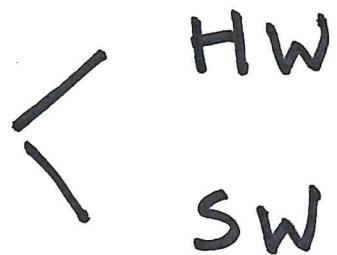


TEORIA DEGLI AUTOMI A STATI FINITI

Applicazioni



- HW: progettazione di circuiti elettronici
modellare semplici sistemi
- SW: costruzione di analizzatori lessicali
soluzione a problemi di ricerca nei testi
modellare pagine web

Un sistema da modellare con automi :

- Abbiamo un Robot che si muove su una griglia
- Il Robot è telecomandato da segnali che indicano la direzione
- Il comportamento del Robot può essere modellato da un automa

posizioni
nella griglia



stati
dell'automa

segnali inviati
al ROBOT



simboli mani di
input dell'automa:
N, S, E, O

Questo ci porta all'automa:

Domanda:

Da cosa dipende la posizione del ROBOT?

Caratteristica principale:

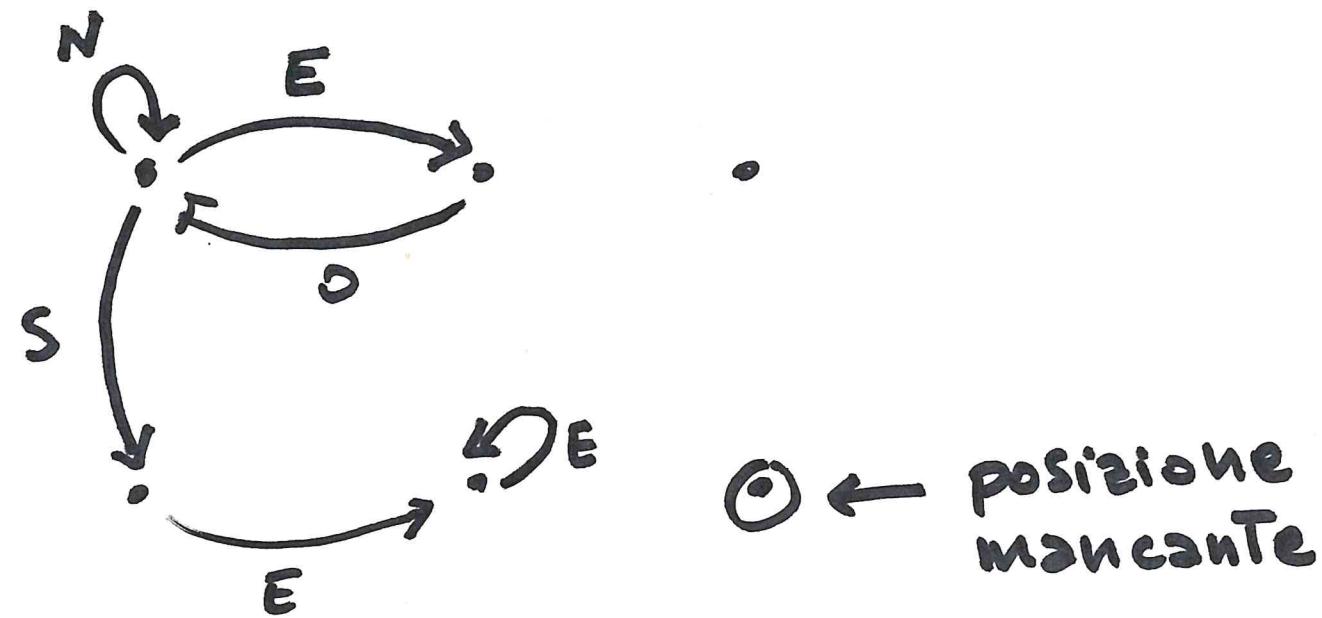
La posizione prossima del ROBOT dipende:

- dalla posizione attuale
- dal simbolo inviato

OSSERVAZIONE:

Un percorso diventa una parola

AUTOMA A STATI FINITI

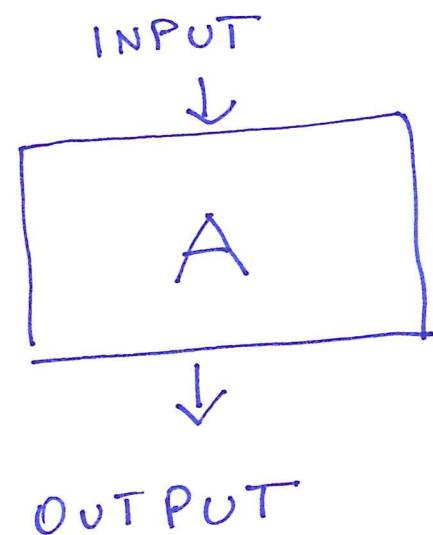


GRAZIE ALL'AUTOMA POSSO STUDIARE
A TAVOLINO PROBLEMI LEGATI
AL SISTEMA

- individuare una parola che mi porta da A a B
- individuare una parola che mi porta da A a B passando attraverso C (o evita C)
- individuare Tutte le parole prive di cicli che mi portano da A a B
- Tra queste ultime individuare la parola più corta

Automi a stati finiti come riconoscitori
di linguaggi formali

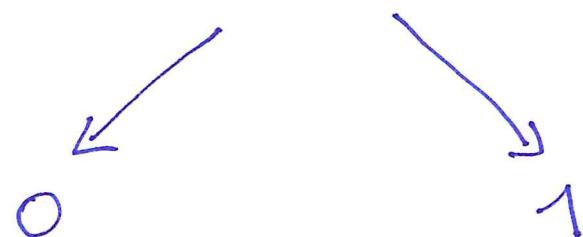
In prima approssimazione si presenta così:



← scatola hera
che interagisce
con l'esterno

Un esperimento consiste in:

- passare $w \in \Sigma^*$ in input
- osservare l'uscita



$\Rightarrow w$ è rifiutata

$\Rightarrow w$ è accettata

Il comportamento di A è dato dall'insieme
di quelle parole che A accetta = Linguaggio accettato



A è un riconoscitore di linguaggi, in particolare
gli automi a stati finiti riconoscono i linguaggi di tipo 3

In dettaglio:

-) In ogni istante di tempo t l'automa si trova in un particolare stato.

Inizialmente A si trova in uno stato iniziale: q_0

-) In funzione del simbolo letto e dello stato attuale A cambia stato

δ = funzione di transizione

$\delta(q, \sigma)$ = stato prossimo di A
essendo in q e leggendo σ

-) Una volta letta l'intera parola w A raggiunge uno stato p e l'uscita dipende da p :

$\lambda(p) = \begin{cases} 0 \\ 1 \end{cases}$ funzione di uscita

Formalizziamo:

Def: Un automa a stati finiti è
una tupla $A = \langle \Sigma, Q, \delta, q_0, \lambda \rangle$

\uparrow
 \downarrow
F

dove:

Σ = alfabeto di input

Q = insieme degli stati

(se Q è finito A è a stati finiti)

δ = funzione di transizione

$\delta: Q \times \Sigma \rightarrow Q$

q_0 = stato iniziale, $q_0 \in Q$

λ = funzione di uscita $\lambda: Q \rightarrow \{0,1\}$
oppure

$F \subseteq Q$ stati finali o accettanti

Osservazione :

- Dato λ posso avere F

$$F = \{ q \in Q \mid \lambda(q) = 1 \}$$

- Dato F posso avere λ

$$\lambda(p) = \begin{cases} 1 & \text{se } p \in F \\ 0 & \text{se } p \notin F \end{cases}$$

Rappresentazioni di δ :

• TABELLA RE

| | | \sum |
|-----|----------|--|
| | | $\delta_j \sigma_1 \sigma_2 \dots \sigma_j \dots \sigma_h$ |
| Q | q_0 | |
| | q_1 | |
| | \vdots | \vdots |
| | q_i | $\cdots \cdots \cdots - \delta(q_i, \sigma_j)$ |
| | \vdots | \vdots |
| | q_K | |

Esempio:

$$Q = \{ q_0, q_1 \} \quad \Sigma = \{ a, b \}$$

q_0 iniziale $F = \{ q_0 \}$

| δ | a | b |
|----------|-------|-------|
| q_0 | q_1 | q_0 |
| q_1 | q_0 | q_1 |

) DIAGRAMMA DEGLI STATI

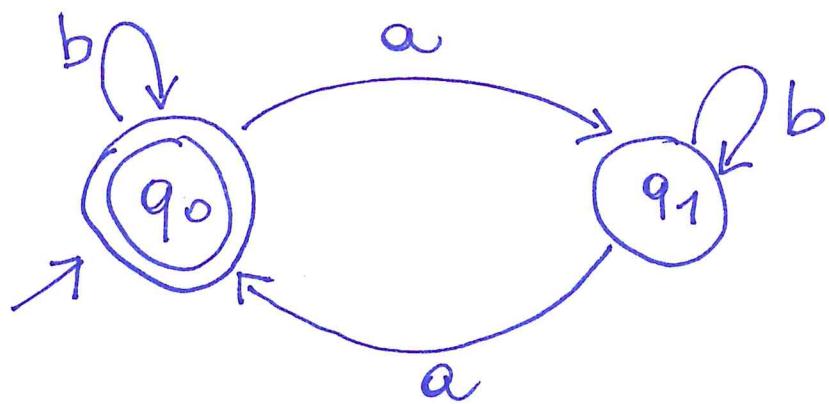
\circ = STATO

\xrightarrow{a} = CAMBIAMENTO DI STATO DOVUTA A a

$\xrightarrow{\quad}$ = STATO INIZIALE

\circlearrowright = STATO FINALE

Esempio :



Una parola è accettata se
partendo da q_0 = STATO INIZIALE
induce un cammino che termina in
uno STATO FINALE

es. di parole :

bbb $q_0 \xrightarrow{bbb} q_0$ è accettata

bab $q_0 \xrightarrow{bab} q_1$ è rifiutata

$\underbrace{aa\dots a}_{\text{lunghezza pari}}$ $q_0 \xrightarrow{aaa\dots a} q_0$ è accettata

lunghezza
pari

Formalizziamo il concetto di
linguaggio riconosciuto

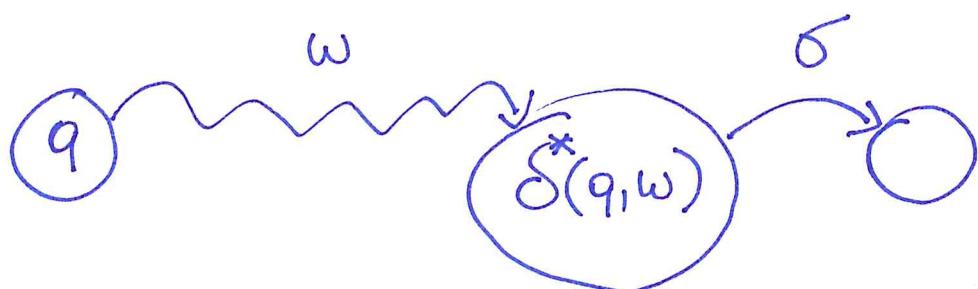
Introduciamo la δ^* che è la δ estesa
alle parole:

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

definizione induttiva:

$$\left\{ \begin{array}{l} \delta^*(q, \varepsilon) = q \\ \delta^*(q, w\sigma) = \delta(\delta^*(q, w), \sigma) \end{array} \right.$$

dove $w \in \Sigma^*$, $\sigma \in \Sigma$



$L(A)$ = linguaggio riconosciuto da A

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

$$= \{ w \in \Sigma^* \mid \lambda(\delta^*(q_0, w)) = 1 \}$$

Esempio

$$L(A) = \{ w \in \{a, b\}^* \mid |w|_a = 2n, n \geq 0 \}$$

$|w|_\sigma$ = numero di σ in w

STATI PARTICOLARI:

- STATO TRAPPOLA : q

se vale - $\forall \sigma \in \Sigma$

$$\delta(q, \sigma) = q$$

- $q \notin F$

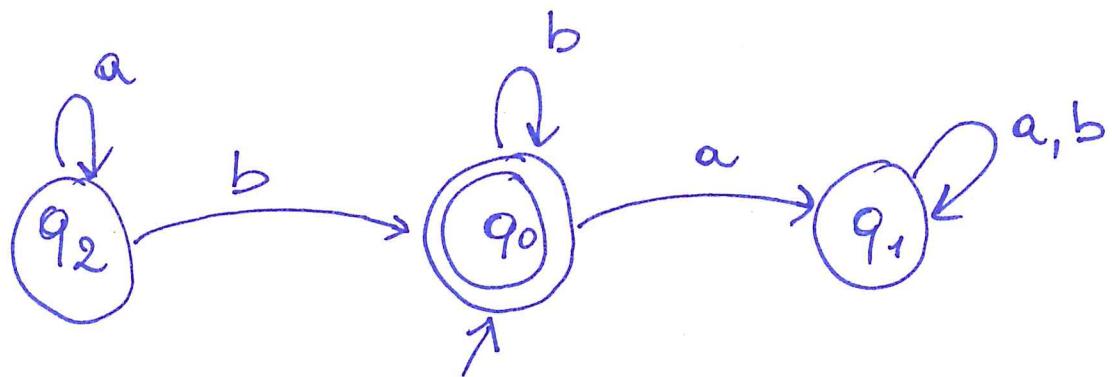
- STATO OSSERVABILE : p

se vale

$$\exists w \in \Sigma^*$$

$$\delta^*(q_0, w) = p$$

Esempio:



Notiamo che:

q_1 è - OSSERVABILE grazie ad "a"

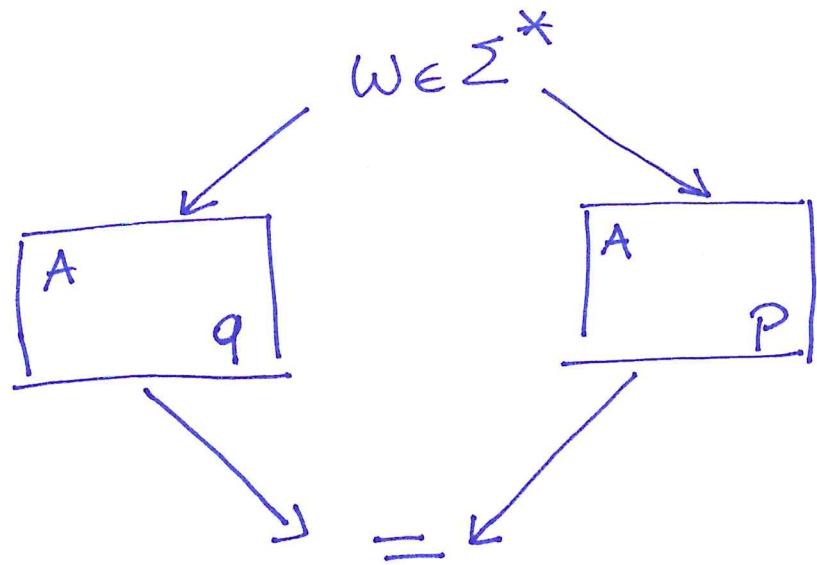
- TRAPPOLA: non esce da q_1
e q_1 non è finale

q_0 è OSSERVABILE grazie ad "ε"

q_2 è NON OSSERVABILE
e si può eliminare!

STATI INDISTINGUIBILI

$q \in P$ si dicono indistinguibili quando



$$\forall w \in \Sigma^* \quad \lambda(\delta^*(q, w)) = \lambda(\delta^*(p, w))$$

STATI DISTINGUIBILI

$q \in P$ si dicono distinguibili quando

$$\exists w \in \Sigma^* \quad \lambda(\delta^*(q, w)) \neq \lambda(\delta^*(p, w))$$

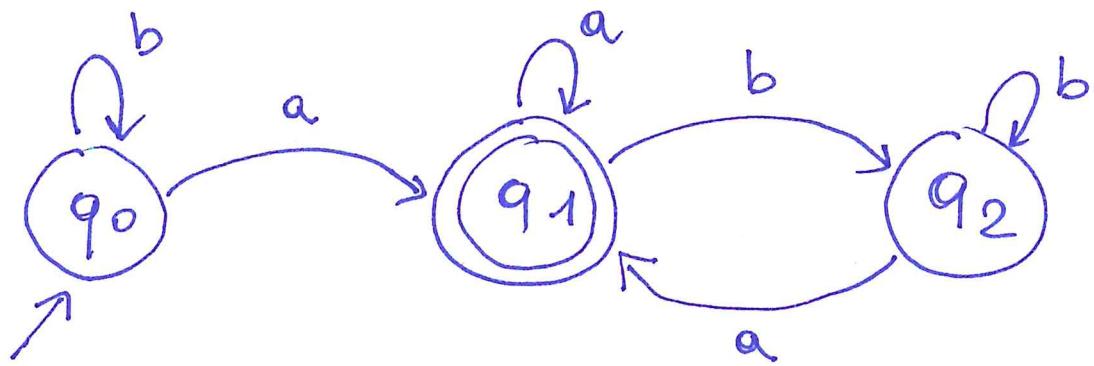
NOTAZIONE

$$q \approx p \quad \text{indistinguibili}$$

$$q \not\approx p \quad \text{distinguibili}$$

Esempio :

A

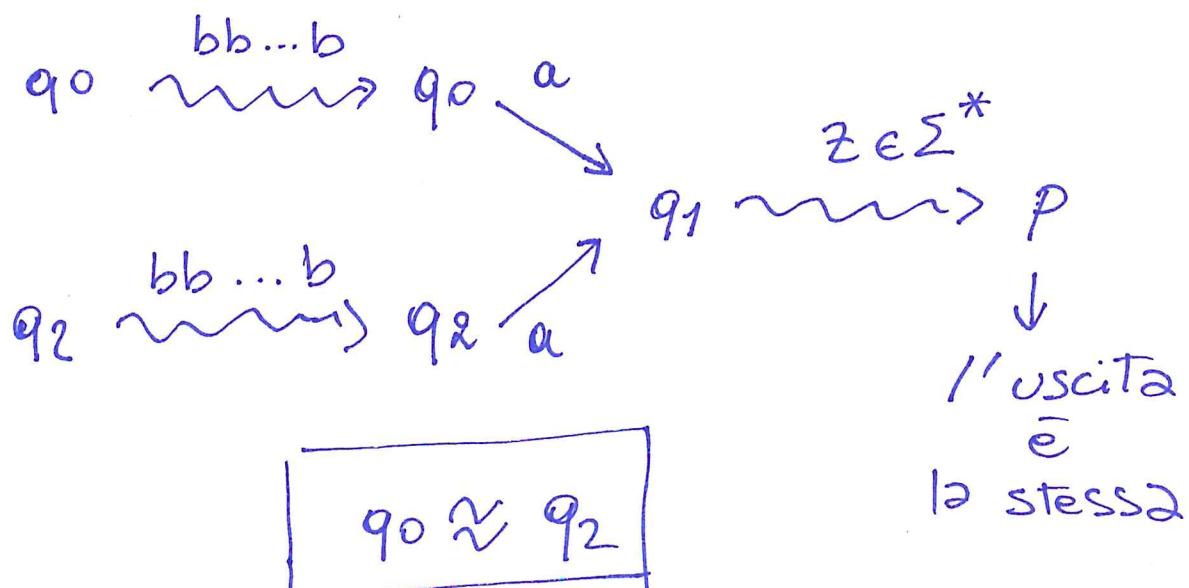


$$L(A) = \{a, b\}^* a$$

$q_0, q_1 : q_0 \not\sim q_1$ grazie ad " ϵ "

$q_1, q_2 : q_1 \not\sim q_2$ "

$q_0, q_2 : \text{osserviamo che}$



RIDUZIONE DEGLI STATI

\approx : relazione binaria su Q

proprietà di \approx :

- riflessiva

$$\forall q \quad q \approx q$$

- simmetrica

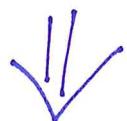
$$\forall q, q' \quad q \approx q' \Rightarrow q' \approx q$$

- transitiva

$$\forall q, q', q'' \quad q \approx q' \text{ e } q' \approx q'' \Rightarrow q \approx q''$$

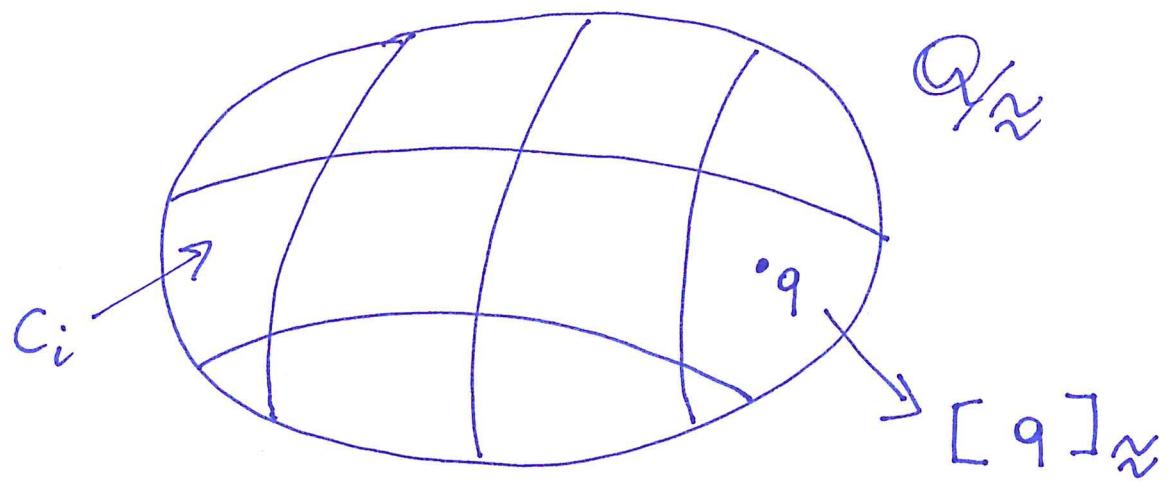
- $\forall q, q' \quad q \approx q' \Rightarrow$

$$\forall z \in \Sigma^* \quad \delta^*(q, z) \approx \delta^*(q', z)$$



\approx è una relazione di equivalenza

Pertanto \approx induce una partizione su \mathbb{Q} :



Partizione in classi : C_i

- $\forall i \quad C_i \neq \emptyset$
- $\forall i, j \quad i \neq j \quad C_i \cap C_j = \emptyset$
- $\bigcup_i C_i = \mathbb{Q}$

Notazione: $[q]_≈ =$ classe di equival.
che contiene
l'elemento q

Dato $A = (Q, \Sigma, q_0, \delta, F)$
 posso costruire A_{\approx} che ha come
 stati le classi di equivalenza: $\frac{Q}{\approx}$.

Formalmente:

$$A_{\approx} = (\Sigma, \frac{Q}{\approx}, [q_0]_{\approx}, \delta_{\approx}, F_{\approx})$$

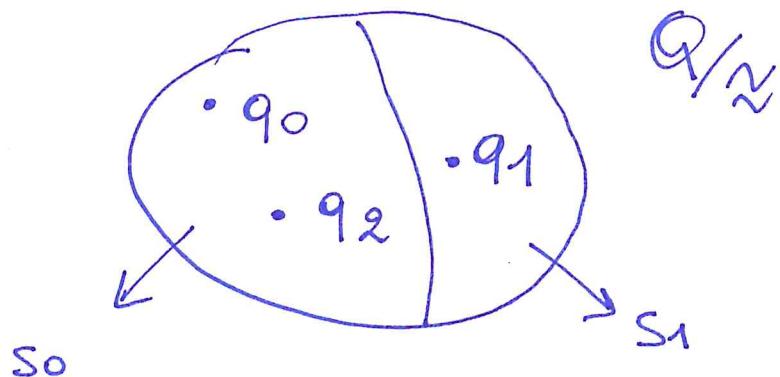
dove:

$$F_{\approx} = \left\{ [p]_{\approx} \mid p \in F \right\}$$

$$\delta_{\approx}: \frac{Q}{\approx} \times \Sigma \rightarrow \frac{Q}{\approx}$$

$$\delta_{\approx}([q]_{\approx}, \sigma) = [\delta(q, \sigma)]_{\approx}$$

Esempio: costruisco $A_{\tilde{\pi}}$



$$Q/\tilde{\pi} = \{ s_0, s_1 \}$$

$$A_{\tilde{\pi}} = (\Sigma = \{a, b\})$$

$$Q/\tilde{\pi} = \{ s_0, s_1 \}$$

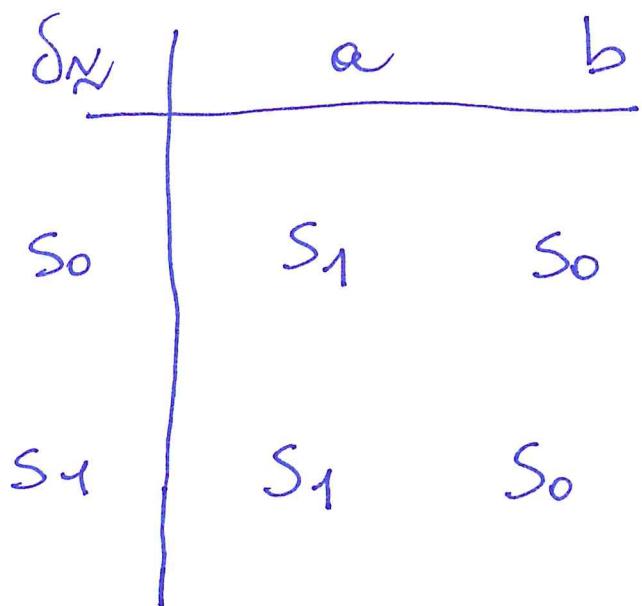
$$[q_0]_{\tilde{\pi}} = s_0$$

$\delta_{\tilde{\pi}}$ da costruire

$$F_{\tilde{\pi}} = \{ [p]_{\tilde{\pi}} \mid p \in F \}$$

$$= \{ [q_1]_{\tilde{\pi}} \} = \{ s_1 \}$$

olefiniamo $\delta_{\tilde{N}}$:



$$\cdot \quad \delta_{\tilde{N}}(s_0, a) = \delta_{\tilde{N}}([q_0]_{\tilde{N}}, a) = [\delta(q_0, a)]_{\tilde{N}}$$

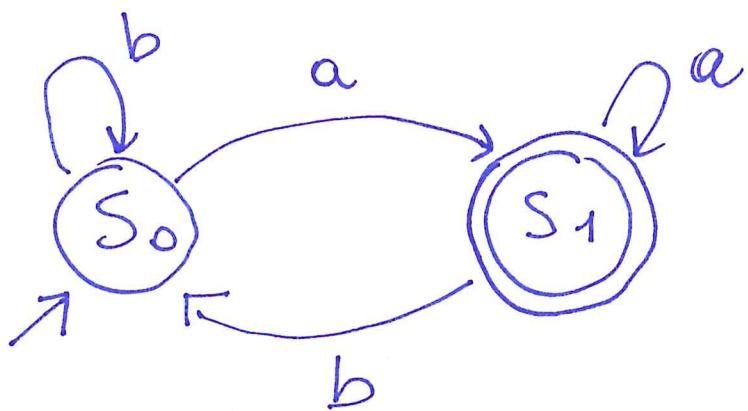
$$= [q_1]_{\tilde{N}} = s_1$$

$$\cdot \quad \delta_{\tilde{N}}(s_1, a) = \delta_{\tilde{N}}([q_1]_{\tilde{N}}, a) = [\delta(q_1, a)]_{\tilde{N}}$$

$$= [q_1]_{\tilde{N}} = s_1$$

...

graficamente A_{\approx} è:



$$L(A_{\approx}) = \{a, b\}^* a = L(A)$$

OSSERVAZIONE:

A_{\approx} è equivalente ad A

ma A_{\approx} ha meno stati di A .

$$3 \rightarrow 2$$

Problema di sintesi di Automi :

- Dato un linguaggio L
- trovare un automa A per L

In particolare:

1. • com'è fatto l'automa massimo per L?
2. • come posso ottenere l'automa minimo per L?

"massimo" = maggior numero di stati

"minimo" = minor numero di stati



nostro obiettivo

1. Automa massimo: G_L

L'idea è: raggiungo uno stato sempre
olvero per ogni parola in input

$$\forall w, w' \in \Sigma^* \quad \delta(q_0, w) \neq \delta(q_0, w')$$

$$\downarrow \\ Q = \Sigma^*$$

Inoltre, per non perdere stati,
tutti gli elementi di Q devono essere
osservabili

$$\forall q \in Q \quad \exists w \text{ t.c. } q = \delta(q_0, w)$$

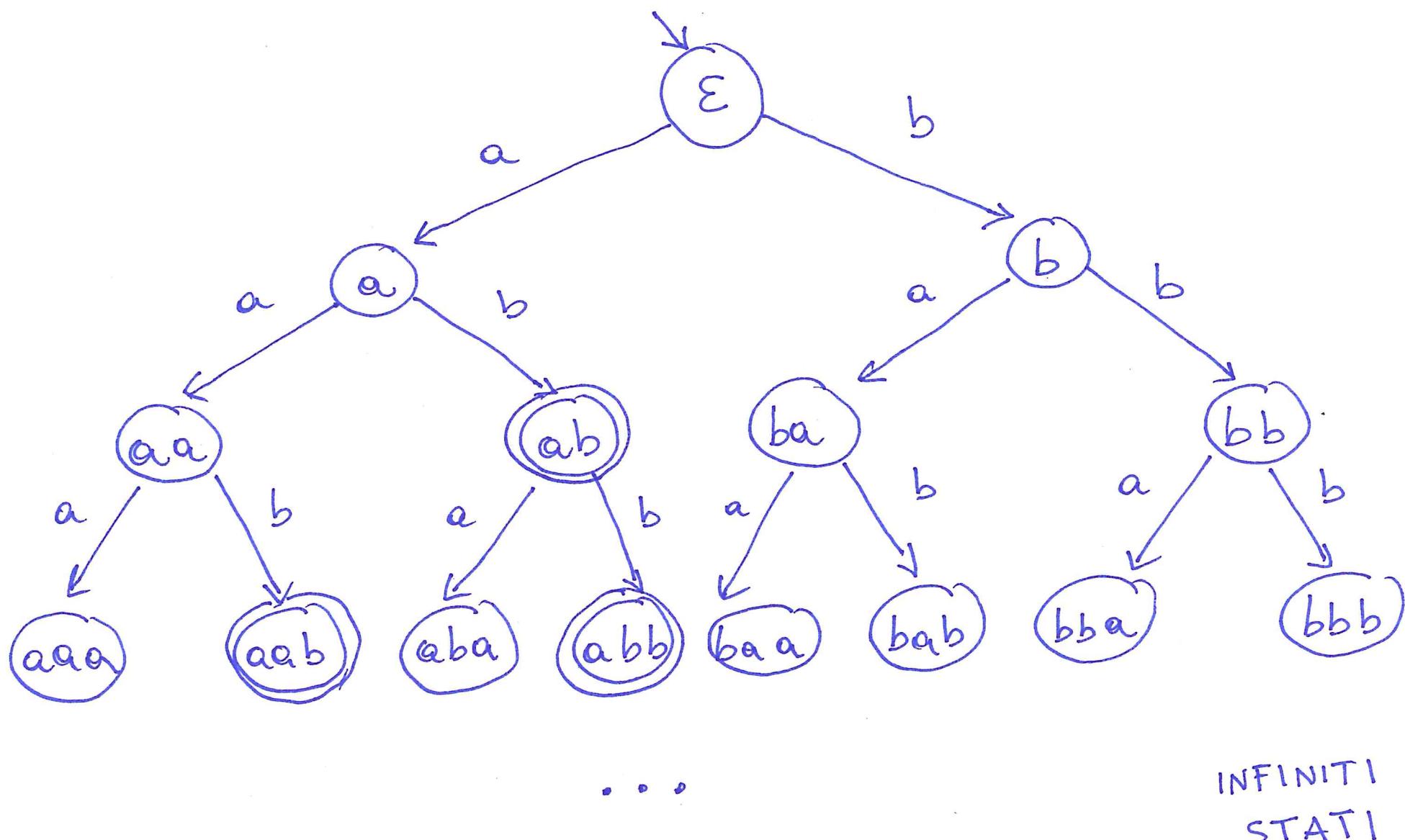
Def: $G_L = (\Sigma^*, \Sigma, [\varepsilon], \delta_{G_L}, F_{G_L})$

dove:

$$F_{G_L} = \{ [w] \in Q \mid w \in L \}$$

$$\delta_{G_L}([w], \sigma) = [w\sigma]$$

Automa massimo G_L per $L = \{ a^n b^m \mid n, m > 0 \}$



2. Automa minimo M_L

Abbiamo 2 tecniche:

- si usa l'automa massimo G_L
- si usa un generico automa A per L
 - 2 stati finiti
- prima tecnica:

Teorema

$G_L \approx$ è l'automa minimo per L
(cioè $G_L \approx = M_L$)

dimo:

Devo mostrare che se A è un automa per L allora:

$$|A| \geq \cancel{X} \cancel{N} \cancel{Y} \cancel{Z} |G_L \approx|$$

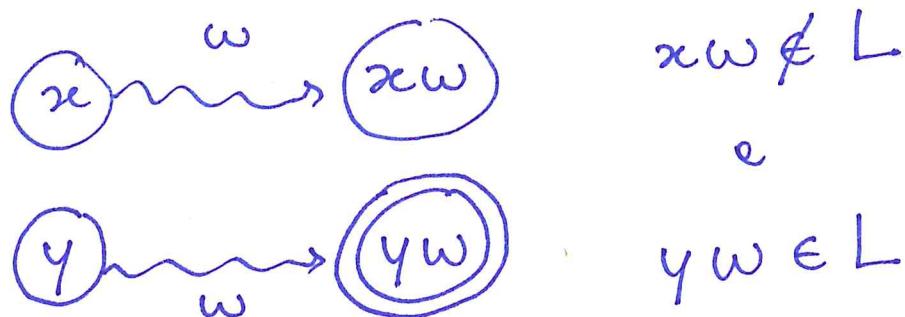
$|A| =$ numero di stati di A

fatto : stati distinguibili in G_L
sono distinti in A :

In fatti :

Siano $[x]$ e $[y]$ distinguibili in G_L

$[x] \neq [y] \Rightarrow \exists w \in \Sigma^*$ t.c.



Allora in A :

$\delta(q_0, x) \neq \delta(q_0, y)$ altrimenti



Ora :

Siano $[x_1]_{\sim}, [x_2]_{\sim}, [x_3]_{\sim}, \dots$

gli stati di $G_{L \sim} \neq \mathcal{M}_L$,

allora $[x_1], [x_2], [x_3], \dots$

sono stati di G_L distinguibili.

In virtù del fatto appena dimostrato

$\delta(q_0, x_1), \delta(q_0, x_2), \delta(q_0, x_3), \dots$

sono distinti in A

(dove A è un generico automa per L)

$\Rightarrow A$ ha almeno tanti stati

quanti $G_{L \sim}$, cioè

$|A| \geq |G_{L \sim}| \Rightarrow$ $\begin{matrix} G_{L \sim} \\ \text{è} \\ \text{minimo} \end{matrix}$

- Seconda Technica:

Teorema

Sia A un automa a stati finiti
per L tale che:

- gli stati di A siano tutti osservabili
- gli stati di A siano tutti distinguibili

Allora A è minimo per L .

Corollario:

A_{\sim} è minimo per L se

A ha tutti stati osservabili.

olim :

Siano $\{q_1, q_2, q_3, \dots, q_n\}$ gli stati di A
↑
iniziale

- sono osservabili, allora

esistono parole $\{w_1, w_2, \dots, w_n\}$

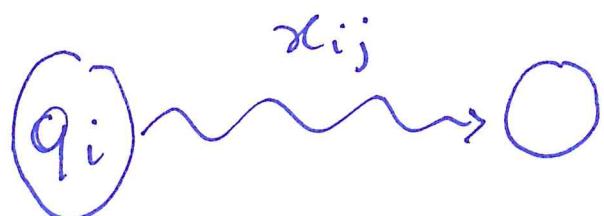
tali che:

$$\forall i \quad q_i = \delta(q_1, w_i)$$

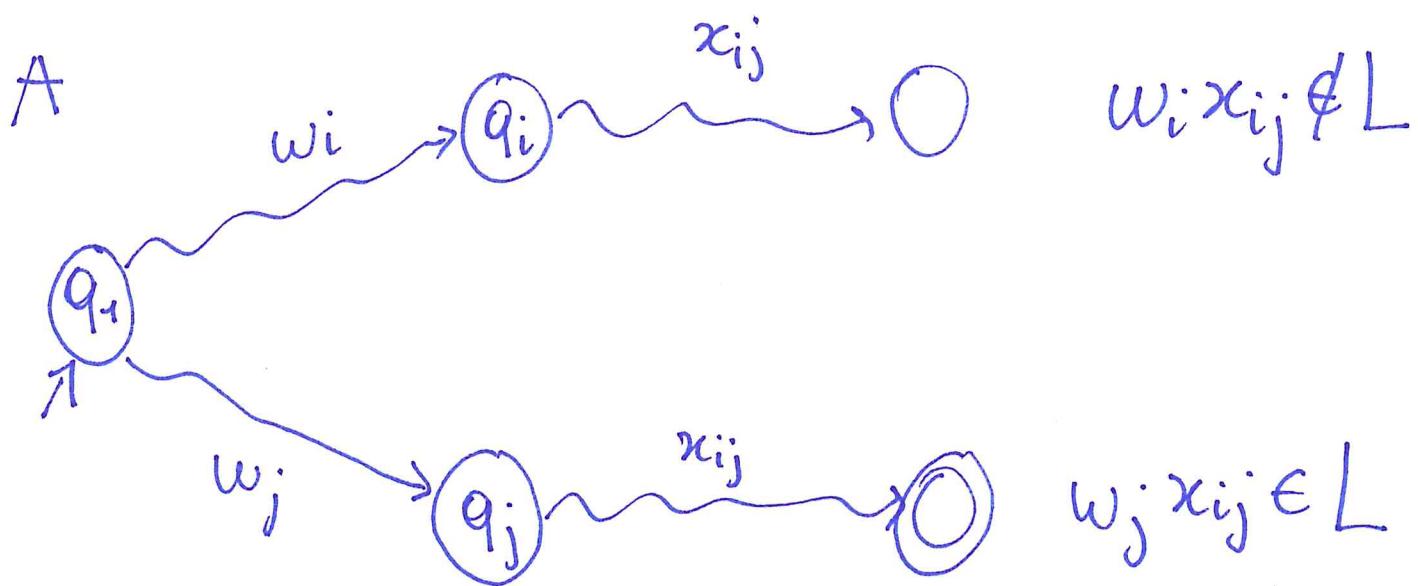
- sono distinguibili, allora

esistono parole $x_{i,j}$ che distinguono

gli stati:



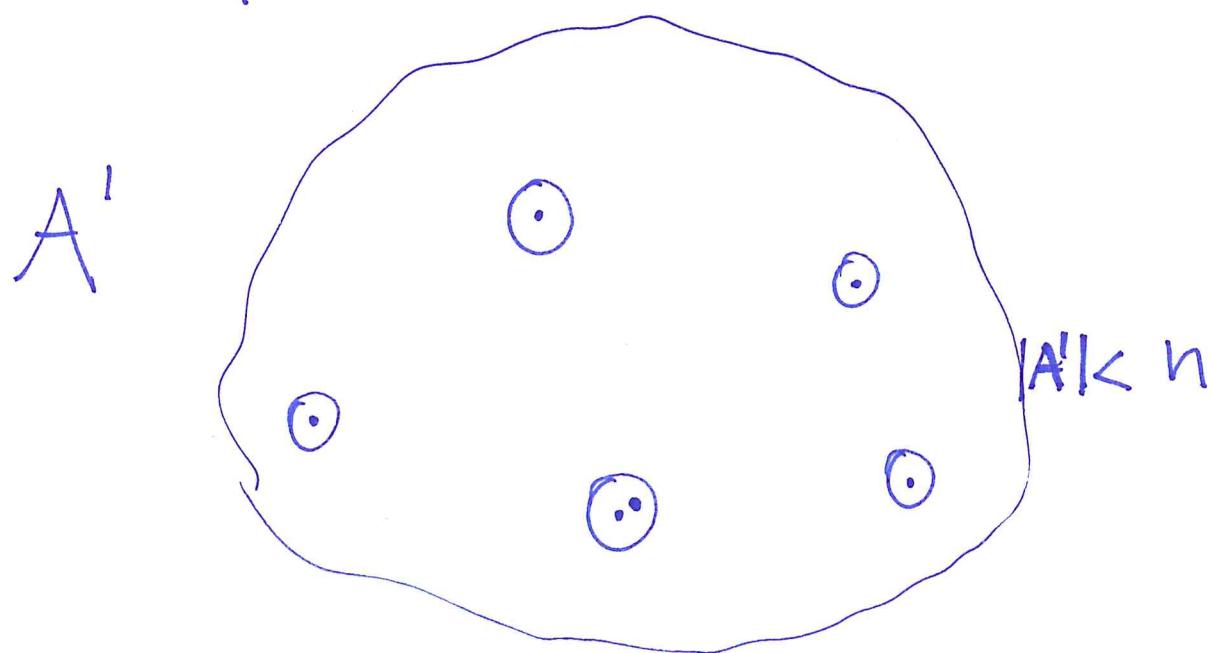
Riassumendo:



Adesso sia A' un automa per L
con meno stati di A .

Diamo in input ad A' le parole

$\{w_1, w_2, \dots, w_n\}$ si ha:

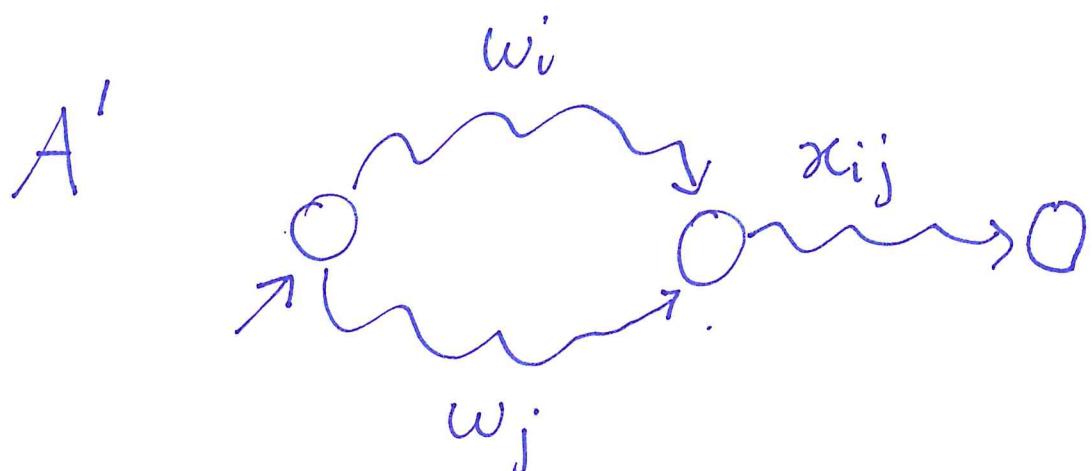


almeno olve parole raggiungono lo stesso stato.

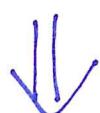
(principio della piccionaia)

Siano w_i, w_j queste due parole.

Allora:



così ottengo $\begin{cases} w_i x_{ij} & | \sigma \text{ in } L \\ w_j x_{ij} & | \sigma \text{ non in } L \end{cases}$



A' non riconosce L



non esiste un automa per L con
meno stati di A

Algoritmi di sintesi ottima di automi

Sintesi ottima di automi

- Dato un linguaggio L
- Trovare l'automa minimo per L

Abbiamo due tecniche:

① Se ho A per L posso (A ha stati finiti):

- eliminare gli stati non osservabili
- costruire A_N

② Se non ho A per L :

- costruisco G_L automa massimo
- costruisco $G_{L \approx} = M_L$

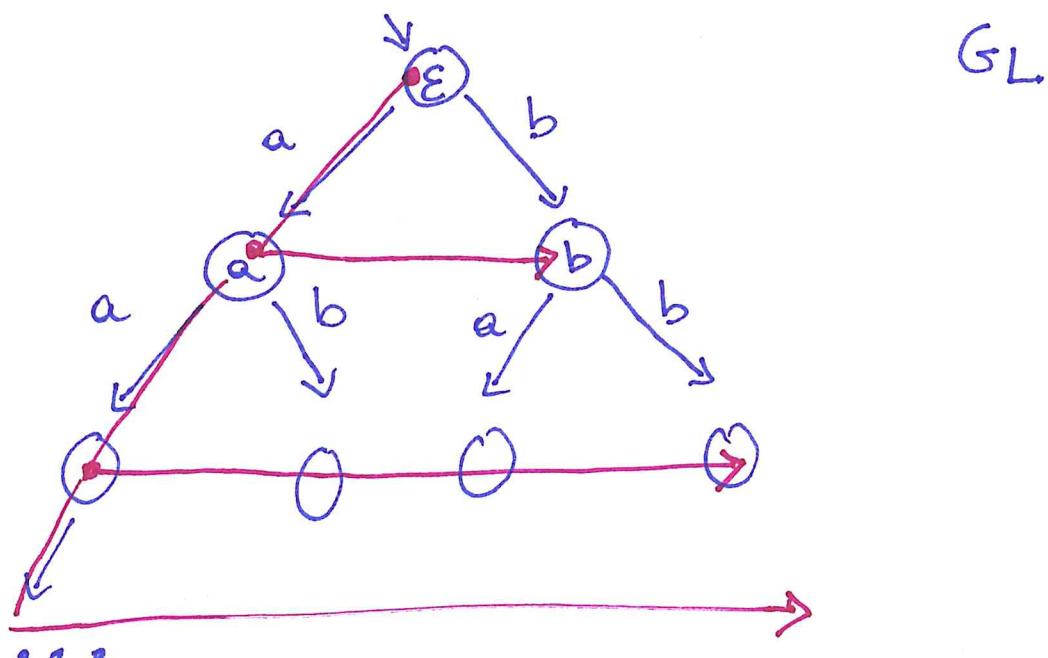
Algoritmi :

① E' possibile confrontare gli stati
seguendo un ordine casuale

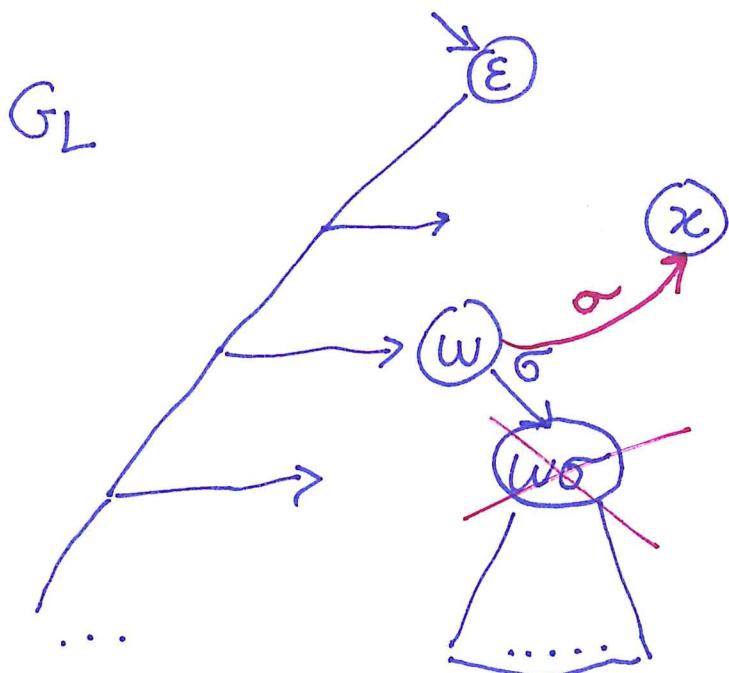
② Devo applicare un algoritmo
sui confronti ben preciso:

Algoritmo per GL

) Si parte dalla radice e si
visitano i nodi (= stati) in
ampiezza:



- La visita consiste nel confrontare il nodo attuale con i nodi precedentemente visitati verificando se i nodi sono indistinguibili oppure no.
- Sia $[\omega\sigma]$ (con $\omega \in \Sigma^*$, $\sigma \in \Sigma$) il nodo attuale. Se accade che $[\omega\sigma]$ è indistinguibile dal nodo $[\omega]$ allora:
 - cancella $[\omega\sigma]$ e il suo sottoalbero
 - resta pendente l'arco uscente da $[\omega]$ etichettato con σ . Ridireziona l'arco verso $[\omega]$.



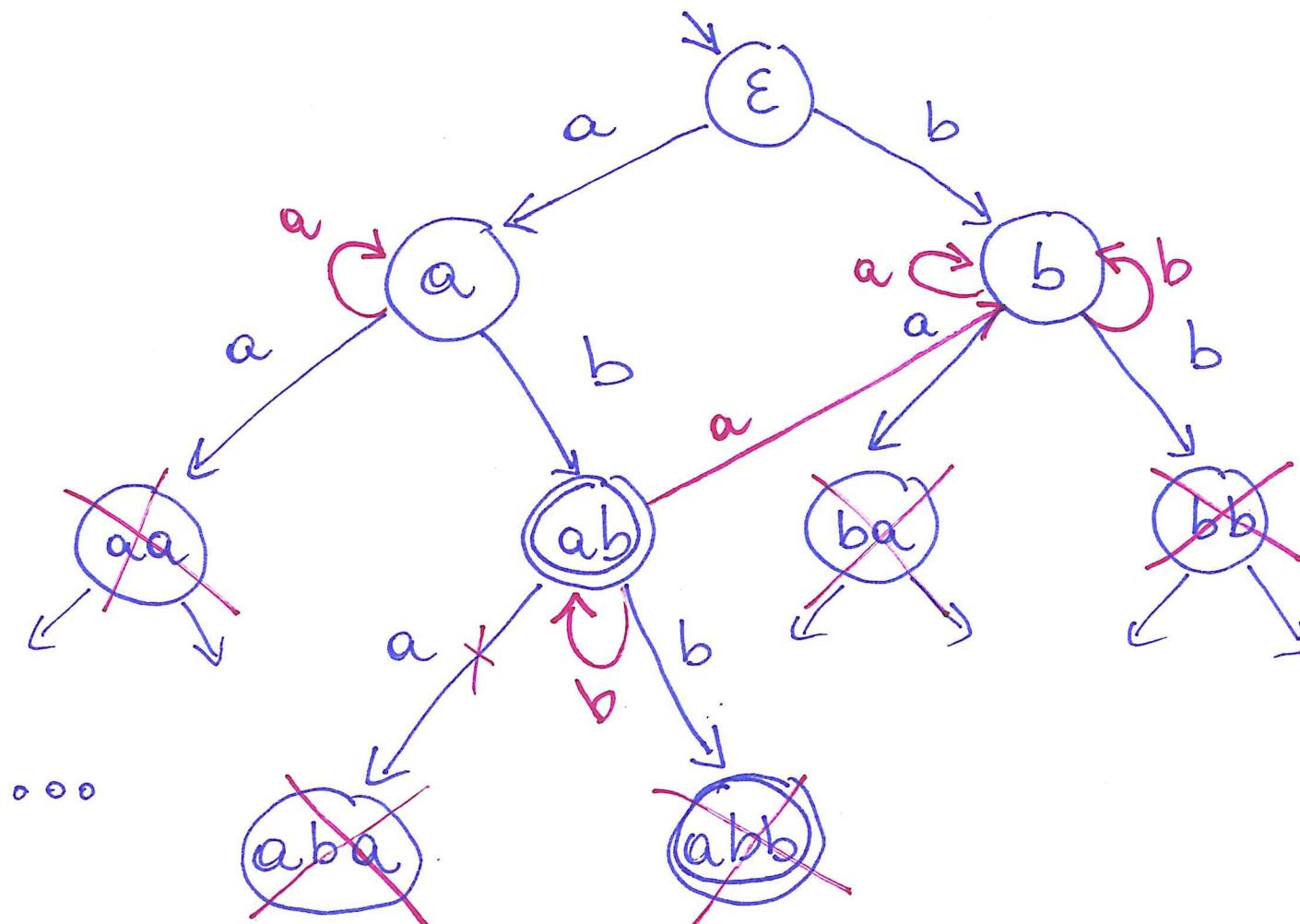
Osservazione:

- lo stato iniziale resta $[\epsilon]$.
- gli stati finali sono i sopra vissuti etichettati con parole di L .

Due prove di esecuzione:

- Per $L = \{a^n b^m \mid n, m > 0\}$
otteniamo un automa a stati finiti : M_L per il Teo mostrato
- Per $L = \{a^n b^n \mid n > 0\}$
otteniamo un automa con infiniti stati
conseguenza: $a^n b^n$ non è regolare

Costruzione di $G_{L\infty}$ per $L = \{a^m b^m \mid m, m > 0\}$



Applicazione dell'algoritmo. Visite:

- $[\varepsilon]$ resta

- $[a]$:

$[a] \not\approx [\varepsilon]$ a causa della p. "b"

- $[b]$:

$[b] \not\approx [\varepsilon]$ a causa della p. "ab"

$[b] \not\approx [a]$ a causa della p. "b"

- $[aa]$:

$[aa] \not\approx [\varepsilon]$ a causa della p. "b"

$[aa] \approx [a]$ si

- $[ab]$: è finale e gli altri no

- $[ba]$:

$[ba] \approx [b]$ si

- $[bb]$:

$[bb] \approx [b]$ si

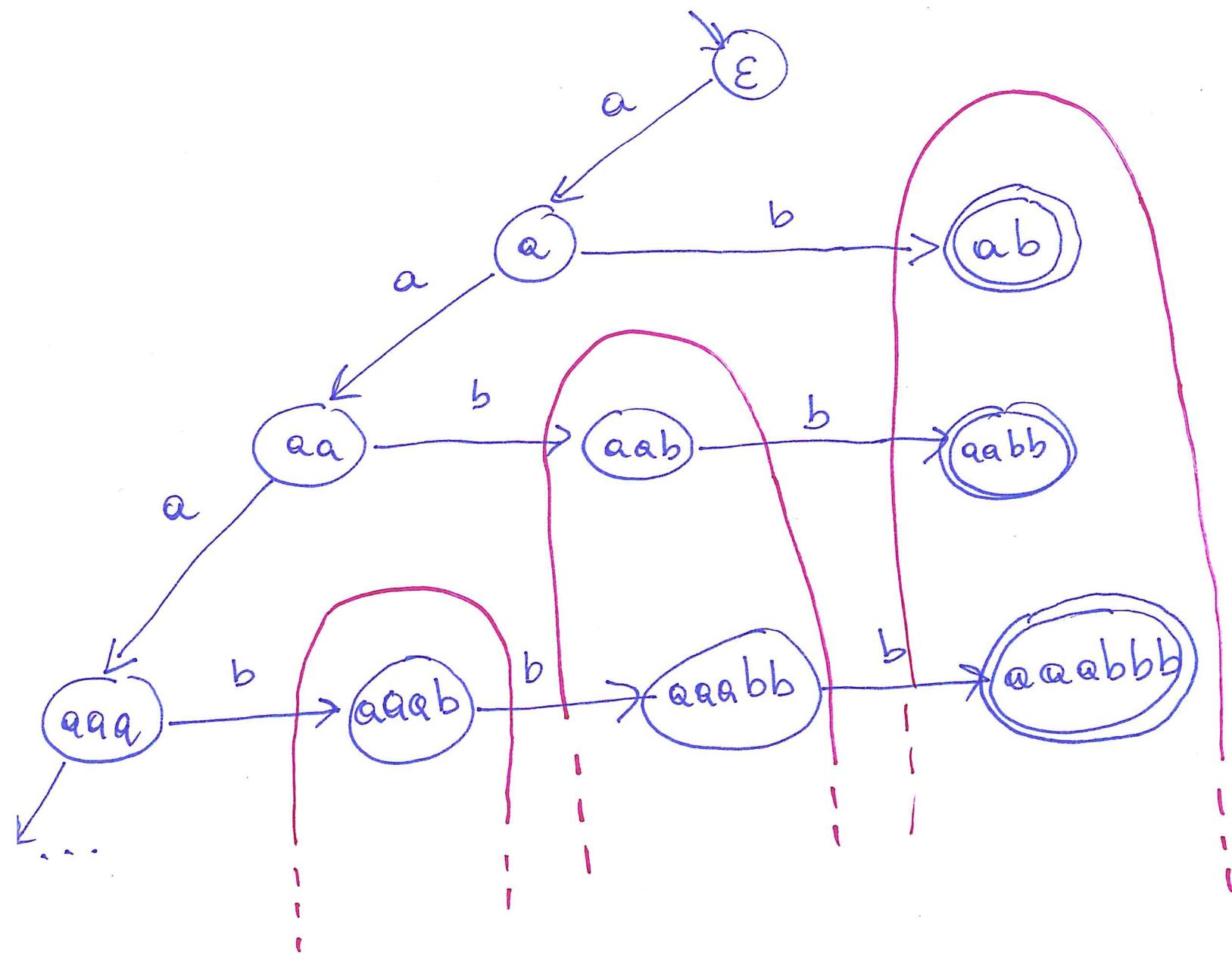
- $[aba]$:

$[aba] \approx [b]$ si

- $[abb]$:

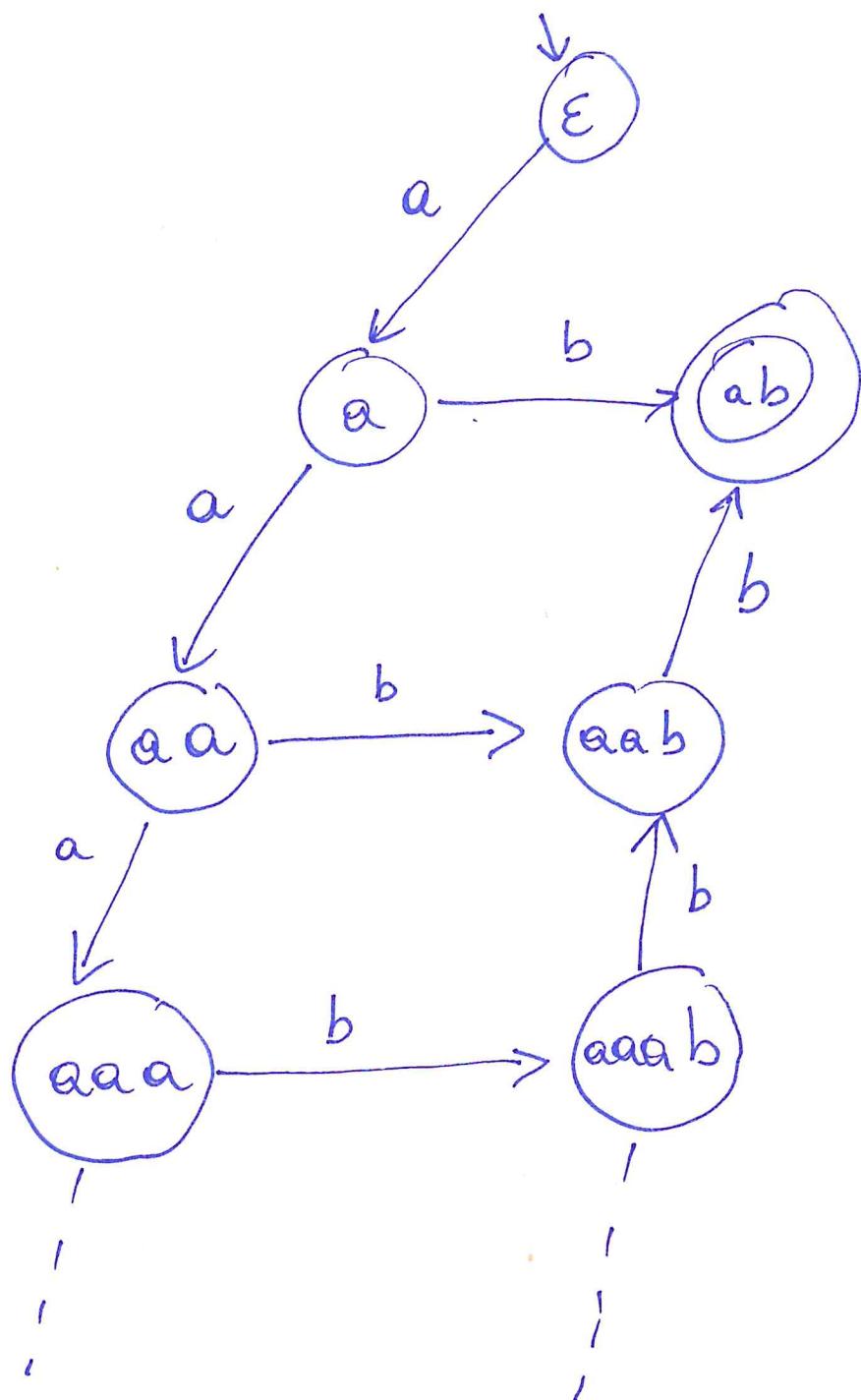
$[abb] \approx [ab]$ si

Costruzione di GL_{Σ} per $\{a^m b^m \mid m > 0\}$



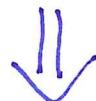
è sottointeso
uno stato trappola

Si ottiene :



$\Rightarrow G_L \approx$

ha infiniti
stati



$a^n b^n$ non
è REGOLARE!

RAMO
INFINITO
CON "a"

RAMO
INFINITO
CON "b"

FATTO:

$a^m b^m$ non è REGOLARE.

dim:

- L'automa minimo $G_{L\approx}$ per $a^n b^n$ ha infiniti stati.
- allora $a^n b^n$ non ammette un automa a stati finiti.
- vista l'equivalenza tra automi a stati finiti e grammatiche di Tipo 3



$a^n b^n$ non è REGOLARE

Teorema

L è generato da g di tipo 3



L è riconosciuto da A a stati finiti

olim: da A ricavo G .

Sia $A = (\Sigma, Q, q_0, S, F)$, posso definire

$$G = (T = \Sigma,$$

$$V = Q,$$

$$S = q_0$$

P:

$$- q \rightarrow \varepsilon \Leftrightarrow q \in F$$

$$- q \rightarrow \sigma p \Leftrightarrow \delta(q, \sigma) = p$$

correttezza della costruzione:

proprietà:

$$\delta(q_0, w) = p \iff q_0 \xrightarrow{*}^w p$$

si dimostra per induzione:

caso base

$$\delta(q_0, \varepsilon) = q_0 \iff q_0 \xrightarrow{*}^\varepsilon q_0 = q_0$$

vero

passo induttivo

suppongo vera la proprietà per n

e lo dimostro per $n+1$

si considera $w\sigma$, con $|w|=n$ e $\sigma \in \Sigma$,

quindi $|w\sigma|=n+1$.

O.K

$$\delta(\underline{q_0}, \underline{\omega}\sigma) = P \Leftrightarrow \delta(\delta(q_0, \omega), \sigma) = P$$

$$\Leftrightarrow \delta(q_0, \omega) = q \text{ e } \delta(q, \sigma) = P \Leftrightarrow$$

↑ ↑
per ipotesi per costruzione

$$\Leftrightarrow q_0 \xrightarrow{*} \omega q \text{ e } q \xrightarrow{*} \sigma P \Leftrightarrow$$

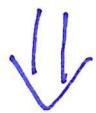
$$\Leftrightarrow q_0 \xrightarrow{*} \underline{\omega}\sigma P$$

A olesso facciamo vedere l'equivalenza:

$$\omega \in L(A) \Leftrightarrow \exists q \in F \text{ s.t. } \delta(q_0, \omega) = q$$

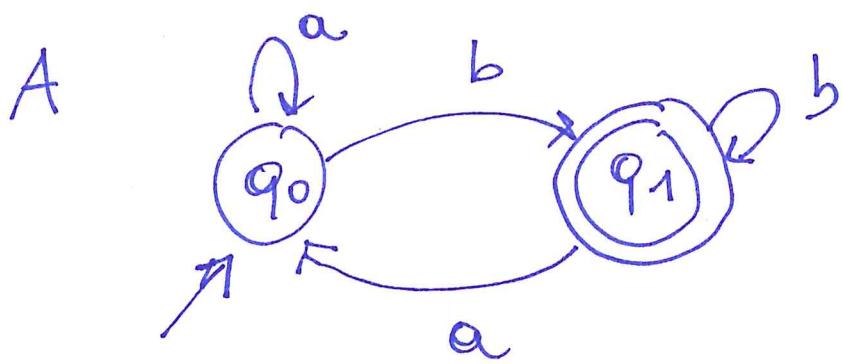
$$\Leftrightarrow q \xrightarrow{*} \varepsilon \text{ e } q_0 \xrightarrow{*} \omega q \Leftrightarrow$$

$$\Leftrightarrow q_0 \xrightarrow{*} \underline{\omega} \Leftrightarrow \omega \in L(G)$$



$$L(A) = L(G)$$

Esempio :



da A ricavo G di tipo 3:

$$T = \{a, b\}, \quad V = \{q_0, q_1\},$$

l'osservazione è q_0 ,

$$P = \{ \cdot : q_1 \rightarrow \epsilon,$$

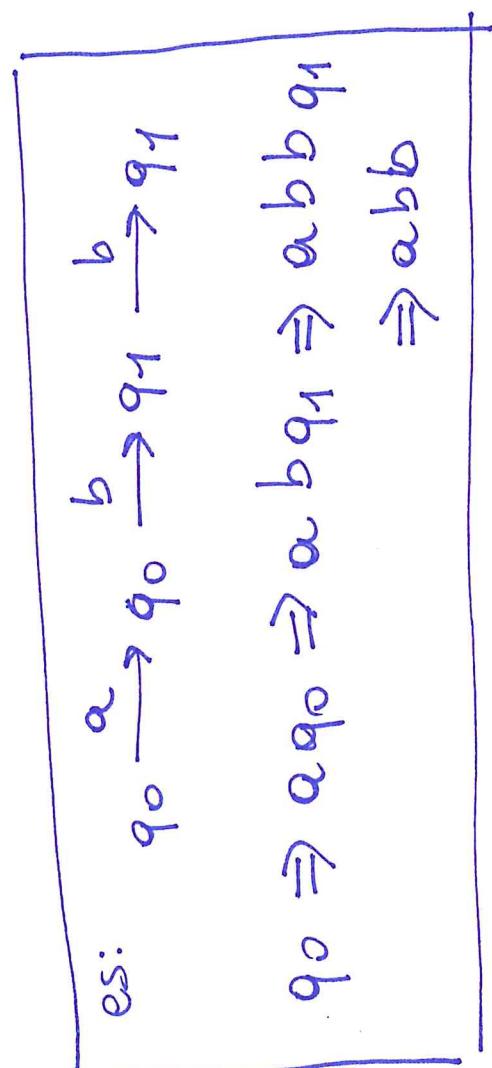
$$\cdot : q_0 \rightarrow a q_0$$

$$\cdot : q_0 \rightarrow b q_1$$

$$\cdot : q_1 \rightarrow b q_1$$

$$q_1 \rightarrow a q_0$$

}



dim: da G di tipo 3 ad A :

- si inverte la costruzione precedente
- ma G deve essere nel formato:

$$\begin{aligned} A &\rightarrow \varepsilon \\ A &\rightarrow \sigma B \end{aligned}$$

Data $G = \langle \Sigma^T, V, S, P \rangle$ definisco A :

$$A = (\Sigma = \Sigma^T$$

$$Q = V$$

~~$$q_0 = S$$~~ assioma

δ è così definita:

$$\delta(A, \sigma) = B \Leftrightarrow A \rightarrow \sigma B$$

$$F = \{ A \mid A \rightarrow \varepsilon \in P\}$$

)

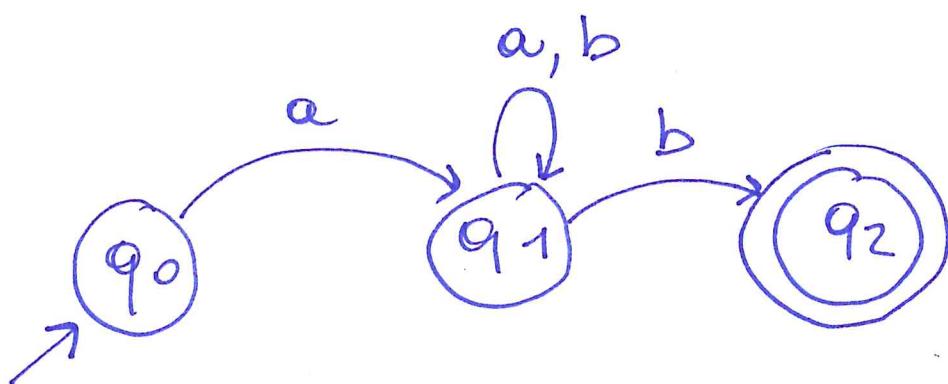
Esempio:

$$G = (\{a, b\}, \{q_1, q_2, q_0\}, q_0, T, V, S)$$

$$P = \left\{ q_0 \xrightarrow{a} q_1, q_1 \xrightarrow{a} q_1, q_1 \xrightarrow{b} q_1, q_1 \xrightarrow{b} q_2, q_2 \xrightarrow{\epsilon} \right\}$$

A

$$L(A) = ?$$



$$\delta(q_1, b) = ? = \begin{cases} q_1 \\ q_2 \end{cases}$$

NOTA: A non è un automa DETERMINIS.
(DFA)

ma è NON DETERMINISTICO
(NFA)

Automi a stati finiti non det. (NFA)

Def. Un NFA è un sistema

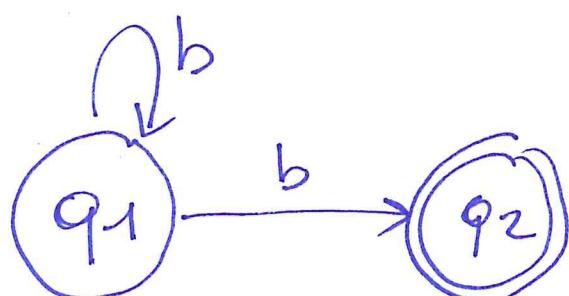
$$A = (\Sigma, Q, q_0, \underline{R}, \bar{F}) \text{ dove}$$

\underline{R} è detta relazione di transizione

$$R: Q \times \Sigma \times Q \rightarrow \{0, 1\}$$

$$R(q, \sigma, p) = \begin{cases} 1 & \text{se } q \xrightarrow{\sigma} p \\ 0 & \text{altrimenti} \end{cases}$$

Esempio:



$$R(q_1, b, q_1) = 1$$

$$R(q_1, b, q_2) = 1$$

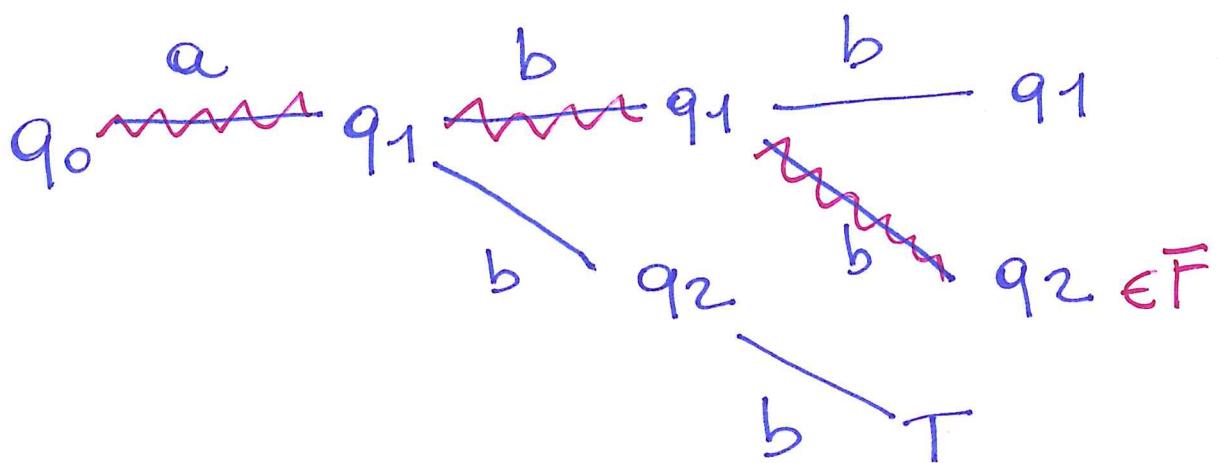
Interpretazione:

A sceglie non det. di transitare
o in q_1 o in q_2

Osservazione:

Data una parola w in input ad A ,
è possibile che induca più di
un cammino possibile

es:



Def: linguaggio riconosciuto da un NFA A

) w è accettata se ammette un
cammino che porta in uno stato finale

) il linguaggio riconosciuto è dato
dalle parole che soddisfano il
criterio di accettazione

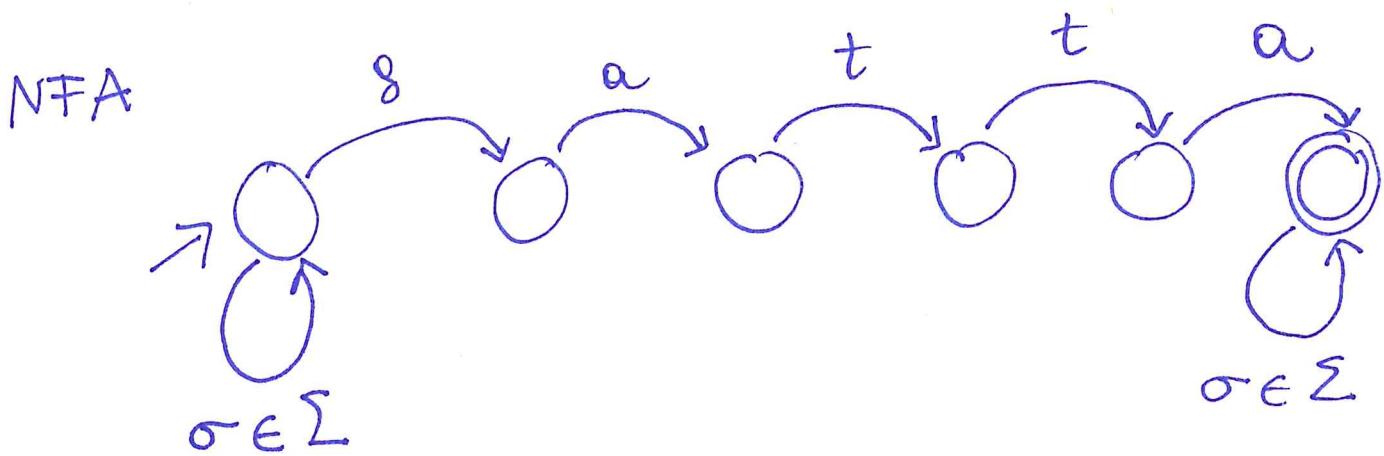
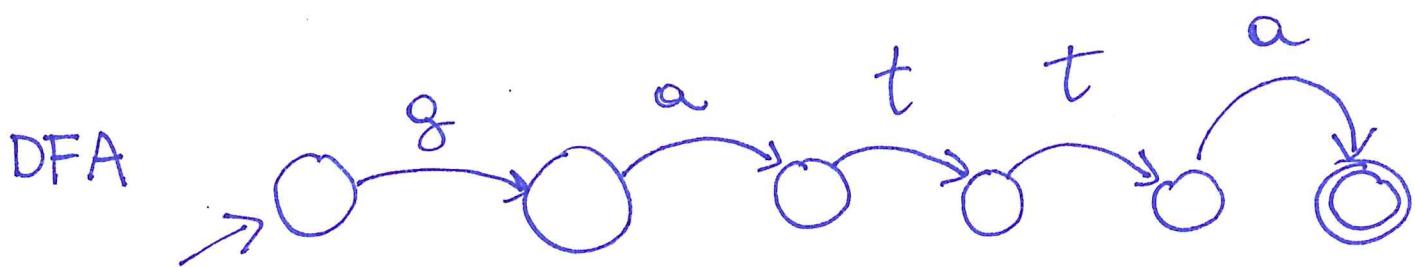
$L(\text{DFA}) = L(\text{NFA})$

Semplici applicazioni:

•) con un DFA riconosco ^{FACILMENTE} una parola

•) con un NFA riconosco ^{FACILMENTE} testi che
contengono una certa parola

Esempio:



esecuzioni:

"La gatta" è accettata

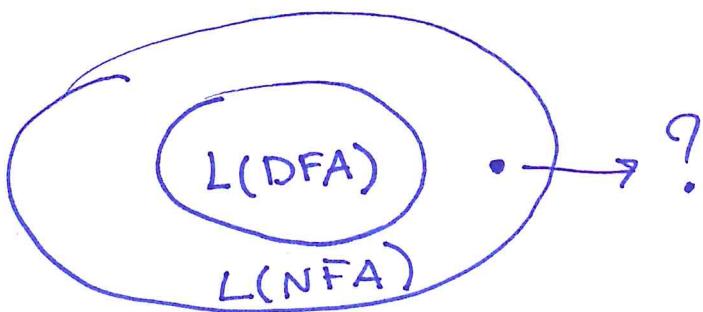
"Il gatto" è rifiutato

"Il gatto della gatta"

$L(\text{DFA})$ = la classe dei linguaggi accettati da DFA

$L(\text{NFA})$ = la classe dei linguaggi accettati da NFA

Vale:



NOTA: I DFA sono un caso particolare di NFA

Risposta alla domanda : NO!

Teorema Per ogni L riconosciuto da un NFA esiste un DFA che lo riconosce (è equivalente)

{ Esiste L t.c. $L \in \text{accettato da un NFA}$, ma non esiste un DFA che lo riconosce }

Corollario:

$$R_3 = L(\text{NFA}) = \underbrace{L(\text{DFA})}$$

dim. del Teorema 2

Dato $A = (\Sigma, Q, q_0, R, F)$ NFA

costruisco A' deterministico.

$A' = (\Sigma, 2^Q, \{q_0\}, S_R, F')$ dove:

↑
l'insieme
dei sottoinsiemi di Q es: $Q = \{q_0, q_1\}$

$F' = \{ Y \in 2^Q \mid Y \cap F \neq \emptyset \}, \quad 2^Q = \dots$

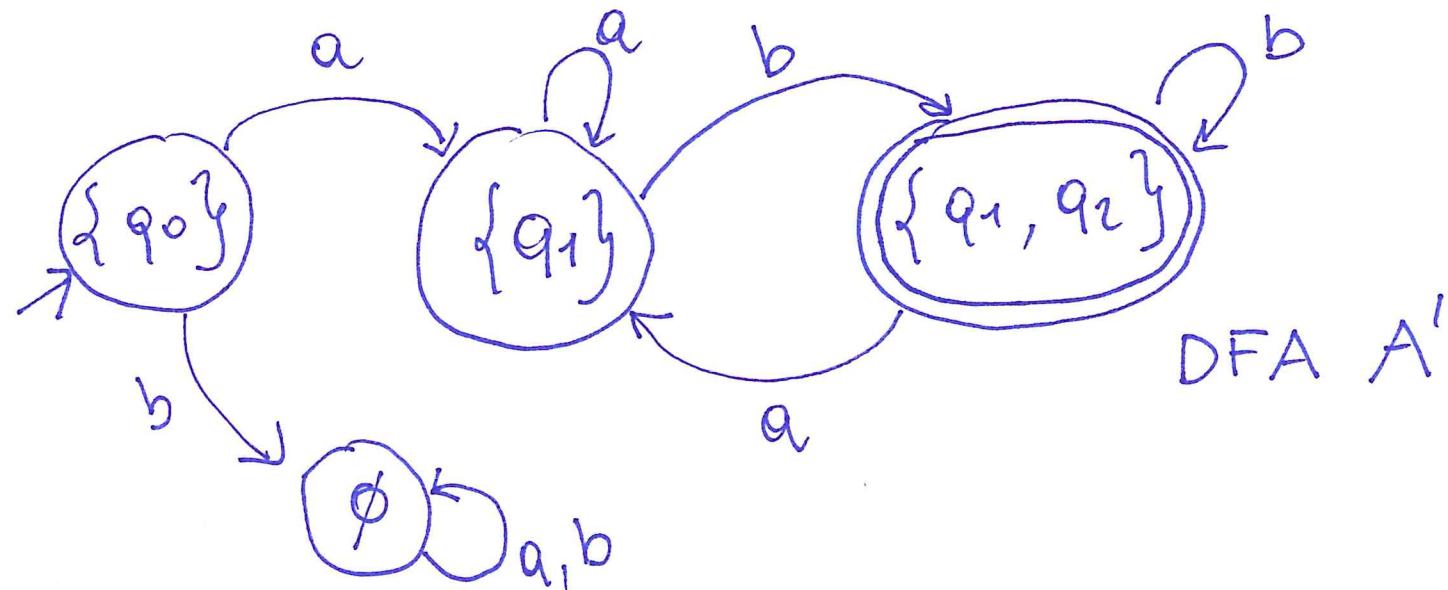
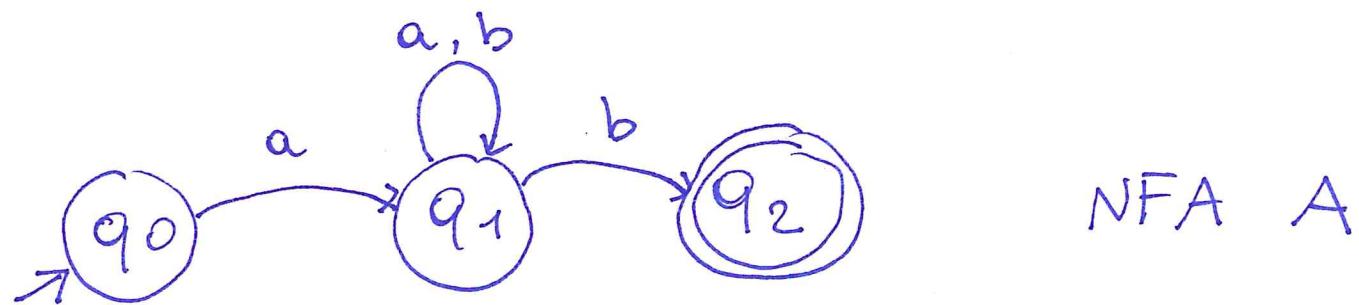
S_R : $2^Q \times \Sigma \rightarrow 2^Q$ che soddisfa

$S_R(X, \sigma) = \bigcup_{q \in X} \left\{ p \in Q \mid R(q, \sigma, p) \right\}$

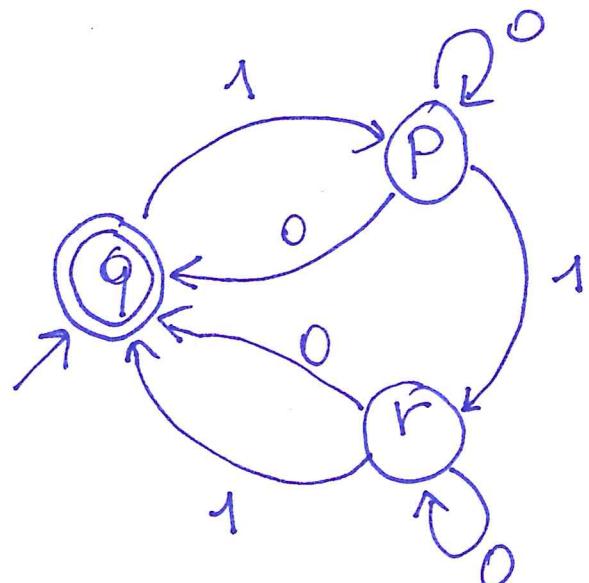
Esempio: ...

$$\delta_R(X, \sigma) = \bigcup_{q \in X} \{ p \in Q \mid R(q, \sigma, p) = 1 \}$$

Esercizio



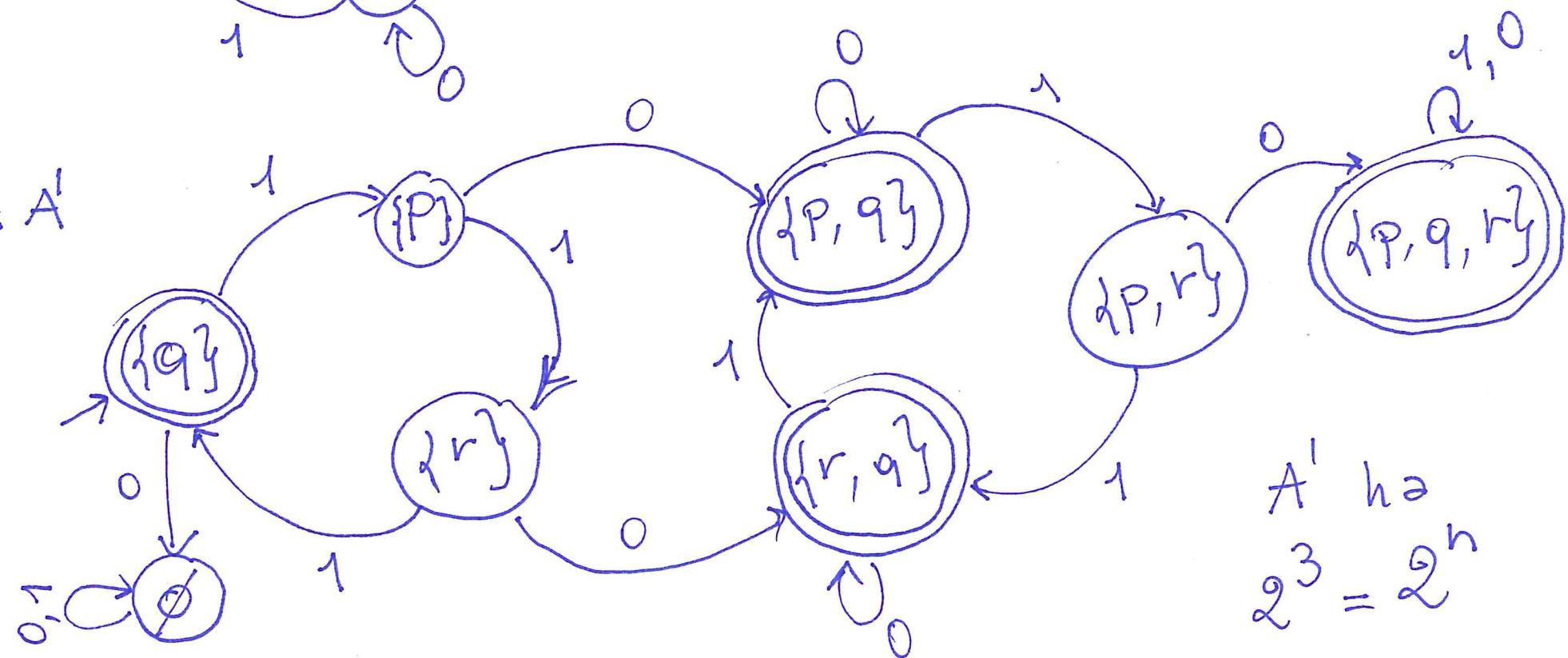
Trasformazione NFA \rightarrow DFA con incremento esponenziale
degli stati



NFA A

$n = 3$

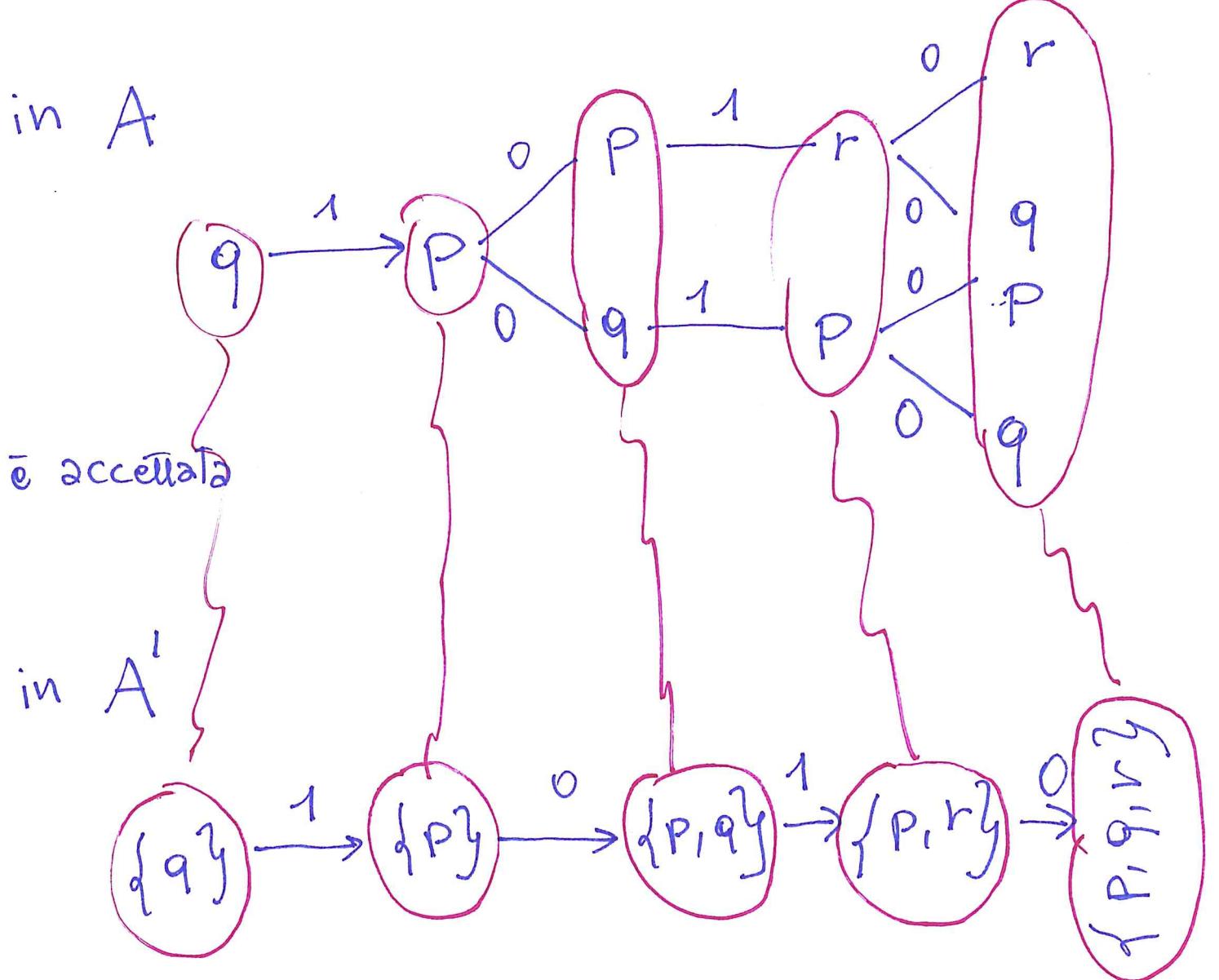
DFA A'



$$A' \text{ ha } 2^3 = 2^n$$

Input : 1010

in A



è accettata

$$\{q, P, r\} \cap F \neq \emptyset$$

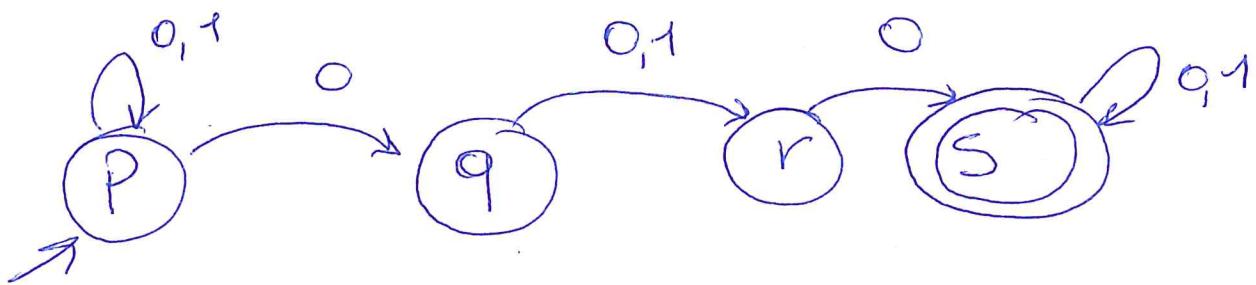
Dato $w \in \Sigma^*$,

indotto da w,

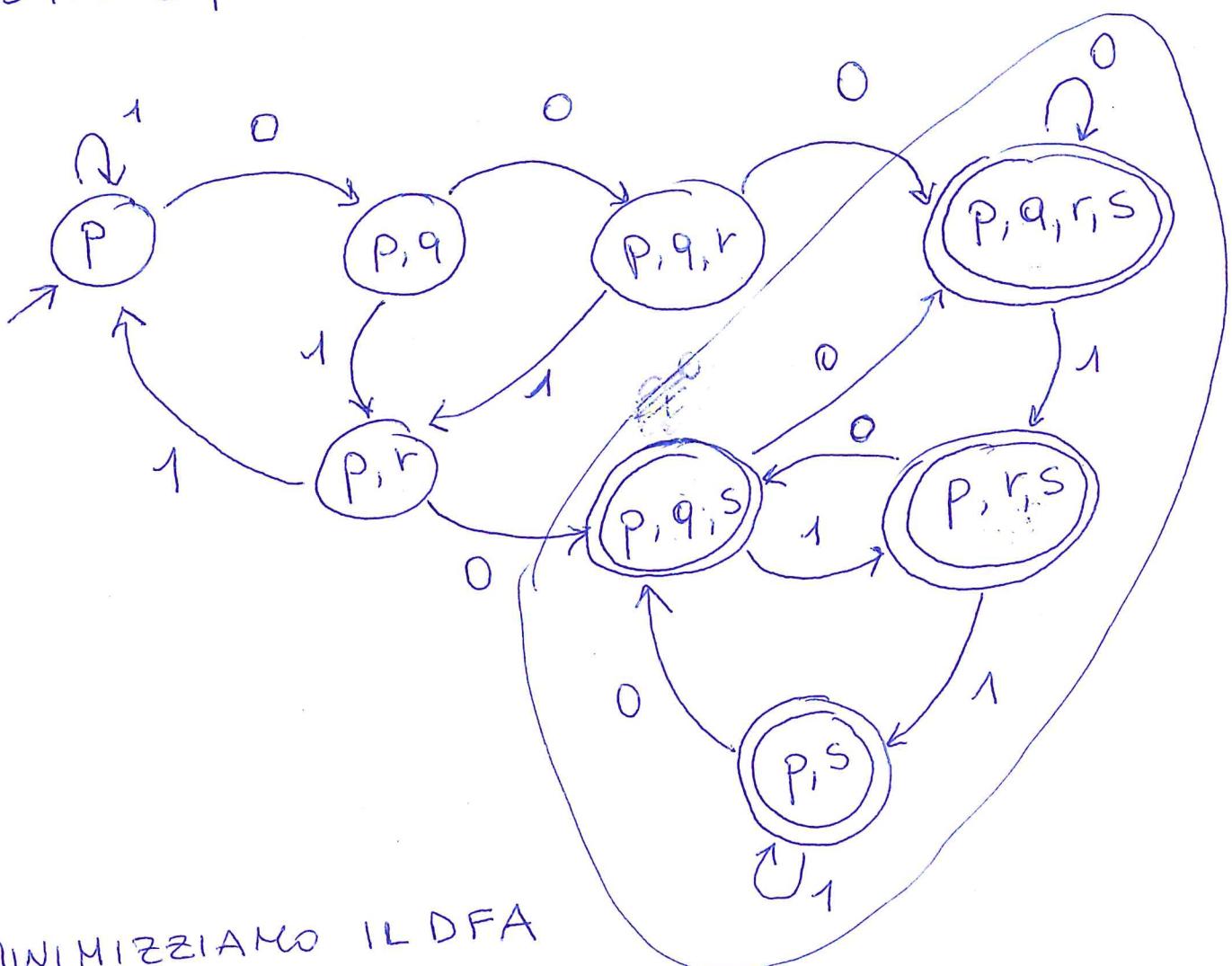
NOTA: Il percorso nel DFA A' simula tutti i cammini possibili di w nell'NFA

ESERCIZIO

NFA:



DFA EQUIVALENTE:



MINIMIZZIAMO IL DFA

$$\{P, q, s\} \approx \{P, q, r, s\}$$

$$\{P, r, s\} \approx \{P, q, r, s\}$$

$$\{P, s\} \approx \{P, q, r, s\}$$



ALTRÒ FORMALISMO per i linguaggi regolari

ESPRESSIONI REGOLARI (ER)

Definizione

Una espressione regolare su Σ è:

- \emptyset
 - ϵ
 - $\sigma \in \Sigma$
- } E.R. base

Se p e q sono espressioni regolari

allora:

- $p + q$
 - $p \cdot q$
 - p^*
- } E.R. complesse

Sono espressioni regolari

Le E.R. denotano dei linguaggi:

E.R.

denota

Linguaggio

\emptyset

\emptyset

ϵ

$\{\epsilon\}$

σ

$\{\sigma\}$

Date le E.R. P, q che denotano L_p e L_q

Allora:

$P + q$

$L_p \cup L_q$

$P \cdot q$

$L_p \cdot L_q$

P^*

L_p^*