UNIVERSITÀ DEGLI STUDI DI TRIESTE

Computabilità, Complessità e Logica

Prof. Adriano Peron

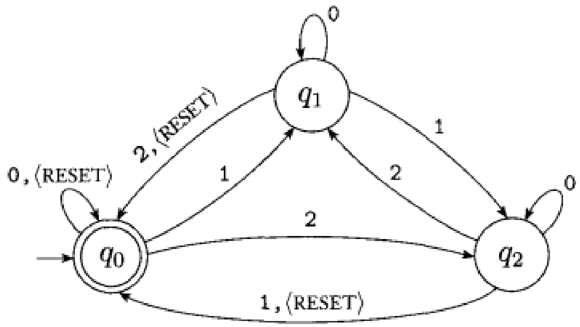
Computabilità: Automi regolari

Automi regolari (Macchine a stati finiti)

- **Modello computazionale semplice ed efficace**
  - **E una macchina stato-transizione con un numero finito di stati.**
  - **Rappresentazione come grafo con gli archi etichettati.**
- **Intuizione: rappresentazione di un sistema che interagisce con il modo esterno.**

Esempio: Contatore modulo 3

- Operazioni con l'ambiente esterno:
- Reset
- Incremento 0
- Incremento 1
- Incremento 2

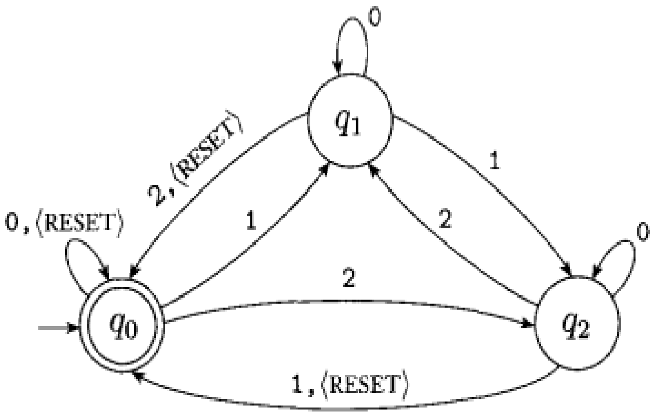


Automi regolari (Macchine a stati finiti)

Stato.

- Nodo del grafo. Rappresentazione la condizione interna del sistema. Il suo stato di controllo, il contenuto delle sue variabili.
- Esempio. Uno stato per ogni valore del contatore

- Transizione.
  - Arco orientato. Rappresenta una variazione della condizione interna del Sistema a seguito di una interazione col mondo esterno.
  - Etichetta. Rappresenta il tipo di sollecitazione del mondo esterno che determina il cambio di stato.
  - Esempio. Operazioni di incremento o reset sul contatore.



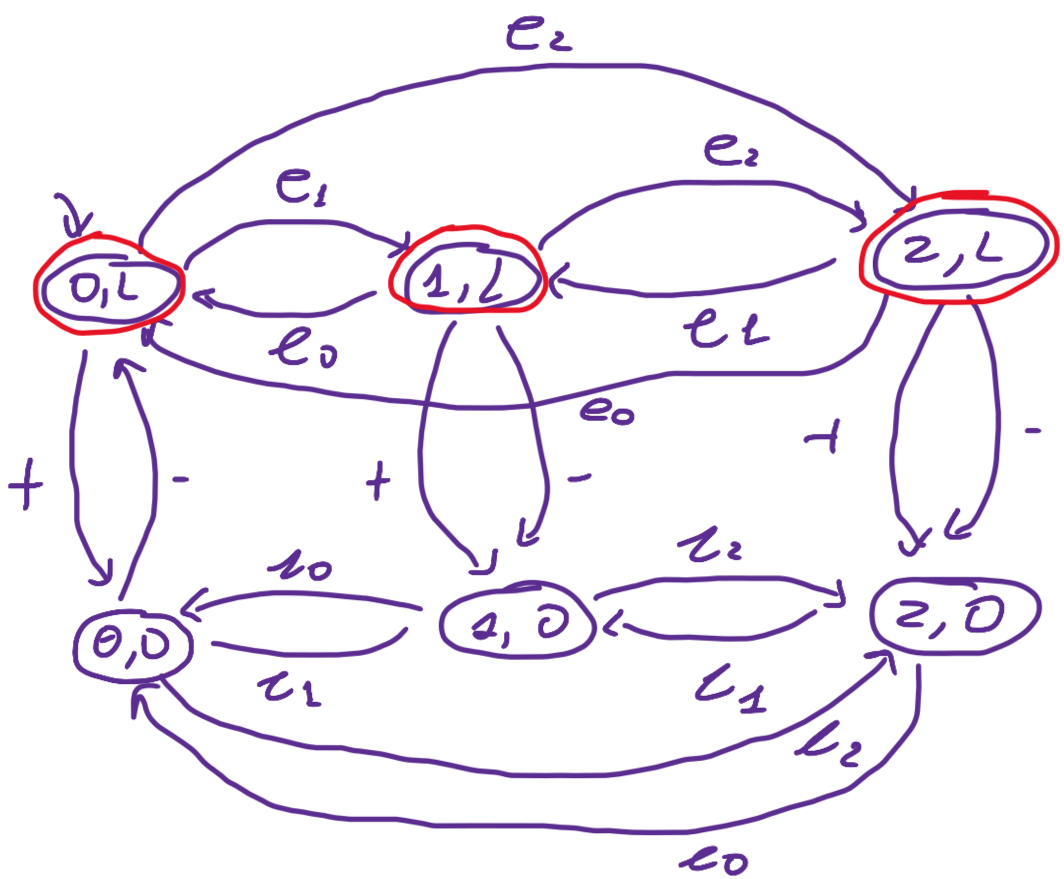
Caratteristiche del modello

- Il numero degli stati è finito;
- Computazione controllata dall'ambiente esterno
  - E possible una evoluzione della macchina senza un trigger esterno?
- Il comportamento della macchina è **deterministico**: fissato lo stato di partenza e l'input dell'ambiente esterno è determinato in maniera univoca lo stato di arrivo.
- Comunicazione unidirezionale Ambiente-Macchina
  - La macchina può comunicare con l'esterno?

Esempio: il controllo di un ascensore

- Lo stato ha due componenti:
  - il piano $P = \{0, 1, 2\}$ e
  - lo stato di occupazione libero/occupato $S = \{L, O\}$.
- Ciascuno stato è un elemento del prodotto $P \times S$
- Lo stato iniziale è $(0, L)$
- Gli stati finali sono $\{(0, L), (1, L), (2, L)\}$
- L'alfabeto contiene simboli per
  - le chiamate dell'ascensore al piano $\Sigma_e = \{e_0, e_1, e_2\}$
  - le indicazioni di direzione interna $\Sigma_i = \{i_0, i_1, i_2\}$
  - Ingresso e uscita dall'ascensore $\Sigma_{IO} = \{+, -\}$

Esempio: il controllo di un ascensore

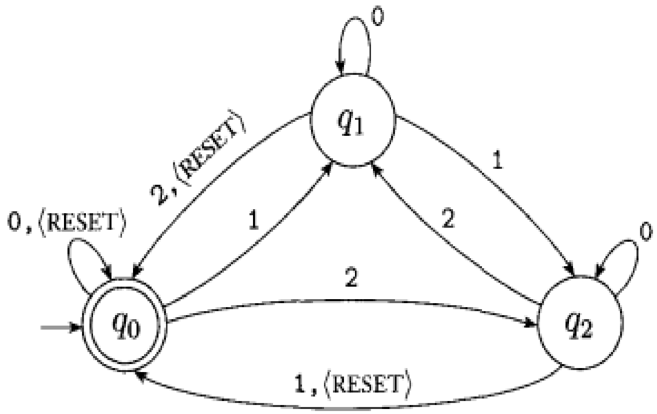


Automi regolari: sintassi

Sintassi formale.

$<Q,\Sigma,\delta,q_0,F>$

- Q è un insieme **finito** di stati.
- Σ è un **alfabeto**, insieme di simboli di etichette per gli archi delle transizioni.
- δ è la **funzione di transizione** $\delta : Q \times \Sigma \rightarrow Q$ che, fissato uno stato (sorgente) e un simbolo dell'alfabeto, determina lo stato successivo (destinazione).
- q_0 è lo **stato iniziale.**
- $F \subseteq Q$ è l'insieme degli **stati finali**.

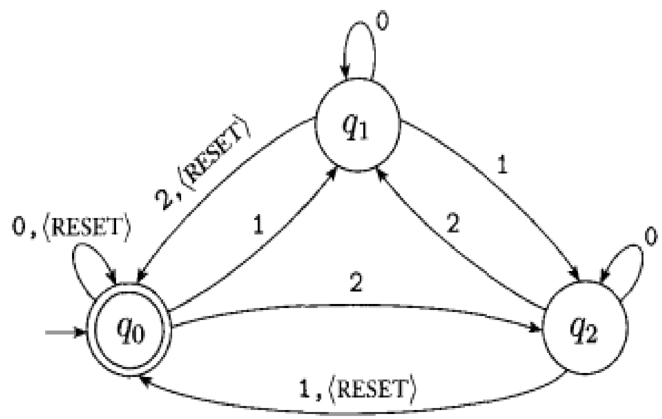


Automi regolari: sintassi

Sintassi formale. Esempio.

$\langle Q,\Sigma,\delta,q_0,F \rangle$

- $Q = \{q_0, q_1, q_2\}$.
- $\Sigma = \{RESET, 0,1,2\}$
- $\delta = \{(q_0,0, q_0), (q_0,1, q_1), (q_0, 2, q_2), (q_0,RESET,q_0),$
  $(q_1,0, q_1), (q_1,1, q_2), (q_1,2, q_0), (q_1,RESET,q_0),$
  $(q_2,0, q_2), (q_2,1, q_1), (q_2,2, q_1), (q_2,RESET,q_0)\}$
- q_0 è lo **stato iniziale.**
- $F =\{q_0, q_1, q_2\}$..



Automi regolari: notazione

Notazione

$<Q,\Sigma,\delta,q_0,F>$

- Σ un alfabeto di simboli. (esempio $\Sigma=\{a,b\}$)
- Σ^* **rappresenta l'insieme delle parole di lunghezza finita sull'alfabeto Σ**
- Le parole su Σ si ottengono sequenzializzando un numero arbitrario ma finito (anche 0) di simboli di Σ.
  - Comprende la parola nulla ϵ (epsilon) (sequenzializzazione di 0 simboli di Σ)
  - Comprende parole della forma

$$w= \sigma_1\ \sigma_2 \ldots\sigma_n,\ \text{con } \sigma_i \in \Sigma^* \text{ per ogni } i,\ 1\leq i \leq n$$

- **$\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$ rappresenta l'insieme delle parole finite non nulle sull'alfabeto Σ**
- **$|w|$ denota (rappresenta) la lunghezza della parola w**
  - $|\ \epsilon\ |=0$
  - $|w|= n$ se $w= \sigma_1 \ldots\sigma_n$

Automi regolari: notazione

Per una parola w,

- w(i) con $1 \leq i \leq |w|$, denota l'i-esimo simbolo della parola w

  (se w= $\sigma_1\ \sigma_2 \ldots.\sigma_n$, allora w(i)= σ_i per $1 \leq i \leq n$)

- w(i,j) con $1 \leq i \leq j \leq |w|$, denota la sottoparola di w (infisso) delimitata dall'i-esimo e j-esimo simbolo.

  (se w= $\sigma_1\ \sigma_2 \ldots.\sigma_n$, allora w(i,j)= $\sigma_i \ldots\ \sigma_j$ per $1 \leq i \leq j \leq n$)

- Concatenazione di parole su Σ^*

  date due parole w,w'$\in \Sigma^*$ sullo stesso alfabeto w·w' denota la concatenazione di w e w'

  se w= $\sigma_1\ \sigma_2 \ldots.\sigma_n$, e w'= $\sigma'_1\ \sigma'_2 \ldots.\sigma'_m$ allora w·w'= $\sigma_1\ \sigma_2 \ldots.\sigma_n\sigma'_1\ \sigma'_2 \ldots.\sigma'_m$

  Si osservi che la parola vuota funge da elemento neutro nella concatenazione

  $w \cdot \epsilon = w$ e $\epsilon \cdot w = w$

Automi regolari: notazione

Un linguagggio L su un alfabeto di simboli Σ è un insieme di parole (possibilmente vuoto) di Σ^*. (In simboli, $L \subseteq \Sigma^*$)

- **L = ∅ è un linguaggio (il linguaggio vuoto)**
- **Attenzione: L = {ϵ} non è il linguaggio vuoto !!!**
- **Esempio per l'alfabeto Σ={a,b}**
- **L_1={ϵ,a,aa,aaa,aaaa,……..} (tutte le sequenze di a di lunghezza arbitraria)**

$$L_1=\{a^n : n \geq 0\}$$

- **L_2={ϵ,ab,aabb,aaabbb,aaaabbbb,……..} (tutte le sequenze di lunghezza arbitraria di a seguito da uno stesso numero di di b)**

$$L_2=\{a^n b^n : n \geq 0\}$$

- **L_3 parole di lunghezza k**

$$L_3=\{w \in \Sigma^*: |w| = k\}$$

Automi regolari: semantica intuitiva

Descrizione di una computazione di un automa regolare $<Q,\Sigma,\delta,q_0,F>$

- L'automa inizia la computazione nello stato iniziale;
- L'evoluzione è guidata da una sequenza di simboli (parola) dell'alfabeto Σ

$$w = \sigma_1\sigma_2 \ldots\sigma_n,\ w \in \Sigma^*$$

- I simboli della parola vengono elaborati in sequenza (da σ_1 a σ_n) un simbolo alla volta
- L'elaborazione di un simbolo produce un cambiamento di stato (transizione)
- Una computazione è la sequenza degli stati che l'automa assume nell'elaborazione dei simboli.
- Una computazione per una parola ha buon fine se lo stato raggiunto è uno stato classificato come finale.

Automi regolari: semantica

Descrizione di una computazione di un automa regolare $<Q,\Sigma,\delta,q_0,F>$

- Intuitivamente è una sequenza di transizioni dallo stato iniziale.
- Viene fissata una sequenza di simboli dell'alfabeto (una parola sull'alfabeto Σ)

$$w = \sigma_1\sigma_2 \ldots.\sigma_n,\ w \in \Sigma^*$$

- Una computazione per la parola w è una sequenza di stati

$$q_1, q_2, \ldots., q_n, q_{n+1}$$

- q_1 è lo stato iniziale.
- $q_i \in Q$ per ogni i, $1 \leq i \leq n+1$
- $(q_i, \sigma_i, q_{i+1}) \in \delta$ per ogni i, $0 \leq i \leq n$ (è possible scrivere anche $(q_i, w(i), q_{i+1}) \in \delta$ per ogni i, $0 \leq i \leq n$.
- La computazione è accettante se $q_{n+1} \in F$.

Automi regolari: semantica

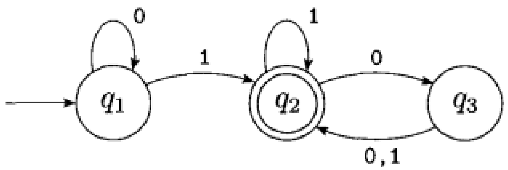
Linguaggio L(A) riconosciuto da automa regolare A=<Q,Σ,δ,q_0,F>

- E' l'insieme delle parole sull'alfabeto per cui l'automa A ha una computazione accettante.

  L(A)= {w ∈ Σ^* : la computazione di A per w è accettante}

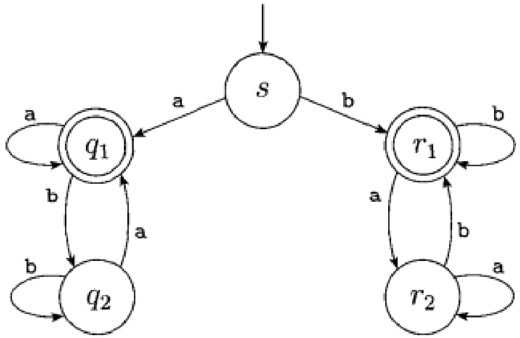
- Un linguaggio è detto **regolare** se è riconosciuto da un automa regolare.
- Attenzione: essendo l'automa **deterministico** (la relazione di transizione è una funzione) la parola **determina in modo univoco** la computazione.
- Un automa astrattamente è visto come **accettore di parole** e, dunque, come un **riconoscitore di linguaggi**.
- Dal punto di vista di un sistema può essere interpretato come una macchina che definisce le modalità accettabili di interazione con la macchina stessa.
- L'interazione è corretta se pilota la macchina ad uno stato di accettazione.

Esempi. Linguaggi accettati



Linguagio L(A) riconosciuto

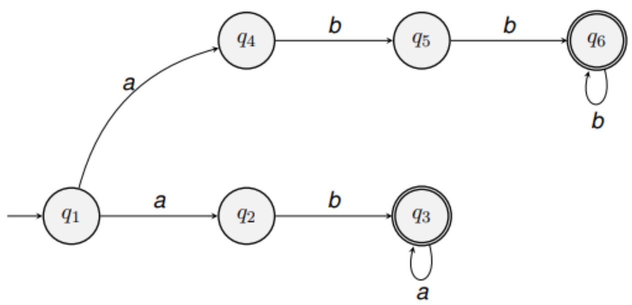
L(A)= $\{w \in \{0,1\}^* : w = w' \cdot 1 \cdot w'',\ w' \in \{0\}^*,\ w'' \in \{00,01,1\}^* \}$

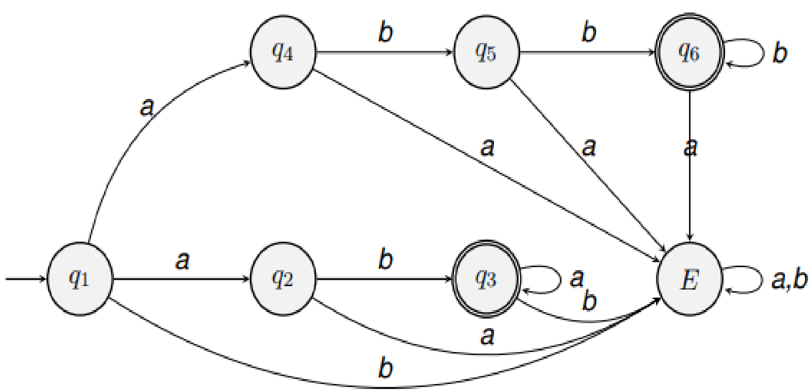


Linguagio L(A) riconosciuto

L(A)= $\{w \in \{a,b\}^* : w = ?$ $\}$

Esempi. Linguaggi accettati





Linguagio L(A) riconosciuto

L(A)= {w ∈ {0,1}* : w=ab· w'', w''∈ {a*,bb*} }

Si osservi che l'automa non è completamente definito poiché ad esempio dallo stato q_1 non ci sono archi etichettati dal simbolo b

La relazione di transizione δ è una funzione totale ed è definita per ogni elemento dell'alfabeto.

La mancanza di un arco etichettato dal simbolo simbolo b indica che se nello stato q_1 si osserva b allora la computazione non potrà essere accettante.

L'automa equivalente con la funzione di transizione completamente esplicitata usa uno stato pozzo E che non può raggiungere gli stati accettanti

I due automi accettano lo stesso linguaggio

Esempi. Linguaggi accettati

Automi per i seguenti linguaggi?

$L_1(A) = \{w \in \{a,b\}^* : w$ ha un numero pari di a$\}$

$L_2(A) = \{w \in \{a,b\}^* : w$ ha un numero pari di a e di b$\}$

E' possibile esprimere un automa regolare per ogni possibile linguaggio?

Tutti i linguaggi su un alfabeto sono regolari?

Ad. Es.

$L_2(A) = \{w \in \{a,b\}^* :$ stesso numero di a e di b$\}$

$L_3(A) = \{w \in \{a,b\}^* :$ maggior numero di a che di b$\}$

$L_4(A) = \{w \in \{a,b\}^* : w = a^n \cdot b^n\}$

$L_3(A) = \{w \in \{a,b\}^* :$ maggior numero di a che di b$\}$

Vedremo che la capacità espressiva degli automi regolari è limitata.

E' possible dimostrare che alcuni linguaggi non sono regolari (non esiste nessun automa regolare che li possa accettare)

Primi problemi sui linguaggi regolari

Consideriamo un automa deterministico A sull'alfabeto Σ

- **Vogliamo stabilire se siamo in grado di risolvere (decidere) i seguenti problemi.**
- **Problema del linguaggio vuoto.**

  E' possible stabilire se $L(A) = \emptyset$?

- **Problema dell'appartenenza di una parola $w \in \Sigma^*$.**

  E' possible stabilire se w $\in L(A)$?

- **Problema dell'universalità.**

  E' possible stabilire se $L(A) = \Sigma^*$?

- **Problema della finitezza**

  E' possible stabilire se $L(A)$ è finito?

Per risolvere i problem elencati ricordiamo che un automa è un grafo.

Equivalenza tra automi

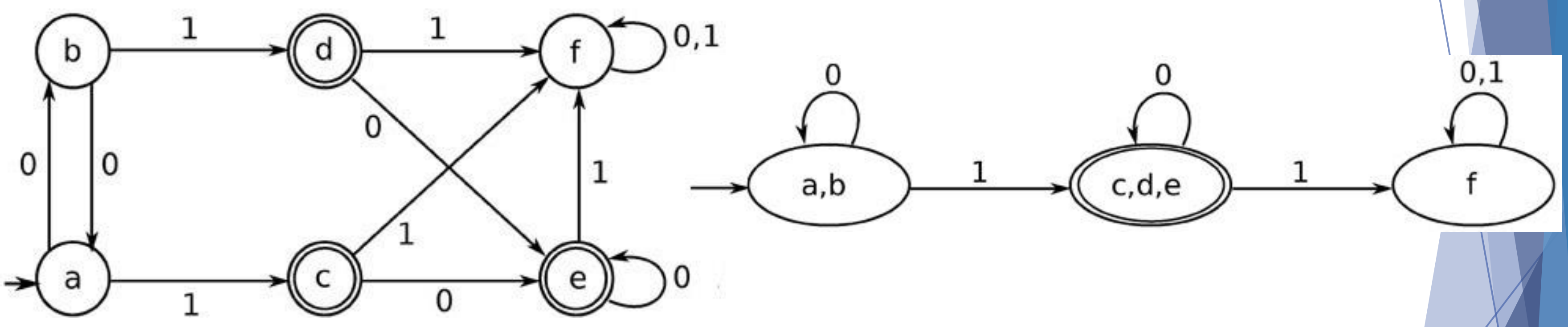
Nozione di equivalenza

Due automi A_1 e A_2 si dicono equivalenti se accettano lo stesso linguaggio:

$$L(A_1) = L(A_2)$$

Lo stesso linguaggio L può essere accettato da automi diversi

Esempio.



Si preferiscono automi che minimizzano il numero di stati.

Perché?

Automi regolari non deterministici

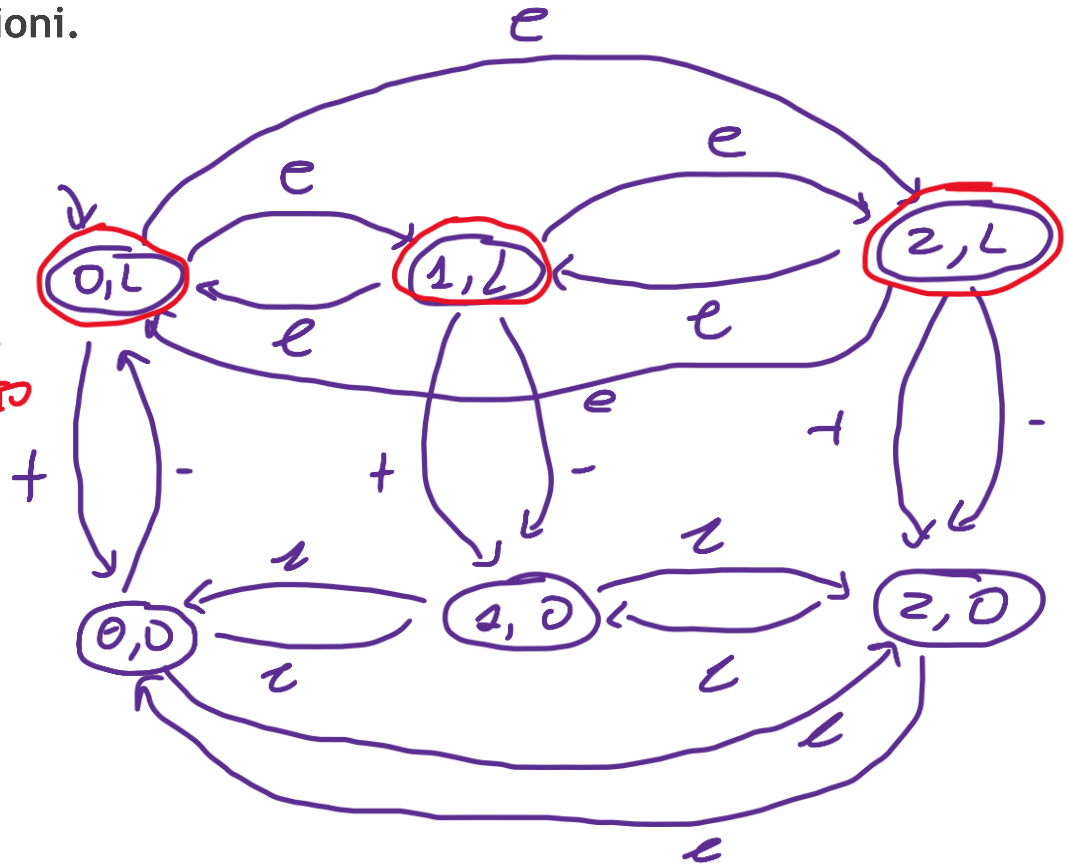
Estensione degli automi regolari deterministici.

- **La transizione da uno stato q non è più univocamente determinata da un simbolo dell'alfabeto.**
- **Se in uno stato vi sono più transizioni etichettate dallo stesso simbolo dell'alfabeto si sceglie non-deterministicamente quale transizione far scattare.**
- **Scelta non deterministica:**
- **Una scelta casuale tra diverse possibili opzioni**
- **Si astrae dal dettaglio che determina la scelta.**
- **Può essere un modo per semplificare la descrizione di un sistema.**
- **Permette di scrivere degli automi più compatti (succinti).**

Esempio: il controllo di un ascensore

- **Versione non deterministica. Trascuro dettagli di chiamate e direzioni.**

$\Sigma_e = \{e\}$
$\Sigma_i = \{i\}$
$\Sigma_{IO} = \{+,-\}$
$\Sigma = \Sigma_e \cup \Sigma_i \cup \Sigma_{IO}$



Automi regolari non deterministici

Estensione degli automi regolari deterministici.

- La transizione da uno stato q non è più univocamente determinata da un simbolo dell'alfabeto.
- δ non è una funzione ma una relazione.

Sintassi formale.

$<Q,\Sigma,\delta,q_0,F>$

- Q è un insieme finito di stati.
- Σ è un alfabeto, insieme di simboli di etichette per gli archi delle transizioni
- δ è la relazione di transizione $\delta \subseteq Q \times \Sigma \times Q$ che, fissato uno stato (sorgente) e un simbolo dell'alfabeto, determina i possibili stati successivi (destinazione).
- q_0 è lo stato iniziale.
- $F \subseteq Q$ è l'insieme degli stati finali.

Automi regolari non-deterministici: semantica

Descrizione di una computazione di un automa non-deterministico $A=<Q,\Sigma,\delta,q_0,F>$ su una parola $w= \sigma_1\sigma_2 \ldots\sigma_n$, $w \in \Sigma^*$

- Una computazione per la parola w è una sequenza di stati

$$q_1, q_2, \ldots, q_n, q_{n+1}$$

- q_1 è lo stato iniziale.
- $q_i \in Q$ per ogni i, $1\leq i \leq n+1$
- $(q_i, \sigma_i, q_{i+1}) \in \delta$ per ogni i, $0\leq i \leq n$ (è possible scrivere anche $(q_i, w(i), q_{i+1}) \in \delta$ per ogni i, $0\leq i \leq n$.
- La computazione è accettante se $q_{n+1} \in F$.
- **Una parola w è accetta da A se esiste una computazione accettante di A per la parola w.**

Automi regolari non-deterministici: semantica

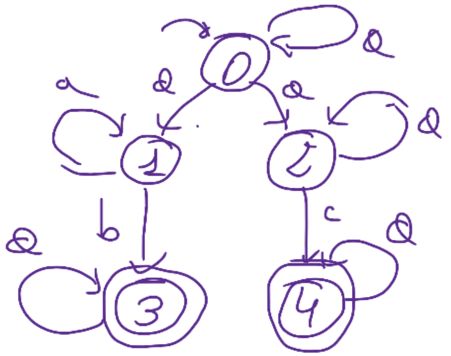
Automa deterministico.

- Una parola w ha **una sola computazione** (univocamente accettante o non accettante)

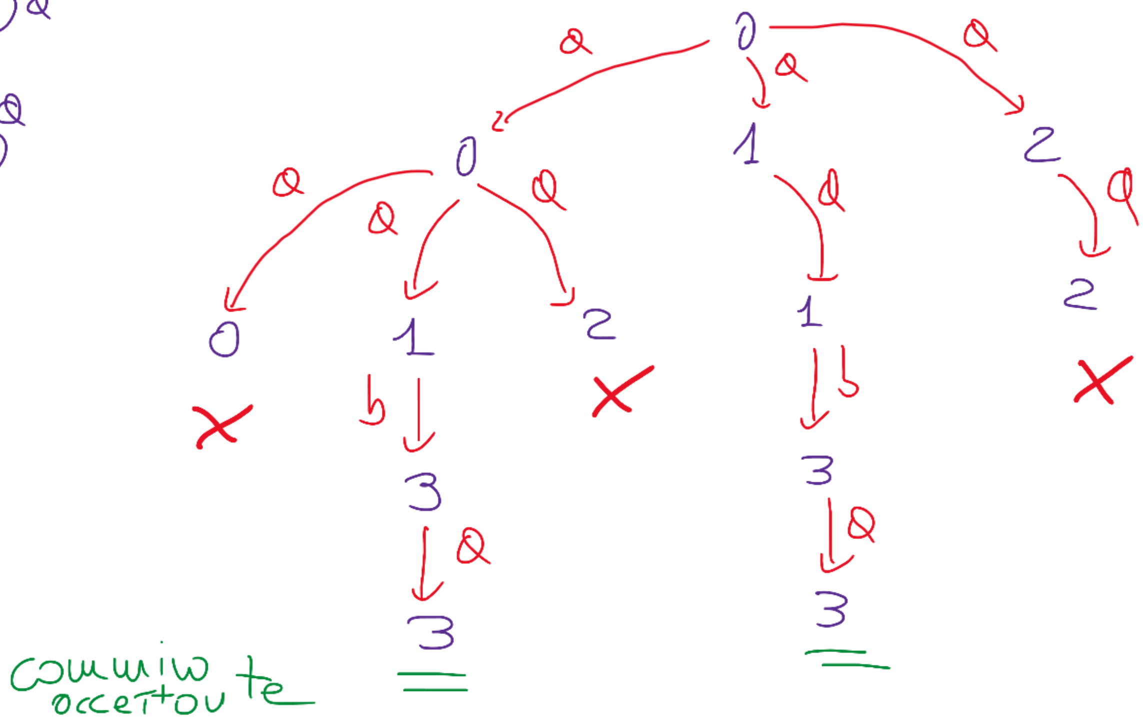
Automa non-deterministico.

- Una parola w può avere più computazioni (diverse).
- Alcune computazioni per w possono essere accettanti ed altre non accettanti
- Per accettare la parola w basta che vi sia **almeno una** computazione accettante (indifferente quale sia).
- Esempio.

Intuizione. Albero delle computazioni per una parola.



PAROLA : aaba



Automi regolari non-deterministici

Esempio

Si consideri il linguaggio L su $\Sigma = \{a_1, \ldots, a_n, \#\}$

$$L = \{v \cdot \# \cdot w : v, w \in H^*,\ v\ usa\ l'insieme\ di\ simboli\ \mathrm{H} \subseteq \{a_1, \ldots, a_n\}\ e\ w\ usa\ l'insieme\ \mathrm{H}' \subseteq \{a_1, \ldots, a_n\}\ con\ \mathrm{H} \neq H'\}$$

Automa deterministico:

- **Due fasi:**
- **La prima fase memorizza nello stato i simboli che compaiono prima di incontrare il simbolo # (insieme H).**
- **Nella seconda fase (dopo aver letto il simbolo #) si controlla che l'insieme di simboli che occorrono dopo # sia diverso da H.**
- **Per ricordare un sottoinsieme di Σ servono 2^n $stati$.**
- **L'automa deterministico richiede un numero di stati esponenziale nella cardinalità dell'alfabeto**

Esempio

Esempio di succintezza.

Si consideri il linguaggio L su $\Sigma = \{a_1, \dots, a_n, \#\}$
$\boldsymbol{L = \{v \cdot \# \cdot w : v, w \in H^*,\ esiste\ \ h \in \{a_1, \dots, a_n\}, h\ occorre\ in\ v\ e\ w\}}$

Automa deterministico:

- **Due fasi:**
- **La prima fase memorizza nello stato I simboli che compaiono prima di incontrare il simbolo #.**
- **Nella seconda fase (dopo aver letto il simbolo #) si controlla che ci sia un simbolo memorizzato nella prima parte che occorre anche nella seconda.**
- **Per ricordare un sottoinsieme di Σ servono 2^n $\boldsymbol{stati}$.**
- **L'automa deterministico richiede un numero di stati esponenziale nella cardinalità dell'alfabeto**

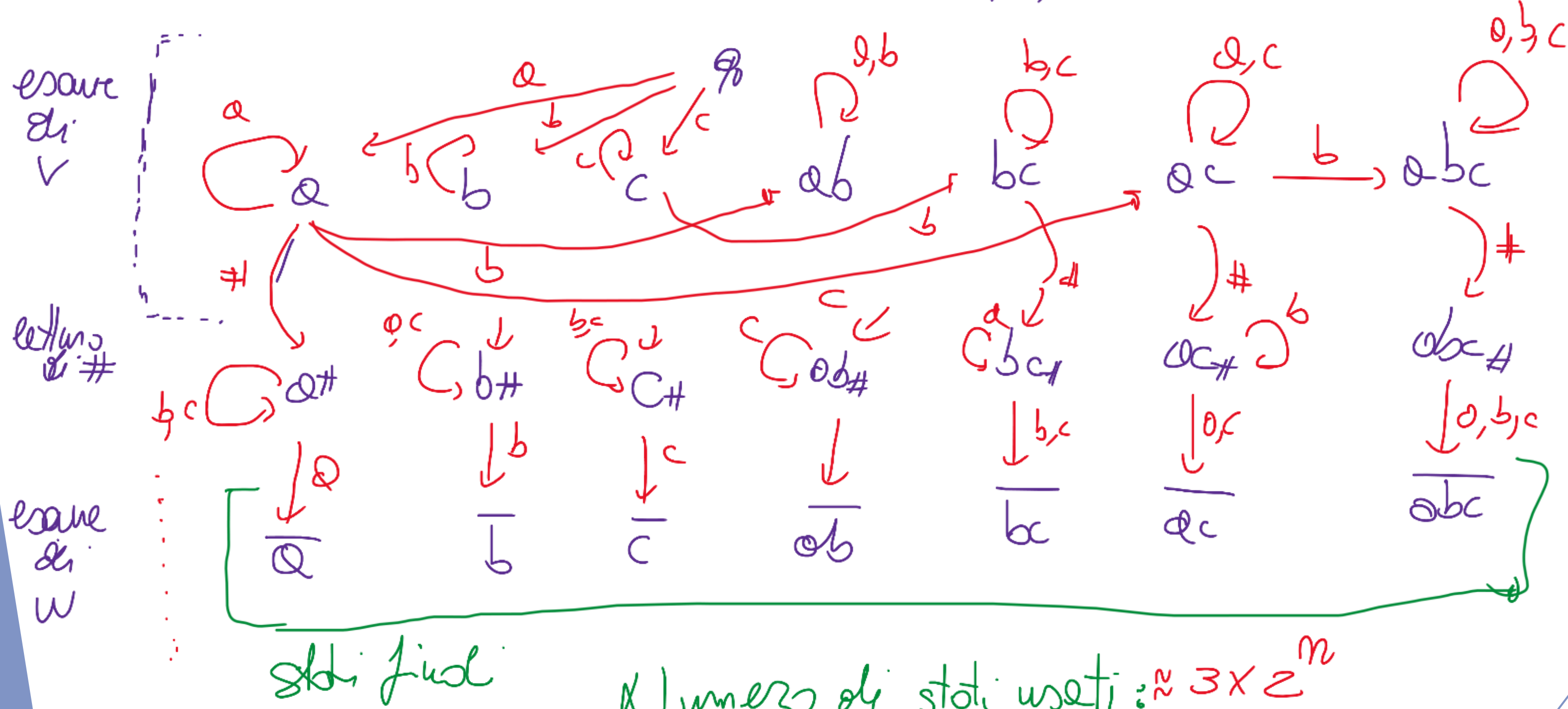
Automi regolari non-deterministici

Esempio di succintezza.

Si consideri il linguaggio L su $\Sigma = \{a_1, \dots, a_n, \#\}$

$$L = \{v \cdot \# \cdot w : v, w \in H^*,\ esiste\ \ h \in \{a_1,\dots,a_n\}\ h\ occorre\ in\ v\ e\ w\}$$

Automa deterministico: (usiamo a,b,c,#



Numero di stati usati: $\approx 3 \times 2^n$

Esempio

Esempio di succintezza.

Si consideri il linguaggio L su $\Sigma = \{a_1, \dots, a_n, \#\}$
$L = \{v \cdot \# \cdot w : v, w \epsilon H^*, \, esiste \;\; h \, \epsilon \{a_1, \dots, an\} \, h \, occorre \; in \; v \; e \; w\}$

Automa non-deterministico:

Si inizia la computazione scegliendo nondeterministicamente il simbolo h (scommessa)

Si verifica la scommessa sia corretta, vale a dire che h occorra sia prima di # sia dopo # (accettazione)

Si scommette per ogni simbolo in $\{a_1, \dots, a_n\}$ quindi almeno una delle scommesse ha successo (e basta che solo una abbia successo).

Per l'automa basta un numero di 2(n+1) stati

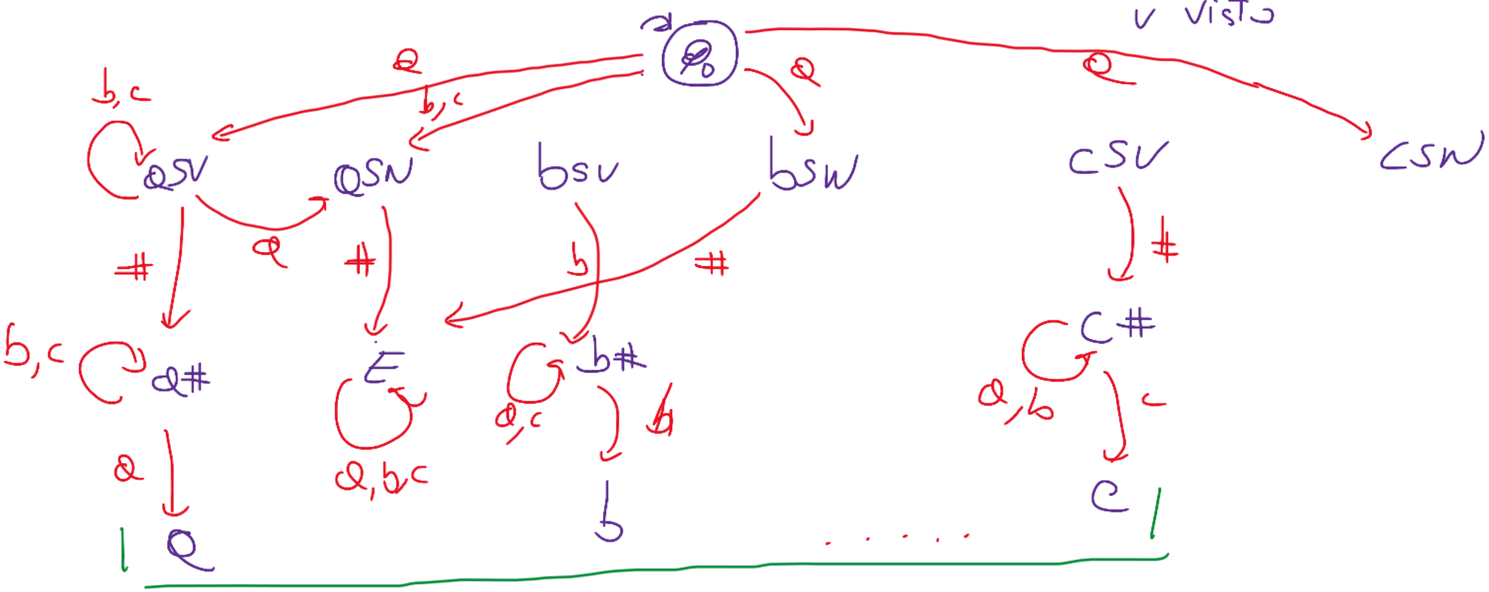
Automi regolari non-deterministici

Esempio di succintezza.

Si consideri il linguaggio L su $\Sigma = \{a_1, \dots, an, \#\}$
$\boldsymbol{L = \{v \cdot \# \cdot w : v, w \in H^*, \text{ esiste } h \in \{a_1, \dots, an\}, h \text{ occorre in } v \text{ e } w\}}$

Automa non-deterministico:

S sono l'ingresso
V visto



Stati finale

NUMERO STATI : ≈ 4×n

Esercizio

Si scriva l'automa non deterministico che riconosce il linguaggio usando un numero di stati polinomiale nella cardinalità dell'alfabeto.

Si consideri il linguaggio L su $\Sigma = \{a_1, \ldots, a_n, \#\}$

$$L = \{v \cdot \# \cdot w : v, w \in H^*, v\ usa\ l'insieme\ di\ simboli\ \mathrm{H} \subseteq \{a_1, \ldots, a_n\}\ e\ w\ usa\ l'insieme\ \mathrm{H}' \subseteq \{a_1, \ldots, a_n\}\ con\ \mathrm{H} \neq H'\}$$

L'Automa deterministico richiede un numero esponenziale di stati.

Idea:

- Si scelga in modo non-deterministico il simbolo a_i tale per cui $\mathrm{H} \neq H'$
- Si verifichi la correttezza della scelta.

Espressività del non-determinismo

- Gli automi regolari non-deterministici sono più espressivi degli automi regolari deterministici?
- (espressività = capacità di riconoscere linguaggi)
- Esistono linguaggi riconosciuti da automi non-deterministici che non siano riconosciuti da automi deterministici?

No!

Teorema. Per ogni automa non-deterministico A esiste un automa deterministico DA tale che L(A)=L(DA).

- La differenza tra determinismo e non-determinismo negli automi regolari non riguarda l'espressività ma la succintezza della rappresentazione.

Determinizzazione di un automa

- Determinizzazione di un automa A: costruzione dell'automa deterministico DA equivalente ad A (L(A)=L(DA))
- Automa non-deterministico A=$<Q,\Sigma,\delta,q_0,F>$
- Costruzione dell'automa deterministico DA=$<Q',\Sigma,\delta',q'_0,F'>$
- Da uno stato posso transire in un insieme di stati mediante un simbolo dell'alfabeto
- Sia $\delta(q,a) = \{q' : < q, a, q' > \epsilon\ \delta\}$ per $q, q' \epsilon Q\ e\ a \epsilon \Sigma$

  Insieme di stati raggiunti dallo stato q mediante il simbolo a
- Lo stato dell'automa deterministico è l'insieme di stati che possono essere raggiunti mediante una parola
- Uno stato di DA è un insieme di stati di A:
- $Q' = 2^Q$ $(insieme\ delle\ parti\ di\ Q)$

Determinizzazione di un automa

- Costruzione DA=<Q',Σ,δ',q'$_0$,F'>
- Stato iniziale

  $q'_0 = \{q_0\}$ singoletto con lo stato iniziale di A
- Stati finali

$F' = \{H \epsilon Q' : H \cap F \neq \emptyset\}$ stati che contengono almeno uno stato finale di A.

- Relazione di transizione

$F' = \{< H, a, K >: H \epsilon\ Q', K = \bigcup_{q \in H} \delta(q, a)\}$ da H si raggiunge l'insieme di stati non-deterministicamente raggiungibili da qualche stato in H.

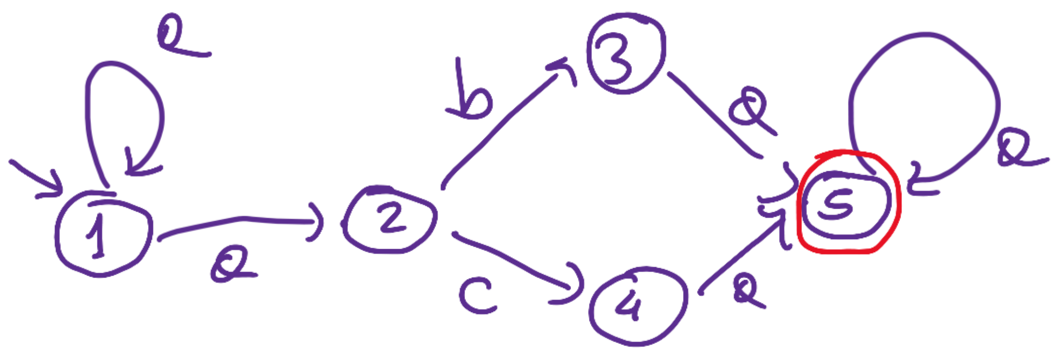
Si può provare che L(A)=L(DA)

- Per ogni computazione accettante di A per w esiste una computazione accettante di DA per w e viceversa.

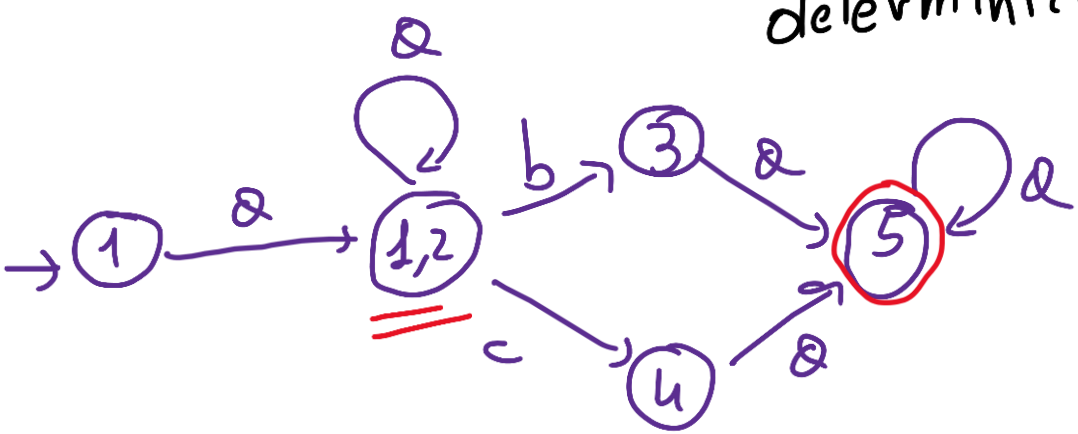
Esempio determinizzazione

Determinizzazione

automa non-deterministico



determinizzazione



in questo caso non c'è esplosione esponenziale

Proprietà di chiusura per automi regolari

- **Gli automi regolari riconoscono linguaggi (insiemi di parole)**
- **Come si comportano rispetto alle operazioni insiemistiche (note anche come chiusure booleane) complemento, unione e intersezione?**

Complementazione: caso deterministico

- **Dato un un linguaggio L riconosciuto da un automa regolare deterministico A, esiste un automa deterministico CA che riconosce il complemento di L?**

Il complemento di un linguaggio L su Σ^* è il linguaggio Σ^*/L .

Teorema. Gli automi deterministici sono chiusi per complementazione.

Nel caso deterministico la complementazione è semplice basta complementare l'insieme degli stati finali

Prova.

- **Dato un automa deterministico $\mathrm{D}A =< Q, \Sigma, q_0, \delta, F >$ l'automa deterministico $L(CDA) = < Q, \Sigma, q_0, \delta, Q/F >$ è tale che $L(CDA) = \Sigma^* / L(DA)$**
- La complementazione degli automi deterministici è semplice.
- Attenzione! Per poter applicare la complementazione l'automa di partenza deve essere completo (da ogni stato deve partire un arco per ogni dimbolo dell'alfatbeto).
- Se l'automa non è completo va prima completato e poi complementato.

Complementazione: caso non-deterministico

- Dato un un linguaggio L riconosciuto da un automa regolare non-deterministico A, esiste un automa non-deterministico CA che riconosce il complemento di L?

Teorema. Gli automi non-deterministici sono chiusi per complementazione.

Nel caso non-deterministico la complementazione è più complessa.

Non è possibile sfruttare la complementazione degli stati finali!

Prova.

- **Determinizzazione.** Dato un automa non-deterministico $A =< Q, \Sigma, q_0, \delta, F >$ è possible trovare un automa deterministico DA tale che L(A)=L(DA).
- Complementazione. DA è deterministico e dunque esiste un automa CDA tale che L(CDA) è il complemento di L(DA)=L(A).
- Nel caso non-determimnistico la complemetazione può richiedere una esplosione esponenziale nella dimensione iniziale dell'automa!

Intersezione

- **Dati due linguaggi L_1 e L_2 riconosciuti da due automi regolari (deterministici/non-deterministici) A_1 e A_2, esiste un automa (deterministico/non-deterministico) A che riconosce il linguaggio $L_1 \cap L_2$?**
- **L'alfabeto dei due linguaggi deve essere lo stesso!**

Teorema. Gli automi regolari sono chiusi per intersezione.

Idea. Possiamo eseguire I due automi A_1 e A_2 sincronamente (per ogni transizione dell'uno ci deve essere una simultanea transizione dell'altro)

Intersezione

- Dati due linguaggi L_1 e L_2 riconosciuti da due automi regolari (deterministici/non-deterministici) A_1 e A_2, esiste un automa (deterministico/non-deterministico) A che riconosce il linguaggio $L_1 \cap L_2$?
- L'alfabeto dei due linguaggi deve essere lo stesso!

Teorema. Gli automi regolari sono chiusi per intersezione.

Esempio.

L_1 il linguaggio su $\Sigma = \{a, b\}$ con parole con un numero pari di a.

L_2 il linguaggio su $\Sigma = \{a, b\}$ con parole con un numero pari di b.

$L_1 \cap L_2$ il linguaggio su $\Sigma = \{a, b\}$ con parole con un numero pari di a e b.

Intersezione

Esempio.

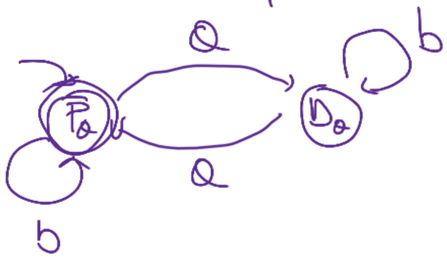
L_1 il linguaggio su $\Sigma = \{a, b\}$ con parole con un numero pari di a.

L_2 il linguaggio su $\Sigma = \{a, b\}$ con parole con un numero pari di b.

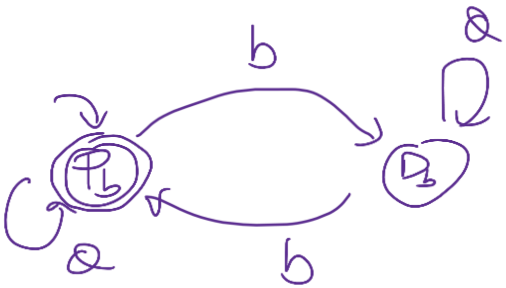
$L_1 \cap L_2$ il linguaggio su $\Sigma = \{a, b\}$ con parole con un numero pari di a e b.

Intersezione

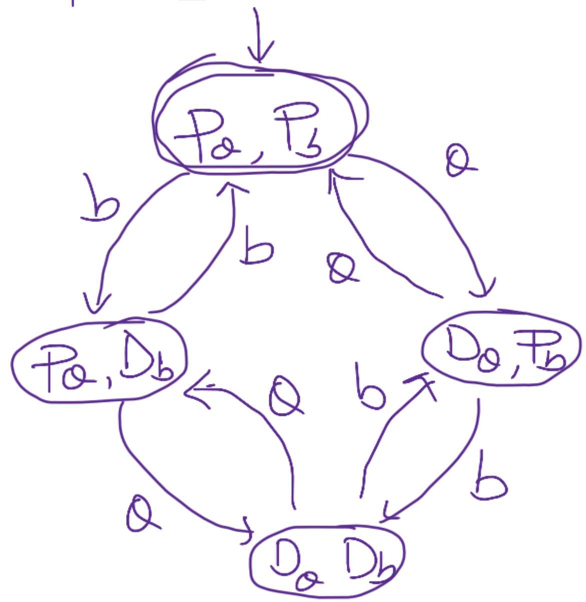
Numero pari di a



Numero pari di b



Intersezione
pari a e pari b



Intersezione: costruzione generale

$$A_1 =< Q_1, \Sigma, q_{01}, \delta_1, F_1 > A_2 =< Q_2, \Sigma, q_{02}, \delta_2, F_2 >$$

$$A \quad =< Q \quad , \Sigma, q_0, \delta \quad , F \quad >,$$

- $Q = Q_1 \times Q_2$

  Coppie di stati che evolvono sincronamente

- $q_0 = (q_{01}, q_{02})$

  Coppia di stati iniziali

- $\delta = \{((q,p), a, (q', p')) : (q, a, q') \in \delta_1, (p, a, p') \in \delta_2 , a \in \Sigma\}$

  Coppia di transizioni sincrone

- $F = F_1 \times F_2$

  Coppia di stati finali

$$L(A) = L(A_1) \cap L(A_2)$$

Unione

- Dati due linguaggi L_1 e L_2 riconosciuti da due automi regolari (non-deterministici) A_1 e A_2, esiste un automa (non-deterministico) A che riconosce il linguaggio $L_1 \cup L_2$?
- L'alfabeto dei due linguaggi deve essere lo stesso!

Teorema. Gli automi regolari (deterministici/non-deterministici) sono chiusi per unione.

Idea.

- Prendiamo i due automi A_1 e A_2 garantendo che abbiano due insiemi disgiunti di stati.
- Aggiungiamo uno stato iniziale che direziona in modo potenzialmente non deterministico la computazione verso l'automa A_1 o l'automa A_2

Unione: costruzione generale

$$A_1 =< Q_1, \Sigma, q_{01}, \delta_1, F_1 > \; A_2 =< Q_2, \Sigma, q_{02}, \delta_2, F_2 >$$

$$A \quad =< Q \quad, \Sigma, q_0, \delta \quad, F \quad >,$$

- $Q = \{1\}xQ_1 \cup \{2\}xQ_2 \cup \{q_0\}$

  Unione disgiunta dei due insiemi di stati

- $\delta = \{\big((i,p), a, (i,p')\big) : (p,a,p') \in \delta_i, a \in \Sigma, i \in \{1,2\}\} \cup$
  $\{\big(q_0, a, (i,p')\big) : (q_{0i}, a, p') \in \delta_i, a \in \Sigma, i \in \{1,2\}\}.$

  Unione disgiunta delle transizioni

- $F = \{1\}xF_1 \cup \{2\}xF_2 \cup \{q_0: se\; q_{01} \in F_1 o\; q_{02} \in F_2\}$

  Unione disgiunta degli stati finali

$$L(A) = L(A_1) \cup L(A_2)$$

Concatenazione

- Dati due linguaggi L_1 e L_2 riconosciuti da due automi regolari (non-deterministici) A_1 e A_2, esiste un automa (non-deterministico) A che riconosce il linguaggio $L_1 \cdot L_2$?
- **Definizione**: $L_1 \cdot L_2 = \{w \cdot v : w \epsilon L_1 \text{ e } v \epsilon L_2\}$

Teorema. Gli automi regolari (deterministici/non-deterministici) sono chiusi per concatenazione.

Idea.

- **Prendiamo un automa che simula** A_1 finché non raggiunge uno stato di accettazione di A_1.
- Nello stato di accettazione di A_1 sceglie non-determisticamente se continuare a simulare A_1 o se simulare A_2

Concatenazione: costruzione generale

$$A_1 =< Q_1, \Sigma, q_{01}, \delta_1, F_1 > A_2 =< Q_2, \Sigma, q_{02}, \delta_2, F_2 >$$

$$A \quad =< Q \quad , \Sigma, q_0, \delta \quad , F \quad >,$$

- $Q = \{1\}\text{x}Q_1 \cup \{2\}\text{x}Q_2$

  Unione disgiunta dei due insiemi di stati

- $q_0 = (1, q_{01})$

  Stato iniziale (della prima componente)

- $\delta = \{((i,p), a, (i,p')) : (p,a,p') \in \delta_i, a \in \Sigma, i \in \{1,2\}\} \cup \{((1,q \quad), a, (2,p \quad)) : (q_{02}, a, p) \in \delta_2, a \in \Sigma, q \in F_1\}$

  Unione disgiunta delle transizioni

- $F = \{2\}\text{x}F_2 \cup \{(1, q_{01}) : se\ q_{01} \in F_1\ e\ q_{02} \in F_2\}$

  Unione disgiunta degli stati finali

$$L(A) = L(A_1) \cdot L(A_2)$$

Stella di Kleene

- Dato un linguaggio L riconosciuto da un automa regolare (non-deterministici) A, esiste un automa (non-deterministico) A che riconosce il linguaggio L^*?
- **Definizione**: L^*= $\{v_1 \cdot v_2 \cdots v_n : v_i \epsilon L$ per ogni 1 $\leq$i$\leq$ $n\} \cup \{\varepsilon\}$
- Concatenazione di un numero arbitrario di parole di L

Teorema. Gli automi regolari (deterministici/non-deterministici) sono chiusi per stella di Kleene.

Idea.

- Prendiamo un automa che simula A finché non raggiunge uno stato di accettazione di A_1.
- Nello stato di accettazione di A sceglie non-determinticamente se continuare a simulare A ripartendo dallo stato iniziale o se fermarsi

Stella di Kleene: costruzione generale

$$A_1 =< Q_1, \Sigma, q_{01}, \delta_1, F_1 >$$

$$A \quad =< Q_1, \Sigma, q_{01}, \delta \quad , F \quad >,$$

- $\delta = \delta_1 \cup \{(q, a, q_{01}) : (q, a, p) \in \delta_1, a \in \Sigma, p \in F_1\}$

  Transizione dagli stati finali all'iniziale

- $F = F_1 \cup \{ q_{01}\}$

  Lo stato iniziale è accettante per includere la parola vuota

$$L(A) = L(A_1)^*$$

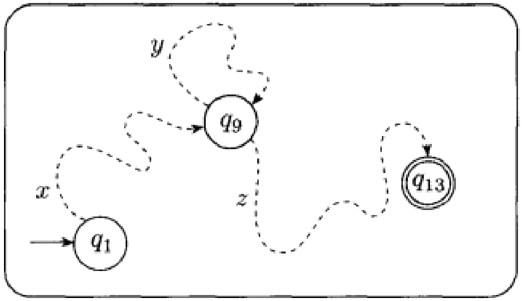
Linguaggi non regolari.

- **Per accertare che un linguaggio sia regolare basta costruire un automa regolare che lo riconosce.**
- **Come possiamo dimostrare che un linguaggio NON è regolare?**
- **Non riuscire a costruire un automa A tale che L(A)=L è un indizio che L non sia regolare ma non è una prova!**

E' possible sfruttare una proprietà dei linguaggi regolari espressa da un teorema noto con il nome di "Pumping Lemma"

Ad esempio:

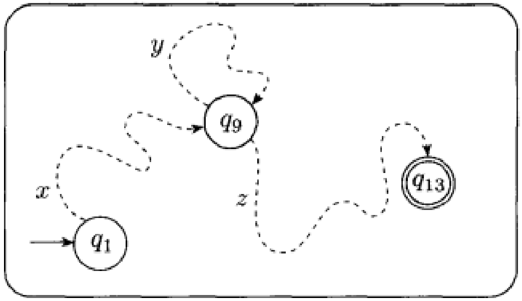
Pumping Lemma: idea.

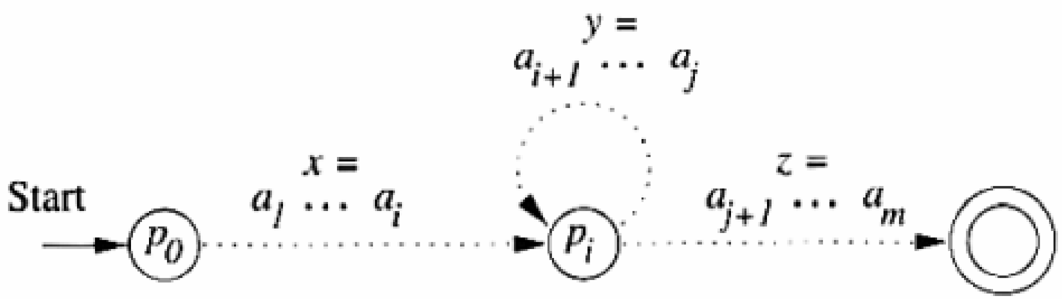


- Supponiamo esista un cammino dallo stato iniziale ad uno stato finale che coinvolge un ciclo.
- Osservazione. Se un cammino è più lungo del numero degli stati dell'automa certamente attraversa e ripete un ciclo!
- Il ciclo può essere percorso un numero arbitrario di volte prima di deviare allo stato di accettazione: la parola è accettata indipendentemente dal numero di iterazioni!
- Conseguenza: Il linguaggio include parole di lunghezza arbitraria (basta aumentare il numero di cicli –pumping)

Pumping Lemma: definizione

- Teorema. Sia L un linguaggio regolare. Esiste un numero p tale che per ogni parola $w \in L$ con $|w| \geq p$ (p è detto periodo di pumping) la parola w può essere divisa in tre parti
- $w = x \cdot y \cdot z$
- e vale che
- $|y|>0$
- $|x \cdot y| \leq p$
- Per ogni $i \geq 1$, $x \cdot y^i \cdot z \in L$





Linguaggi non regolari: applicazione del pumping lemma

- **Per accertare che un linguaggio L NON è regolare si sfrutta il punping lemma.**

1. **Si assume per ipotesi che L sia regolar**
2. **Si applica il pumping lemma per derivare una contraddizione.**

Ad esempio sia L = $\{a^n \cdot b^n : n \geq 0\}$

1. **Assumiamo che L sia regolare.**
2. **Allora esiste un period di pumping p per il lingiaggio.**
3. **Prendiamo la parola** $w = a^p \cdot b^p$
4. **Possiamo frammentare la parola** $w = x \cdot y \cdot z$ **con** $|x \cdot y| \leq p$
5. **se** $|x \cdot y| \leq p$ **allora** $x \cdot y = a^k \cdot a^j$ **per qualche** $1 \leq k + j \leq p$
6. **Per il lemma** $a^k \cdot a^{2j} \cdot a^{k+j-p} \cdot b^n$ **è una parola di L**
7. **Una contraddizione! Infatti** $k + 2j + k + j - p \neq p$

Linguaggi non regolari: applicazione del pumping lemma

Provare che i seguenti linguaggi non sono regolari

1. **L il linguaggio su $\Sigma = \{a, b\}$ dove le parole hanno un ugual numero di occorrenze di a e di b**
2. **L il linguaggio su $\Sigma = \{a\}$ dove le parole hanno lunghezza pari a una potenza di due**
3. **L il linguaggio su Σ delle parole speculari: $L = \{w \cdot w^R : w \epsilon \Sigma^*\}$**

Automi regolari con transizioni interne

- **In un automa regolare una transizione è legata ad un simbolo dell'alfabeto**
- **Una transizione dell'automa non può avvenire in assenza di uno stimolo esterno che controlla l'evoluzione dell'automa.**
- **Non è possibile che l'automa faccia transizioni di stato autonome (elaborazioni interne non visibili dall'ambiente esterno)**
- **Idea:**
- **Permettere speciali transizioni non controllate dall'alfabeto**
- **Si una uno speciale simbolo ϵ dell'alfabeto che rappresenta il simbolo neutro (azione interna).**

Automi regolari con transizioni interne

- **In un automa regolare una transizione è legata ad un simbolo dell'alfabeto**
- **Una transizione dell'automa non può avvenire in assenza di uno stimolo esterno che controlla l'evoluzione dell'automa.**
- Non è possibile che l'automa faccia transizioni di stato autonome (elaborazioni interne non visibili dall'ambiente esterno)
- **Idea:**
- Permettere speciali transizioni non controllate dall'alfabeto
- Si una uno speciale simbolo ϵ dell'alfabeto che rappresenta il simbolo neutro (azione interna).

$q \xrightarrow{\varepsilon} q'$

Esempio: controllore dell‘ascensore con transizioni interne che modellano il cambio del piano

Automi regolari con transizioni interne: sintassi e semantica

Piccola variante della sintassi standard

$<Q,\Sigma,\delta,q_0,F>$

- Q è un insieme **finito** di stati.
- Σ è un **alfabeto,** insieme di simboli di etichette per gli archi delle transizioni
- δ è la **funzione di transizione** $\delta : Q \times \Sigma \cup \{\varepsilon\} \rightarrow Q$ che, fissato uno stato (sorgente) e un simbolo dell'alfabeto, determina lo stato successivo (destinazione).
- q_0 è lo **stato iniziale.**
- $F \subseteq Q$ è l'insieme degli **stati finali.**

$q \xrightarrow{\varepsilon} q'$

Automi regolari con transizioni interne: semantica

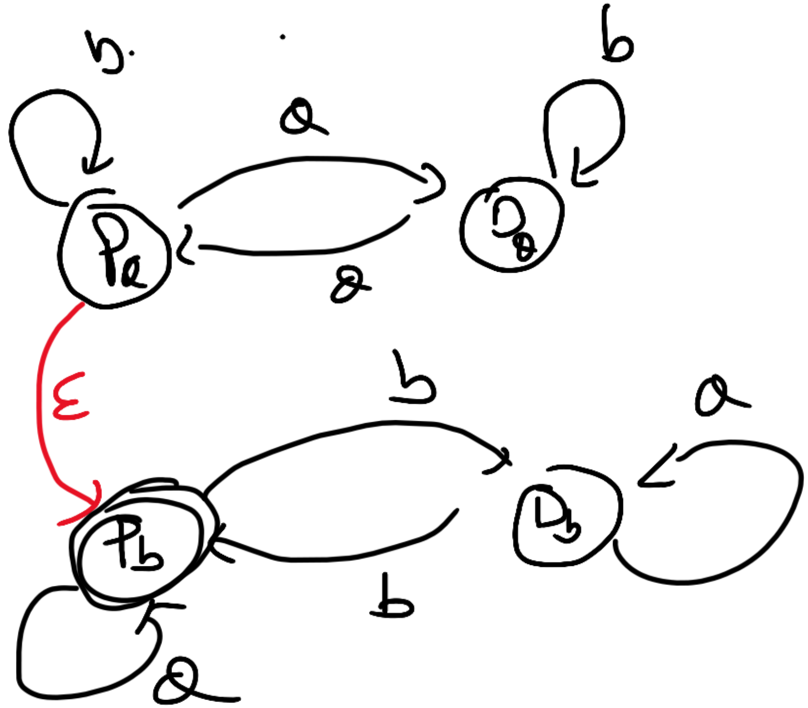
Descrizione di una computazione di un automa regolare $<Q,\Sigma,\delta,q_0,F>$ con transizioni interne

- Intuitivamente è una sequenza di transizioni dallo stato iniziale.
- Una computazione è una sequenza alternata di stati e simboli dell'alfabeto $\Sigma \cup \{\varepsilon\}$

$$q_1\ \sigma_1\ ,\ q_2,\ \sigma_2\ ,\dots,\ q_n,\ \sigma_n\ q_{n+1}$$

- q_1 è lo **stato iniziale.**
- $q_i \in Q$ per ogni i, $1 \leq i \leq n+1$
- $(q_i, \sigma_i, q_{i+1}) \in \delta$ per ogni i, $0 \leq i \leq n$ con $\sigma_i \in \Sigma \cup \{\varepsilon\}$
- La **computazione è accettante** se $q_{n+1} \in F$.
- **La parola accettata dalla computazione è $w \in \Sigma^*$** dove w è la parola ottenuta da $\sigma_1 \cdot \sigma_2 \cdots \sigma_n$ rimuovendo tutte le occorrenze di ε

Esempio:



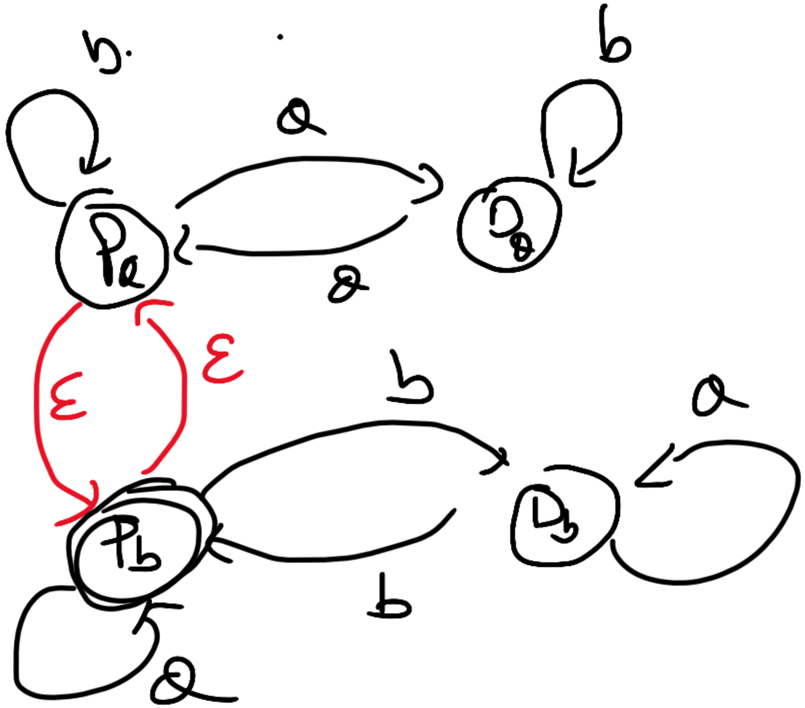
Che linguaggio accetta?

Computazione:

$P_a \; b \; P_a \; a \; D_a \; b \; D_a \; a \; P_a \; \varepsilon \; P_b \; a \; P_b \; b \; D_b \; b \; P_b$

Parola accettata $b\,a\,b\,a \,|\, a\,b\,b$

Esempio:



$P_a \; \varepsilon \; P_b \; \varepsilon \; P_a \; \varepsilon \; P_b \; \varepsilon \; \ldots\ldots$

Automi regolari con transizioni interne

- **L'aggiunta di transizioni interne non aumenta l'espressività degli automi regolari**
- **Teorema. Per ogni automa regolare con transizioni interne A esite un automa senza transizioni interne A' equivalente, L(A)=L(A').**

Idea della costruzione.

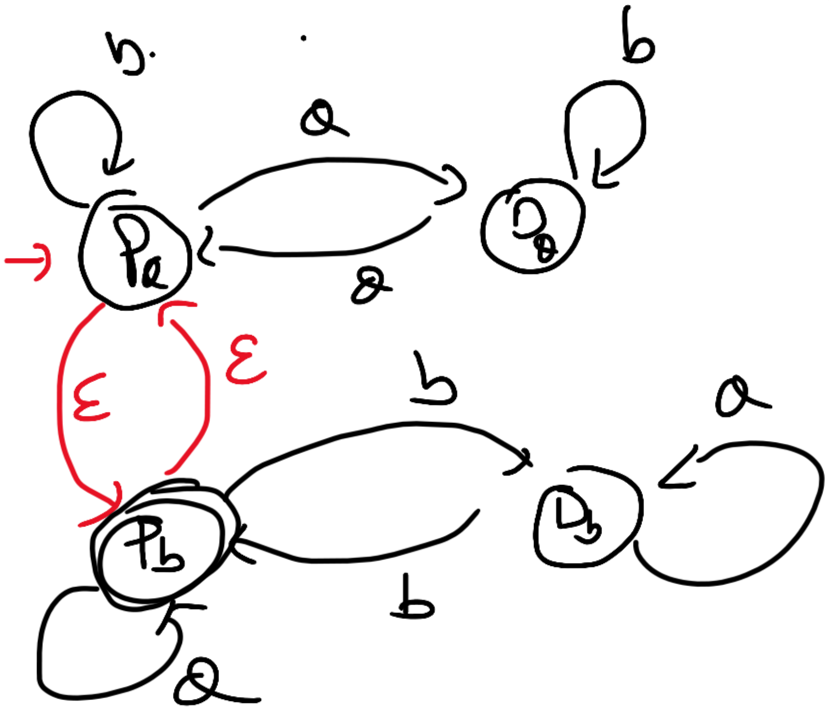
- **A'= <Q,Σ,δ',q$_0$,F'> ha gli stessi stati di A= <Q,Σ,δ,q$_0$,F>**
- **C'e' una transizione $(q, a, q') \in \ \delta'$ $con\ a\epsilon\Sigma\ e\ q, q'\epsilon Q\ se$ Esiste una sequenza di transizioni in δ della forma**

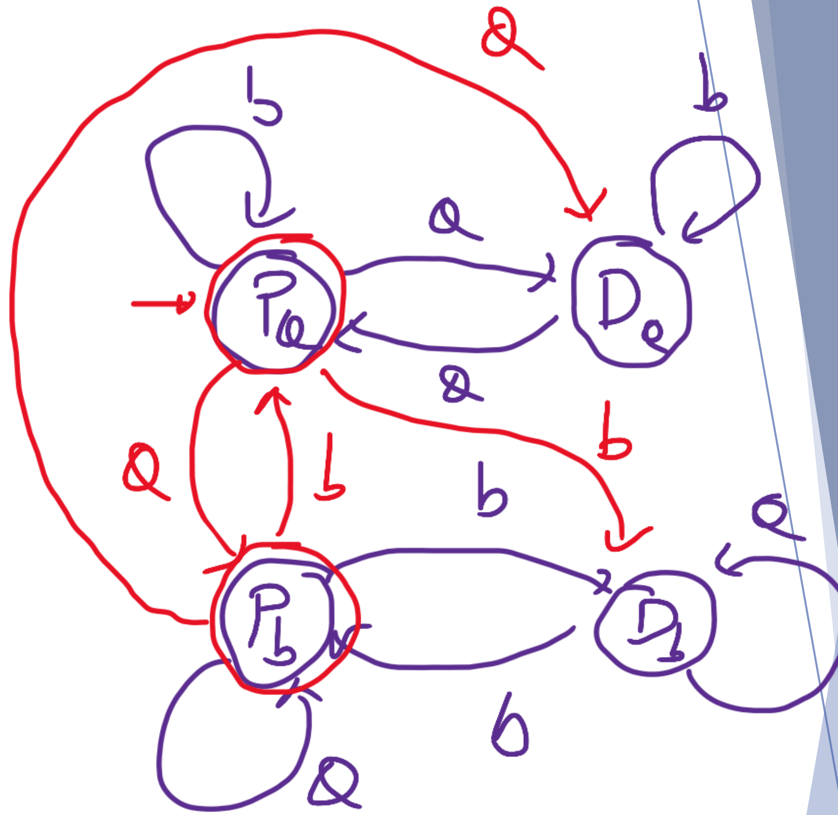
$$q \xrightarrow{\varepsilon} q_1 \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} q_m \xrightarrow{a} q'$$

(caso particolare $q \xrightarrow{a} q'$)

- **Gli stati finali sono quelli raggiungibili da stati finali di A tramite transizioni interne**

Esempio:

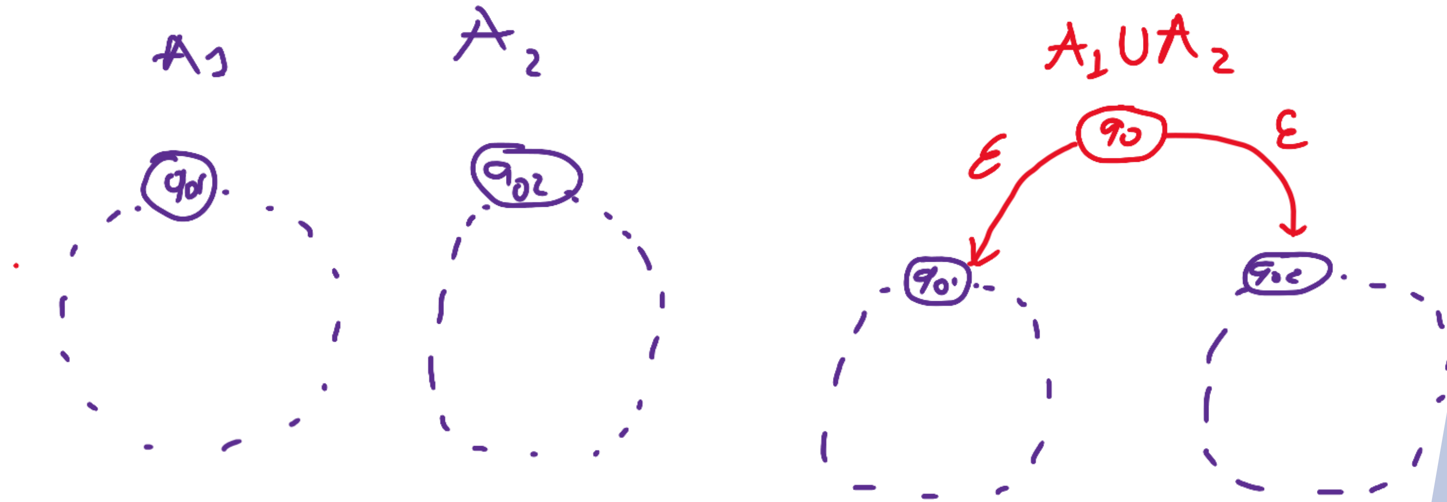




P_a e P_b sono stati finali

Automi regolari con transizioni interne

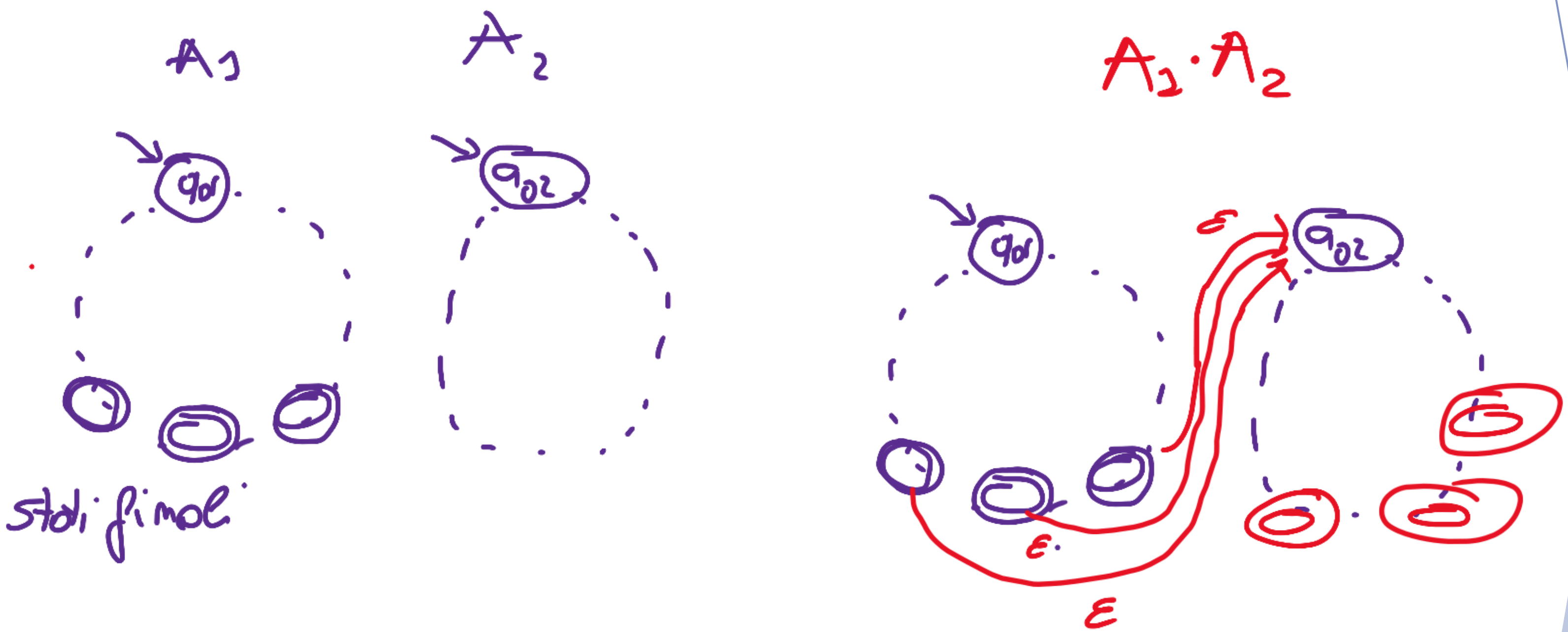
- **L'aggiunta di transizioni interne non aumenta l'espressività degli automi regolari**
- **Le transizioni interne permettono tuttavia descrizioni più agevoli.**
- **Esempio:** Automa per l'unione di due automi A_1 e A_2



Automi regolari con transizioni interne

Le transizioni interne permettono tuttavia descrizioni più agevoli.

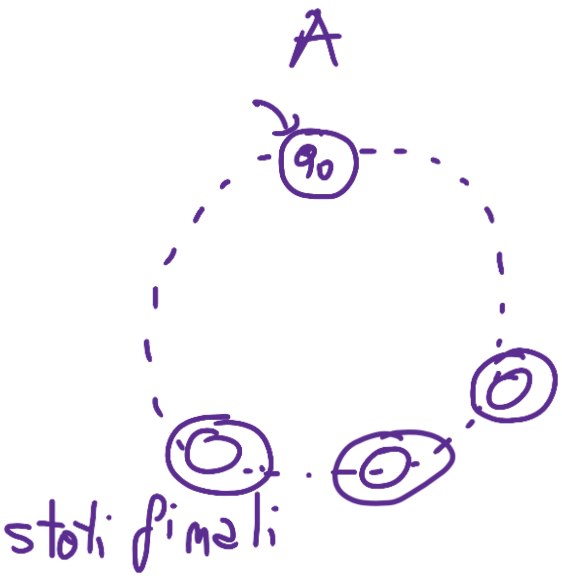
- **Esempio:** Automa per la concatenazione di due automi A_1 e A_2

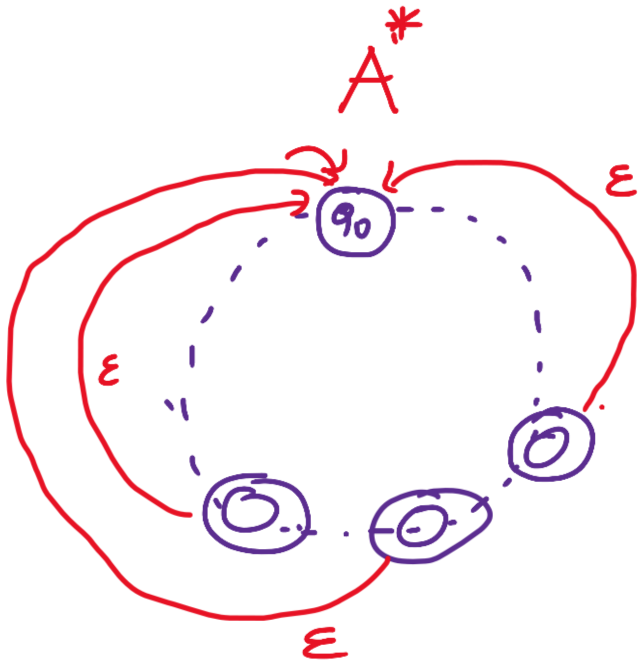


Automi regolari con transizioni interne

Le transizioni interne permettono tuttavia descrizioni più agevoli.

- **Esempio:** Automa per la stella di Kleene di un automa



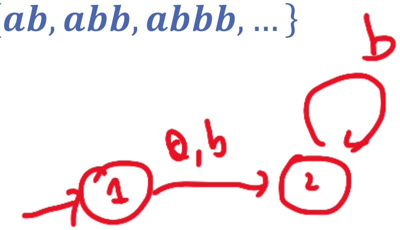


Espressioni regolari: espressioni regolari

- Le espressioni regolari forniscono un modo alternativo per descrivere i linguaggi regolari.
- Idea.
- Il linguaggio viene definito a partire da linguaggi regolari elementari
- I linguaggi vengono trasformati mediante operazioni che preservano la regolarità del linguaggio
- Unione, intersezione, complemento, concatenazione, * di Kleene sono esempi di operazioni che trasformano linguaggi regolari in linguaggi regolari.
- Anziché definire il linguaggio tramite un automa che lo riconosce lo si definisce tramite l'espressione.

Esempi di espressioni regolari

- Si consideri un alfabeto $\Sigma = \{\boldsymbol{a}, \boldsymbol{b}\}$
- $(\boldsymbol{a} \cup \boldsymbol{b}) \cdot \boldsymbol{b}^*$
  - La valutazione di una espressione è un linguaggio
  - $\boldsymbol{a}$ è **il linguaggio** $\{\boldsymbol{a}\}$
  - $\boldsymbol{b}$ è **il linguaggio** $\{\boldsymbol{b}\}$
  - $(\boldsymbol{a} \cup \boldsymbol{b})$ è **il linguaggio** $\{\boldsymbol{a}, \boldsymbol{b}\}$
  - $\boldsymbol{b}^*$ è **il linguaggio** $\{\varepsilon, \boldsymbol{b}, \boldsymbol{bb}, \boldsymbol{bbb}, \ldots\}$
  - Il linguaggio definito è $\{\boldsymbol{b}, \boldsymbol{bb}, \boldsymbol{bbb}, \ldots\} \cup \{\boldsymbol{ab}, \boldsymbol{abb}, \boldsymbol{abbb}, \ldots\}$
  - Stesso linguaggio definito dall'automa



Sintassi delle espressioni regolari

- Si consideri un alfabeto Σ
- $\boldsymbol{\varepsilon}$ è una espressione regolare
- $\boldsymbol{a}$ è una espressione regolare per ogni $\boldsymbol{a} \in \boldsymbol{\Sigma}$
- $\boldsymbol{\phi}$ è una espressione regolare
- Se $\boldsymbol{R}$ $\boldsymbol{e}$ $\boldsymbol{R'}$ sono espressioni regolari lo sono anche
  - $\boldsymbol{R} \cup \boldsymbol{R'}$
  - $\boldsymbol{R} \cdot \boldsymbol{R'}$
  - $\boldsymbol{R^*}$
- L(R) denota il linguaggio ottenuto valutando l'espressione R

Esempi di espressioni regolari

- $\mathbf{0}^* \cdot \mathbf{1} \cdot \mathbf{0}^* =$
- $\mathbf{\Sigma}^* \cdot \mathbf{1} \cdot \mathbf{\Sigma}^* =$
- $\mathbf{\Sigma}^* \cdot \mathbf{0} \cdot \mathbf{0} \cdot \mathbf{1} \cdot \mathbf{\Sigma}^* =$
- $(\mathbf{0} \cdot \mathbf{1}^*)^* =$
- $(\mathbf{\Sigma} \cdot \mathbf{\Sigma})^* =$
- $(\mathbf{\Sigma} \cdot \mathbf{\Sigma} \cdot \mathbf{\Sigma})^* =$
- $(\boldsymbol{\phi})^* = \boldsymbol{\varepsilon}$
- $\mathbf{0}^* \cdot \boldsymbol{\phi} = \boldsymbol{\phi}$
- $(\mathbf{0} \cup \boldsymbol{\varepsilon}) \cdot (\mathbf{1} \cup \boldsymbol{\varepsilon}) =$
- $R \cdot R^* = ?$
- *L'espressione per l'insieme dei numeri decimali?*
- *L'espressione per l'insieme delle stringhe che rispettano il formato di un codice fiscale?*

Confronto tra espressioni regolari e automi regolari.

Teorema. Per ogni espressione regolare R esiste un automa regolare A tale che L(R) = L(A)

- **La prova è per induzione strutturale sulla espressione R**

Casi base.

- $\boldsymbol{R = \varepsilon, A=?}$
- $\boldsymbol{R = \phi, A=?}$
- $\boldsymbol{R = a, A=?}$

Casi induttivi. Supponiamo che A e A' siano gli automi che riconoscono I linguaggi di delle espressioni R e R' rispettivamente.

- $\boldsymbol{R \cup R',}$

$\boldsymbol{A \cup A'}$ è $\boldsymbol{un\ automa\ regolare\ e\ L(A \cup A') = L(A) \cup L(A') = L(R) \cup L(R') = L(R \cup R')}$

- $\boldsymbol{R \cdot R'}$

$\boldsymbol{A \cdot A'}$ è $\boldsymbol{un\ automa\ regolare\ e\ L(A \cdot A') = L(A) \cdot L(A') = L(R) \cdot L(R') = L(R \cdot R')}$

- $\boldsymbol{R^*}$

$\boldsymbol{A^*}$ è $\boldsymbol{un\ automa\ regolare\ e\ L(A^*) = L(A)^* = L(R^*)}$

Esempio

automa per $(a \cdot b \cup a)^*$

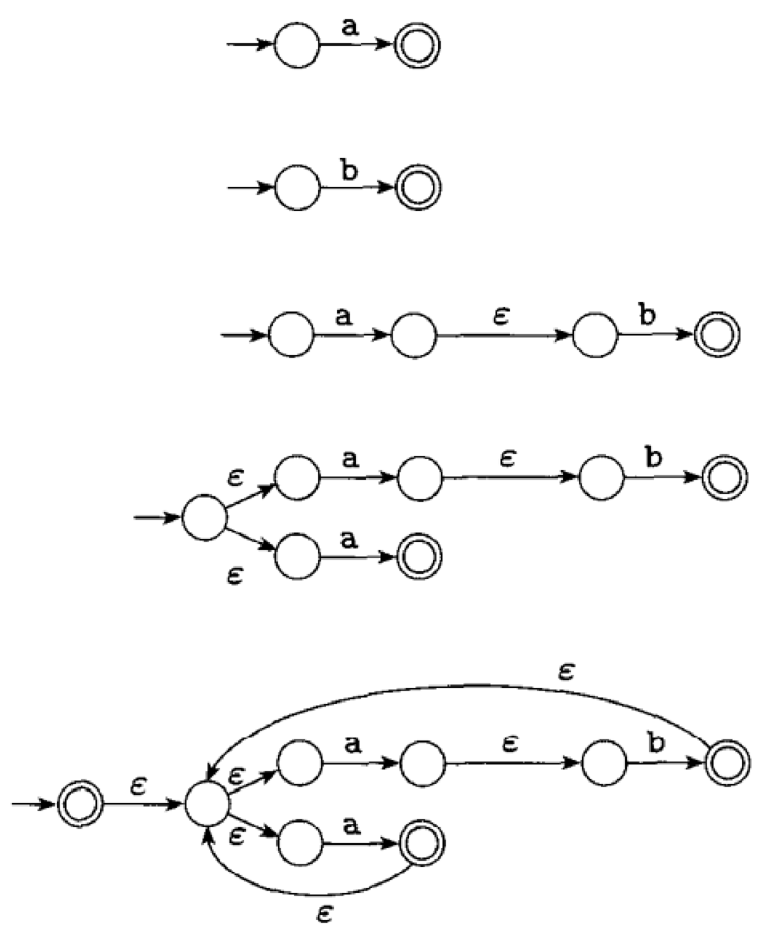
a

b

ab

ab ∪ a

(ab ∪ a)*



Confronto tra espressioni regolari e automi regolari.

Il terorema precedente permette di affermare che i linguaggi regolari includono i linguaggi definiti da espressioni regolari.

Vale anche l'inclusione simmetrica

Teorema. Per ogni automa regolare A esiste una espressione regolare R tale che L(R) = L(A)

- **Come corollario si ha che espressioni regolari e automi definiscono la stessa classe di liguaggi (sono ugualmente espressivi)**
- **La prova richiede l'uso di una variante degli automi regolari nondeterministici: gli automi non-determinsistici generalizzati**

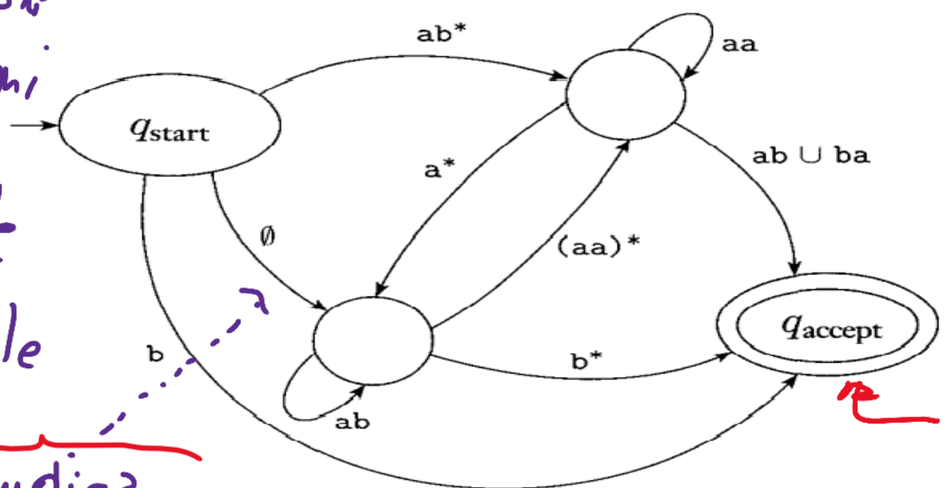
Automi non-deterministici generalizzati

Una transizione non viene attivata da un singolo simbolo di input ma da una sequenza di simboli

Le transizioni sono etichettate da espressioni regolari.

stato iniziale
solo transizioni
uscenti

stato iniziale
≠
stato finale



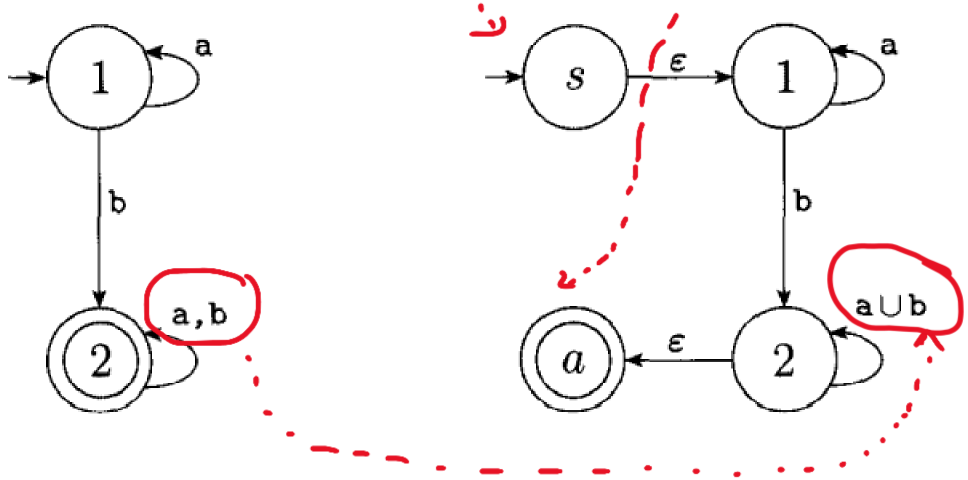
- Un solo stato finale
- solo transizioni entranti

∅ indica
arco non percorribile!

Da automi deterministici a automi non-deterministici generalizzati

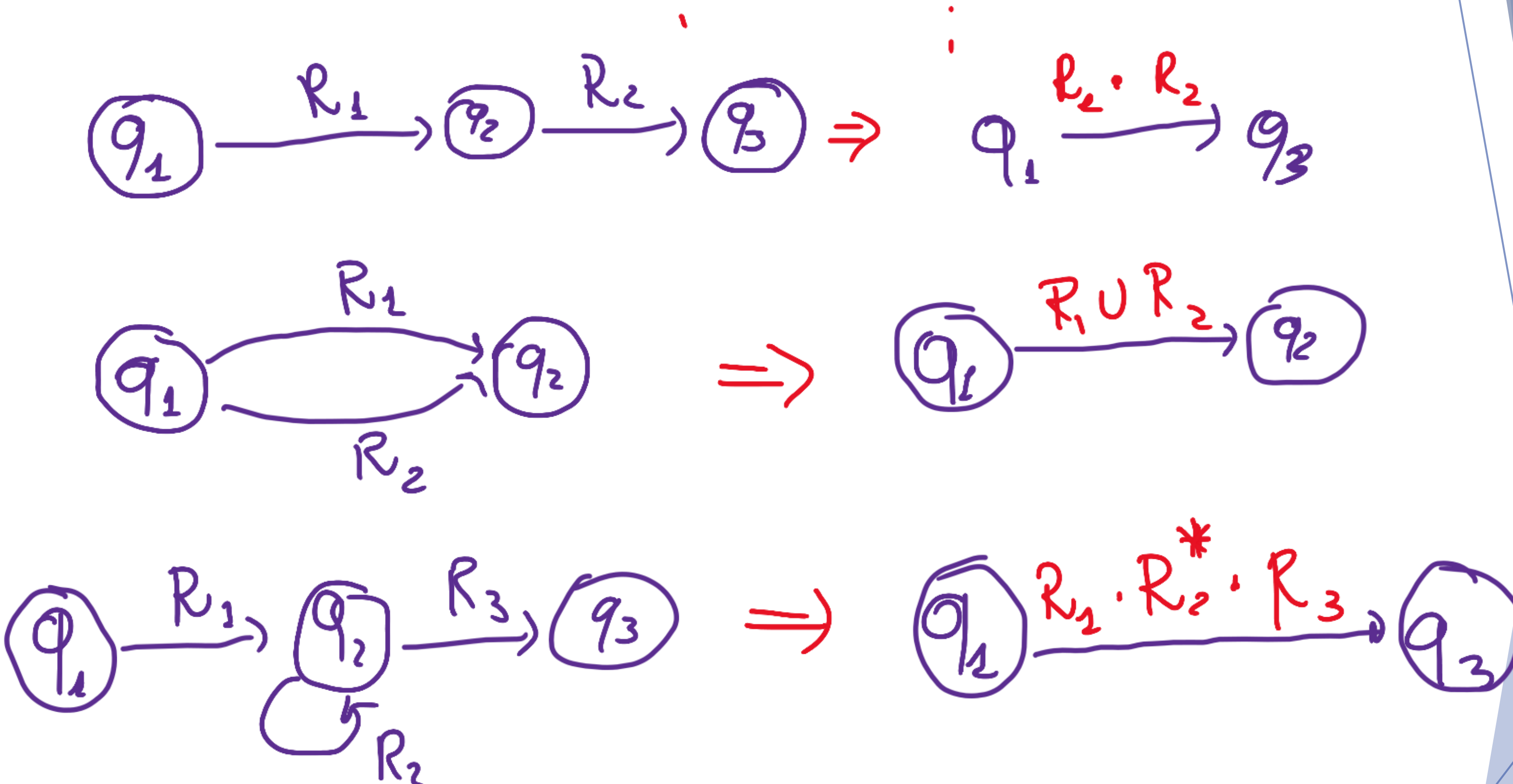
Si introduce un nuovo stato iniziale e uno finale.

automa deterministico



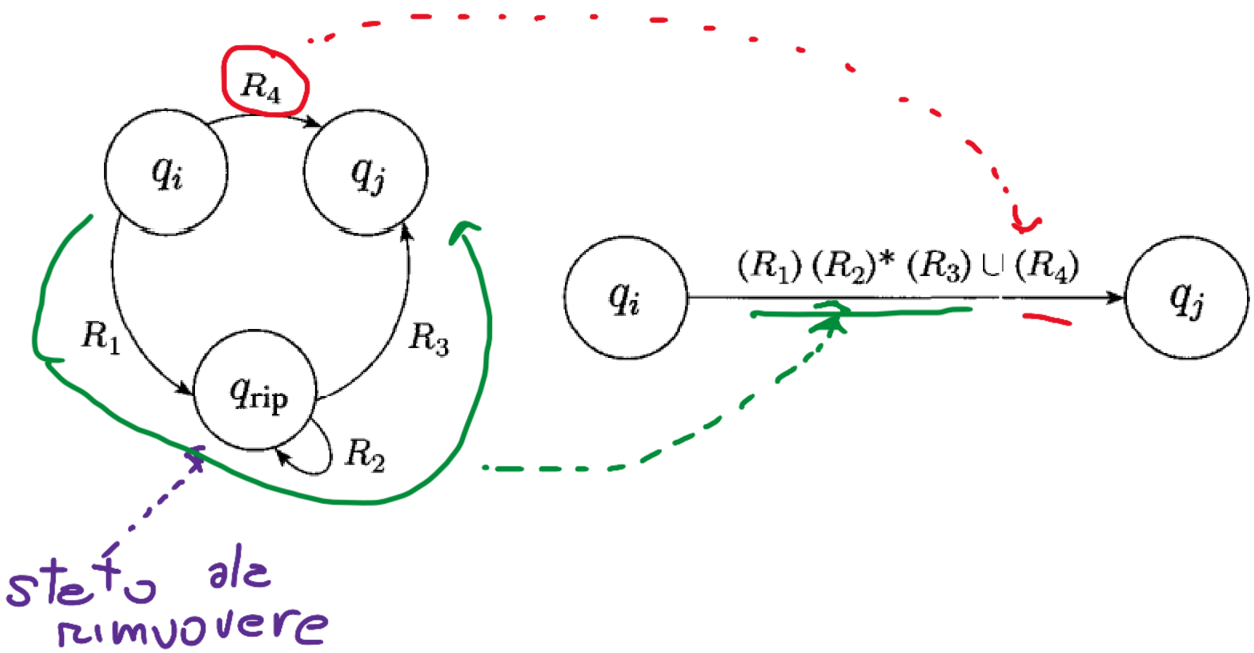
automi non-deterministici generalizzati: rimozione degli stati non iniziali o finali

Regole di trasformazione.



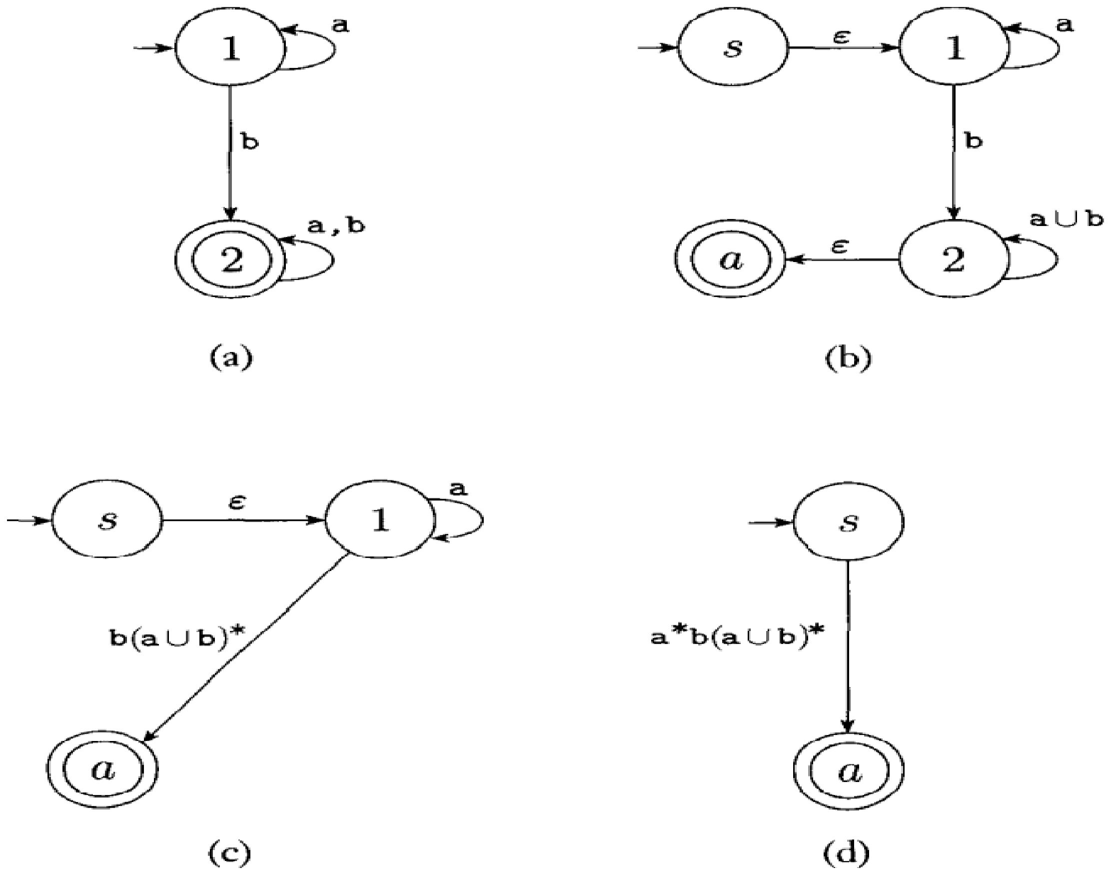
Automi non-deterministici generalizzati: rimozione degli stati non iniziali o finali

Uno alla volta vengono rimossi iterativamente gli stati non iniziale e non finale trasformando le etichette delle transizioni. Idea generale.

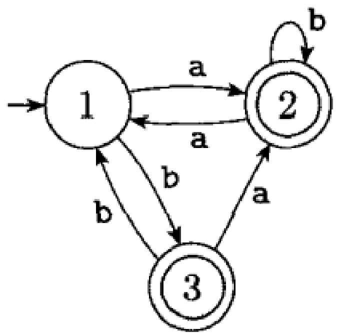


Da automa a espressione. Esempio

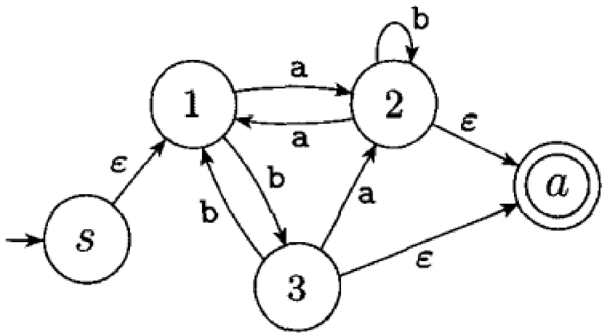
Uno alla volta vengono rimossi iterativamente gli stati non iniziale e non finale trasformando le etichette delle transizioni.



Da automa a espressione. Esempio

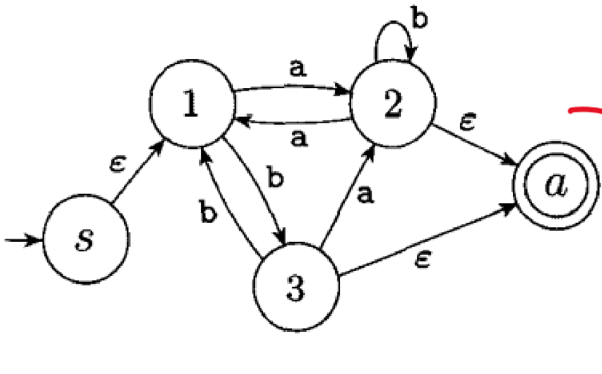


(a)

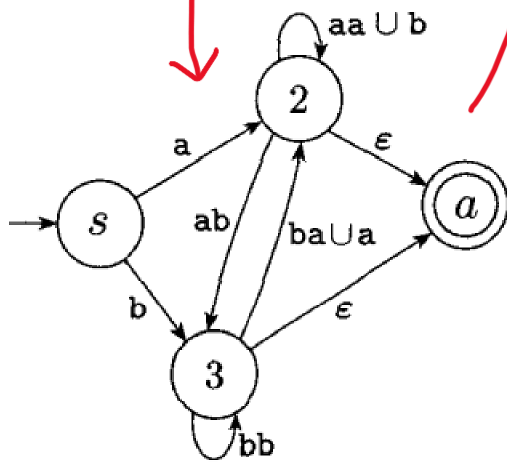


(b)

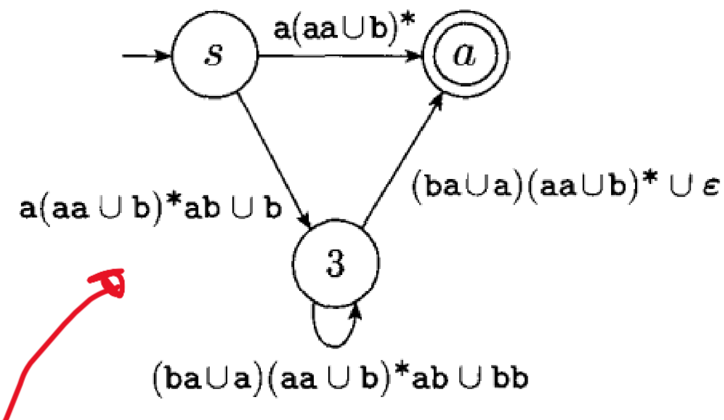
Da automa a espressione. Esempio



(b)

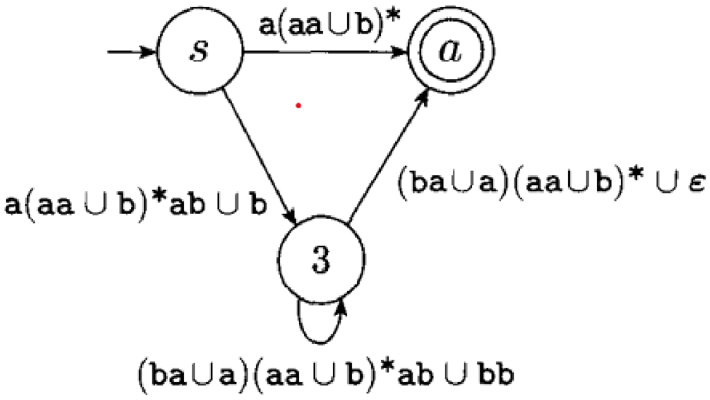


(c)

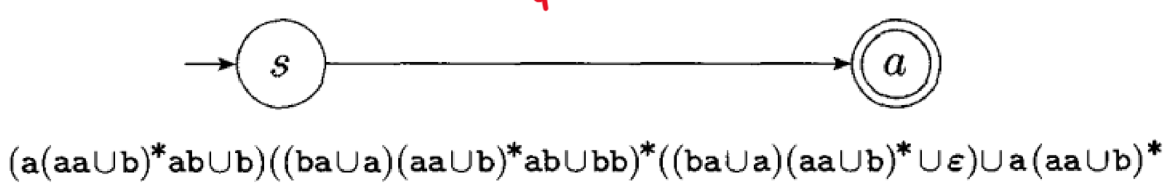


(d)

Da automa a espressione. Esempio



(d)



(e)