Logica e Modelli Computazionali

Operazioni e Espressioni Regolari

Marco Console

Ingegneria Informatica e Automatica, Sapienza Università di Roma

Operazioni Regolari

Operazioni su Linguaggi

- **Definizione**. Un **linguaggio** è un insieme di stringhe
- **Dati due linguaggi** $\mathcal{L}_1, \mathcal{L}_2$ possiamo costruire altri linguaggi applicando le seguenti operazioni
- **Operazioni Insiemistiche.**
  - **Unione.** $\mathcal{L}_1 \cup \mathcal{L}_2 = \{ s \mid s \in \mathcal{L}_1 \; oppure \; s \in \mathcal{L}_2 \}$
  - **Intersezione.** $\mathcal{L}_1 \cap \mathcal{L}_2 = \{ s \mid s \in \mathcal{L}_1 \; e \; s \in \mathcal{L}_2 \}$
  - **Complemento.** $\overline{\mathcal{L}_1} = \{ s \mid s \notin \mathcal{L}_1 \}$
- **Operazioni su Stringhe.**
  - **Concatenazione.** $\mathcal{L}_1 \circ \mathcal{L}_2 = \{ c_1 \dots c_k d_1 \dots d_l \mid c_1 \dots c_k \in \mathcal{L}_1 \; e \; d_1 \dots d_l \in \mathcal{L}_2 \}$
  - **Star.** $\mathcal{L}_1^* = \{ s_1 \dots s_k \mid con \; k \geq 0 \; e \; s_1 \dots s_k \in \mathcal{L}_1 \}$

Operazioni su Linguaggi Regolari

- **Definizione 2**. Un **linguaggio $\mathcal{L}$ è detto regolare** se esiste un **ASFD A tale che $L(A) = \mathcal{L}$**

- **Teorema 1**. Dati due **linguaggi regolari $\mathcal{L}_1, \mathcal{L}_2$ tutti i seguenti linguaggi sono regolari**.
  1. **Unione.** $\mathcal{L}_1 \cup \mathcal{L}_2 = \{ s \mid s \in \mathcal{L}_1 \; oppure \; s \in \mathcal{L}_2 \}$
  2. **Intersezione.** $\mathcal{L}_1 \cap \mathcal{L}_2 = \{ s \mid s \in \mathcal{L}_1 \; e \; s \in \mathcal{L}_2 \}$
  3. **Complemento.** $\overline{\mathcal{L}_1} = \{ s \mid s \notin \mathcal{L}_1 \}$
  4. **Concatenazione.** $\mathcal{L}_1 \circ \mathcal{L}_2 = \{ c_1 \dots c_k d_1 \dots d_l \mid c_1 \dots c_k \in \mathcal{L}_1 \; e \; d_1 \dots d_l \in \mathcal{L}_2 \}$
  5. **Star.** $\mathcal{L}_1^* = \{ s_1 \dots s_k \mid con \; k \geq 0 \; e \; s_1 \dots s_k \in \mathcal{L}_1 \}$

- Tali operazioni vengono spesso chiamate operazioni regolari

Automi Epsilon

- Per dimostrare Teorema 1, introduciamo la famiglia **ϵ-ASFND** di Automi a Stati Finiti
  - Come vedremo gli **ϵ-ASFND** sono equivalenti agli **ASFND** e quindi agli **ASFD** ☺
  - Utilizzare questi automi nelle nostre prove le renderà molto più snelle
  - Inoltre (Sipser, 2022) utilizza questa definizione per gli ASFND

- **Definizione**. Un **ϵ-ASFND** $A =< \Sigma, Q, \delta, q_0, F >$ è un ASFND tale che
  1. $\Sigma = \{a_1, \dots, a_n\}$ è l'***alfabeto*** di input, assumiamo che il simbolo ϵ non sia in Σ;
  2. Q è un insieme finito detto ***insieme degli stati*** ;
  3. $I \in Q$ è lo ***stato iniziale*** ;
  4. $F \subseteq Q$ è un ***insieme degli stati finali***;
  5. $\delta: Q \times \Sigma \cup \{\epsilon\} \to P(Q)$ è una funzione da $Q \times (\Sigma \cup \{\epsilon\})$ a $P(Q)$, chiamata ***funzione di transizione***

- Intuitivamente, un **ϵ-ASFND** può **cambiare stato** utilizzando una transizione col simbolo ϵ **senza consumare** un simbolo della stringa dell'input.
  - Se utilizzando una transazione con simbolo ϵ se disponibile nello stato

Esecuzioni di un Automa a Pila – Preliminari

- Sia s una stringa sull'alfabeto Σ tale che $\epsilon \notin \Sigma$ e $\mathrm{T} \subseteq \Sigma$ un alfabeto.

- **Definizione**. La **restrizione di s ad Σ'** ($s_{|\mathrm{T}}$) è la stringa ottenuta eliminando da s tutti i simboli $c \notin T$

- **Definizione**. Una **ϵ-estensione** di s è una stringa s' sull'alfabeto $\Sigma \cup \{\epsilon\}$ tale che ${s'}_{|\Sigma} = s$
  - La restrizione di s' su Σ coincide con s
  - In altre parole, s' può aggiungere solamente il simbolo ϵ ad s ma un numero arbitrario di volte

- **Esempio**. La restrizione $s_{|T}$ della stringa $s = "asd"$ su $\Sigma = \{a, s, d\}$ all'alfabeto $T = \{a, d\}$ è la stringa $"ad"$

- **Esempio**. La stringa $s' = "a\epsilon s\epsilon d"$ è una ϵ-estensione di $s = "asd"$ su $\Sigma = \{a, s, d\}$ ${s'}_{|\Sigma} = s$
- **Esempio**. La stringa $s'' = "a\epsilon ss\epsilon d"$ non è una ϵ-estensione di $s = "asd"$ su $\Sigma = \{a, s, d\}$ ${s''}_{|\Sigma} = "assd"$

Esecuzioni

- Siano $A =< \Sigma, Q, \delta, I, F >$ un ϵ-ASFND e $s =$ "$c_1 c_2 \dots c_n$" $\in \Sigma^*$ una stringa con $|s| = n$

- **Definizione**. Una **esecuzione di A su S** è una sequenza $(q_1, \dots, q_{k+1}) \in Q^{k+1}$ di $k+1$ elementi di Q tale che esiste una ϵ-estensione "$x_1 x_2 \dots x_k$" di s tale che
  - $\boldsymbol{q_1 = I}$ (**intuitivamente**, il primo stato è quello iniziale)
  - $\boldsymbol{q_{i+1} \in \delta(q_i, x_i)}$ **per** $\boldsymbol{i = 1, \dots, k}$ (**intuitivamente**, ogni stato appartiene all'insieme di quelli che possono essere raggiunti consumando il simbolo corrente della ϵ-estensione considerata)

- **Definizione**. Una esecuzione di $(q_1, \dots, q_{k+1})$ di A su S è **accettante** se il suo stato finale $\boldsymbol{q_k}$ **è in F**

Linguaggio Riconosciuto

- **Definizione**. Dato un **ϵ-ASFND** $A =< \Sigma, Q, \delta, q_0, F >$ e una stringa $x \in \Sigma^*$
  - x è **accettata** da A se **esiste almeno una** esecuzione accettante di A su x
  - Altrimenti, x è **rifiutata**

- **Definizione**. Sia $A =< \Sigma, Q, \delta, q_0, F >$ un **ϵ-ASFND** . **Il linguaggio riconosciuto da A** è il linguaggio $L(A)$ sull'alfabeto Σ tale che

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

- **Domanda**. È vero che ogni linguaggio riconosciuto da un **ϵ-ASFND** è regolare?
  - Ovvero, possiamo sempre definire un **ASFD** equivalente a un **ASFND**?

Equivalenza

- **Teorema**. Per ogni **ϵ-ASFND** A esiste un **ASFND** A' tale che $L(A) = L(A')$
- **Corollario**. Il linguaggio riconosciuto da un è **ϵ-ASFND** è regolare

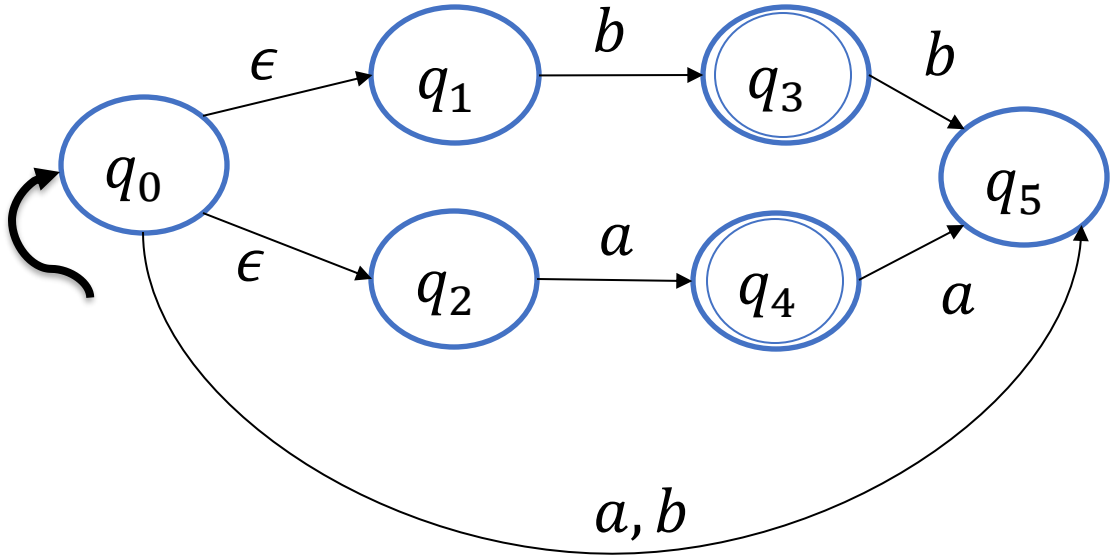
- **Dimostrazione.** L'idea della dimostrazione è quella di costruire un **ASFND** A' equivalente a A

- **Definizione**. Uno stato **$q' \in Q$ è ϵ-ragiungibile da $q \in Q$ in A se** esiste una sequenza di stati $(q_1, q_2, \dots, q_n) \in Q^n$ tale che
  - $q_1 \in \delta(q, \epsilon)$
  - $q_i \in \delta(q_{i-1}, \epsilon)$ per $i = 2, \dots, n$
  - $q' \in \delta(q_n, \epsilon)$

- **Intuizione.** **$q' \in Q$ è ϵ-ragiungibile da $q \in Q$** se può essere raggiunto solo con passi ϵ, senza cioè consumare simboli dell'input

- **Definizione**. La **ϵ-chiusura E(q, A) di $q \in Q$ su A** è l'insieme di tutti gli stati ϵ-raggiungibili da q in A

Esempio di Computazione



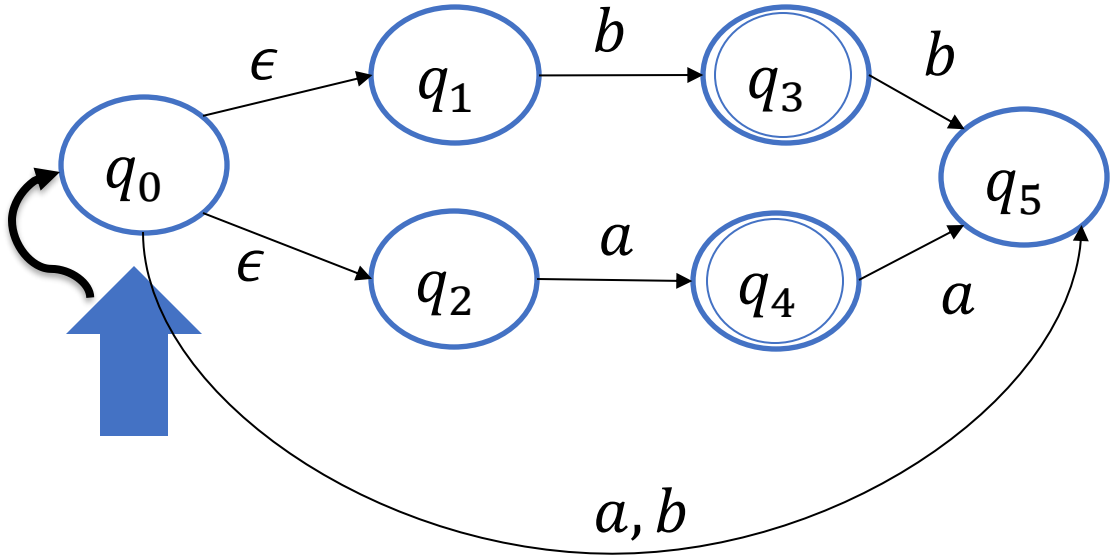
(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

INPUT

A	B	B	B

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INP

A	B	B	B

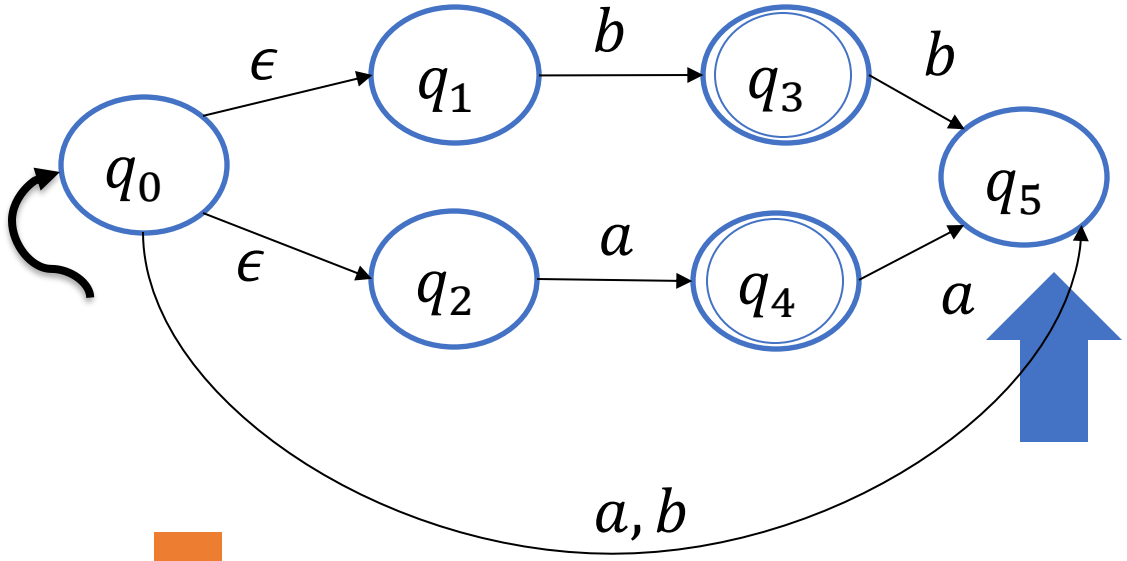
STATO CORRENTE

q_0

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT

A	B	B	B

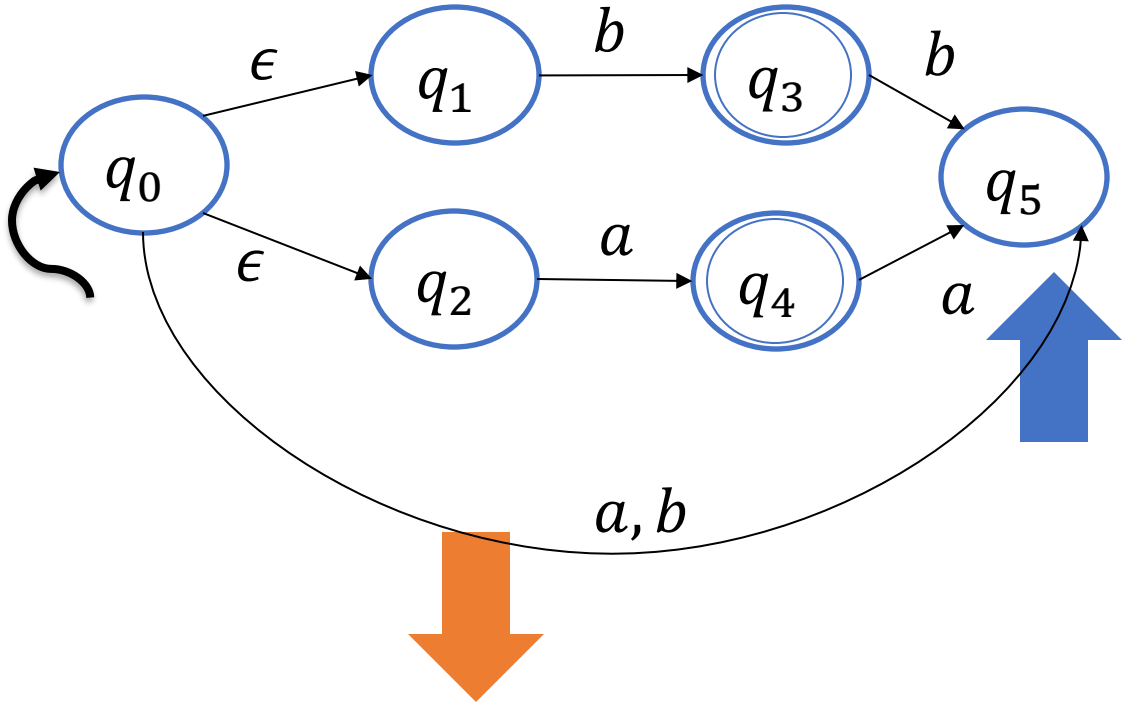
STATO CORRENTE

q_5

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT

A	B	B	B

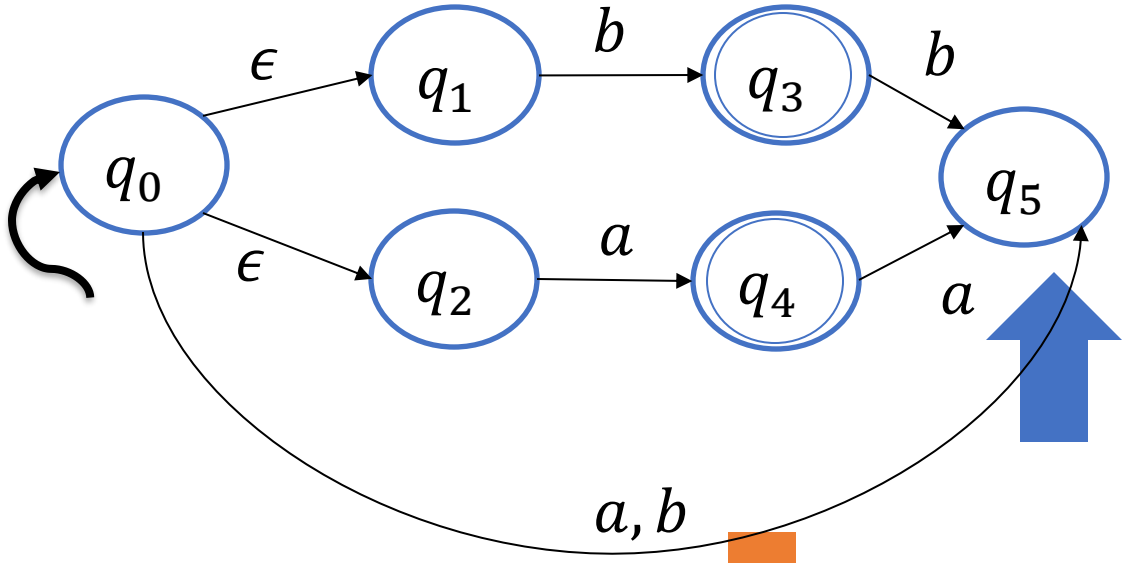
STATO CORRENTE

q_5

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

INPUT

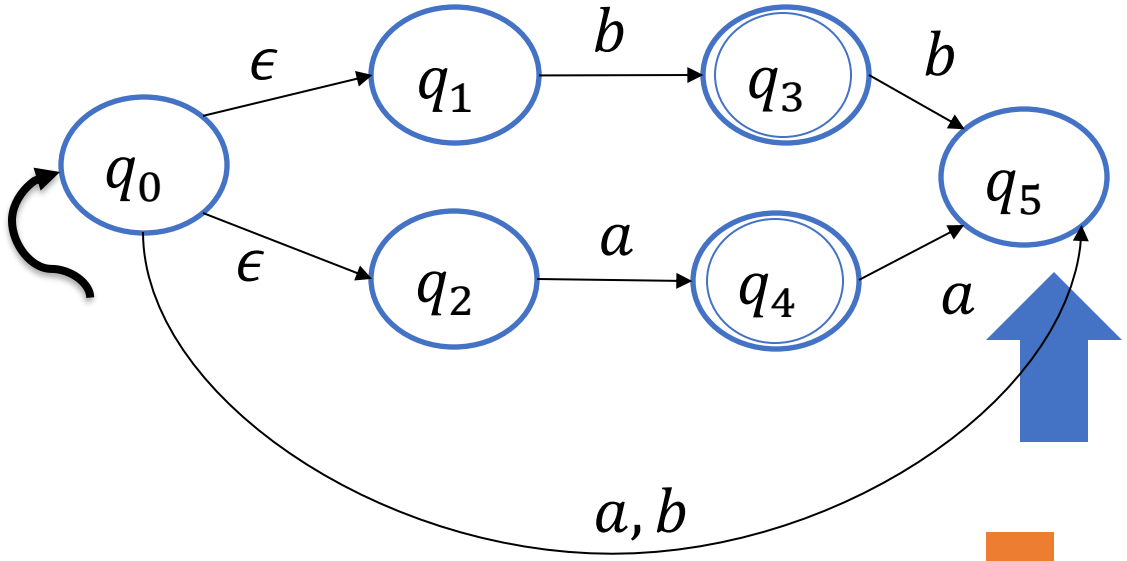
A	B	B	B

STATO CORRENTE

q_5

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

INPUT

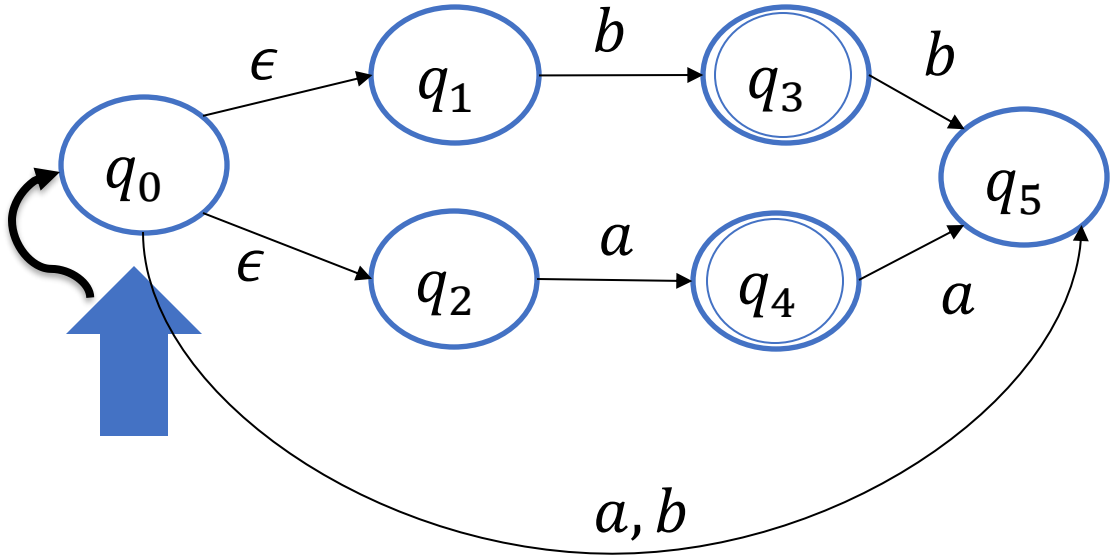
A	B	B	B

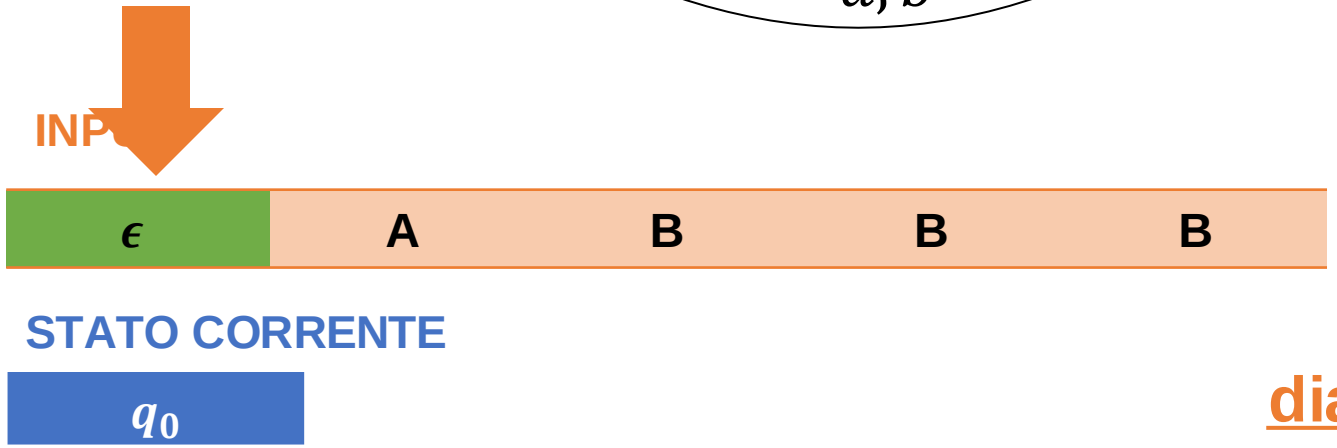
STATO CORRENTE

q_5

Esecuzione non Accettante

Esempio di Computazione

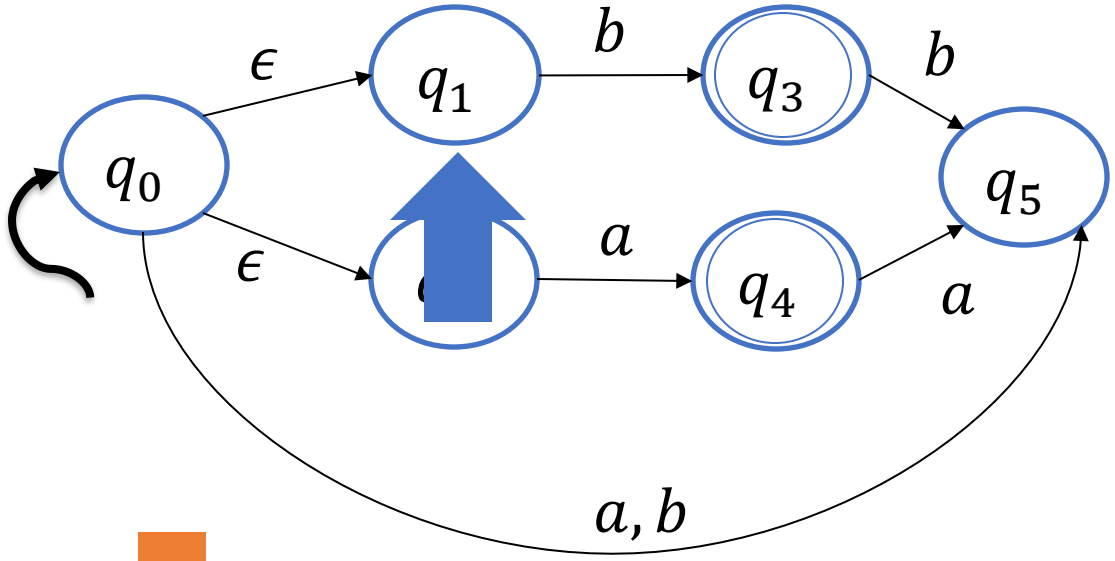




(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT

ϵ	A	B	B	B

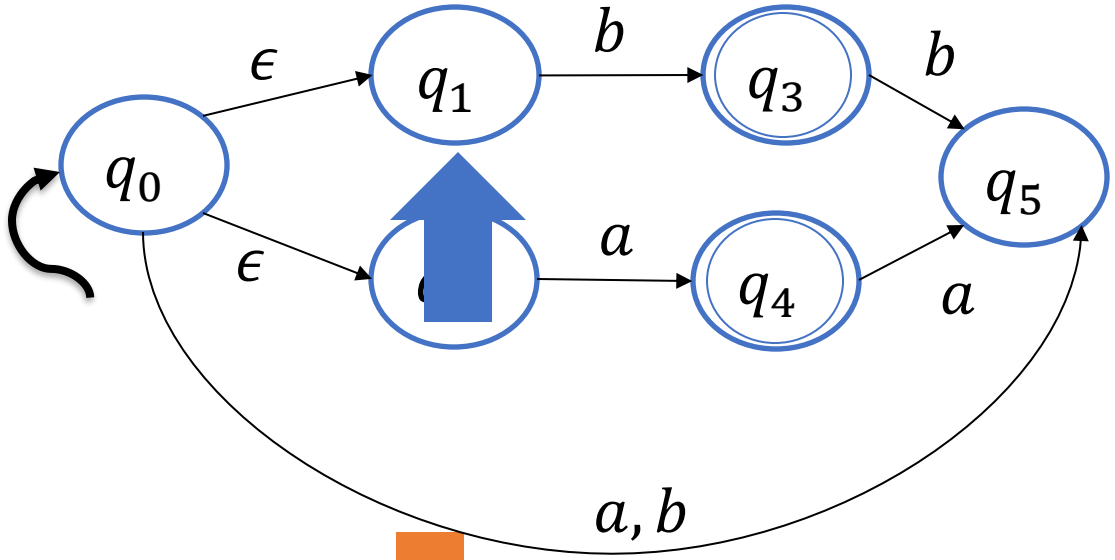
STATO CORRENTE

q_1

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT

ϵ	A	B	B	B

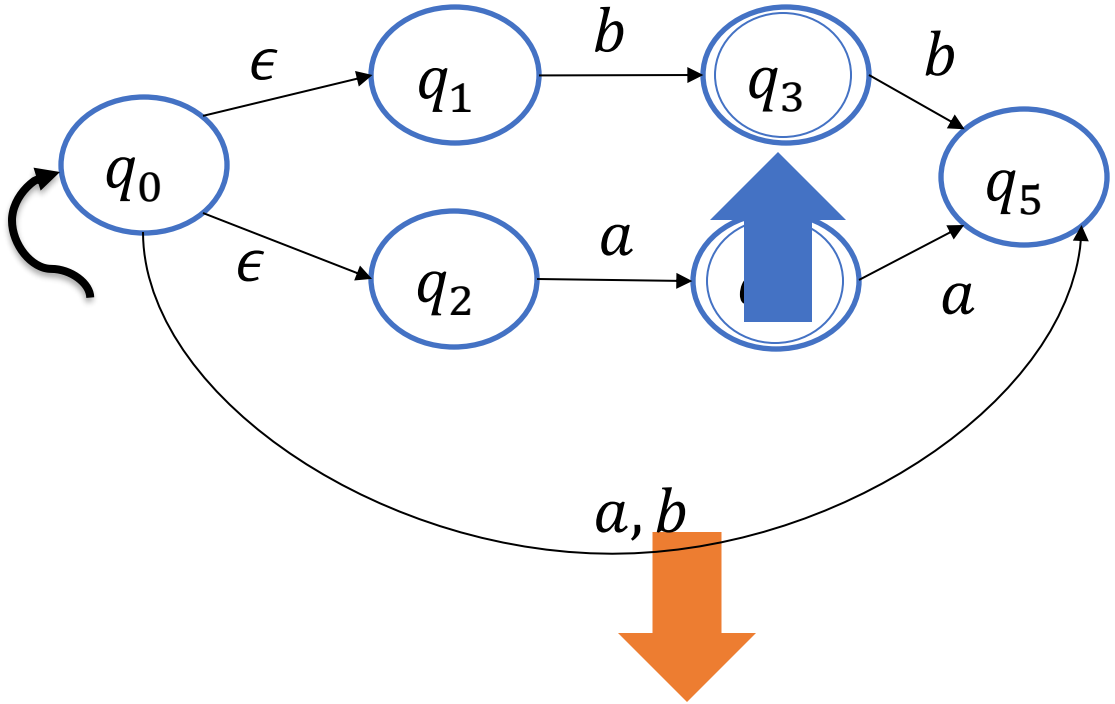
STATO CORRENTE

q_1

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT

ϵ	A	B	B	B

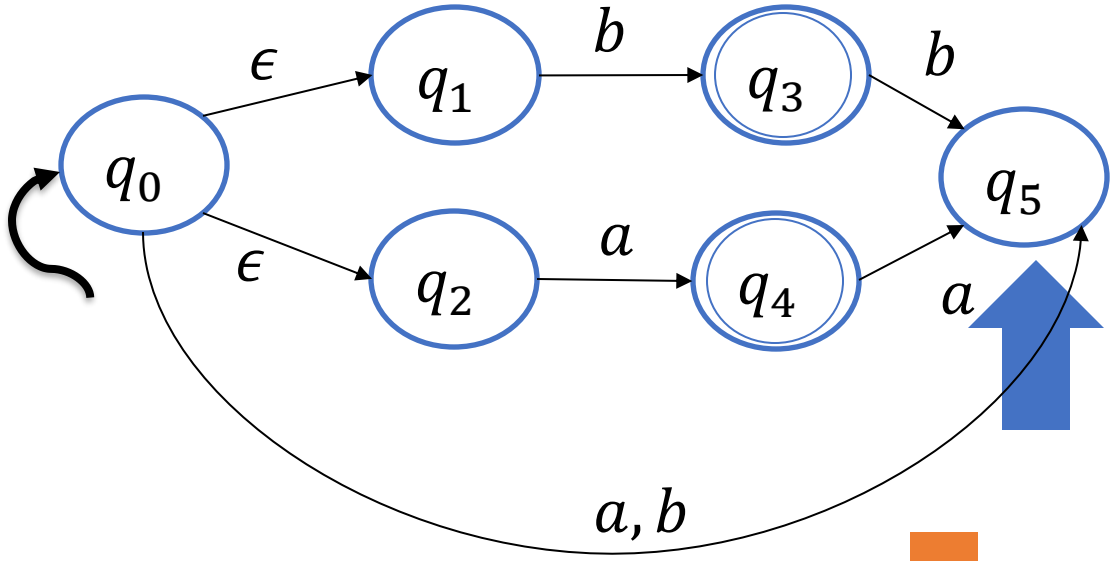
STATO CORRENTE

q_3

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

INPUT

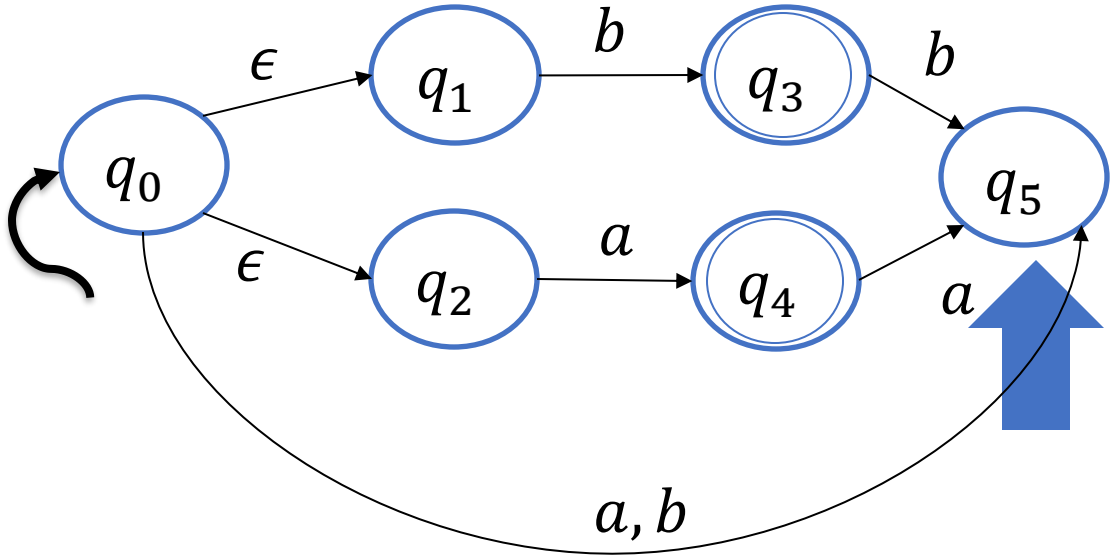
ϵ	A	B	B	B

STATO CORRENTE

q_5

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

INPUT

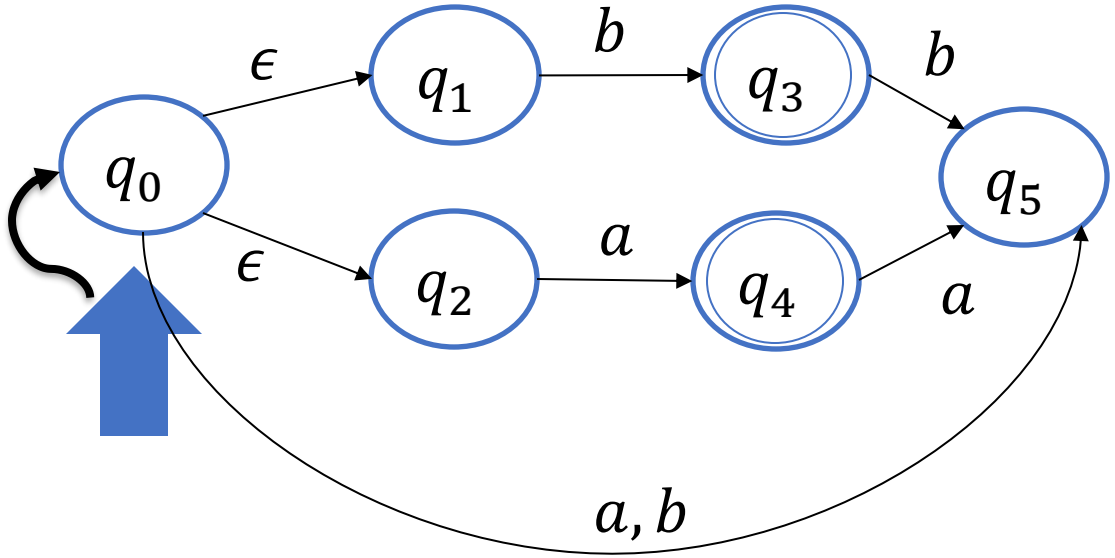
ϵ	A	B	B	B

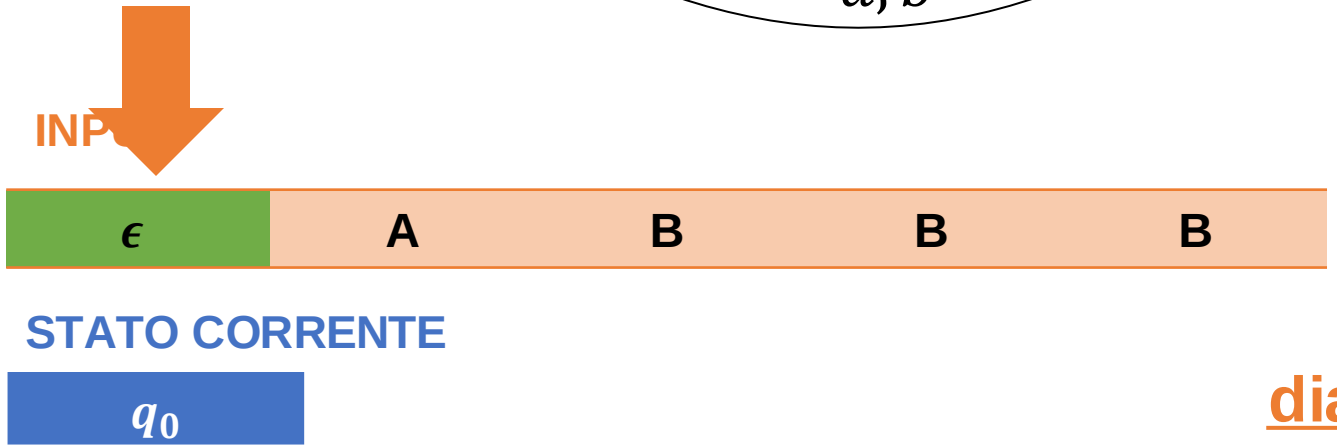
STATO CORRENTE

q_5

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione

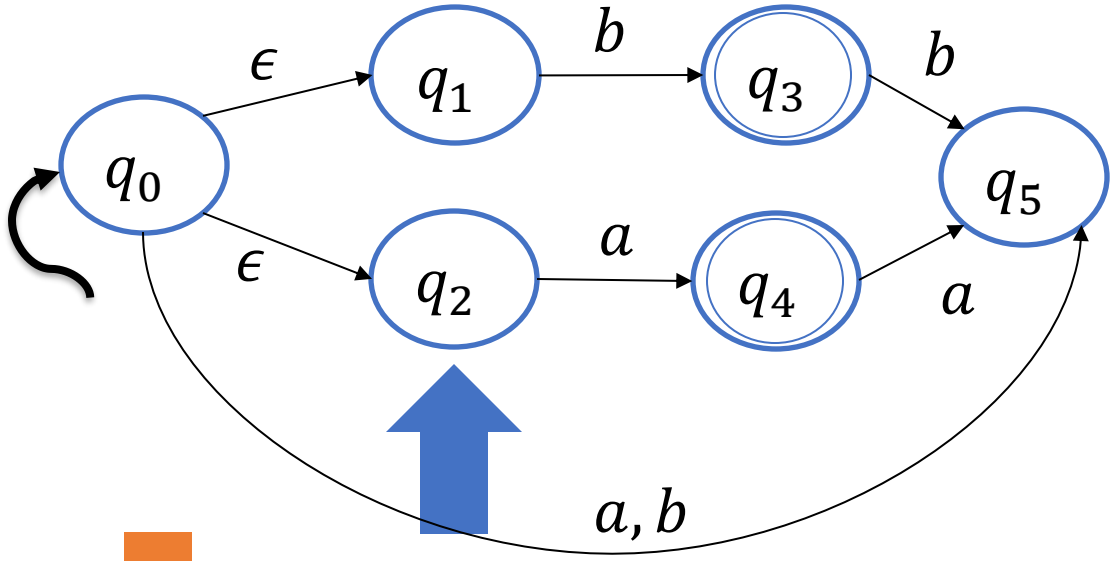




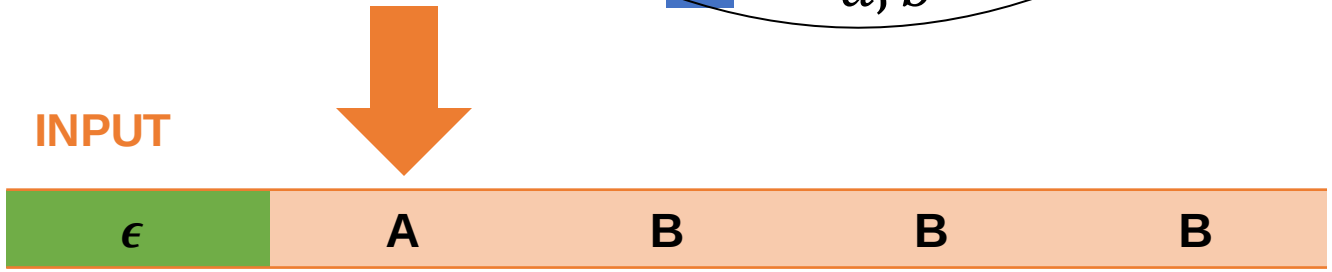
(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT



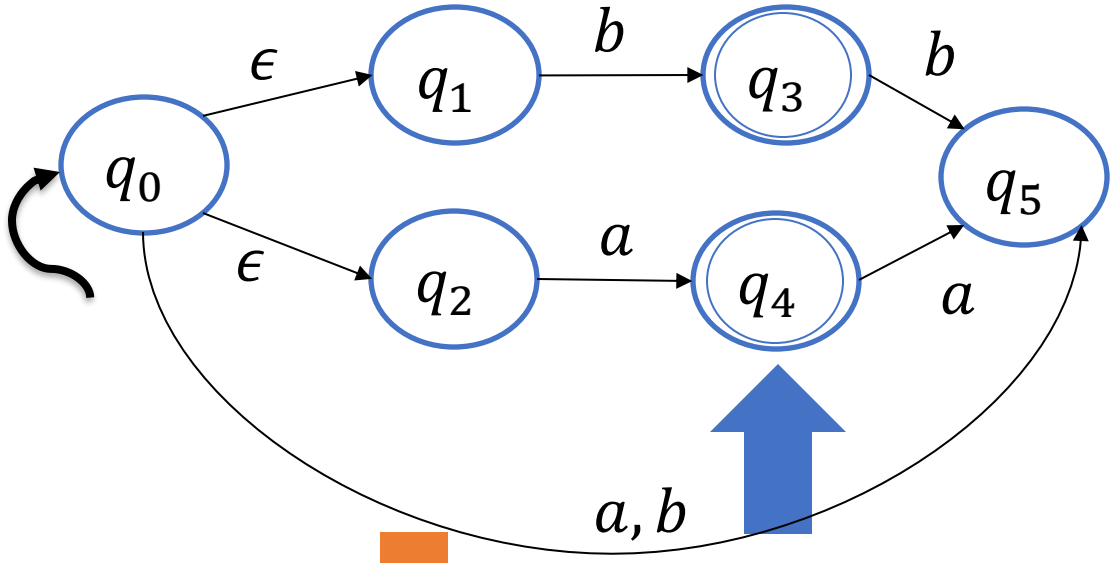
STATO CORRENTE

q_1

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT

ϵ	A	B	B	B

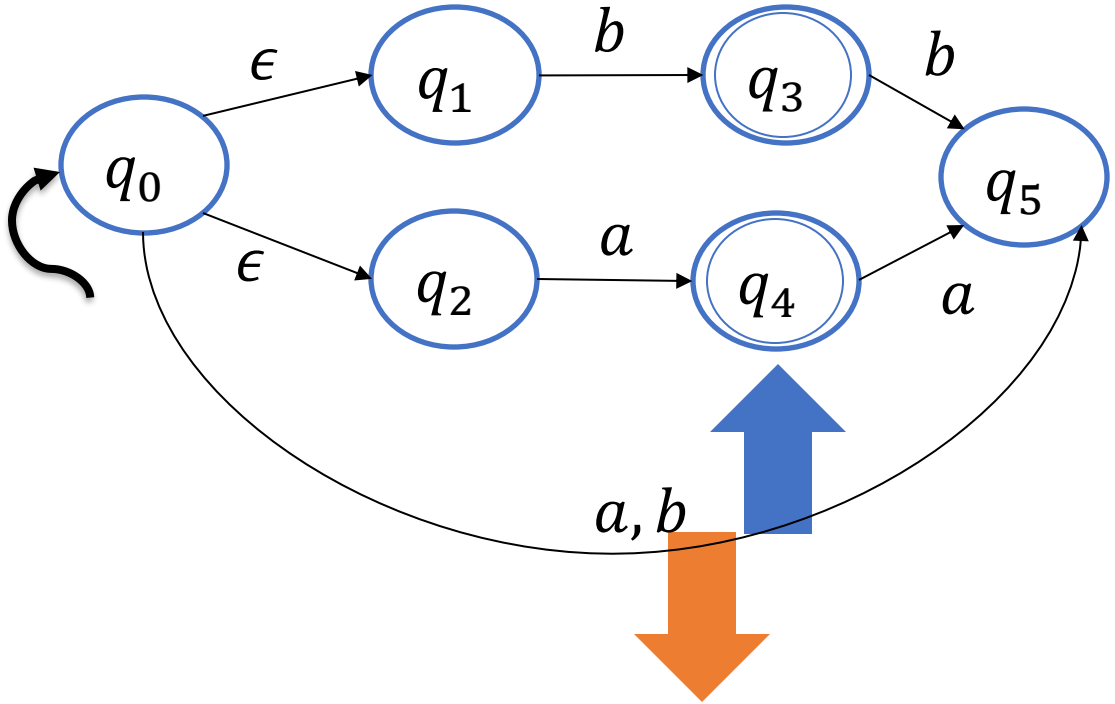
STATO CORRENTE

q_4

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



INPUT

ϵ	A	B	B	B

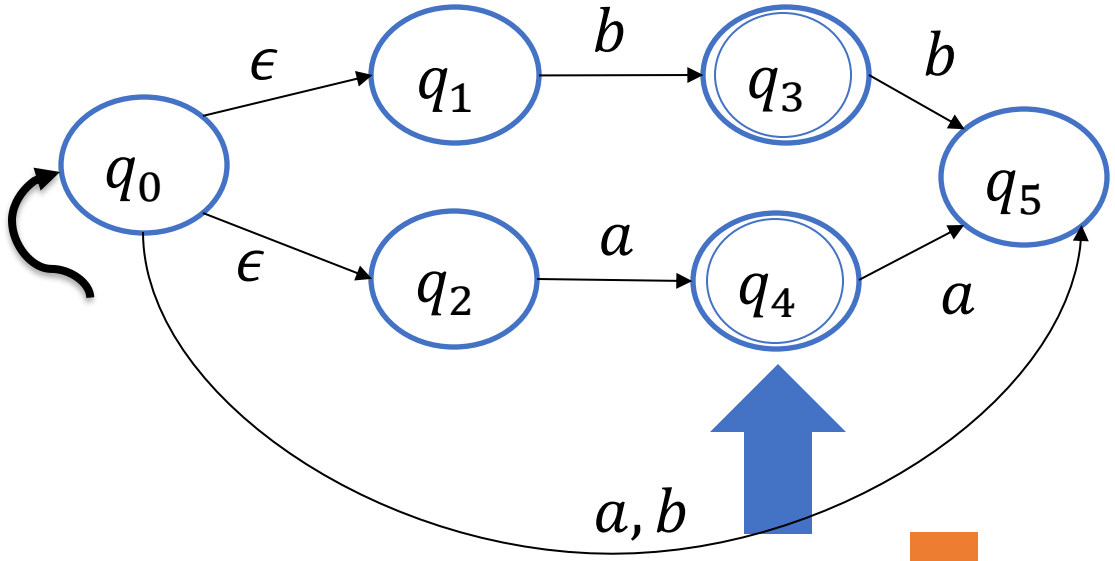
STATO CORRENTE

q_4

(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

INPUT

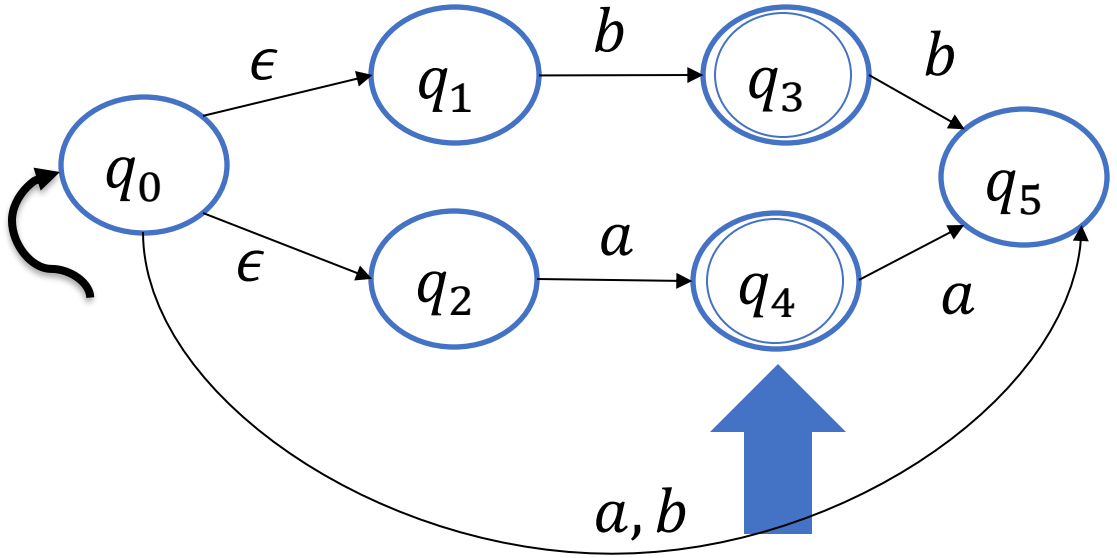
ϵ	A	B	B	B

STATO CORRENTE

q_4

Le transizioni mancanti dal diagramma vanno considerate cappi

Esempio di Computazione



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

INPUT

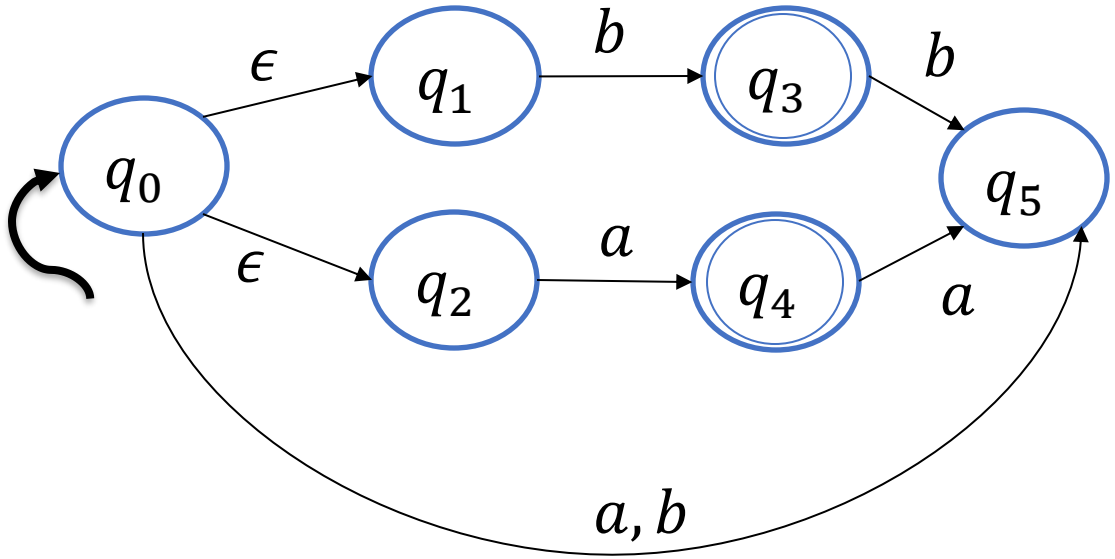
ϵ	A	B	B	B

STATO CORRENTE

q_4

Esecuzione Accettante

Esecuzioni



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

ϵ-Estensione dell'Input

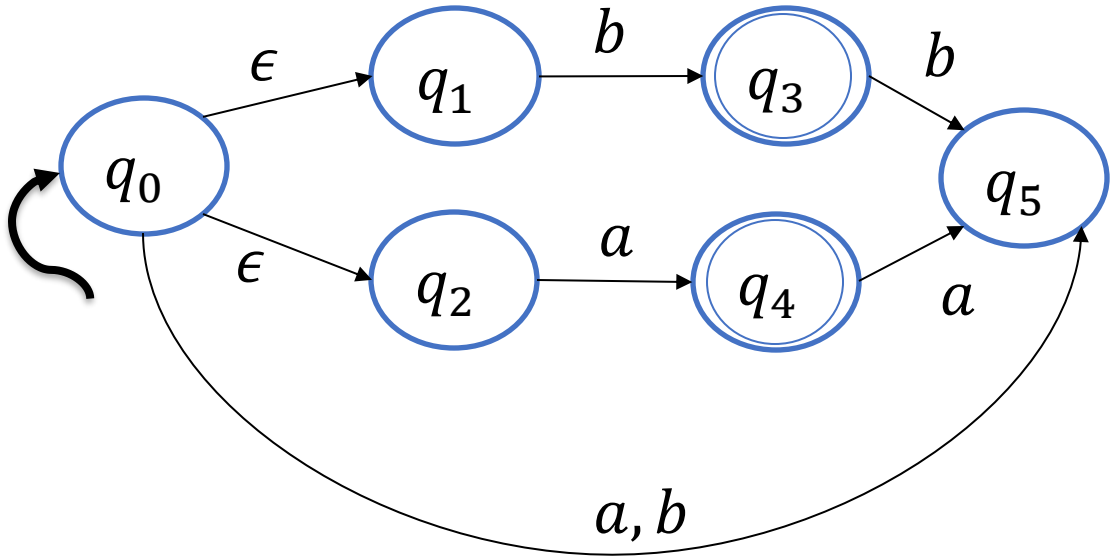
ϵ	A	B	B	B

Esecuzione

q_2	q_4	q_4	q_4	q_4	q_4

Esecuzione Accettante

Esecuzioni



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

ϵ-Estensione dell'Input

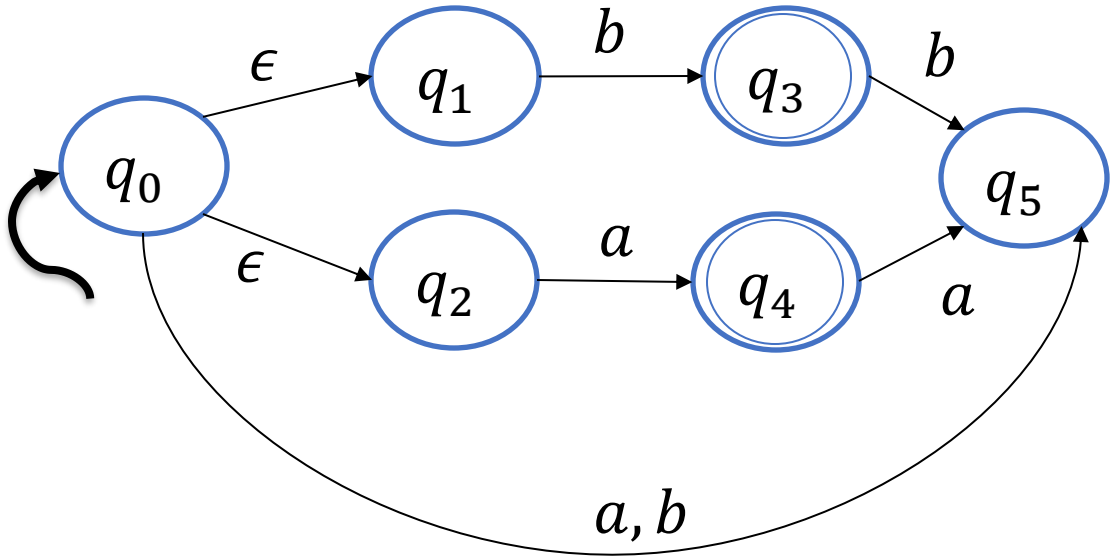
ϵ	A	B	B	B

Esecuzione

q_1	q_1	q_3	q_5	q_5	q_5

Esecuzione Accettante

Esecuzioni



(q, σ)	$\delta(q, \sigma)$	(q, σ)	$\delta(q, \sigma)$
(q_0, ϵ)	$\{q_1, q_2\}$	(q_3, ϵ)	$\{q_3\}$
(q_0, a)	$\{q_5\}$	(q_3, a)	$\{q_3\}$
(q_0, b)	$\{q_5\}$	(q_3, b)	$\{q_5\}$
(q_1, ϵ)	$\{q_1\}$	(q_4, ϵ)	$\{q_4\}$
(q_1, a)	$\{q_1\}$	(q_4, a)	$\{q_5\}$
(q_1, b)	$\{q_5\}$	(q_4, b)	$\{q_4\}$
(q_2, ϵ)	$\{q_2\}$	(q_5, ϵ)	$\{q_5\}$
(q_2, a)	$\{q_4\}$	(q_5, a)	$\{q_5\}$
(q_2, b)	$\{q_2\}$	(q_5, b)	$\{q_5\}$

ϵ-Estensione dell'Input

A	**B**	**B**	**B**

Esecuzione

q_5	q_5	q_5	q_5	q_5

Esecuzione NON Accettante

Equivalenza – Costruzione

- **Teorema**. Per ogni **ϵ-ASFND** A esiste un **ASFND** A' tale che $L(A) = L(A')$

- **Dimostrazione.** Definiamo $A =< \Sigma, Q, \delta', q_0, F >$ a partire da A cambiando solo la funzione di transizione (per eliminare le ϵ-transizioni)
- **Definiamo** $\delta' : Q \times \Sigma \to P(Q)$ tale che $\delta'(q, a)$ coincide con l'unione degli insiemi degli stati raggiungibili dalla ϵ-chiusura di q in A. Più formalmente

$$\delta'(q, a) = \bigcup_{k \in E(q,A)} \delta'(k, a)$$

- Per concludere la prova dimostriamo $L(A) = L(A')$. Forniamo solo una intuizione.

1. Se $s \in L(A)$, esiste una **esecuzione accettante X di A per s** e quindi una ϵ-estensione s' di s con le proprietà desirate. Possiamo utilizzare s' per dimostrare che **X è anche una accettante di A' su s**.
2. Se $s \in L(A')$, esiste una **esecuzione accettante X di A' per s**. Possiamo dimostrare che **X è una esecuzione accettante di A su s** costruendo una ϵ-estensione s' di s.

Dimostrazione Teorema 1 – Osservazione Generale

- **Teorema 1**. Dati due **linguaggi regolari $\mathcal{L}_1, \mathcal{L}_2$ tutti i seguenti linguaggi sono regolari**.
  1. **Unione.** $\mathcal{L}_1 \cup \mathcal{L}_2 = \{ s \mid s \in \mathcal{L}_1 \; oppure \; s \in \mathcal{L}_2 \}$
  2. **Intersezione.** $\mathcal{L}_1 \cap \mathcal{L}_2 = \{ s \mid s \in \mathcal{L}_1 \; e \; s \in \mathcal{L}_2 \}$
  3. **Complemento.** $\overline{\mathcal{L}_1} = \{ s \mid s \notin \mathcal{L}_1 \}$
  4. **Concatenazione.** $\mathcal{L}_1 \circ \mathcal{L}_2 = \{ c_1 \dots c_k d_1 \dots d_l \mid c_1 \dots c_k \in \mathcal{L}_1 \; e \; d_1 \dots d_l \in \mathcal{L}_2 \}$
  5. **Star.** $\mathcal{L}_1^* = \{ s_1 \dots s_k \mid con \; k \geq 0 \; e \; s_1 \dots s_k \in \mathcal{L}_1 \}$

- **Dimostrazione**. Dobbiamo dimostrare le **cinque proposizioni separatamente.**

- Tutte le proposizioni hanno la stessa forma. **Se $\mathcal{L}_1, \mathcal{L}_2$ sono regolari allora anche $\mathcal{L}_1 \star \mathcal{L}_2$ lo è**
  - Proprietà di **chiusura** della famiglia dei linguaggi regolari

- Per tutte le proposizioni utilizzeremo la seguente strategia di prova.
  - Se **$\mathcal{L}_1, \mathcal{L}_2$ sono regolari allora esistono** due ASFD **A_1 e A_2 tali che $\mathcal{L}_1 = L(A_1) \; e \; \mathcal{L}_2 = L(A_2)$**
  - Dimostreremo che **A_1 e A_2** possono essere composti per formare un ASFND A **tale che $L(A) = \mathcal{L}_1 \star \mathcal{L}_2$**

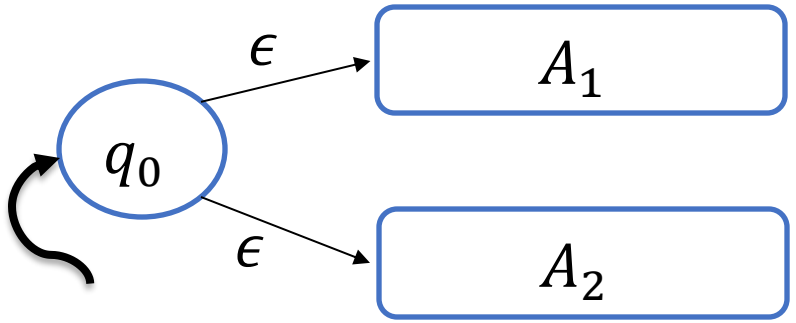
Dimostrazione Proposizione 1.1 (1/3)

- **Proposizione 1.1**. Siano $\mathcal{L}_1, \mathcal{L}_2$ **linguaggi regolari. Il linguaggio** $\mathcal{L}_1 \cup \mathcal{L}_2$ **è regolare (unione)**
- **Dimostrazione**. Procediamo con la nostra strategia di prova
- Se $\mathcal{L}_1, \mathcal{L}_2$ **sono regolari allora esistono** due ASFD A_1 **e** A_2 **tali che** $\mathcal{L}_1 = L(A_1)$ **e** $\mathcal{L}_2 = L(A_2)$
- Procediamo a costruire un ASFND A **tale che** $L(A) = \mathcal{L}_1 \cup \mathcal{L}_2$

1. Assumiamo $A_1 =< \Sigma_1, Q_1, \delta_1, q_0^1, F_1 >$ e $A_2 =< \Sigma_2, Q_2, \delta_2, q_0^2, F_2 >$ **tali che** $Q_1 \cap Q_2 = \emptyset$
2. Definiamo l'ASFND $A =< \Sigma, Q, \delta, q_0, F >$ come segue:
   - $\Sigma = \Sigma_1 \cup \Sigma_2$ (**alfabeto**)
   - $Q = Q_1 \cup Q_2 \cup \{q_0\}$ con $q_0 \notin Q_1 \cup Q_2$ (**stati**)
   - $F = F_1 \cup F_2$

Dimostrazione Proposizione 1.1 (2/3)

- La funzione di transizione δ è definita come segue
  - $\delta(q,a) = \{\delta_1(q,a)\}, se\, q \in Q_1$
  - $\delta(q,a) = \{\delta_2(q,a)\}, se\, q \in Q_2$
  - $\delta(q_0,\epsilon) = \{q_0^1, q_0^2\}$
  - $\delta(q_0,x) = \{\}, per\, ogni\, x \in \Sigma_1 \cup \Sigma_2$
  - $\delta(q,\epsilon) = \{\}, per\, ogni\, q \in Q_1 \cup Q_2$



Dimostrazione Proposizione 1.1 (3/3)

- Per concludere la prova, osserviamo quanto segue
  1. $L(A_1) \cup L(A_2) \supseteq L(A)$. Se $s \in L(A)$, allora esiste una esecuzione accettante per A su s la ϵ-estensione ϵs di s tale che $s \in A_1$ $oppure$ $s \in A_2$. Ne consegue che $s \in L(A_1) \cup L(A_2)$

  2. $L(A_1) \cup L(A_2) \subseteq L(A)$. Se $s \in L(A_1) \cup L(A_2)$, allora esiste una esecuzione accettante per A_1 (oppure A_2) su s. Sia $(q_0^i, q_1^i, \ldots, q_n^i)$ tale esecuzione. Per costruzione,$(q_0, q_0^i, q_1^i, \ldots, q_n^i)$ è una esecuzione accettante per ϵs. Ne consegue che $s \in L(A)$

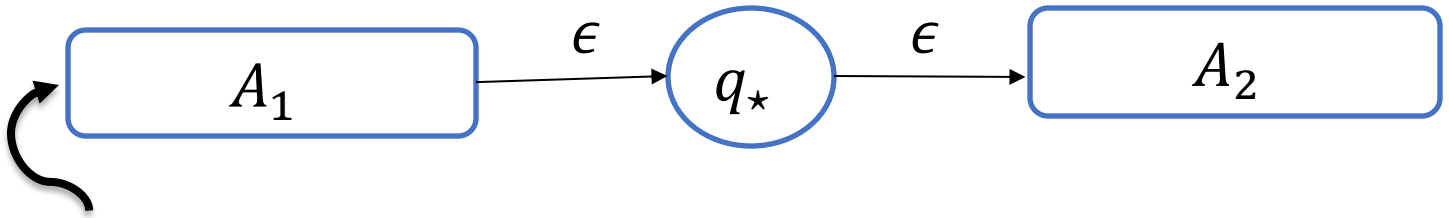
Dimostrazione Proposizione 1.4 (1/3)

- **Proposizione 1.1**. Siano $\mathcal{L}_1, \mathcal{L}_2$ **linguaggi regolari. Il linguaggio** $\mathcal{L}_1 \circ \mathcal{L}_2$ **è regolare (concat)**
- **Dimostrazione**. Procediamo con la nostra strategia di prova
- Se $\mathcal{L}_1, \mathcal{L}_2$ **sono regolari allora esistono** due ASFD A_1 **e** A_2 **tali che** $\mathcal{L}_1 = L(A_1)$ **e** $\mathcal{L}_2 = L(A_2)$
- Procediamo a costruire un ASFND A **tale che** $L(A) = \mathcal{L}_1 \circ \mathcal{L}_2$

1. Assumiamo $A_1 =< \Sigma_1, Q_1, \delta_1, q_0^1, F_1 >$ e $A_2 =< \Sigma_2, Q_2, \delta_2, q_0^2, F_2 >$ **tali che** $Q_1 \cap Q_2 = \emptyset$
2. Definiamo l'ASFND $A =< \Sigma, Q, \delta, q_0^1, F >$ come segue:
   - $\Sigma = \Sigma_1 \cup \Sigma_2$ (**alfabeto**)
   - $Q = Q_1 \cup Q_2 \cup \{q_\star\}$ con $q_\star \notin Q_1 \cup Q_2$ (**stati**)
   - $F = F_2$

Dimostrazione Proposizione 1.4 (2/3)

- La funzione di transizione δ è definita come segue
  - $\delta(q,a) = \{\delta_1(q,a)\}, se\, q \in Q_1\ e\ a \in \Sigma_1$
  - $\delta(q,a) = \{\delta_2(q,a)\}, se\, q \in Q_2\ e\ a \in \Sigma_2$
  - $\delta(q,\epsilon) = \{\ \ \}, se\, q \in (Q_1 \cup Q_2) \setminus F_1$
  - $\delta(q,\epsilon) = \{q_\star\}, se\, q \in F_1$
  - $\delta(q_\star,\epsilon) = \{q_0^2\}$



Dimostrazione Proposizione 1.4 (3/3)

- Per concludere la prova, osserviamo quanto segue
  1. $L(A_1) \cap L(A_2) \supseteq L(A)$. Se st $\in L(A)$, allora esiste una esecuzione accettante per A sulla ϵ-estensione $s\epsilon\epsilon t$ di s tale che $s \in A_1$ $e\, s \in A_2$. Ne consegue che $s \in L(A_1) \cap L(A_2)$

  2. $L(A_1) \cap L(A_2) \subseteq L(A)$. Se $st \in L(A_1) \cap L(A_2)$, allora esiste una esecuzione accettante per A_1 $su\, s$ e per A_2) su s. Siano $(q_0^1, q_1^1, \ldots, q_n^1)$ e $(q_0^2, q_1^2, \ldots, q_m^2)$ tali esecuzioni. Per costruzione $(q_0^1, q_1^1, \ldots, q_n^1, q_\star, q_0^2, q_1^2, \ldots, q_m^2)$ è una esecuzione accettante per $s\epsilon\epsilon t$. Ne consegue che $st \in L(A)$

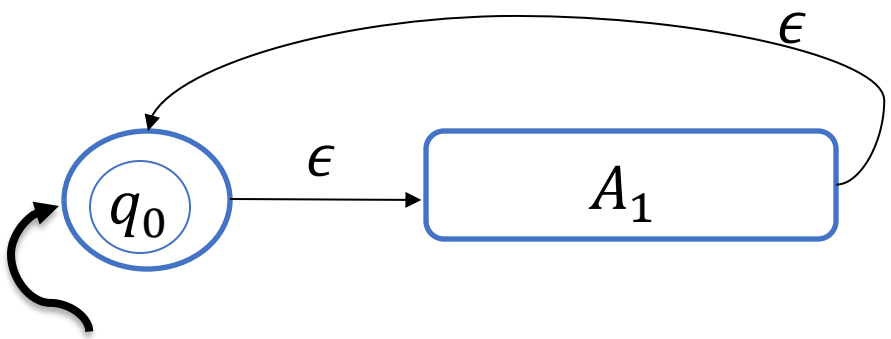
Dimostrazione Proposizione 1.5 (1/3)

- **Proposizione 1.5**. Sia $\mathcal{L}_1$ un **linguaggio regolari. Il linguaggio $\mathcal{L}_1^*$ è regolare (star)**
- **Dimostrazione**. Procediamo con la nostra strategia di prova
- Se $\mathcal{L}_1$ **è regolare allora esistono** un ASFD $A_1 =< \Sigma_1, Q_1, \delta_1, q_0^1, F_1 >$ **tali che** $\mathcal{L}_1 = L(A_1)$
- Procediamo a costruire un ASFND A **tale che** $L(A) = \mathcal{L}_1^*$

1. Definiamo l'ASFND $A =< \Sigma_1, Q, \delta, q_0, F >$ come segue:
   - $Q = Q_1 \cup \{q_0, q_\star\}$ con $q_0, q_\star \notin Q_1$ (**stati**)
   - $F = \{q_0\} \cup F_1$
   - $q_0 \notin Q_1$

Dimostrazione Proposizione 1.5 (2/3)

- La funzione di transizione δ è definita come segue
  - $\delta(q, a) = \{\delta_1(q, a)\}$, *per ogni* $q \in Q_1$ *e* $a \in \Sigma_1$
  - $\delta(q, \epsilon) = \{\ \}$, *se* $q \in Q_1 \setminus F_1$
  - $\delta(q, \epsilon) = \{q_0\}$, *se* $q \in F_1$
  - $\delta(q_0, \epsilon) = \{q_0^1\}$
  - $\delta(q_0, a) = \{\ \}$ *per ogni* $a \in \Sigma_1$



Dimostrazione Proposizione 1.5 (3/3)

- Per concludere la prova, dimostriamo che $s \in L(A)$ se e solo se $s \in L(A_1)^*$
- Consideriamo due casi distinti.

- Caso $|s| = 0$. $s \in L(A_1)^*$ e $s \in L(A)$ per costruzione.
- Caso $|s| > 0$.

1. $L(A) \subseteq L(A_1)^*$. Se $s_1 s_2 \ldots s_k \in L(A)$, allora esiste una esecuzione accettante per A sulla ϵ-estensione $\epsilon s_1 \epsilon s_2 \ldots \epsilon s_k$ della stringa $s_1 s_2 \ldots s_k$.Possiamo concludere che $s_1, s_2, \ldots, s_k \in L(A_1)$, e dunque $s \in L(A_1)^*$

2. $L(A) \supseteq L(A_1)^*$ Se $s = s_1 s_2 \ldots s_k \in L(A_1)^*$, allora esiste una esecuzione accettante di A_1 per s_i, per ogni $i = 1, \ldots, k$. Quindi la ϵ-estensione $\epsilon s_1 \epsilon s_2 \ldots s_k$ di $s_1 s_2 \ldots s_k$ *è* accettata da A. Possiamo concludere che $s_1 s_2 \ldots s_k \in L(A)$.

Espressioni Regolari

Le Espressioni Regolari

- Introduciamo adesso una nuova notazione, questa volta di natura algebrica, per definire i linguaggi.

- **Def:** Sia Σ un alfabeto, e sia $\mathcal{O} = \{+, *, (,), \cdot, \emptyset\}$ con $\mathcal{O} \cap \Sigma = \emptyset$. Una **espressione regolare su Σ** è stringa r non vuota su $\Sigma \cup \mathcal{O}$ tale che r è uguale a una delle seguenti
  1. $\emptyset$
  2. $r \in \Sigma$
  3. $(s + t)$, dove sia s che t sono espressioni regolari su Σ
  4. $(s \cdot t)$, dove sia s che t sono espressioni regolari su Σ
  5. $(s)^*$, dove s è una espressione regolari su Σ

Corrispondenza tra Espressioni Regolari e Linguaggi

- Le espressioni regolari consentono di rappresentare linguaggi mediante una opportuna interpretazione dei simboli che le compongono.
- La seguente tabella mostra la corrispondenza tra un'espressione regolare e il linguaggio che essa rappresenta

Espr. regolari	Linguaggi
$\emptyset$	Λ
a	$\{a\}$
$(s+t)$	$\mathcal{L}(s) \cup \mathcal{L}(t)$
$(s \cdot t)$	$\mathcal{L}(s) \circ \mathcal{L}(t)$
s^*	$(\mathcal{L}(s))^*$

Assunzioni

- Come già fatto per le stringhe, alcune volte scriveremo st al posto di $s \cdot t$.
- Assumendo che il simbolo $*$ abbia precedenza sul simbolo $\cdot$ e questo abbia precedenza sul simbolo $+$, e tenendo conto dell'associatività di queste operazioni, si possono spesso eliminare alcune parentesi
  - Esempio: L'espressione regolare $(a + (b \cdot (c \cdot d)))$ su $\{a, b, c, d\}$ si può scrivere come $a + bcd$
- Inoltre, per questioni di succintezza, utilizzeremo $(r)^+$ per indicare $r(r)^*$, dove r è un'espressione regolare su un alfabeto Σ
- N.B: il linguaggio $\{\epsilon\}$ può essere descritto dall'espressione regolare $\emptyset^*$

Espressioni Regolari – Esercizio 1

Definire il linguaggio $L(r)$ rappresentanto dall'espressione regolare $r = (a + b)^* a$ sull'alfabeto $\{a, b\}$

Esercizio Espressioni Regolari 1 – Soluzione

Definire il linguaggio $L(r)$ rappresentato dall'espressione regolare $r = (a + b)^* a$ sull'alfabeto $\{a, b\}$

$$L\big((a+b)^* a\big) = L\big((a+b)^*\big) \circ L(a) =$$
$$\big(L(a+b)\big)^* \circ L(a) =$$
$$\big(L(a) \cup L(b)\big)^* \circ L(a) =$$
$$(\{a\} \cup \{b\})^* \circ \{a\} =$$
$$\{a, b\}^* \circ \{a\} = \{x \mid x \in \{a, b\}^+ \; e \; x \; termina \; con \; a\}$$

Espressioni Regolari – Esercizio 2

Sia $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, \ .\}$

Definire un’espressione regolare r su Σ che rappresenta tutti i numeri razionali

Espressioni Regolari – Esercizio 2

Sia $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, \ .\}$

Definire un'espressione regolare r su Σ che rappresenta tutti i numeri razionali

$$\big((1+2+3+4+5+6+7+8+9)(0+1+2+3+4+5+6+7+8+9)^* + 0\big).(0+1+2+3+4+5+6+7+8+9)^+$$

Relazione tra Espressioni Regolari e ASF(N)D

- Dimostreremo ora che la classe dei linguaggi accettati da automi a stati finiti **coincide** con quella dei linguaggi descritti da espressioni regolari **ovvero che…**

- **Teorema**. Le espressioni regolari descrivono tutti e soli i linguaggi regolari

- **Prova**. In termini più precisi dimostreremo che

1. Per ogni espressione regolare r esiste un **ϵ-ASFND A_r** che riconosce il linguaggio descritto da r, ovvero tale che $L(r) = L(A_r)$
2. Viceversa, per ogni **ASFD** A esiste una espressione regolare r_A che descrive il linguaggio riconosciuto da A, ovvero tale che $L(A) = L(r_A)$

ASFND catturano le Espressioni Regolari – Parte 1

- Per ogni espressione regolare r esiste un ASFD A_r che riconosce il linguaggio descritto da r, ovvero tale che $L(r) = L(A_r)$

- Consideriamo una espressione regolare r su un alfabeto Σ, ovvero r è di una delle tre forme:
  - 1.$r = \emptyset$, 2. $r \in \Sigma$, 3. $r = (s + t)$, oppure $r = (s \cdot t)$, oppure $r = (s)^*$, dove sia s che t sono espressioni regolari su Σ

- I casi 1. e 2. sono triviali. Rimane quindi da dimostrare il caso 3. Per costruzione di $L(r)$, è quindi sufficiente dimostrare quanto segue:

  a) Dati due ASFD A_1 e A_2, esiste un ASFD A tale che $L(A) = L(A_1) \cup L(A_2)$
     - **Proposizione 1.1** dimostrata in precedenza

  b) Dati due ASFD A_1 e A_2, esiste un ASFD A tale che $L(A) = L(A_1) \circ L(A_2)$
     - **Proposizione 1.4** dimostrata in precedenza

  c) Dato un ASFD A, esiste un ASFD A' tale che $L(A') = \big(L(A)\big)^*$
     - **Proposizione 1.5** dimostrata in precedenza

Le Espressioni Regolari catturano gli ASFD – Parte 2

- Per ogni ASFD A esiste una espressione regolare r_A che descrive il linguaggio riconosciuto da A, ovvero tale che $L(A) = L(r_A)$

- Esiste un **algoritmo** che dato un ASFND costruisce una espressione regolare equivalente tramite una serie di passi di semplificazione
  1. Elimina gli stati dell'ASFND uno a uno e…
  2. Li rimpiazza con archi che rappresentano espressioni regolari equivalenti

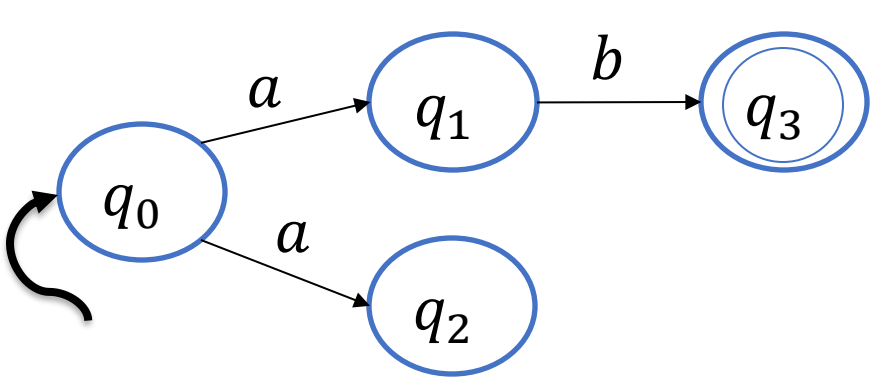
- **Non vedremo i dettagli della prova… ☺**

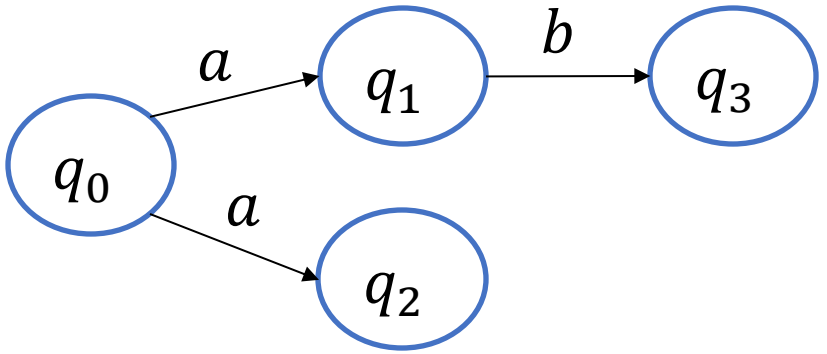
- **Ma vedremo l'algoritmo ;)**

Da ASFND a Espressione Regolare – Algoritmo

- **Definizione**. Un **grafo etichettato** sul linguaggio $\mathcal{L}$ è una coppia $G = (N, E)$ dove
  - N è un insieme (detto insieme dei **nodi** di G)
  - E è un insieme di terne (n_1, n_2, e) dove $n_1, n_2 \in N$ e $e \in \mathcal{L}$ (detto insieme degli **archi** di G)

- **Definizione**. Sia $A =< \Sigma, Q, \delta, q_0, F >$ un **ϵ-ASFND**. Il **grafo indotto da A** è il grafo etichettato (Q, E) tale che $E = \{(q, q', e) \mid q' \in \delta(q, e)\}$
  – Sostanzialmente è il grafo indotto dal diagramma degli stati dell'automa



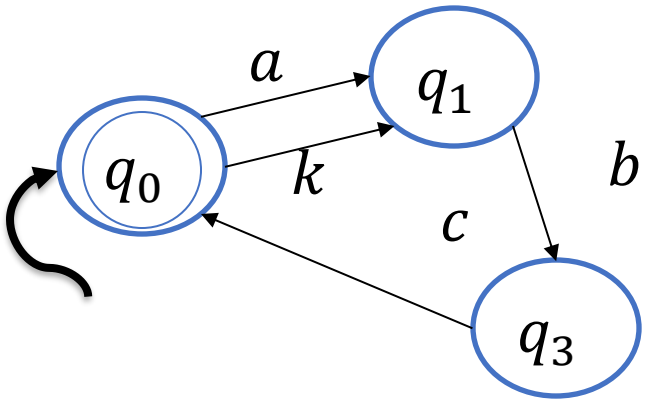


Da ASFND a Espressione Regolare – Algoritmo

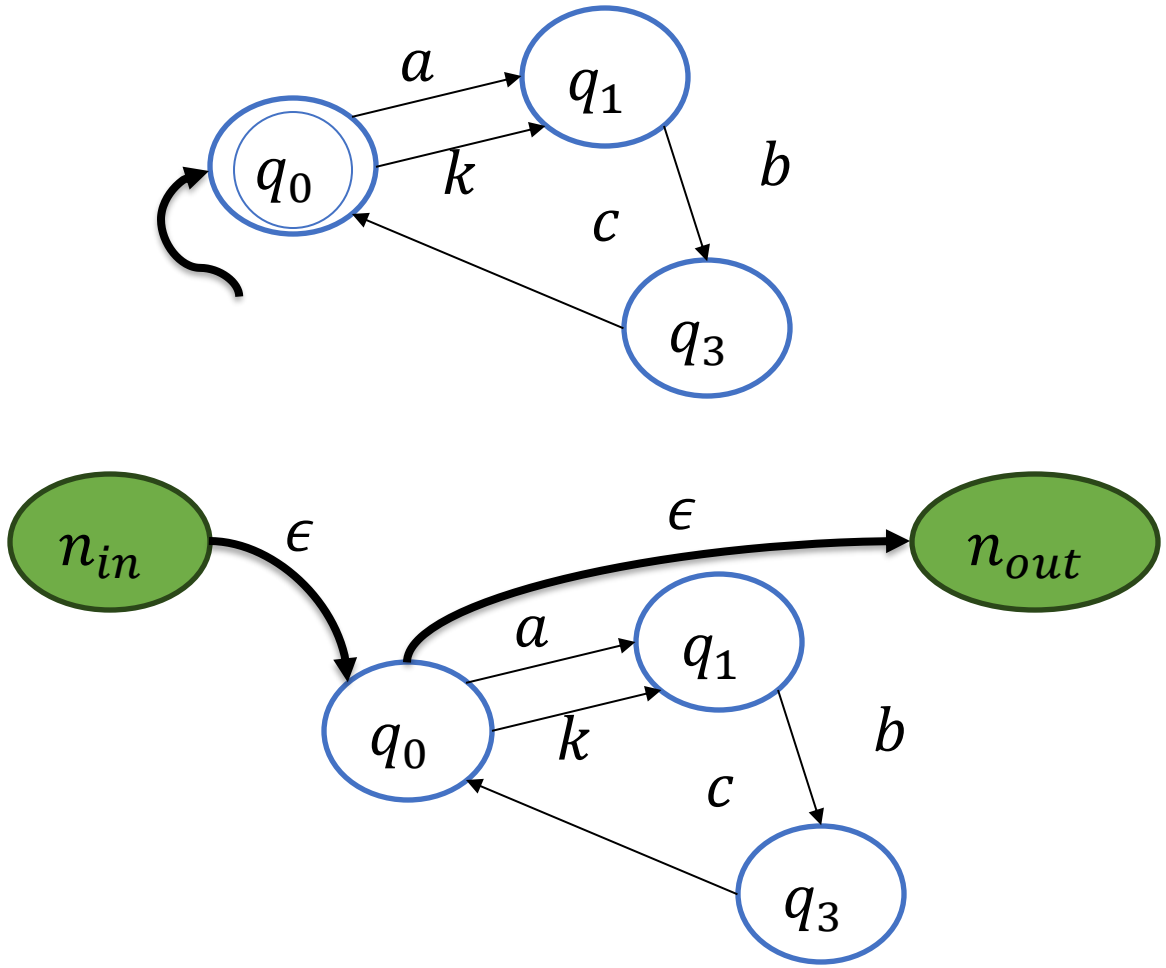
- **Input un ASFND $A =< \Sigma, Q, \delta, q_0, F >$**
- **Una Espressione Regolare r**

1. Sia $G = (N, E)$ il grafo indotto da A
2. **Aggiungi** a G i nodi n_{in} , $n_{out} \notin N$ e gli archi (n_{in}, q_0, ϵ) e (q_f, n_{out}, ϵ), **per ogni** $q_f \in F$
3. **Finché** ($|N| > 2$ **oppure** $|E| > 1$) (**G contiene più di 2 nodi o più di un arco**) **esegui i seguenti passi**
   1. **Per ogni coppia di nodi** $x, y \in N$ (**non necessariamente distinti**)
      1. Sia $E = \{e_1, e_2, \dots, e_k\}$ l'insieme delle etichette di tutti gli archi da x a y (archi paralleli)
      2. **Rimuovi** tutti gli archi da x a y
      3. **Aggiungi** un arco da x a y con l'etichetta $(e_1) \cup (e_2) \cup \cdots \cup (e_k)$
   2. **Scegli un nodo** $n \in N$ **che non sia** n_{in} **o** n_{out}
      1. **Per ogni** coppia di archi $e_{in}^n = (x, n, e) \in E$ e $e_{out}^n = (n, y, e') \in E$ (e_{in}^n entrante in n, e_{out}^n uscente da n)
         1. **Aggiungi** un arco $(x, y, (e) \circ (e_n)^* \circ (e'))$ dove e_n **è l'etichetta dell'unico cappio di** n (**se esiste**)
      2. **Rimuovi** da G n e tutti gli archi entranti e uscenti da n
4. **Restituisci l'etichetta dell'unico arco in** G

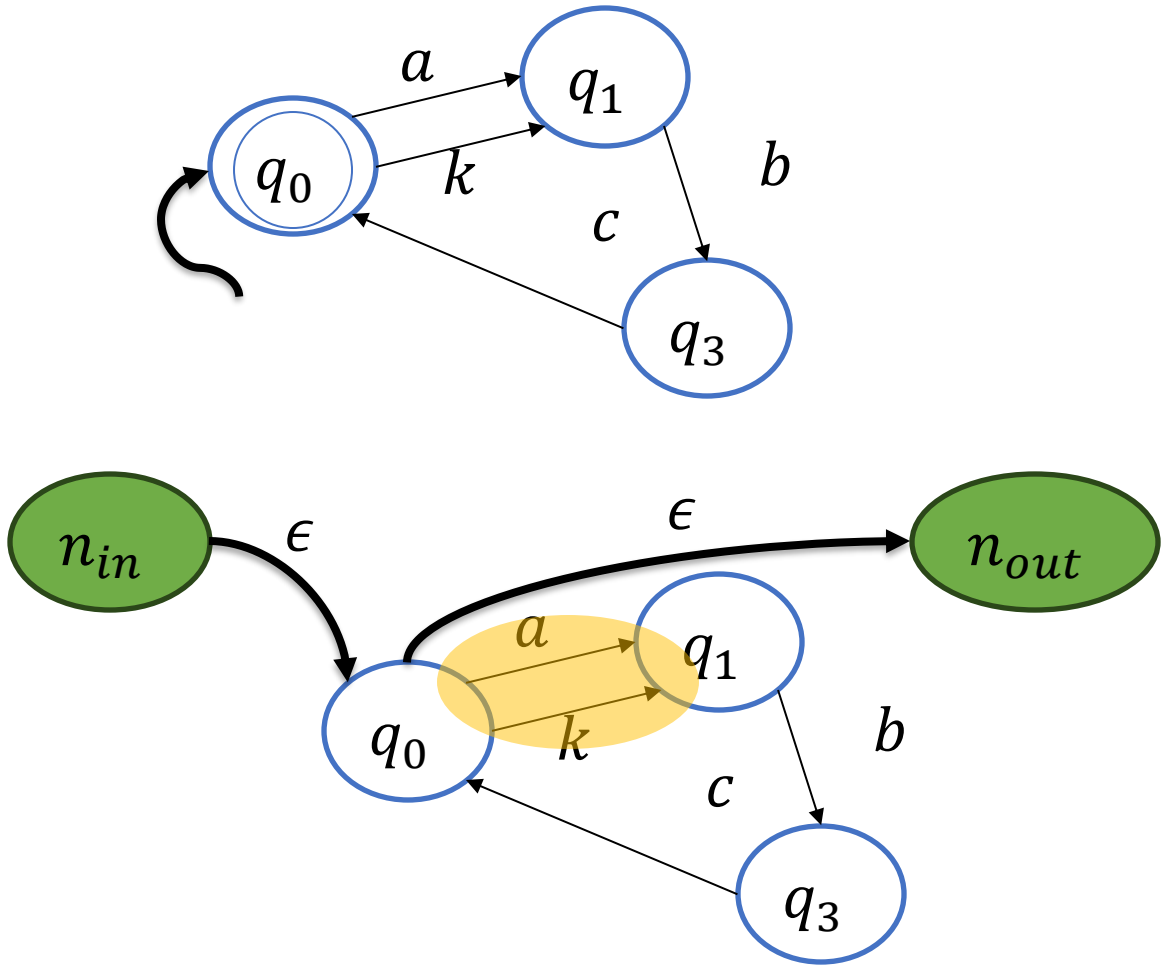
Da ASFND a Espressione Regolare – Esempio



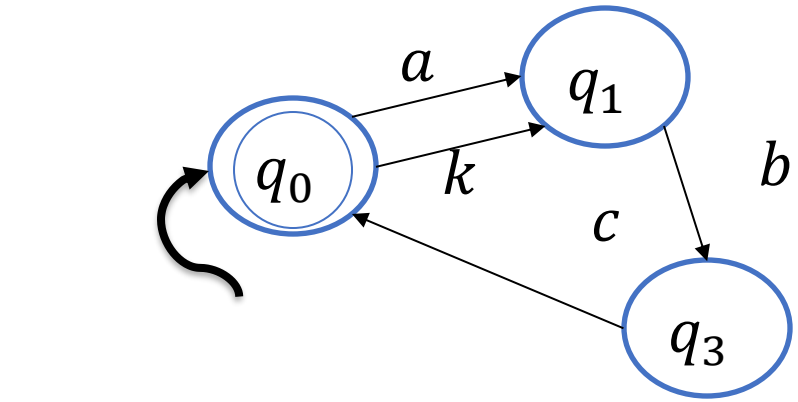
Da ASFND a Espressione Regolare – Esempio

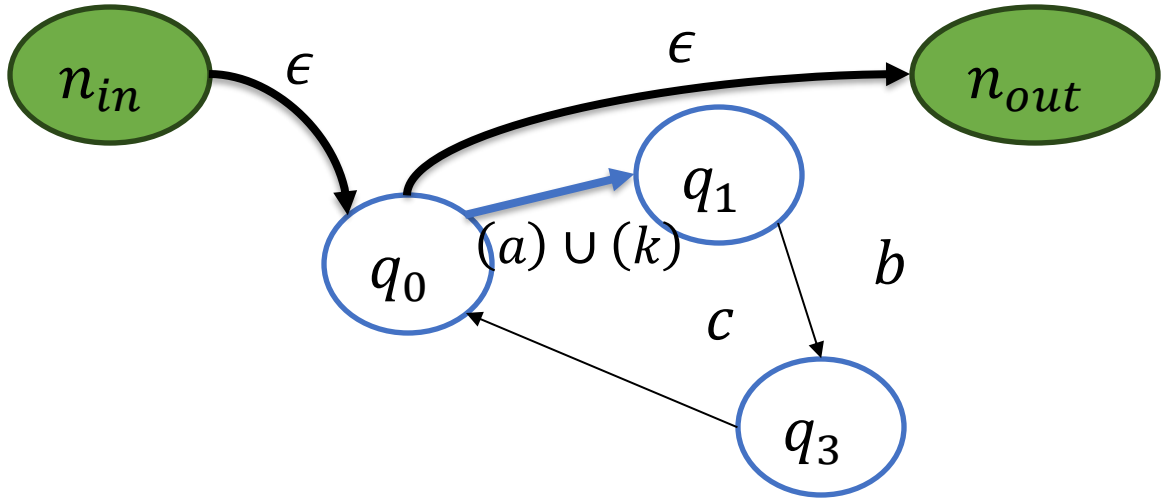


Da ASFND a Espressione Regolare – Esempio

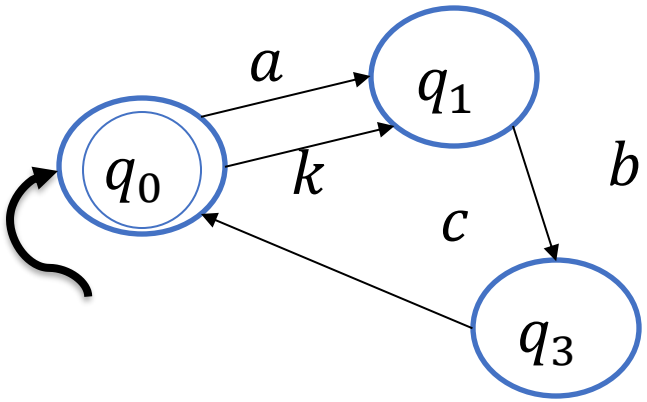


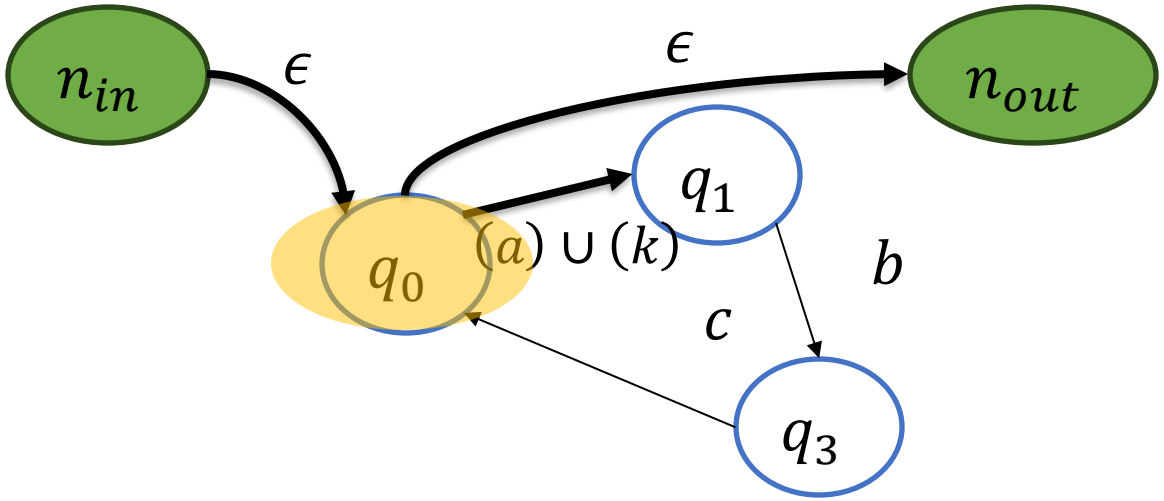
Da ASFND a Espressione Regolare – Esempio



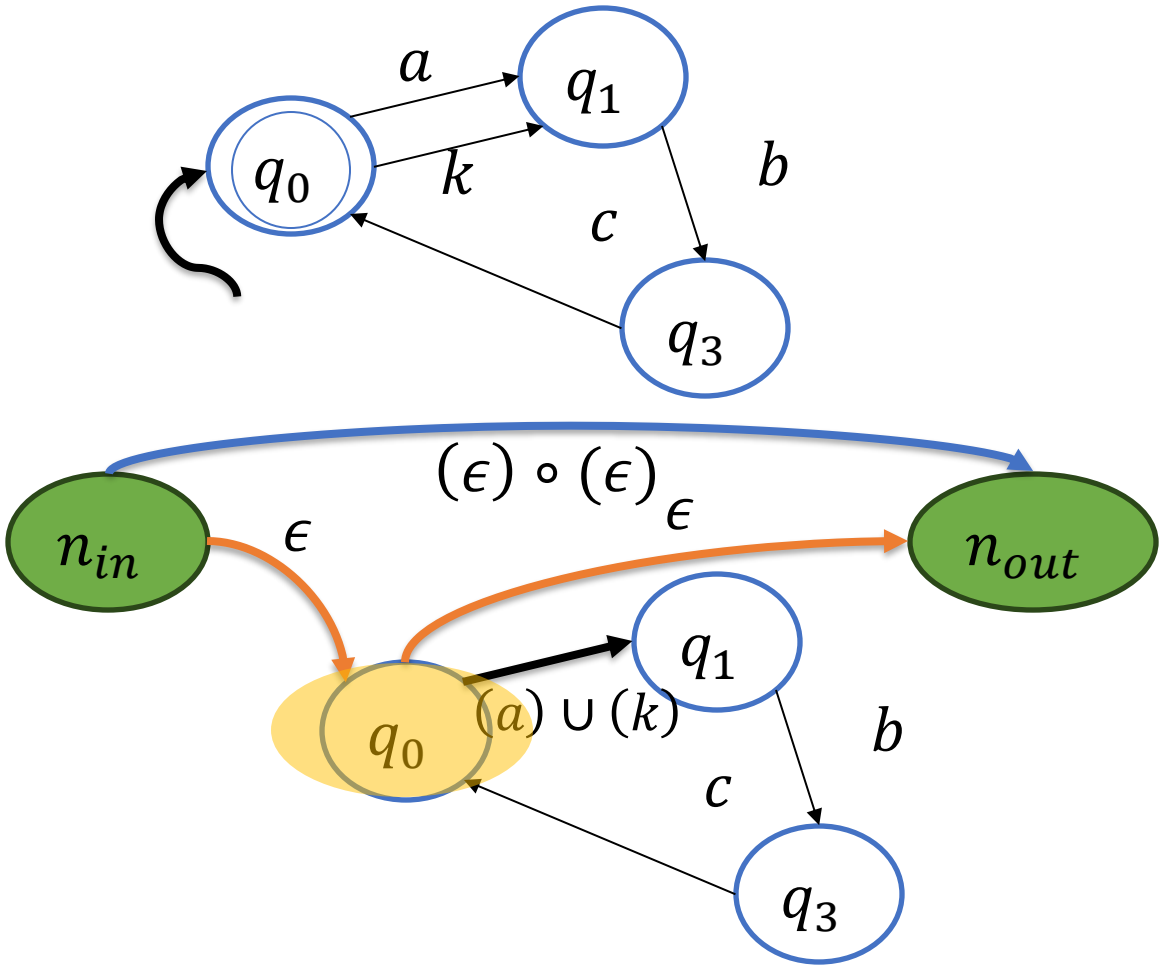


Da ASFND a Espressione Regolare – Esempio

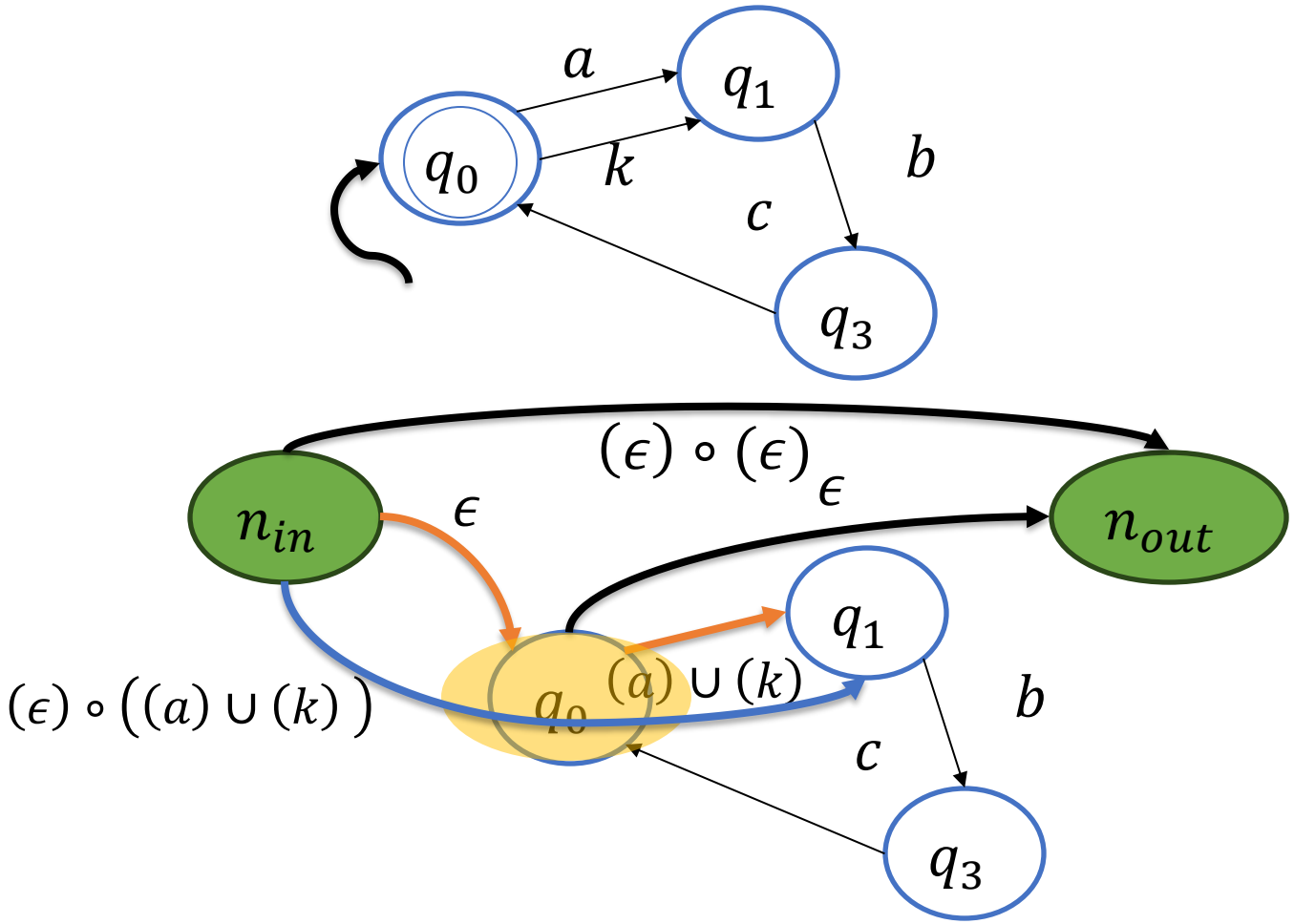




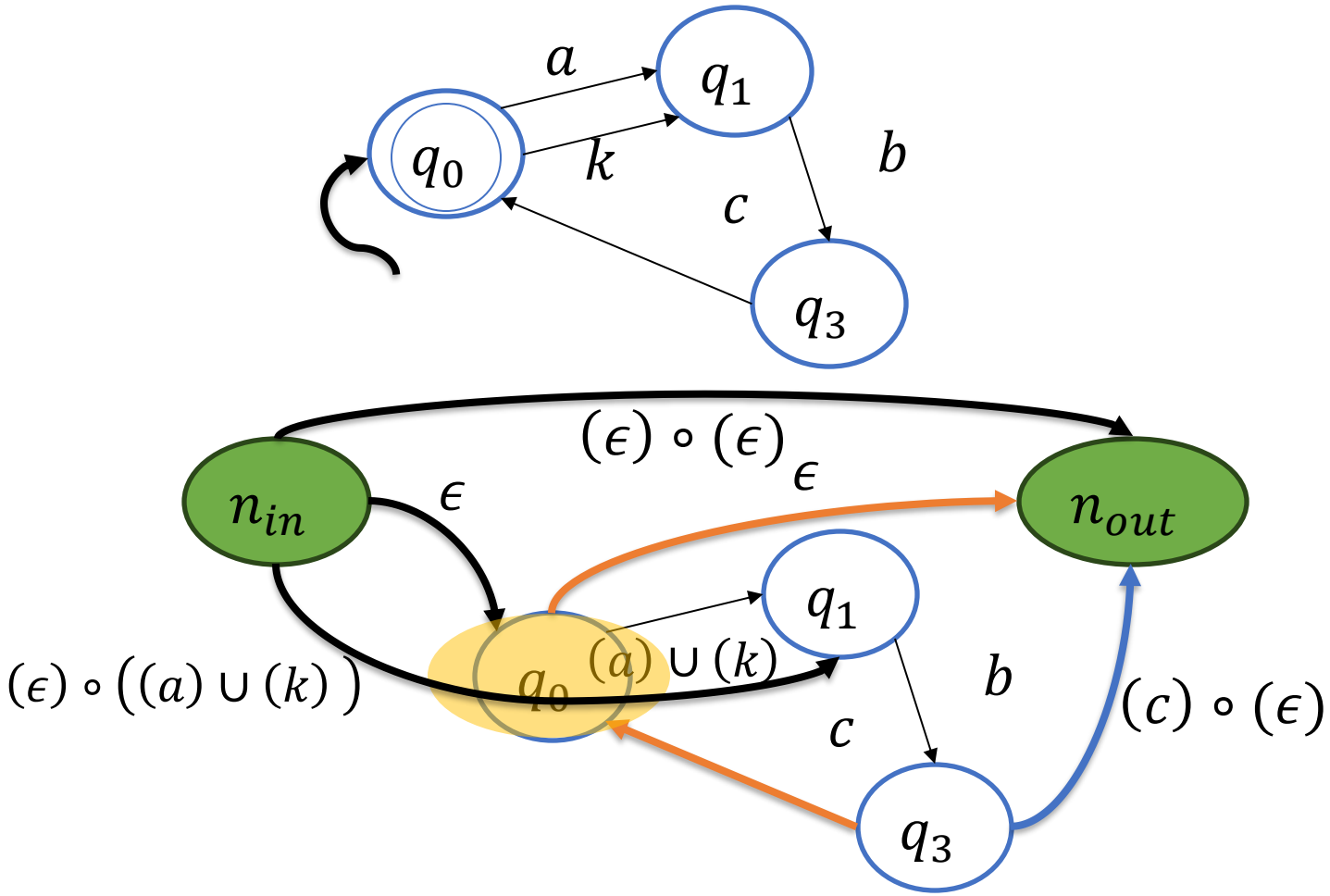
Da ASFND a Espressione Regolare – Esempio



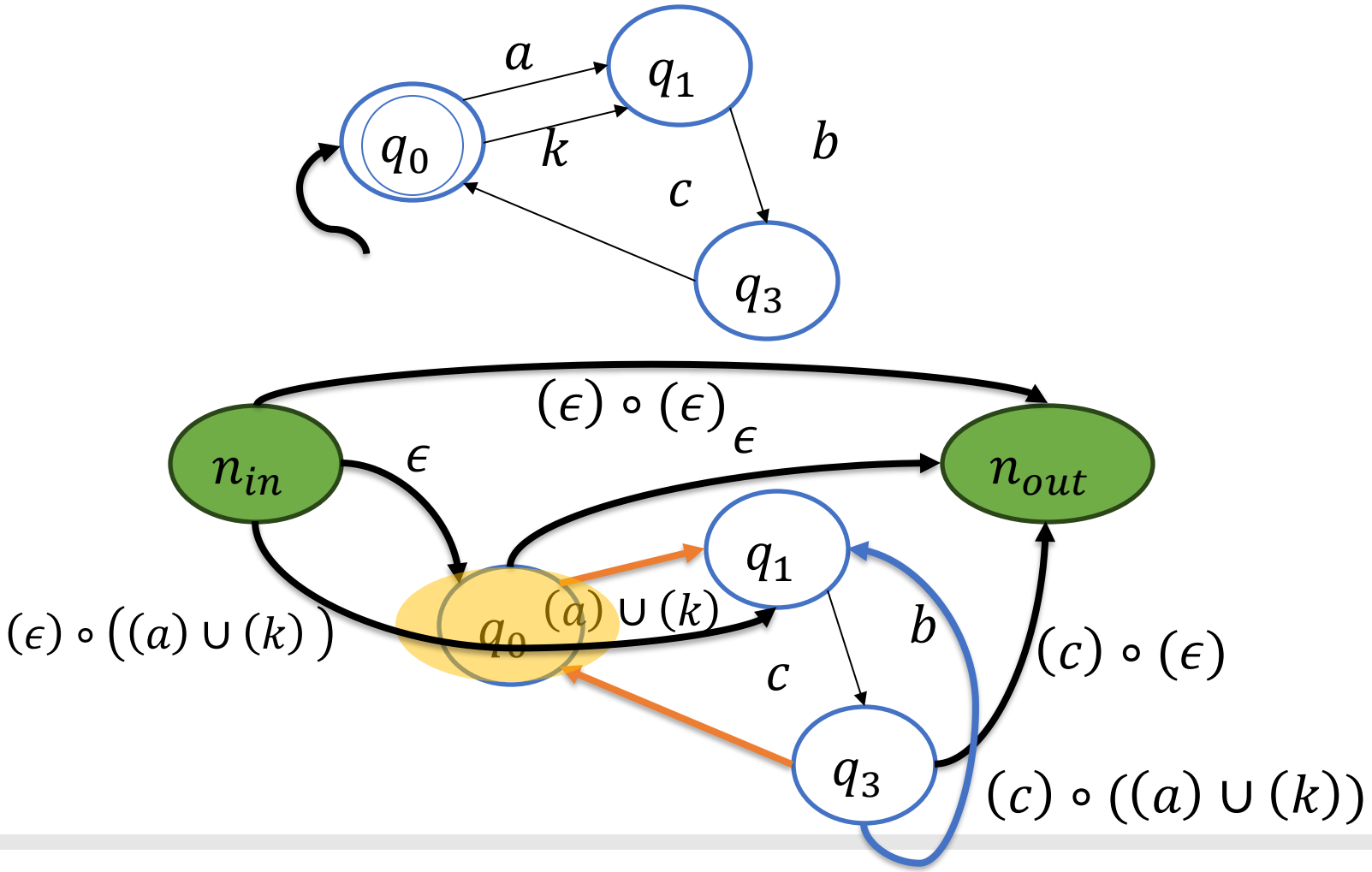
Da ASFND a Espressione Regolare – Esempio



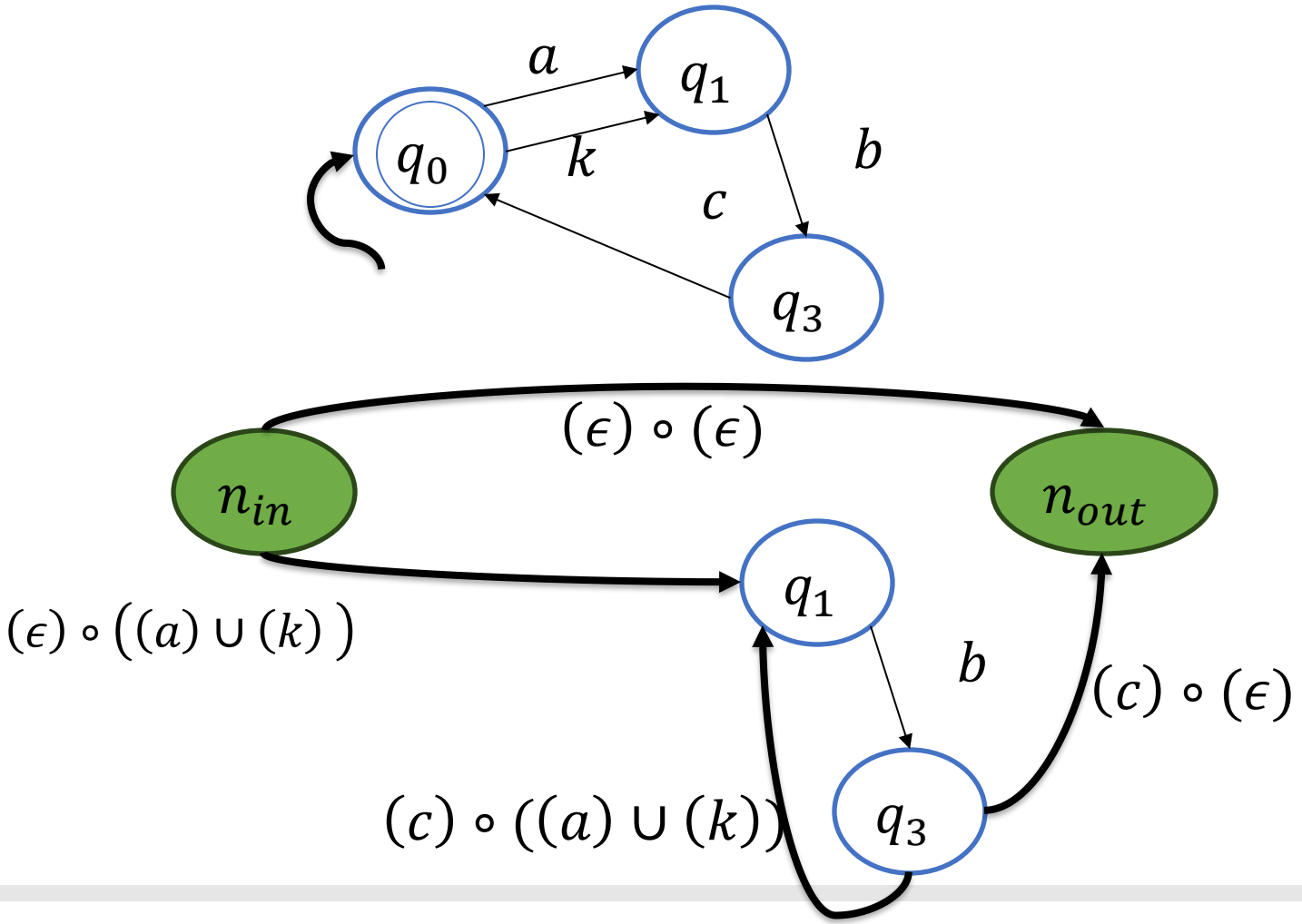
Da ASFND a Espressione Regolare – Esempio



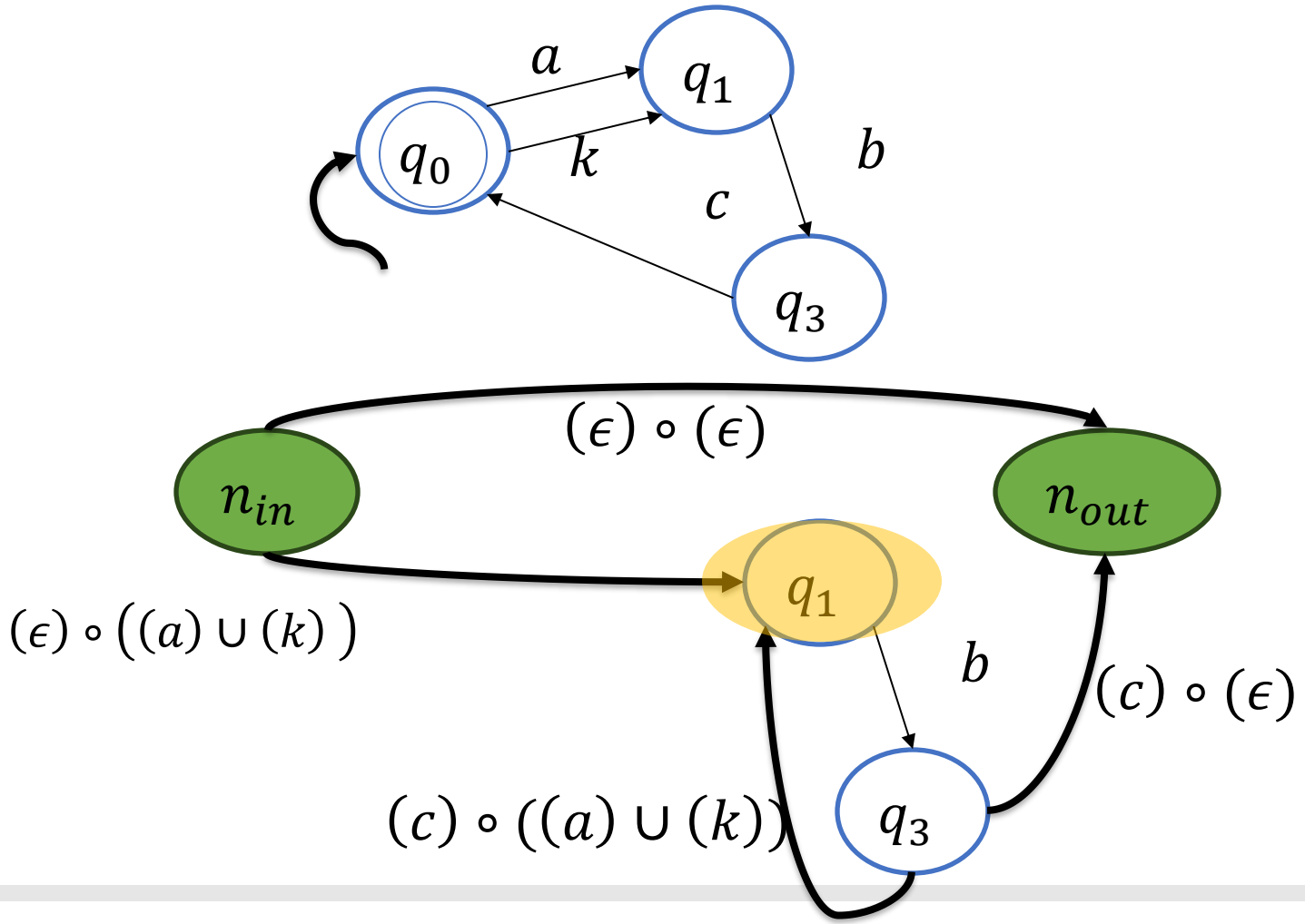
Da ASFND a Espressione Regolare – Esempio



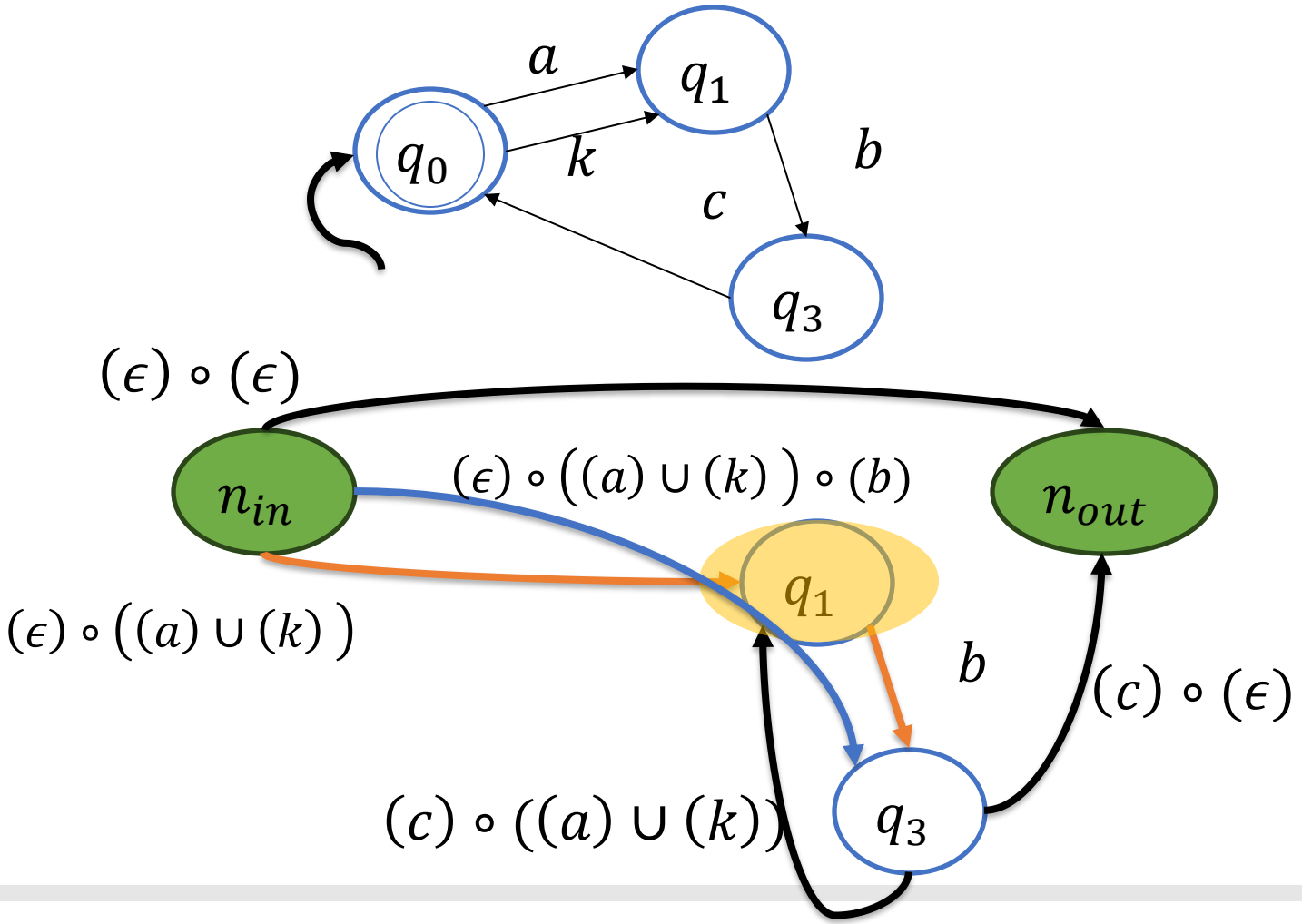
Da ASFND a Espressione Regolare – Esempio



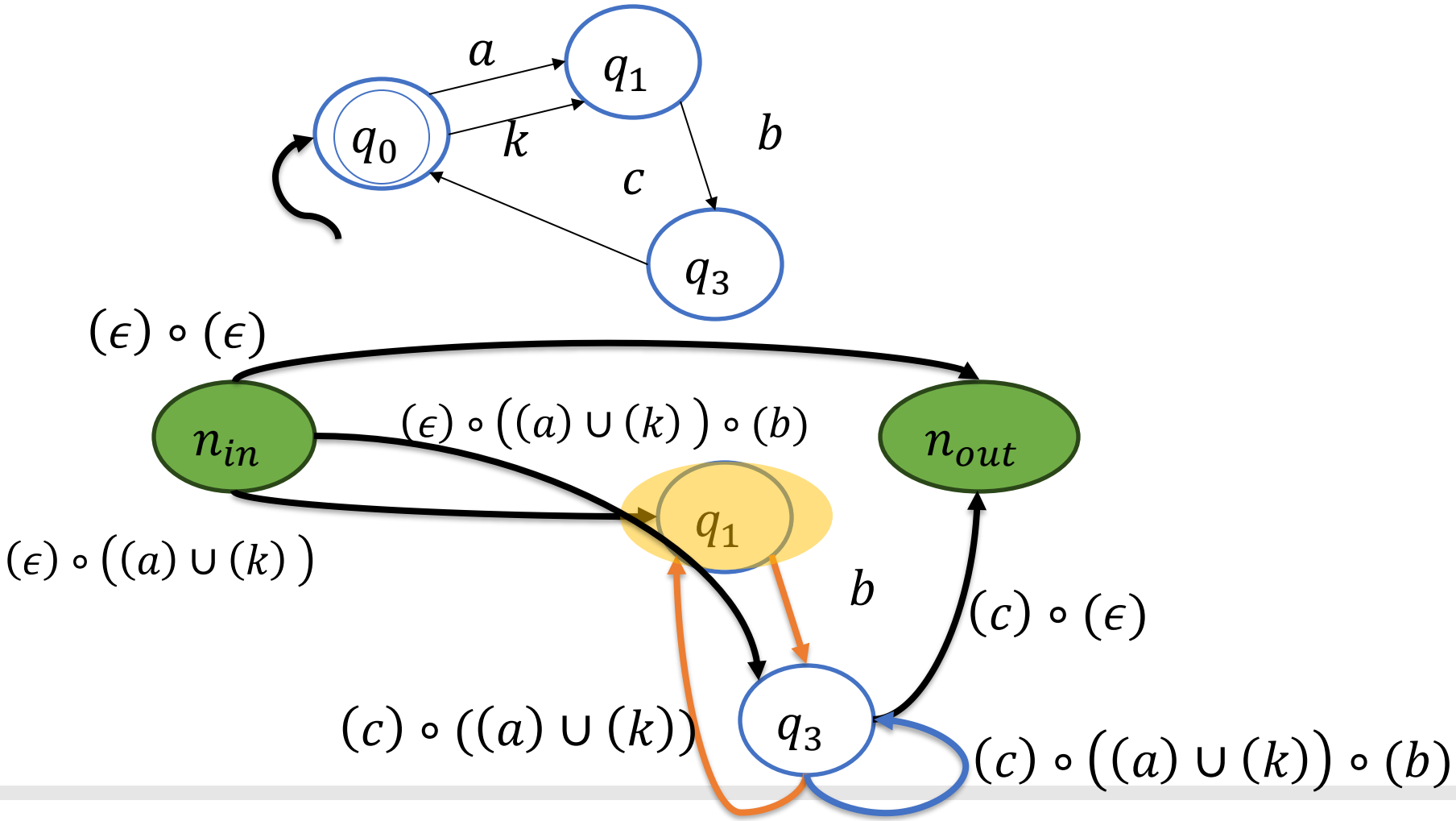
Da ASFND a Espressione Regolare – Esempio



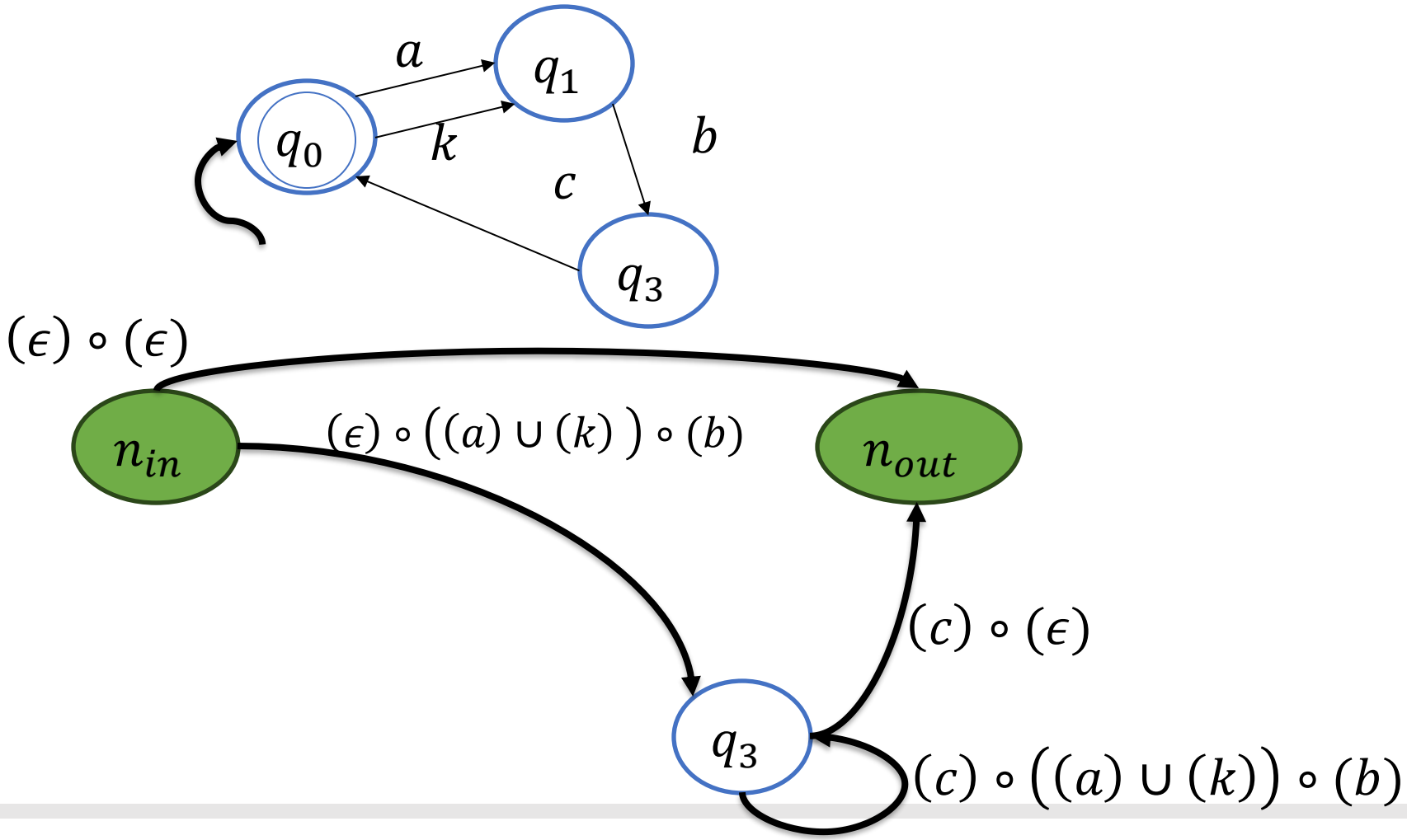
Da ASFND a Espressione Regolare – Esempio



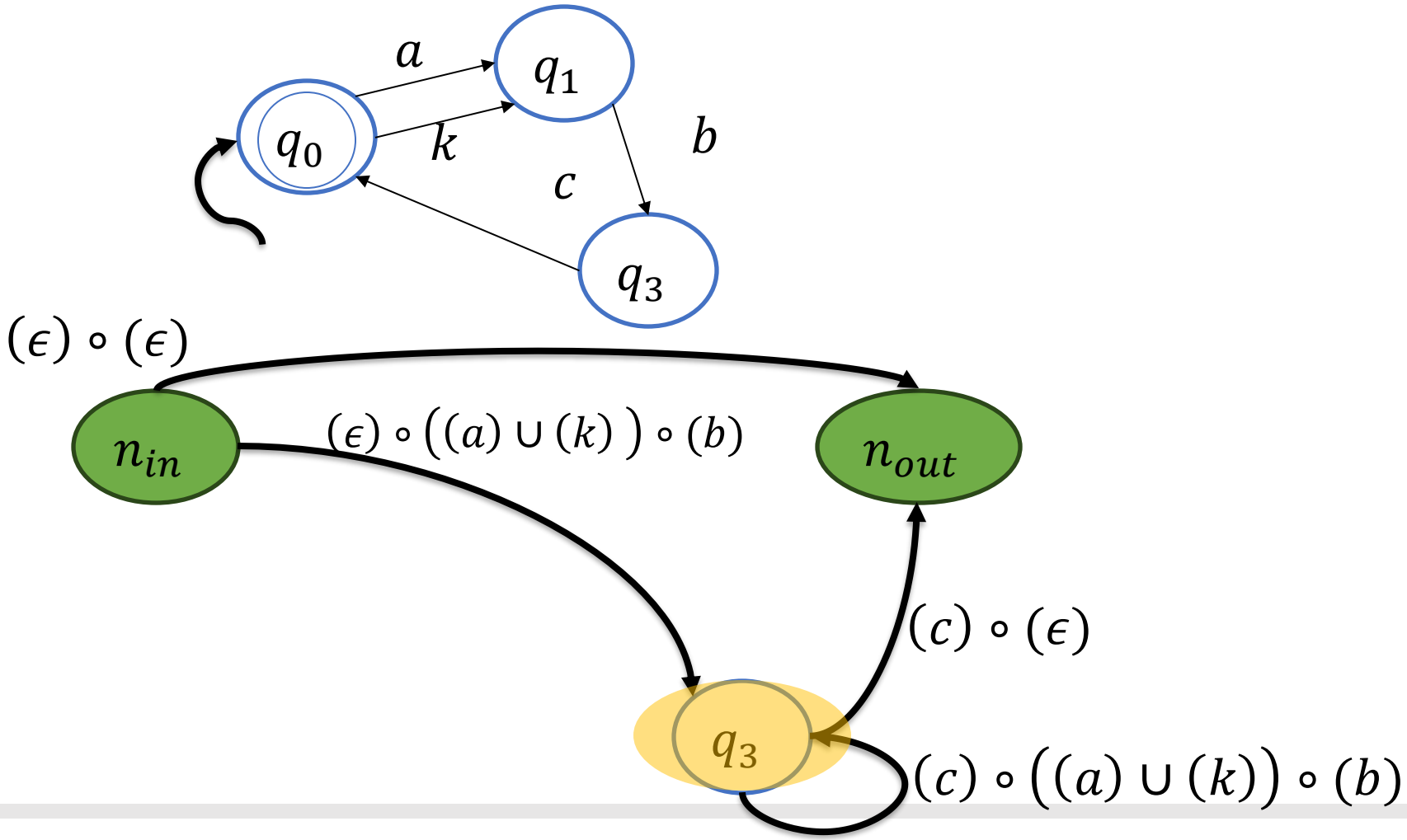
Da ASFND a Espressione Regolare – Esempio



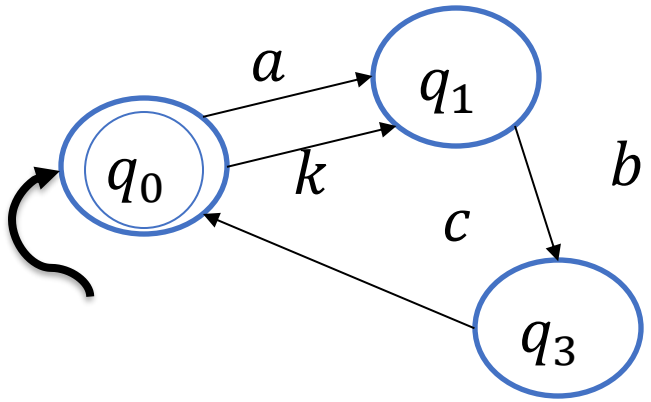
Da ASFND a Espressione Regolare – Esempio



Da ASFND a Espressione Regolare – Esempio



Da ASFND a Espressione Regolare – Esempio



$(\epsilon) \circ (\epsilon)$

n_{in}

$(\epsilon) \circ \big((a) \cup (k)\big) \circ (b)$

n_{out}

$(\epsilon) \circ \big((a) \cup (k)\big)$

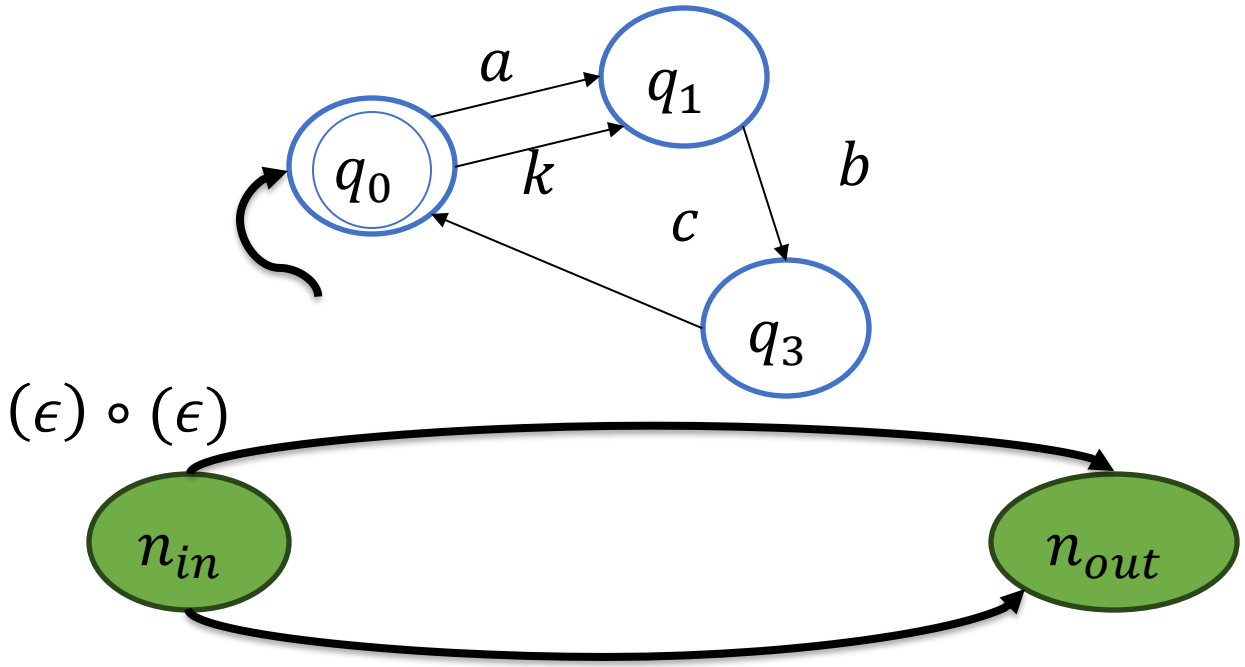
$\circ\, (b) \circ \Big((c) \circ \big((a) \cup (k)\big) \circ (b)\Big)^{*} \circ \big((c) \circ (\epsilon)\big)$

q_3

$(c) \circ (\epsilon)$

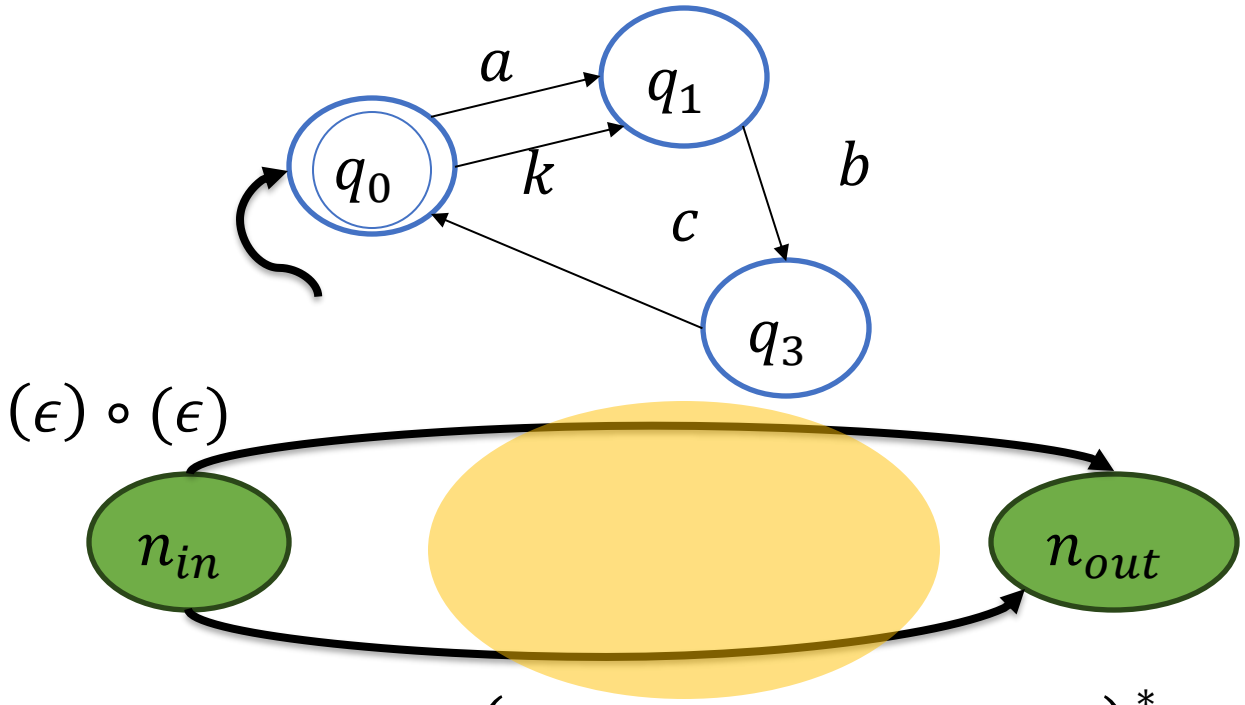
$(c) \circ \big((a) \cup (k)\big) \circ (b)$

Da ASFND a Espressione Regolare – Esempio



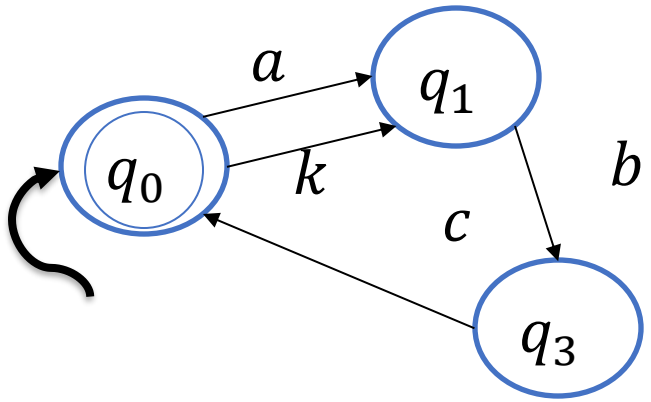
$$(\epsilon) \circ \big((a) \cup (k)\big) \circ (b) \circ \Big((c) \circ \big((a) \cup (k)\big) \circ (b)\Big)^{*} \circ \big((c) \circ (\epsilon)\big)$$

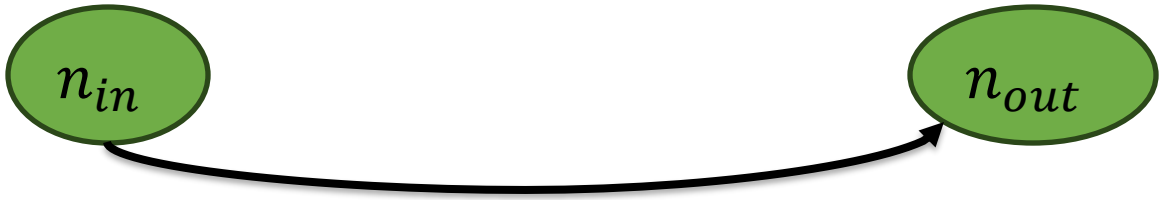
Da ASFND a Espressione Regolare – Esempio



$$(\epsilon) \circ \big((a) \cup (k)\big) \circ (b) \circ \Big(\big((c) \circ \big((a) \cup (k)\big) \circ (b)\big)^{*} \circ \big((c) \circ (\epsilon)\big)\Big)$$

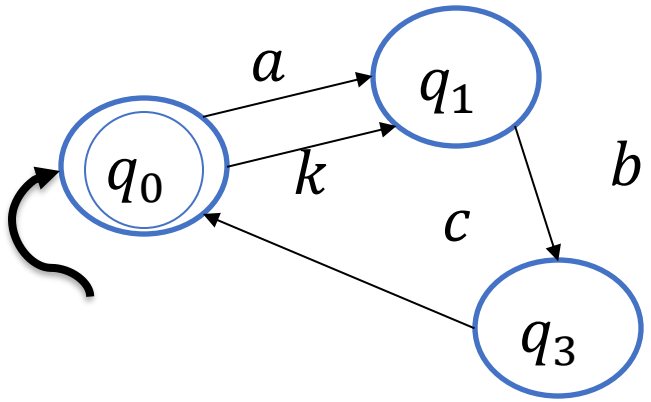
Da ASFND a Espressione Regolare – Esempio





$$\big((\epsilon)\circ(\epsilon)\big)\cup\Big((\epsilon)\circ\big((a)\cup(k)\big)\circ(b)\circ\big((c)\circ\big((a)\cup(k)\big)\circ(b)\big)^{*}\circ\big((c)\circ(\epsilon)\big)\Big)$$

Da ASFND a Espressione Regolare – Esempio



$$\Big((\epsilon)\circ(\epsilon)\Big)\cup\Big((\epsilon)\circ\big((a)\cup(k)\big)\circ(b)\circ\Big((c)\circ\big((a)\cup(k)\big)\circ(b)\Big)^{*}\circ\big((c)\circ(\epsilon)\big)\Big)$$

$$(\epsilon)\cup\Big(\big((a)\cup(k)\big)\circ(b)\circ\Big((c)\circ\big((a)\cup(k)\big)\circ(b)\Big)^{*}\circ(c)\Big)$$

$$\Big(\big((a)\cup(k)\big)\circ(b)\circ(c)\Big)^{*}$$

Le Espressioni Regolari catturano gli ASFD – Parte 2

- Per ogni ASFD A esiste una espressione regolare r_A che descrive il linguaggio riconosciuto da A, ovvero tale che $L(A) = L(r_A)$

- **Proposizione**. L'algoritmo che abbiamo mostrato con input A produce una espressione regolare r_a tale che $L(A) = L(r_A)$
- **Dimostrazione**. Dobbiamo dimostrare due proprietà.
  1. L'algoritmo termina per ogni input. Banale, il numero di stati decresce sempre di uno.
  2. L'algoritmo ritorna l'espressione regolare corretta. L'intuizione dietro alla prova formale è che ogni volta che rimuoviamo uno stato X costruiamo le espressioni regolari che sono definite da X e i suoi vicini nel grafo indotto dall'automa

Altre Proprietà di Chiusura

Altre Proprietà di Chiusura

- Precedentemente, abbiamo dimostrato che la famiglia dei linguaggi regolari è chiusa rispetto:
  - **L'unione (Proposizione 1.1)**
  - **La concatenazione (Proposizione 1.4)**
  - **L'iterazione (Proposizione 1.5)**

- Dobbiamo ancora dimostrare che la classe dei linguaggi regolari è chiusa anche rispetto:
  - **La complementazione**
  - **L'intersezione**

Chiusura rispetto la Complementazione

- Dato un ASFD $A =< \Sigma, Q, \delta, q_0, F >$, esiste un ASFD A' tale che $L(A') = \overline{L(A)} = \Sigma^* \setminus L(A)$

- Sia $A =< \Sigma, Q, \delta, q_0, F >$ un ASFD. Definiamo adesso l'ASFD A' nel seguente modo $A' = < \Sigma', Q', \delta', q_0', F' >$ tale che:
  - $\Sigma' = \Sigma$
  - $Q' = Q$
  - $q_0' = q_0$
  - $\delta' = \delta$
  - $F' = Q \setminus F$

- E' immediato verificare che $L(A') = \overline{L(A)} = \Sigma^* \setminus L(A)$. Più precisamente, ogni stringa che porta l'automa A in uno stato finale porterà l'automa A' in uno stato non finale; viceversa, ogni stringa che porta l'automa A in uno stato non finale porterà l'automa A' in uno stato finale

Chiusura rispetto l'Intersezione

Dati due ASFD A_1 e A_2, esiste un ASFD A tale che $L(A) = L(A_1) \cap L(A_2)$

Dimostrazione: Basta applicare la legge di De Morgan, ovvero

$$L(A_1) \cap L(A_2) = \overline{\overline{L(A_1)} \cup \overline{L(A_2)}}$$