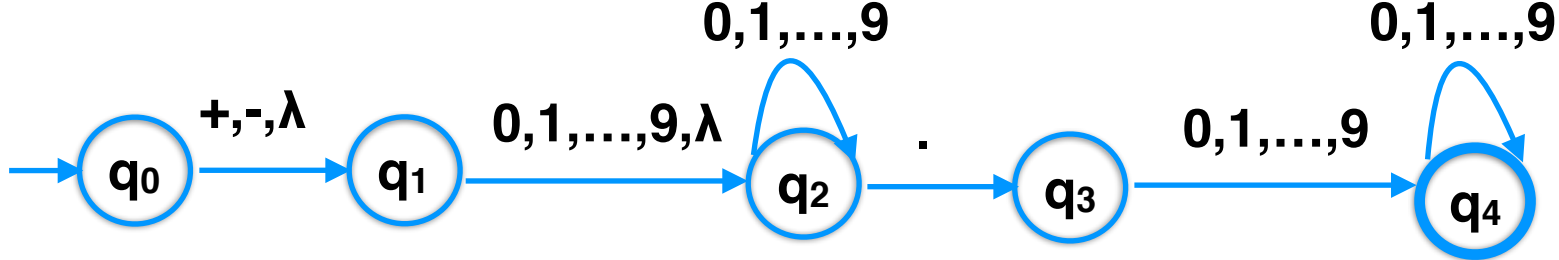
$\delta(q, a) = \emptyset$ , quindi la parola non viene tutta letta.

Questo è un cammino bloccato che non può influire sull'accettazione o meno di  $x$ .

# Esempio di NFA con $\epsilon$ -mosse

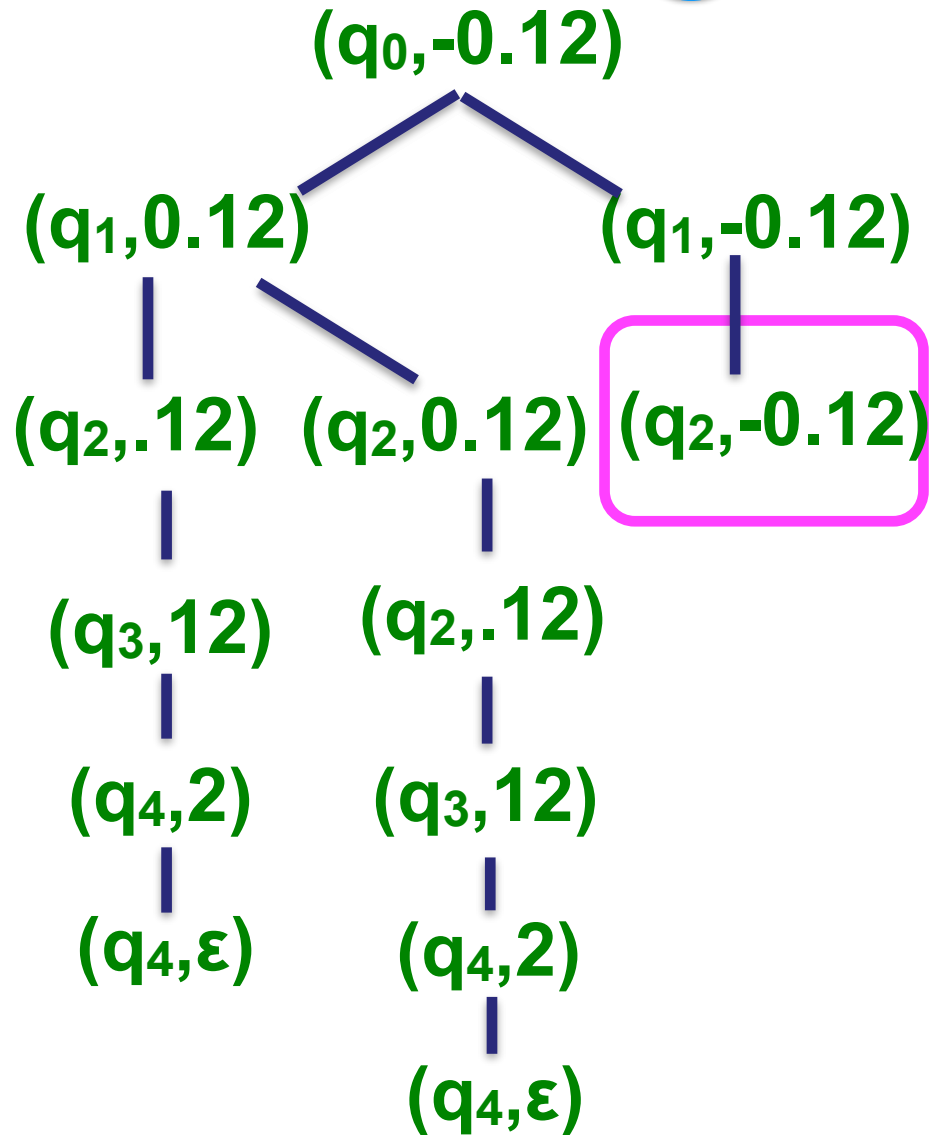
Costruiamo un NFA che accetta le stringhe che rappresentano i numeri decimali, nella notazione anglosassone:





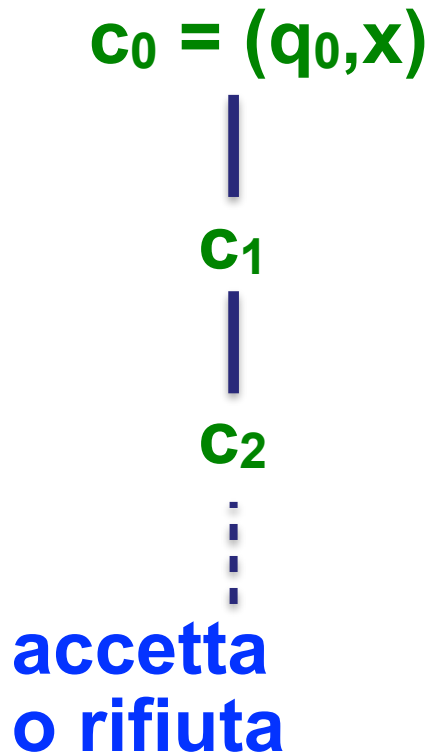
**Esempio di  
computazione  
su -0.12**

Invece la parola  $+ - 0.1$   
non è accettata  
perché la funzione di  
transizione dà  
l'insieme vuoto sulla  
coppia  $(q_1, -)$

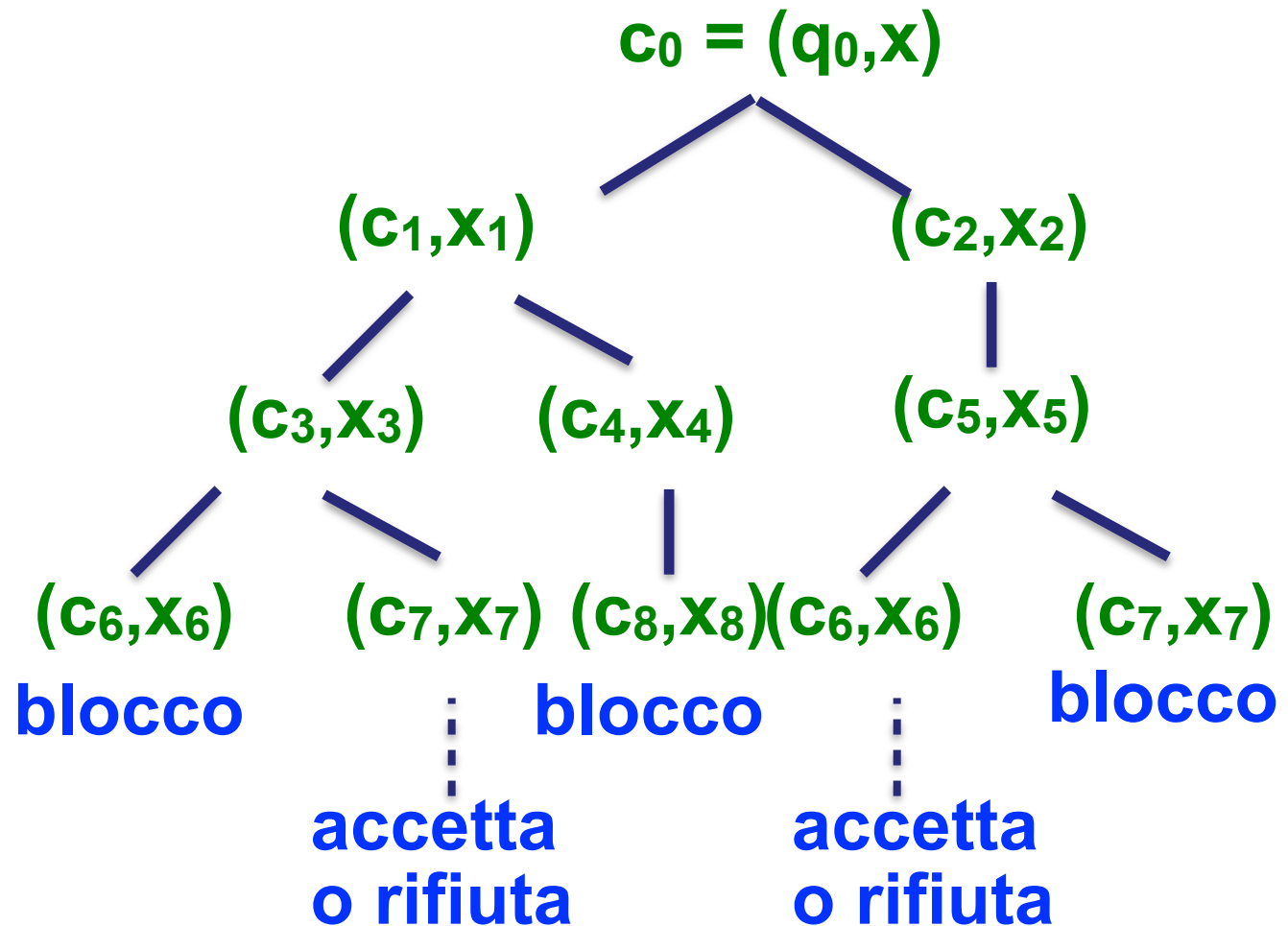


# DETERMINISMO E NON

Una computazione  
deterministica



Una computazione  
non deterministica



# Non determinismo e potere computazionale per gli automi a stati finiti

Nel caso degli automi a stati finiti il non determinismo non aumenta il potere computazionale, detta  $\mathcal{L}(\text{NFA})$  la classe dei linguaggi accettati dagli automi finiti non deterministici e  $\mathcal{L}(\text{DFA})$  quella dei linguaggi accettati dai deterministici:

**TEOREMA**  $\mathcal{L}(\text{DFA}) = \mathcal{L}(\text{NFA})$

# CONFRONTO TRA NFA e DFA

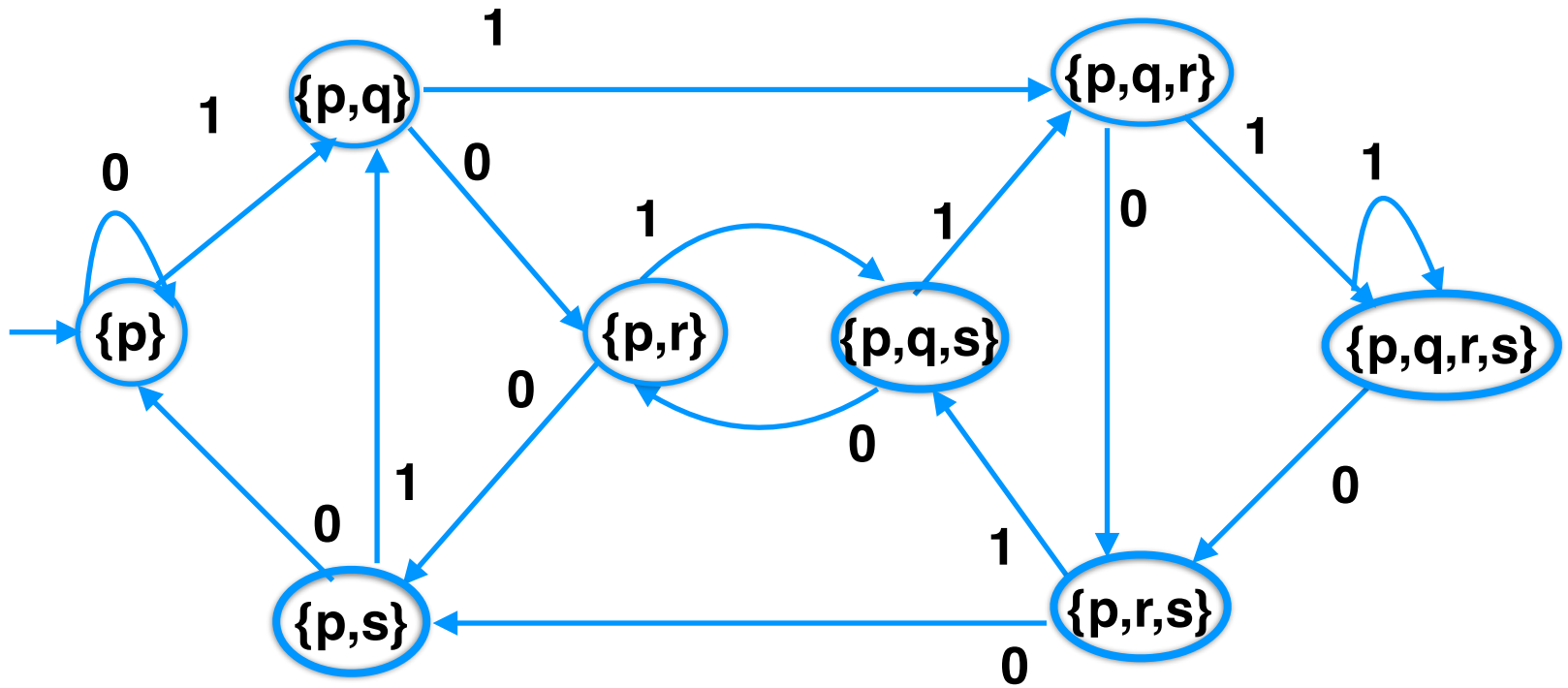
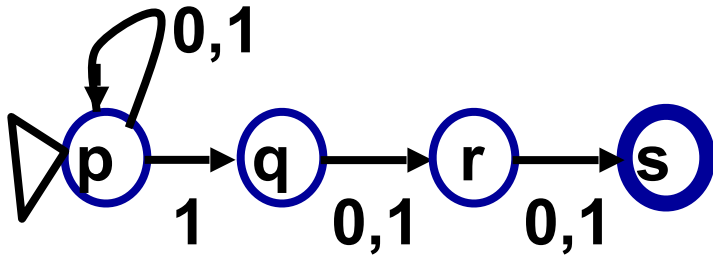
Ogni DFA può essere visto come un caso particolare di NFA e quindi

$$\mathcal{L}(\text{DFA}) \subseteq \mathcal{L}(\text{NFA})$$

Ma possiamo anche dimostrare che per ogni NFA  $M$ , esiste un DFA  $M1$  **equivalente** a  $M$ , cioè tale che  $L(M)=L(M1)$  e cioè che

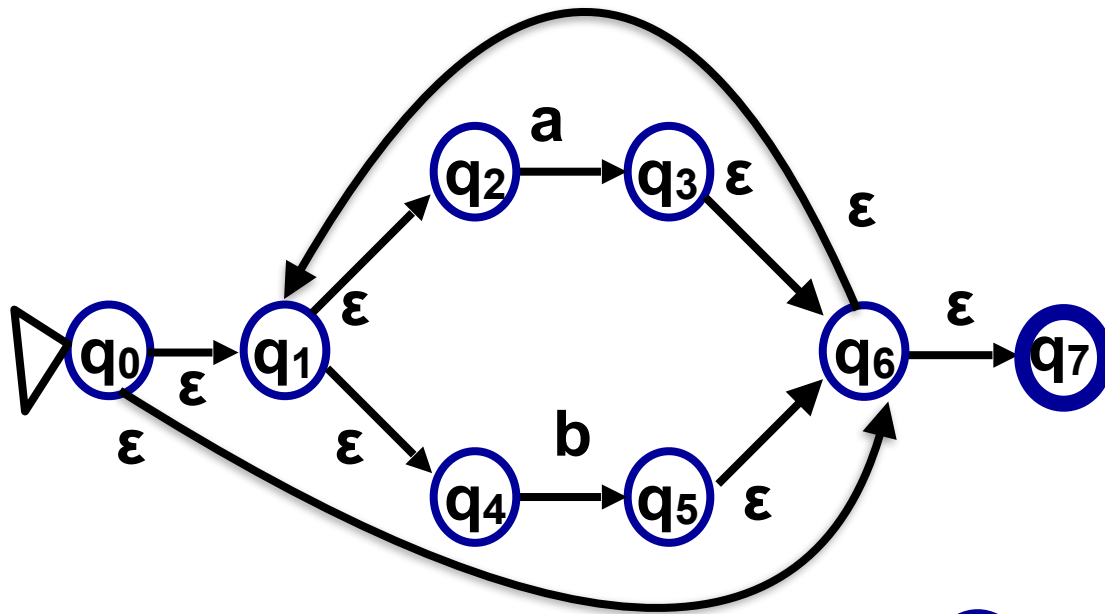
$$\mathcal{L}(\text{NFA}) \subseteq \mathcal{L}(\text{DFA})$$

# ESEMPIO 1

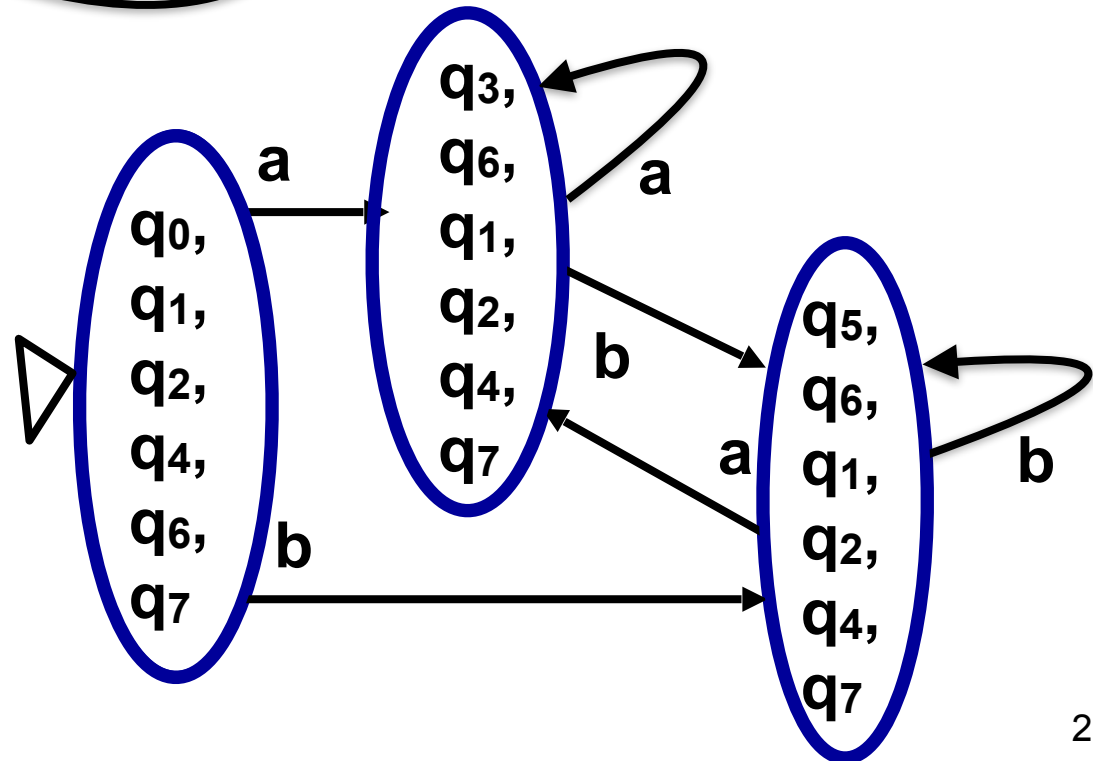




# ESEMPIO



$\epsilon$ -closure( $q$ )=  
l'insieme degli  
stati  
raggiungibili da  
 $q$  con solo  $\epsilon$ -  
mosse.  
Qui le parentesi  
graffe sono  
omesse



# Teorema di esistenza di un DFA equivalente a un NFA

Dato un NFA  $M = (Q, \Sigma, \delta, q_0, F)$  costruiamo un DFA  
equivalente  $A = (Q', \Sigma, \delta', q'_0, F')$ :

$Q' = P(Q)$  all'insieme che contiene

$q'_0 = \varepsilon\text{-closure}(q_0)$

$F' = \{ X \mid X \text{ è in } P(Q) \text{ e } X \text{ contiene uno stato di } F \}$

Per ogni  $X$  in  $P(Q)$  e  $a$  in  $\Sigma$ , lo stato raggiunto da  $X$   
leggendo  $a$  è l'unione delle  $\varepsilon$ -closure di tutti gli stati  
raggiunti leggendo  $a$  da uno stato di  $X$ :

$$\delta'(X, a) = \bigcup_{p \in X} \left( \bigcup_{q \in \delta(p, a)} \varepsilon\text{-closure}(q) \right)$$

Poiché  $(q_0, x) \Rightarrow_M^* (q, \varepsilon)$  con  $q$  in  $F$  sse  $(q'_0, x) \Rightarrow_A^* (X, \varepsilon)$

con  $X$  in  $F'$ , i due automi sono equivalenti.

# Algoritmo di costruzione di un DFA equivalente a un NFA

Algoritmo DFAeqNFA

**input** un NFA con  $\varepsilon$ -mosse  $M=(\Sigma, Q, \delta, q_0, F)$

**output** un DFA  $M'=(\Sigma, Q', \delta', q_0', F')$  equivalente

Inizializza  $Q'$  all'insieme che contiene

$\varepsilon$ -closure( $q_0$ ) come unico elemento non marcato.

**while** c'è uno stato non marcato  $T$  in  $Q'$  **do**

  {marca  $T$ ;

**for** ogni simbolo input  $a$  in  $\Sigma$  **do**

$\{V = \cup_{q \in T} \delta(q, a)\};$

$V' = \cup_{q \in V} \varepsilon\text{-closure}(q);$

**if**  $V'$  non appartiene a  $Q'$  **then**

      aggiungi  $V'$  come stato non marcato in  $Q'$ ;

      poni  $\delta'(T, a) = V'$

    }

  }

# Costruzione di un DFA equivalente a un NFA: conclusione

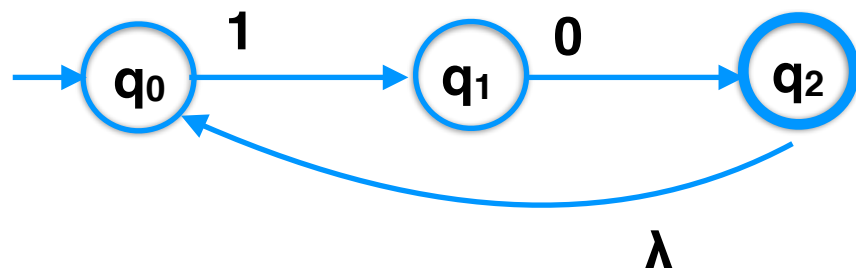
Con l'algoritmo DFAeqNFA abbiamo costruito l'insieme degli stati e la funzione di transizione del DFA  $M'=(\Sigma,Q',\delta',q_0',F')$  equivalente al dato NFA  $M=(\Sigma,Q,\delta,q_0,F)$ .

Lo stato iniziale di  $M'$ ,  $q_0'$ , è l'insieme degli stati raggiungibili da quello iniziale di  $M$  utilizzando solo  $\epsilon$ -mosse, cioè  $q_0' = \epsilon\text{-closure}(q_0)$ .

Gli stati finali di  $M'$  sono gli stati di  $Q'$  che contengono uno stato finale di  $M$

Gli stati di  $Q'$  sono tutti gli stati raggiungibili dallo stato iniziale. Gli stati di  $Q'$  sono al più  $2^n$ , se  $n$  è il numero degli stati dell'NFA di partenza  $M$ .

# Esempio di automa deterministico con $\lambda$ mosse



**Il linguaggio accettato è  $\{(10)^n \mid n > 0\}$ . Ogni cammino di computazione è lineare perché da  $q_2$  non c'è altra possibilità che l'esecuzione della  $\lambda$  - mossa. Potrebbe modellare un sistema in cui due sole azioni sono ammesse una di seguito all'altra e poi il sistema "automaticamente" si rimette nello stato iniziale.**

# Le epsilon mosse (da sole) non fanno nondeterminismo

Un automa a stati finiti deterministico con epsilon mosse, in breve  $\epsilon$ -DFA, è una quintupla  $M = (Q, \Sigma, \delta, q_0, F)$  dove

- $Q$  e  $\Sigma$  sono l'insieme degli stati e dei simboli di input;
- $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow P(Q)$  è la funzione di transizione, tale che
  - $|\delta(q, a)| \leq 1$  per ogni  $a$  in  $\Sigma \cup \{\epsilon\}$  e ogni  $q$  in  $Q$
  - se  $|\delta(q, \epsilon)| = 1$  allora  $\delta(q, a) = \emptyset$
- $q_0$  è lo stato iniziale e  $F \subseteq Q$  è l'insieme degli stati finali;

# Esempi di NFA

**Si costruisca un NFA che accetta il linguaggio:**

$$L = \{x010y \mid x,y \text{ in } \{0,1\}^*\}$$

**e il DFA equivalente**