PDA sono equivalenti ai linguaggi Context-Free

Teorema:

$$\left\{\begin{array}{c}\text{Context-Free}\\ \text{linguaggi}\\ \text{(grammatiche)}\end{array}\right\} = \left\{\begin{array}{c}\text{linguaggi}\\ \text{accettati da}\\ \text{PDA}\end{array}\right\}$$

dimostrazione - Step 1:

$$\left\{\begin{array}{c}\text{Context-Free}\\ \text{linguaggi}\\ \text{(grammatiche)}\end{array}\right\} \subseteq \left\{\begin{array}{c}\text{linguaggi}\\ \text{accettati by}\\ \text{PDAs}\end{array}\right\}$$

Traduci ogni grammatica context-free G
In un PDA M con: $L(G) = L(M)$

dimostrazione - step 1

trasforma le

grammatiche Context-Free
in
PDAs

Prendiamo una grammatica context-free G

Tradurremo G in un PDA M tale che:

$$L(G) = L(M)$$

Def. : Una derivazione $S \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_{n-1} \Rightarrow \alpha_n$ si dice leftmost (sinistra) se:

$\forall i = 1,\ldots,n-1 \quad si\ ha \quad \alpha_i = uX\beta_i \quad e \quad \alpha_{i+1} = u\gamma\beta_i, \quad con\ u \in \Sigma^*, X \in N, (X \to \gamma)\ in\ P$

[redacted]

Nel primo caso si scrive : $\alpha_i \underset{L}{\Longrightarrow} \alpha_j \quad (i < j)$.

[redacted]

Useremo solo leftmost

Grammatica consideriamo le Derivazioni Leftmost

Terminali

Leftmost variabile

Variabile o terminali

$$\Rightarrow \cdots$$

$$\Rightarrow xAy$$

$$\Rightarrow \sigma_i \cdots \sigma_j Bzy$$

produzione Applicata

$$A \rightarrow \underbrace{\sigma_i \cdots \sigma_j}_{\text{Terminali}} Bz$$

Variabili o Terminali

Variabile

Procedura di conversione:

per ogni produzione in G

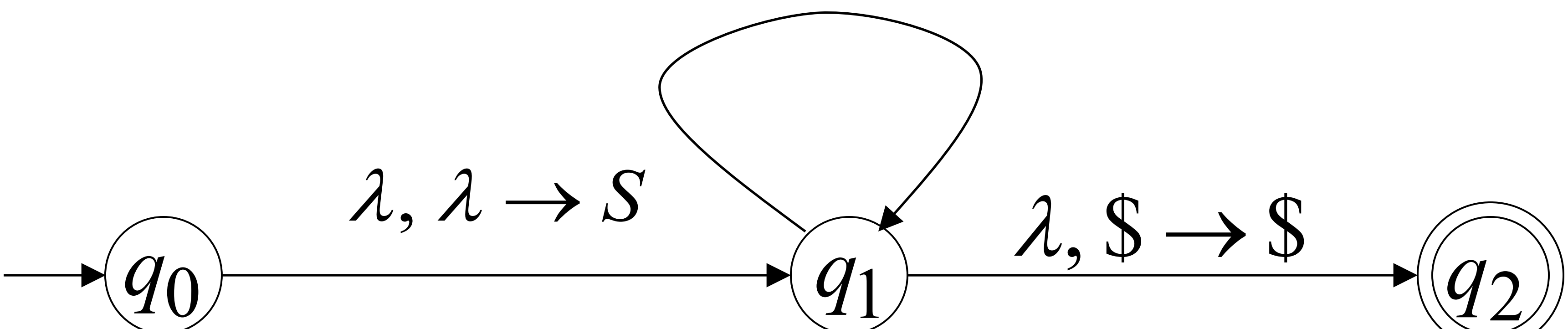
$$A \to w$$

per ogni terminale in G

$$a$$

Addiziona le transizioni

$$\lambda, A \to w \qquad a, a \to \lambda$$



esempio

grammatica

$S \rightarrow aSTb$

$S \rightarrow b$

$T \rightarrow Ta$

$T \rightarrow \lambda$

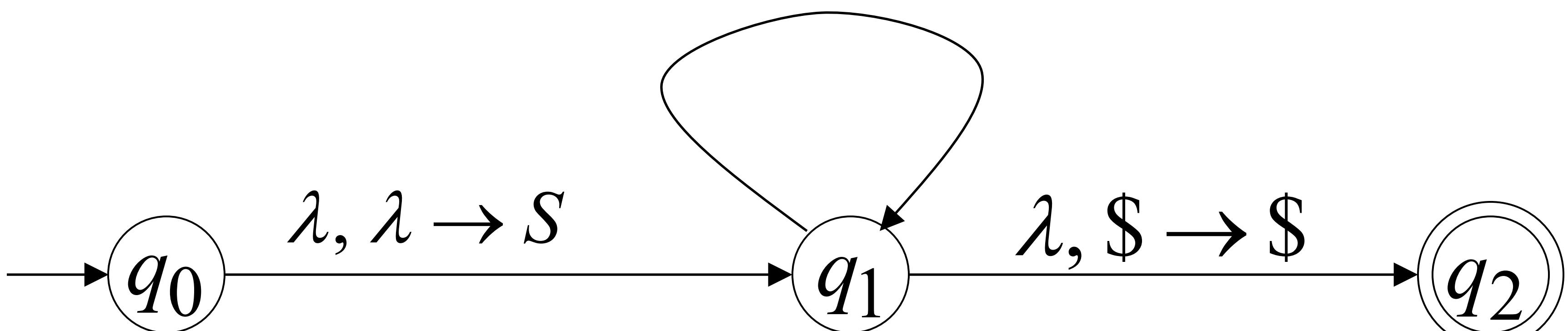
PDA

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta \qquad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda \qquad b, b \rightarrow \lambda$



Esempio:

Input

a	b	a	b

Time 0

$$\lambda, S \to aSTb$$

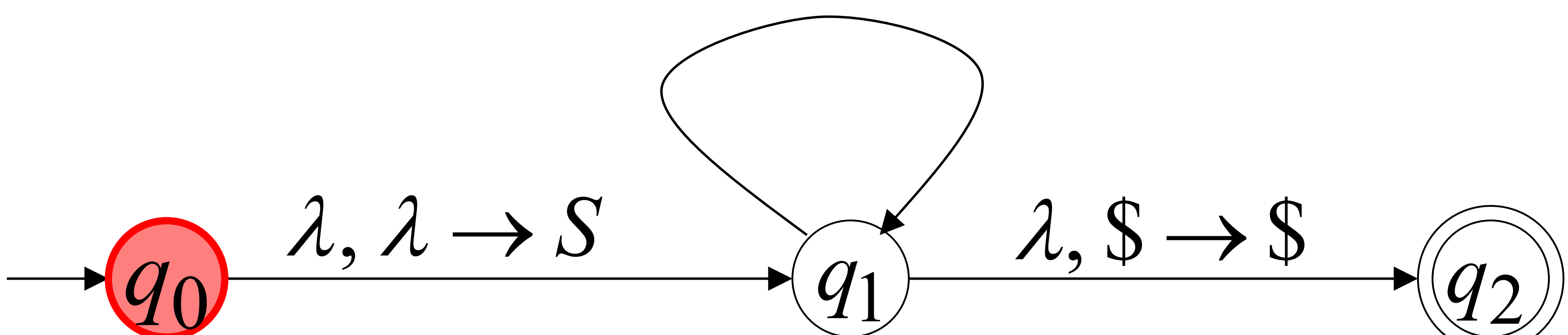
$$\lambda, S \to b$$

$$\lambda, T \to Ta \qquad a, a \to \lambda$$

$$\lambda, T \to \lambda \qquad b, b \to \lambda$$

$\$$

Stack



derivazione: S

Input

a	b	a	b

Time 1

Stack

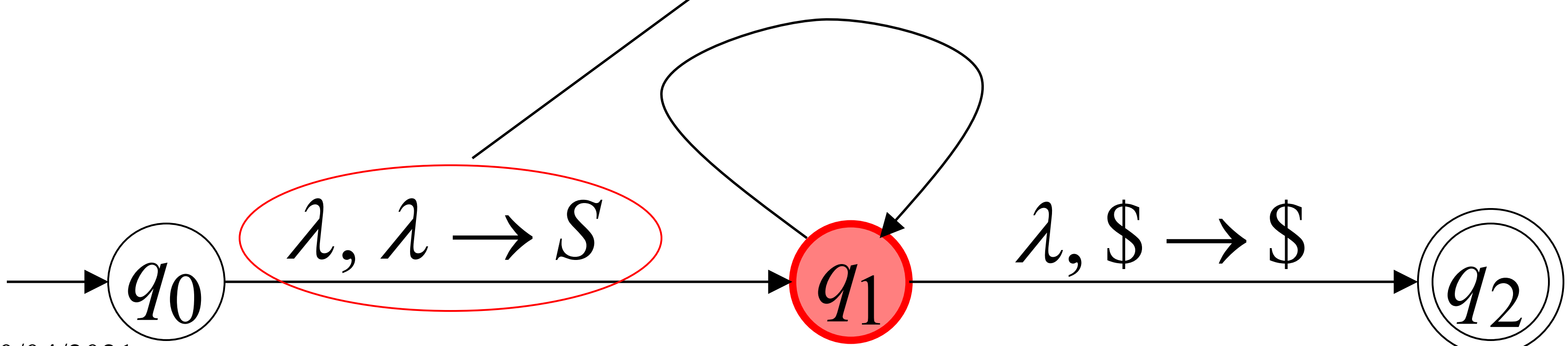
S
$\$$

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\qquad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\qquad b, b \rightarrow \lambda$



derivazione: $S \Rightarrow aSTb$

Input

a	b	a	b

Stack

a
S
T
b
$\$$

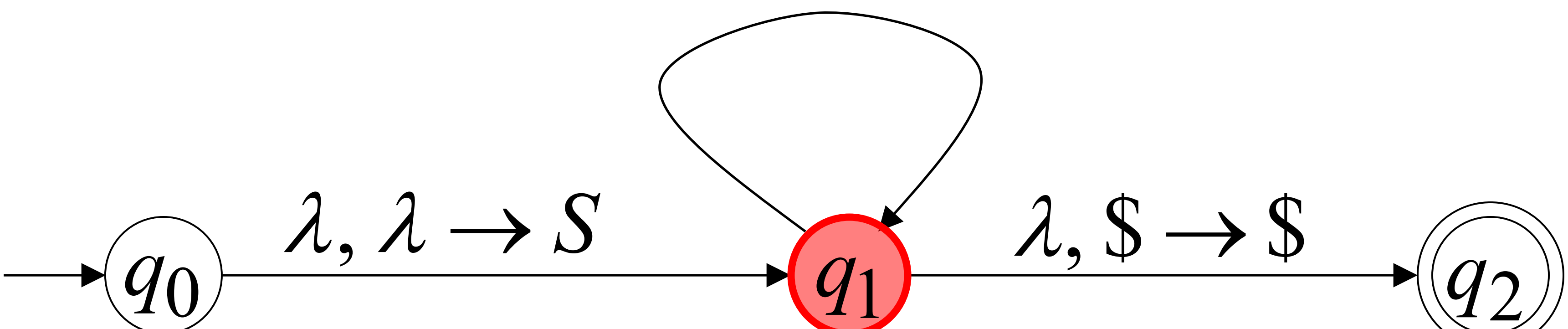
Time 2

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$



derivazione: $S \Rightarrow aSTb$

Input

a	b	a	b

Stack

$\cancel{a}$
S
T
b
$\$$

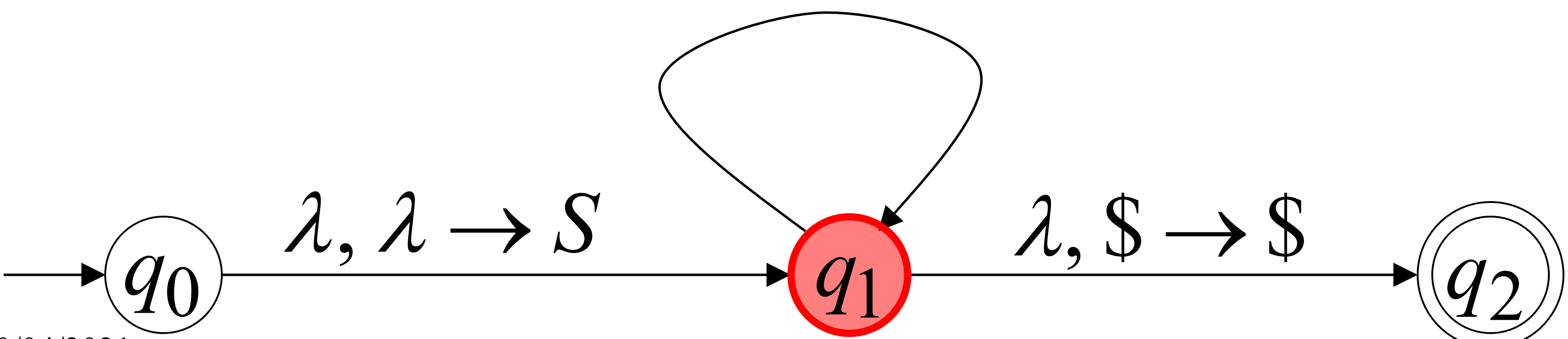
Time 3

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad$ $a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad$ $b, b \rightarrow \lambda$



derivazione: $S \Rightarrow aSTb \Rightarrow abTb$

Input

a	b	a	b

Time 4

$\lambda, S \rightarrow aSTb$

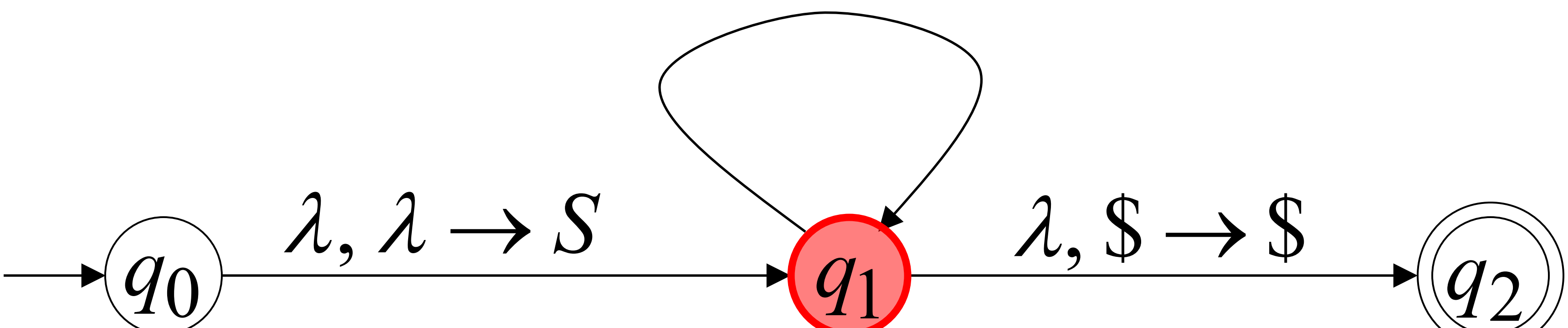
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad$ $a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad$ $b, b \rightarrow \lambda$

b
T
b
$\$$

Stack



derivazione: $S \Rightarrow aSTb \Rightarrow abTb$

Input

a	b	a	b

Time 5

$\lambda, S \to aSTb$

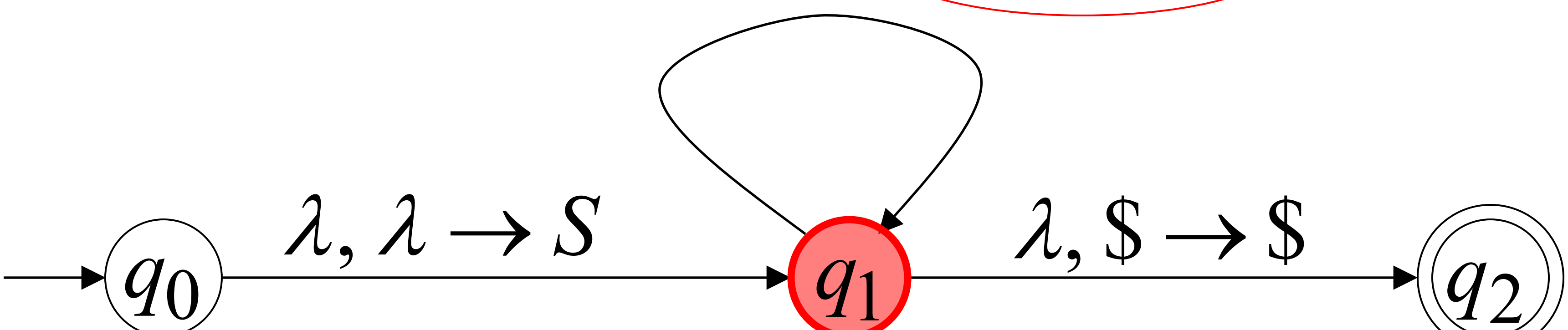
$\lambda, S \to b$

$\lambda, T \to Ta$ $\quad a, a \to \lambda$

$\lambda, T \to \lambda$ $\quad b, b \to \lambda$

$\cancel{b}$
T
b
$\$$

Stack



derivazione: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab$

Input

a	b	a	b

Time 6

$\lambda, S \rightarrow aSTb$

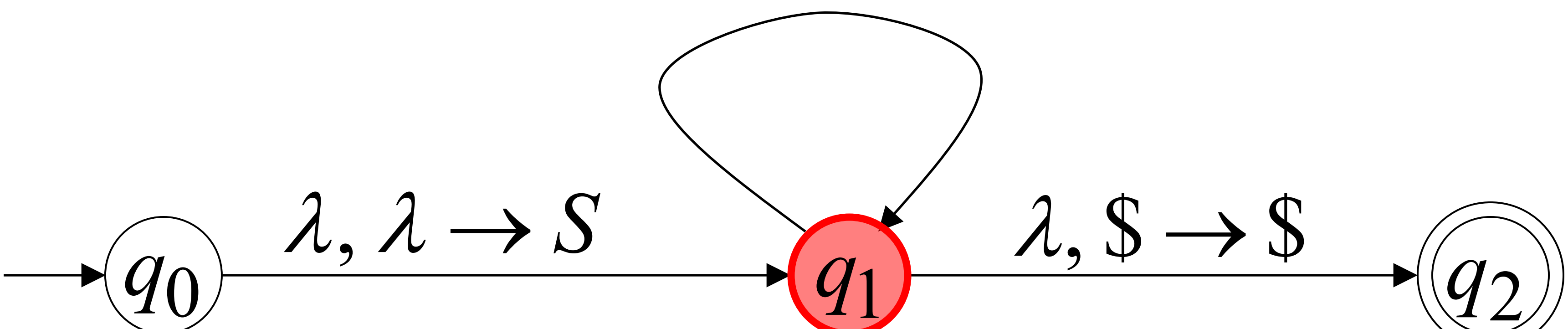
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\qquad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\qquad b, b \rightarrow \lambda$

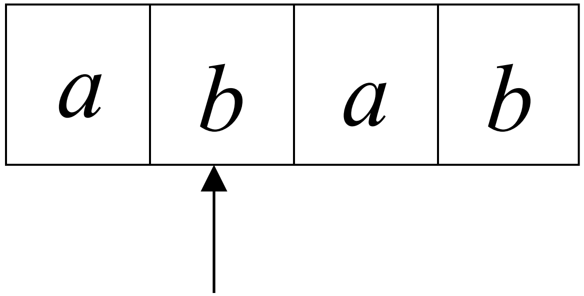
T
a
b
$\$$

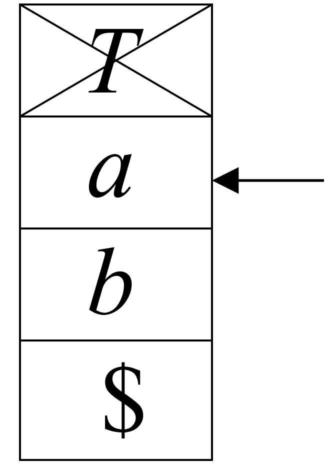
Stack



derivazione: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input





Stack

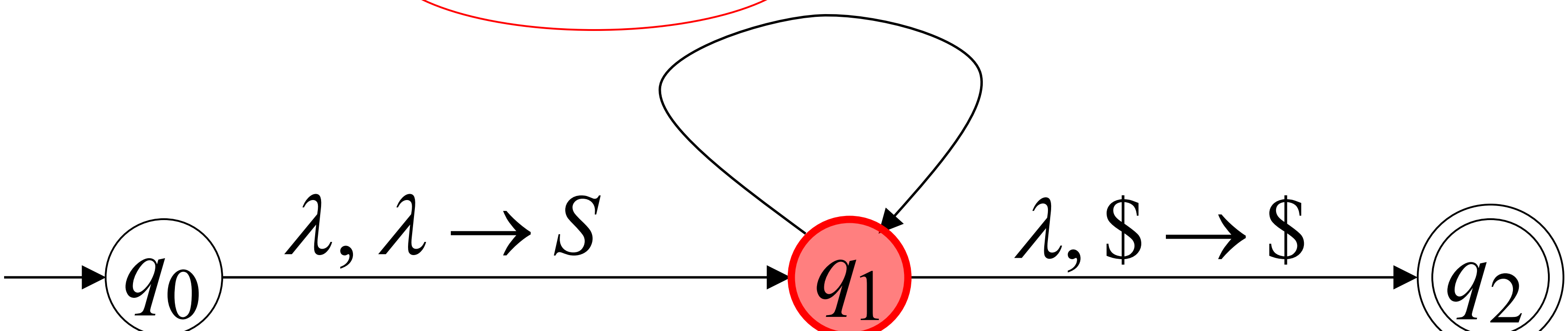
Time 7

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$



derivazione: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input

a	b	a	b

Stack

$\cancel{a}$
b
$\$$

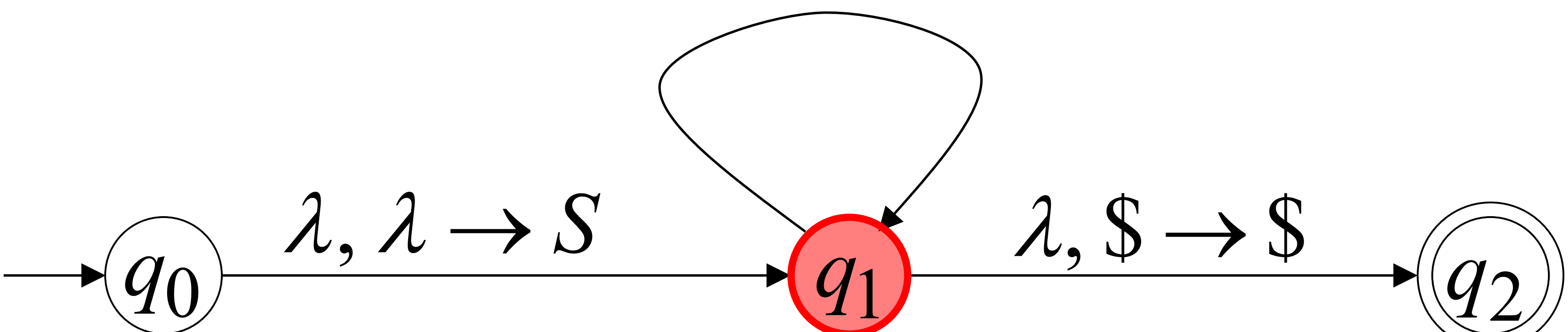
Time 8

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$



derivazione: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input

a	b	a	b

Time 9

$\cancel{b}$

$\$$

Stack

$\lambda, S \rightarrow aSTb$

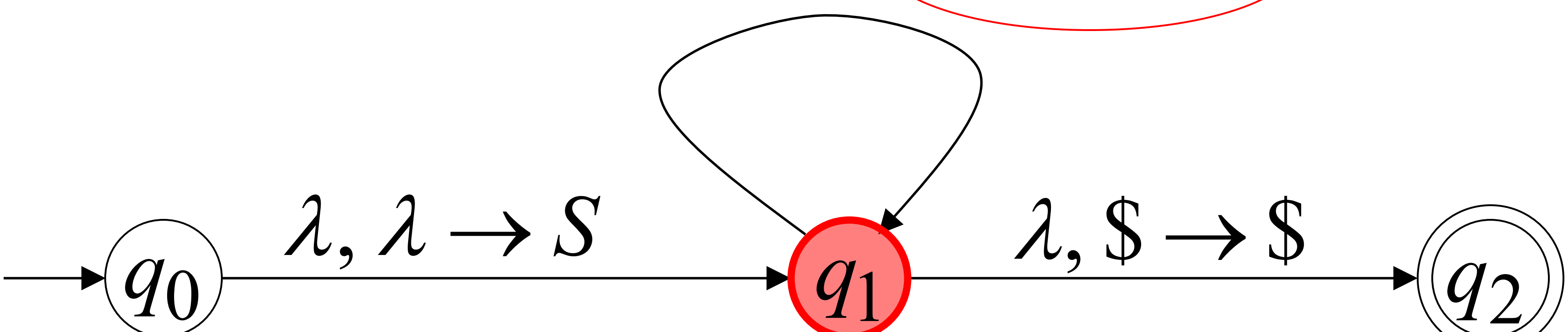
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$

$\lambda, T \rightarrow \lambda$

$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



derivazione: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input

a	b	a	b

Time 10

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

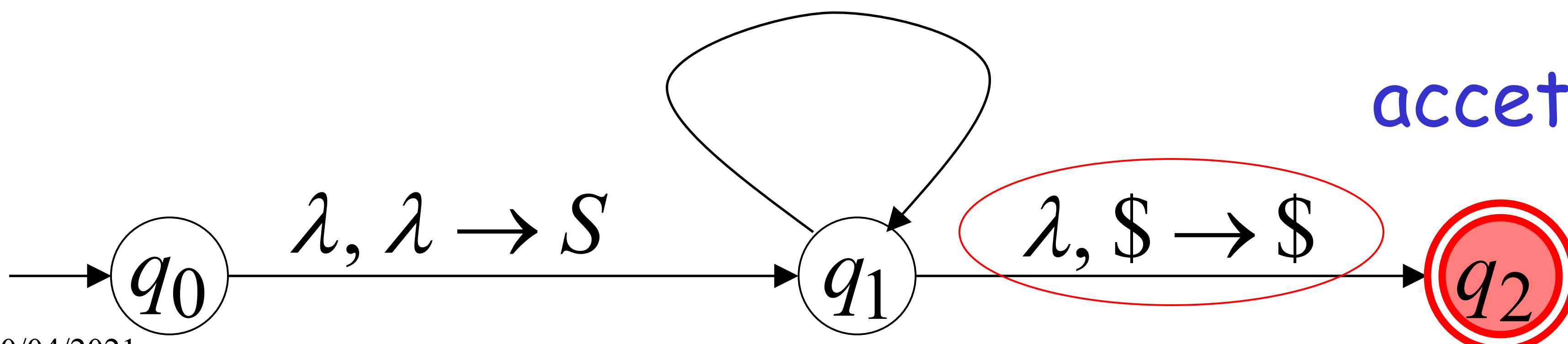
$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$

$\$$

Stack

accettaz



Procedura di conversione:

per ogni produzione in G

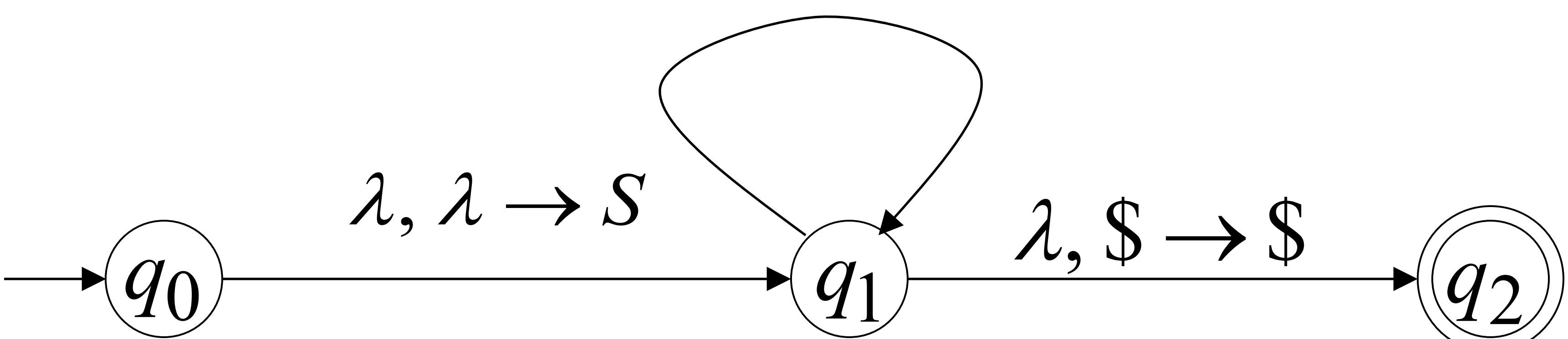
$$A \to w$$

per ogni terminale in G

$$a$$

Addiziona le transizioni

$$\lambda,\ A \to w \qquad a,\ a \to \lambda$$



esempio

grammatica

$S \to aSTb$

$S \to b$

$T \to Ta$

$T \to \lambda$

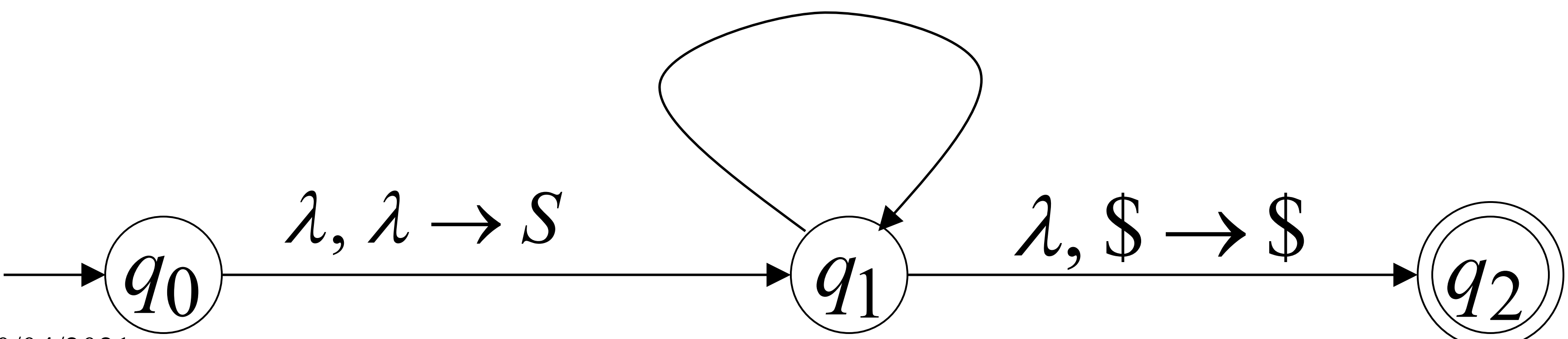
PDA

$\lambda, S \to aSTb$

$\lambda, S \to b$

$\lambda, T \to Ta$ $\quad a, a \to \lambda$

$\lambda, T \to \lambda$ $\quad b, b \to \lambda$



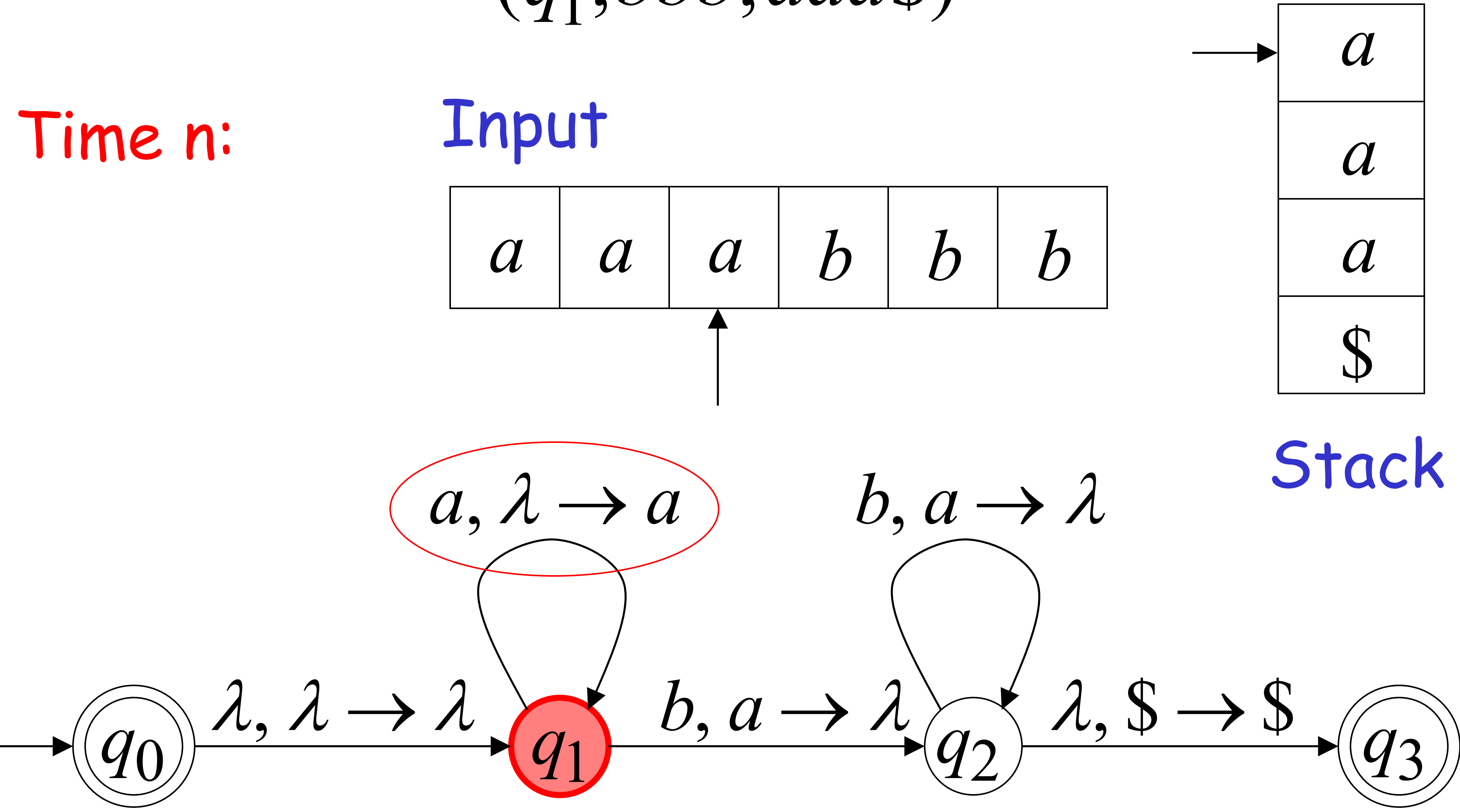
Proviamo a dimostrare il teorema

Ricordiamo la notazione di configurazione istantanea e di passo di computazione.

configurazione istantanea definizione:

$$(q_1, bbb, aaa\$)$$

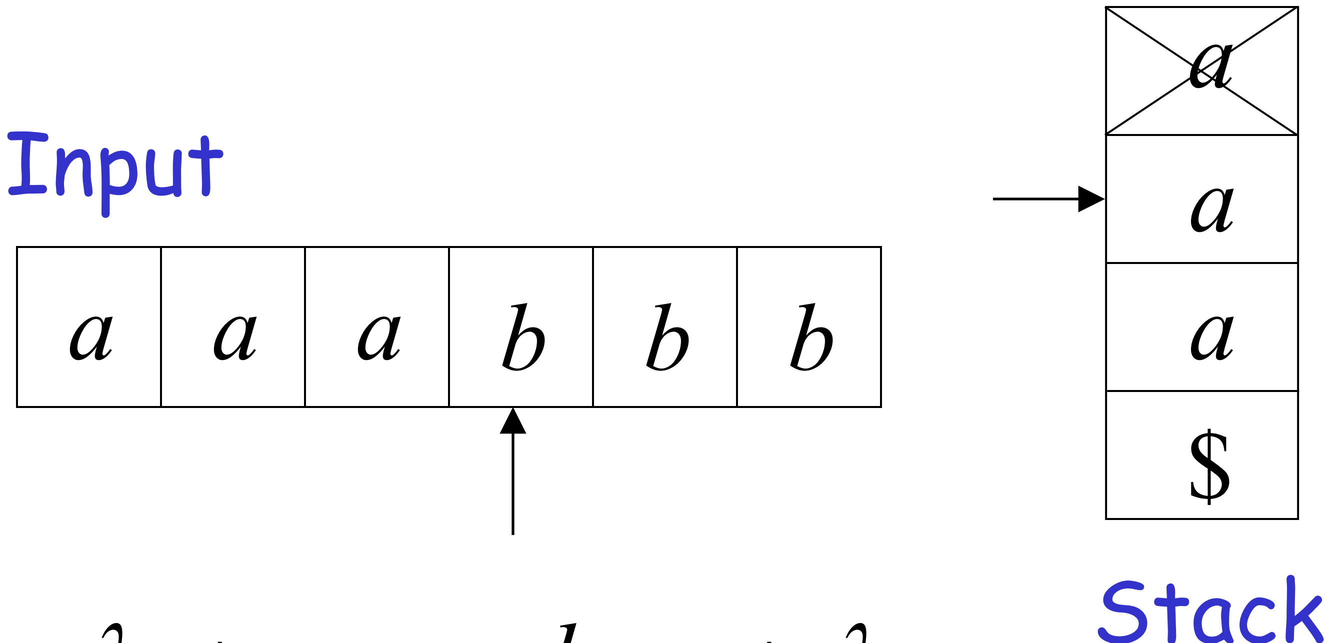
Time n:

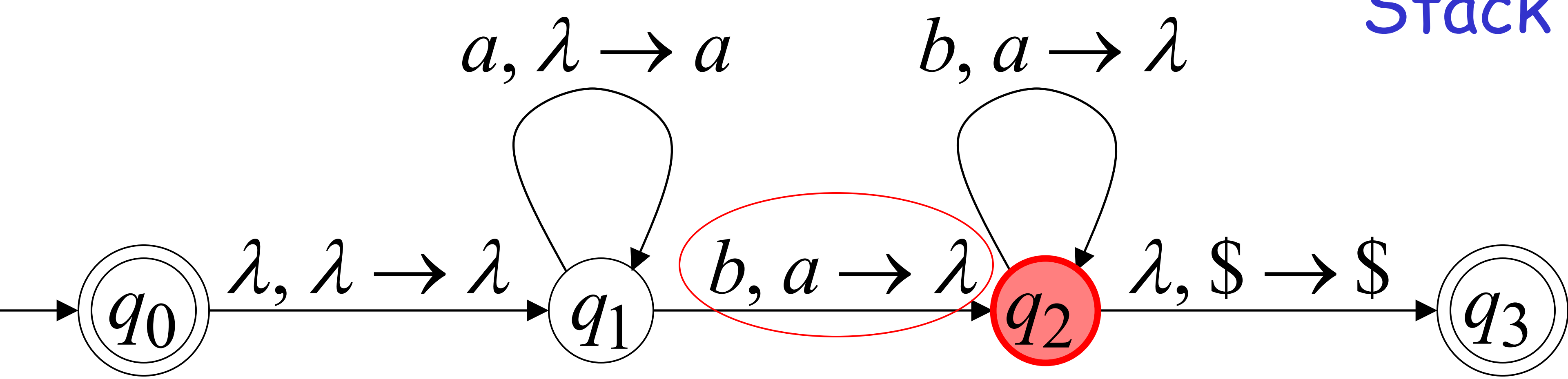


configurazione istantanea

$$(q_2, bb, aa\$)$$

Time n+1:





passo di computazione.

$$(q_1, bbb, aaa\$) \succ (q_2, bb, aa\$)$$

Time n

Time n+1

PDA simula le derivazioni leftmost

grammatica
Leftmost derivazione

$$S$$
$$\Rightarrow \cdots$$
$$\Rightarrow \boxed{\sigma_1 \cdots \sigma_k} \boxed{X_1 \cdots X_m}$$
$$\Rightarrow \cdots$$
$$\Rightarrow \sigma_1 \cdots \sigma_k \sigma_{k+1} \cdots \sigma_n$$

PDA calcolo

$$(q_0, \sigma_1 \cdots \sigma_k \sigma_{k+1} \cdots \sigma_n, \$)$$
$$\succ (q_1, \boxed{\sigma_1 \cdots \sigma_k} \sigma_{k+1} \cdots \sigma_n, S\$)$$
$$\succ \cdots$$
$$\succ (q_1, \sigma_{k+1} \cdots \sigma_n, \boxed{X_1 \cdots X_m} \$)$$
$$\succ \cdots$$
$$\succ (q_2, \lambda, \$)$$

simboli
esaminati

Contenuti
Dello Stack

Procedura di conversione:

per ogni produzione in G

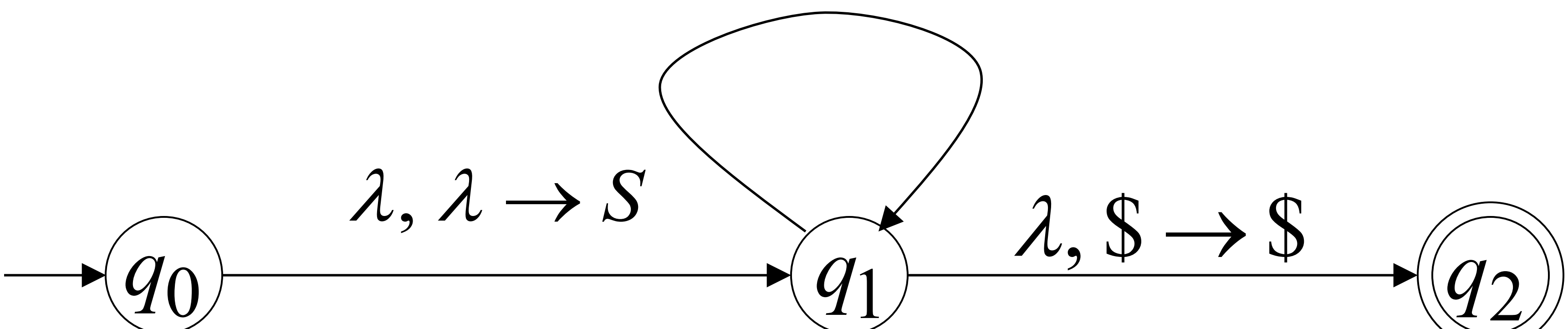
$$A \rightarrow w$$

per ogni terminale in G

$$a$$

Addiziona le transizioni

$$\lambda, A \rightarrow w \qquad a, a \rightarrow \lambda$$



esempio

grammatica

$S \rightarrow aSTb$

$S \rightarrow b$

$T \rightarrow Ta$

$T \rightarrow \lambda$

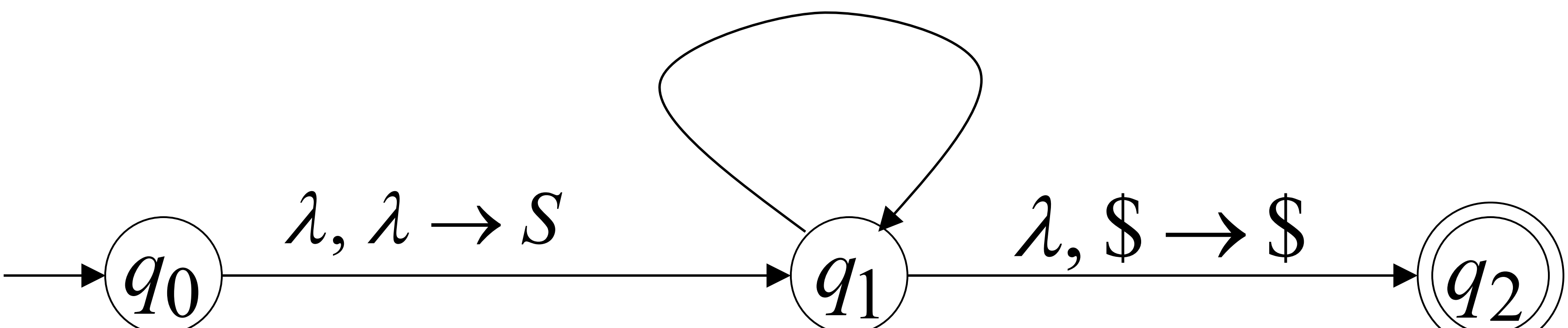
PDA

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\qquad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\qquad b, b \rightarrow \lambda$



grammatica

Derivazione Leftmost

Calcolo PDA

$$S$$

$$\Rightarrow aSTb$$

$$\Rightarrow abTb$$

$$\Rightarrow abTab$$

$$\Rightarrow abab$$

$$(q_0, abab, \$)$$

$$\succ (q_1, abab, S\$)$$

$$\succ (q_1, bab, STb\$)$$

$$\succ (q_1, bab, bTb\$)$$

$$\succ (q_1, ab, Tb\$)$$

$$\succ (q_1, ab, Tab\$)$$

$$\succ (q_1, ab, ab\$)$$

$$\succ (q_1, b, b\$)$$

$$\succ (q_1, \lambda, \$)$$

$$\succ (q_2, \lambda, \$)$$

grammatica
Leftmost derivazione

calcolo del PDA

$\Rightarrow \cdots$

$\Rightarrow xAy$

$\Rightarrow \sigma_i \cdots \sigma_j Bzy$

$\succ \cdots$

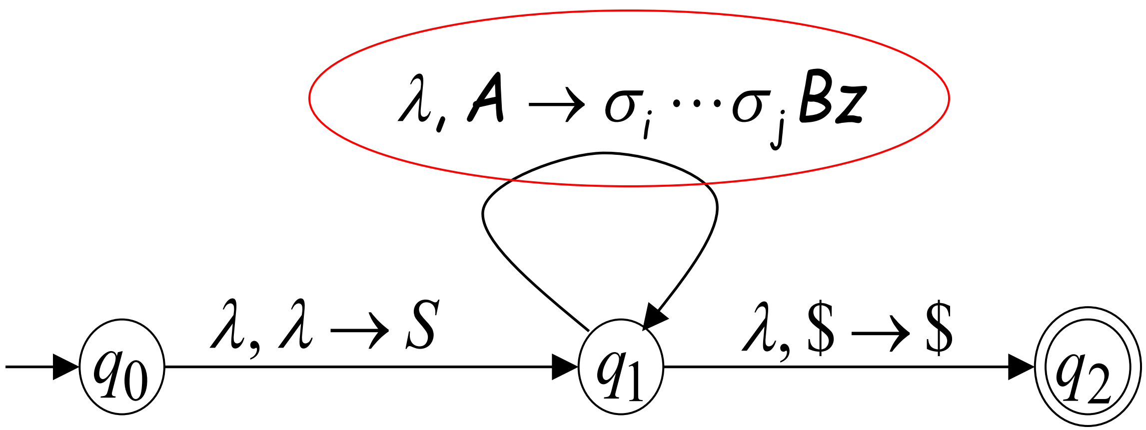
$\succ (q_1, \sigma_i \cdots \sigma_n, Ay\$)$

$\succ (q_1, \sigma_i \cdots \sigma_n, \sigma_i \cdots \sigma_j Bzy\$)$

Produzione Applicata

$A \to \sigma_i \cdots \sigma_j Bz$

Transizione applicata



grammatica
Leftmost derivazione

calcolo del PDA

$$\Rightarrow \cdots$$

$$\Rightarrow xAy$$

$$\Rightarrow \sigma_i \cdots \sigma_j Bzy$$

$$\succ \cdots$$

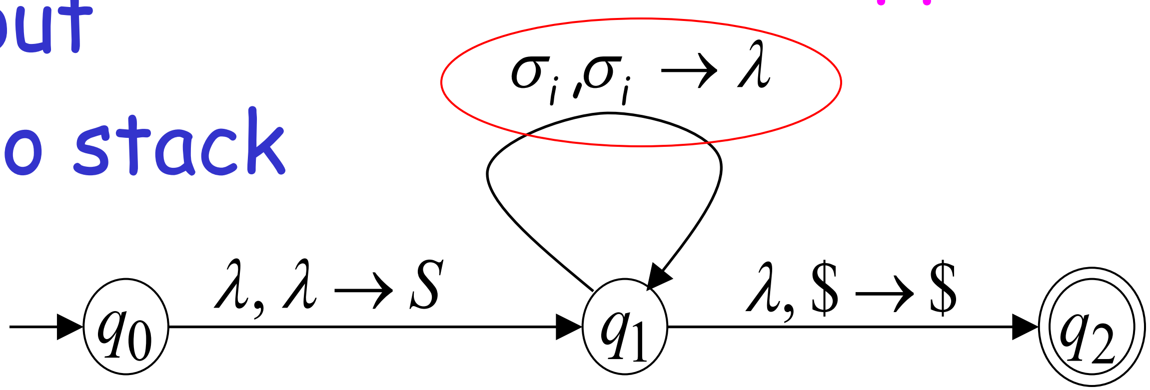
$$\succ (q_1, \sigma_i \cdots \sigma_n, Ay\$)$$

$$\succ (q_1, \sigma_i \cdots \sigma_n, \sigma_i \cdots \sigma_j Bzy\$)$$

$$\succ (q_1, \sigma_{i+1} \cdots \sigma_n, \sigma_{i+1} \cdots \sigma_j Bzy\$)$$

Transizione applicata

leggi σ_i dall'input
E rimuovilo dallo stack



grammatica

Leftmost derivazione

$\Rightarrow \cdots$

$\Rightarrow xAy$

$\Rightarrow \sigma_i \cdots \sigma_j Bzy$

PDA calcolo

$\succ \cdots$

$\succ (q_1, \sigma_i \cdots \sigma_n, Ay\$)$

$\succ (q_1, \sigma_i \cdots \sigma_n, \sigma_i \cdots \sigma_j Bzy\$)$

$\succ (q_1, \sigma_{i+1} \cdots \sigma_n, \sigma_{i+1} \cdots \sigma_j Bzy\$)$

$\succ \cdots$

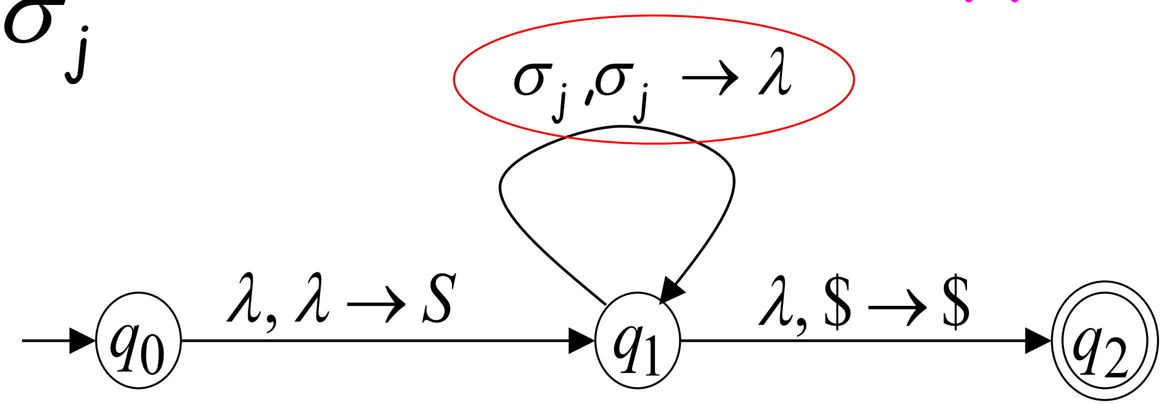
$\succ (q_1, \sigma_{j+1} \cdots \sigma_n, Bzy\$)$

ultima Transizione applicat

tutti simboli $\sigma_i \cdots \sigma_j$

Sono stati rimossi

Dal top dello stack



Il processo viene ripetuto con successiva variabile leftmost

$$\Rightarrow \cdots$$
$$\Rightarrow xAy$$
$$\Rightarrow \sigma_i \cdots \sigma_j Bzy$$
$$\Rightarrow \sigma_i \cdots \sigma_j \sigma_{j+1} \cdots \sigma_k Cpzy$$

$$\succ \cdots$$
$$\succ (q_1, \sigma_{j+1} \cdots \sigma_n, Bzy\$)$$
$$\succ (q_1, \sigma_{j+1} \cdots \sigma_n, \sigma_{j+1} \cdots \sigma_k Cpzy\$)$$
$$\succ \cdots$$
$$\succ (q_1, \sigma_{k+1} \cdots \sigma_n, Cpzy\$)$$

Produzione applicata

$$B \rightarrow \sigma_{j+1} \cdots \sigma_k Cp$$

E cosi via

dimostrato che:

grammatica G
Genera la
stringa w

$$S \overset{*}{\Rightarrow} w$$

Se e solo se

PDA M
accetta w

$$(q_0, w, \$) \succ (q_2, \lambda, \$)$$

quindi $L(G) = L(M)$

Proof - step 2

Tradurre
i PDA
in
grammatiche Context-Free

Prendi un qualsiasi PDA M

Traduremmo M

In una grammatica G context-free

tale che: $L(M) = L(G)$

Prima di tutto modifica PDA M tale che:

1. PDA ha un solo stato di accettazione

2. Svuota tutta la pila prima di accettare

3. Per ogni transizione o

push un simbolo

oppure

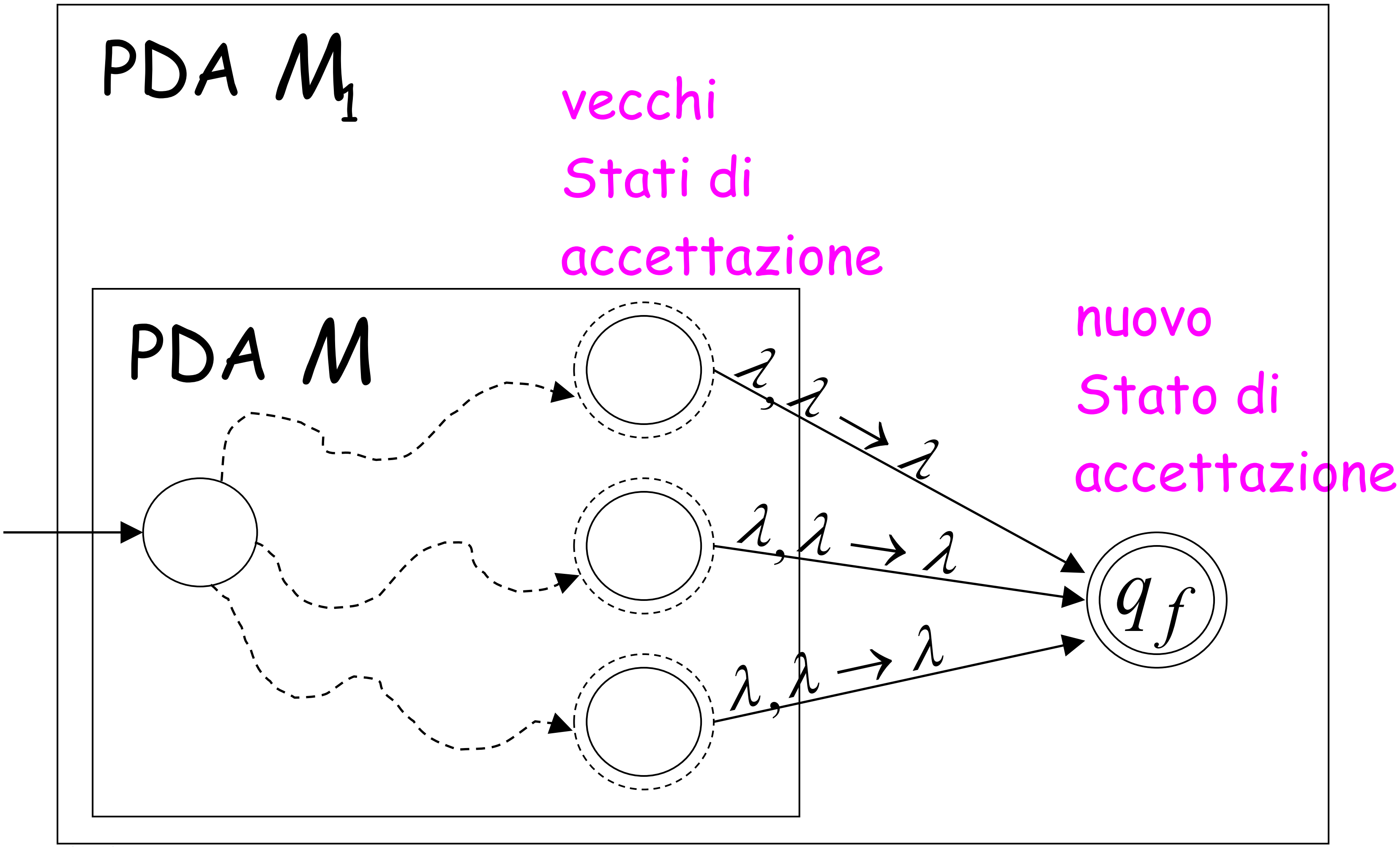
pop un simbolo

ma non entrambe le cose

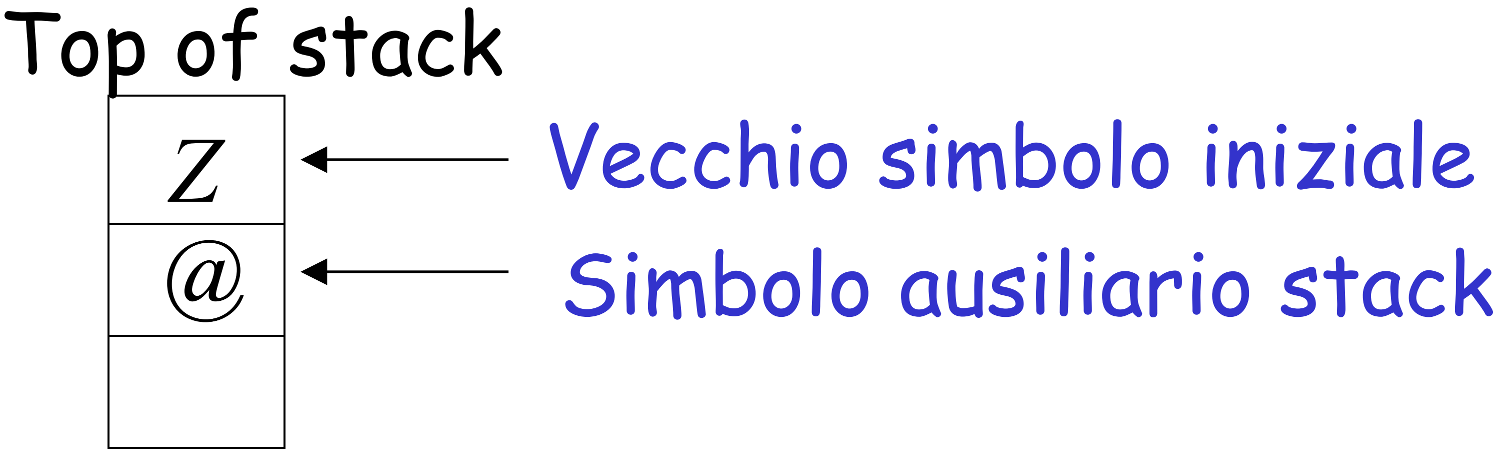
Inoltre abbiamo:

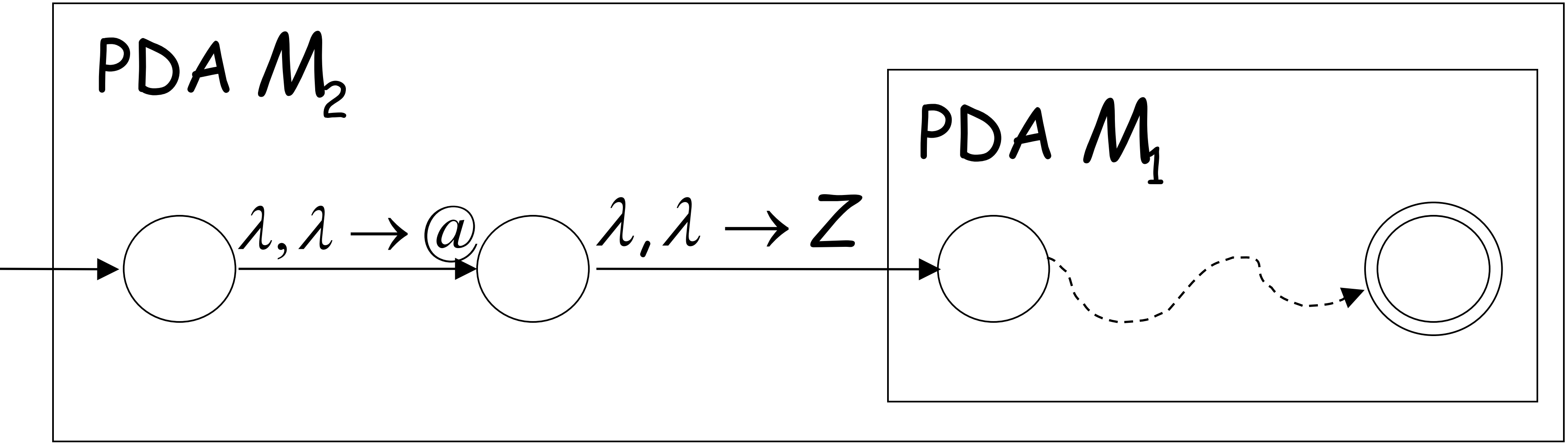
- \- nuovo simbolo iniziale @
- \- stato di accettazione nello stack solo @

1. il PDA deve avere un solo stato di accettazione



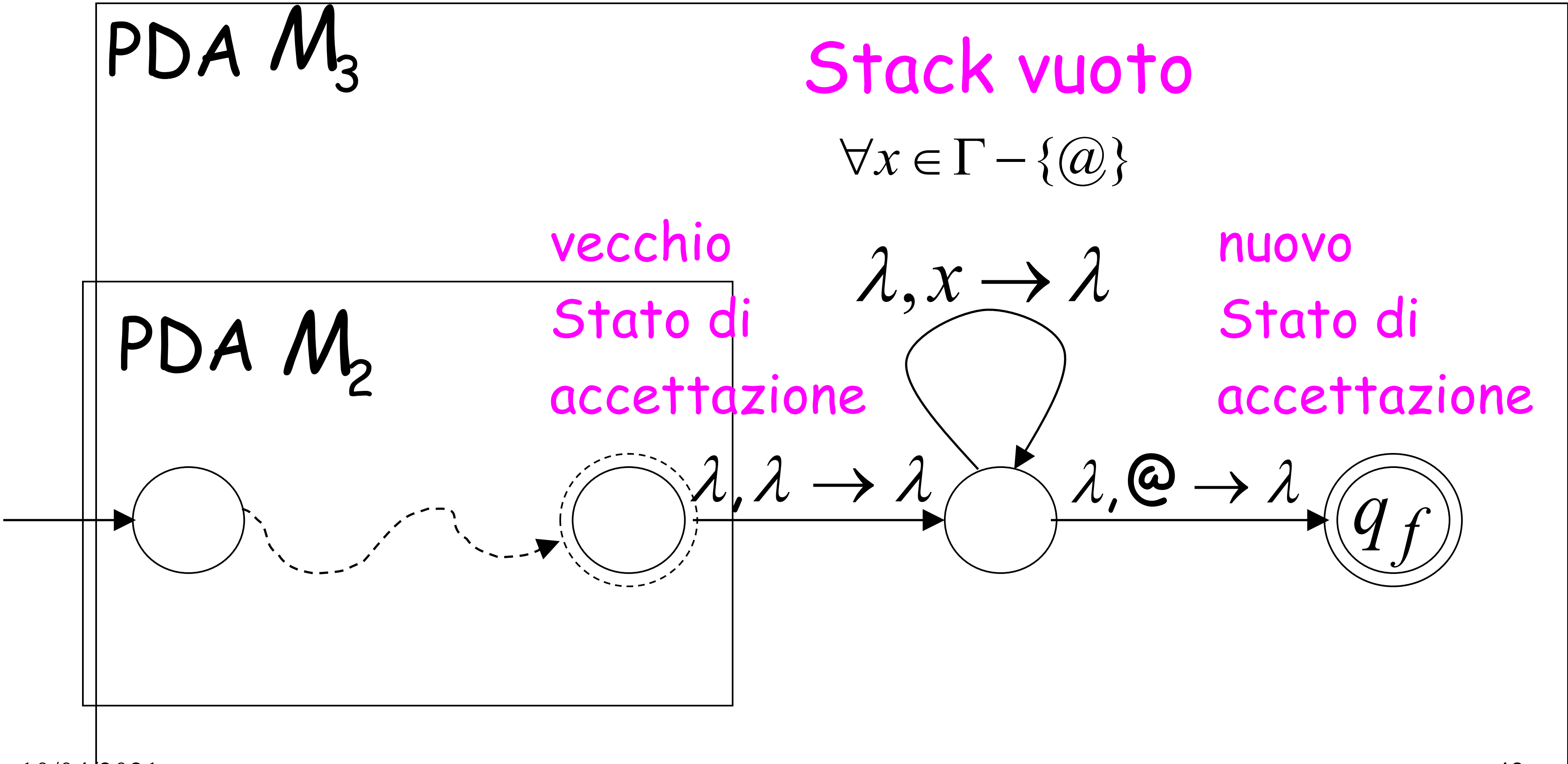
2. Nuovo simbolo iniziale dello stack





M_1 pensa ancora che Z è il simbolo iniziale

3. Nello stato di accetazione lo stack contiene solo il simbolo @

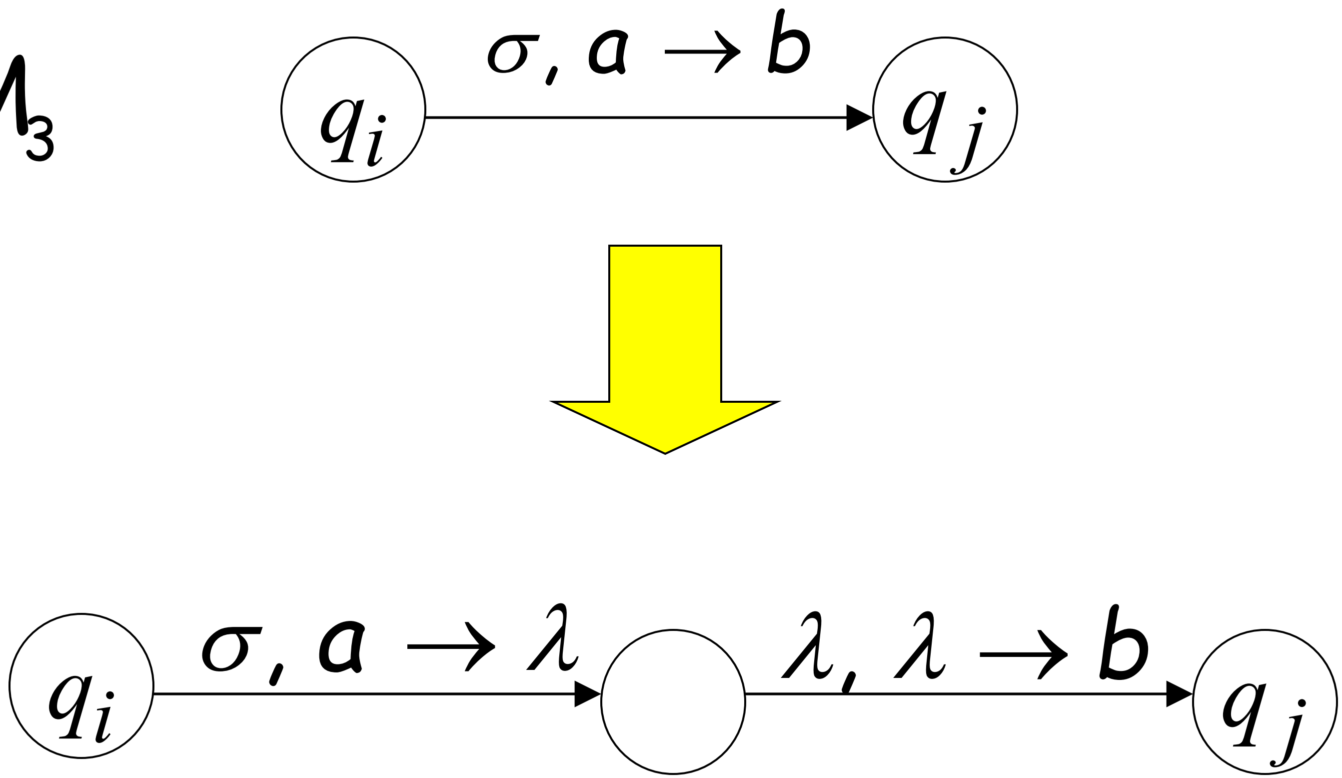


4. Ogni transizione ha un push oppure un pop

Mai le due cose insieme.

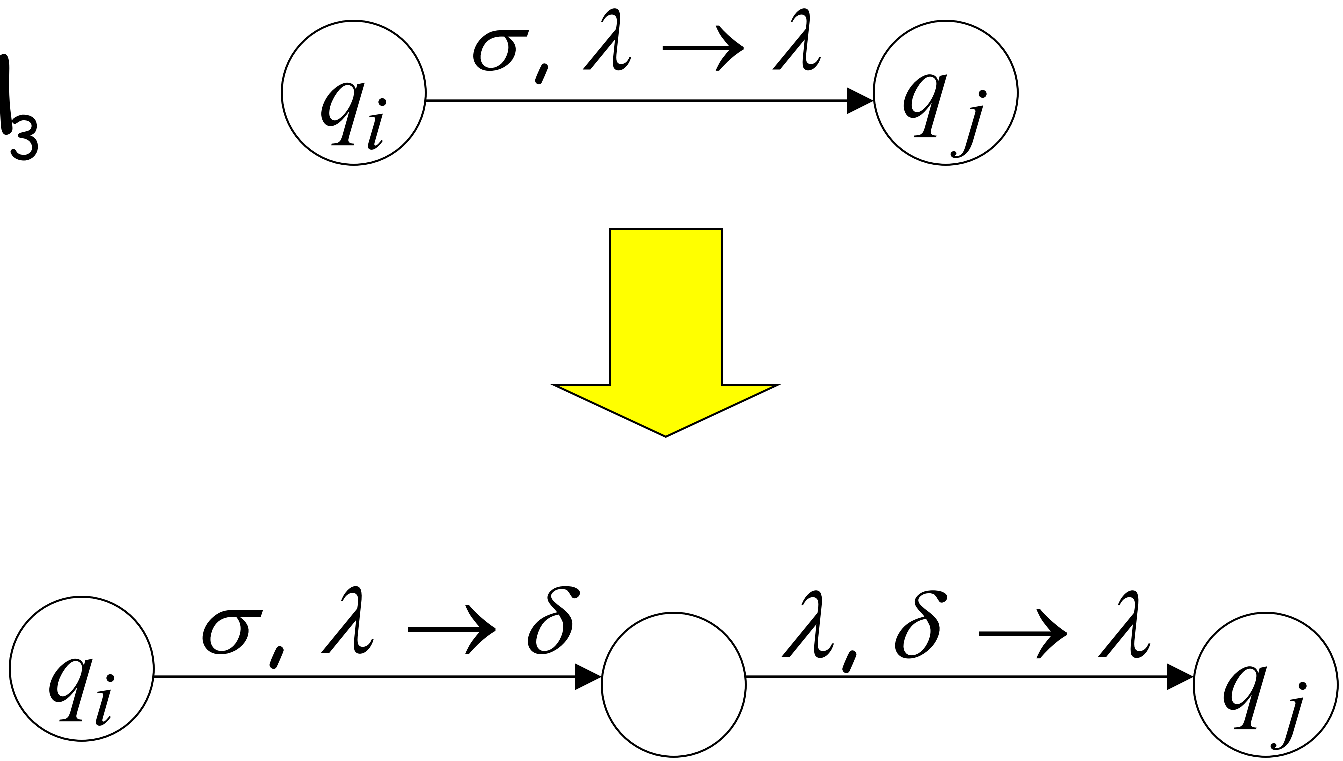
PDA M_3

PDA M_4



PDA M_3

PDA M_4

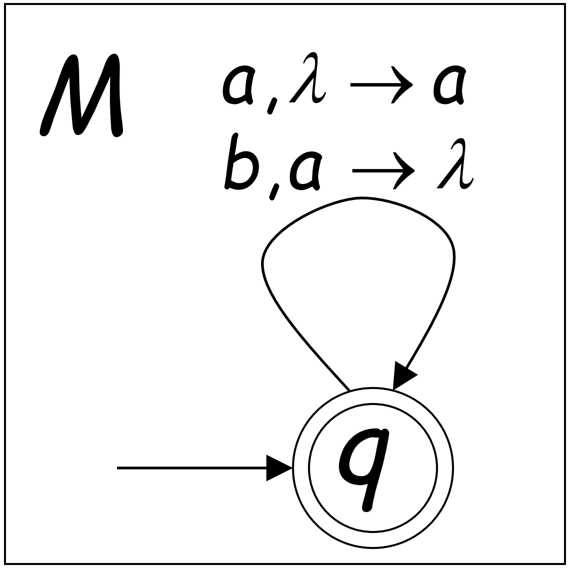


dove δ è un qualsiasi simbolo dell'alfabeto di input

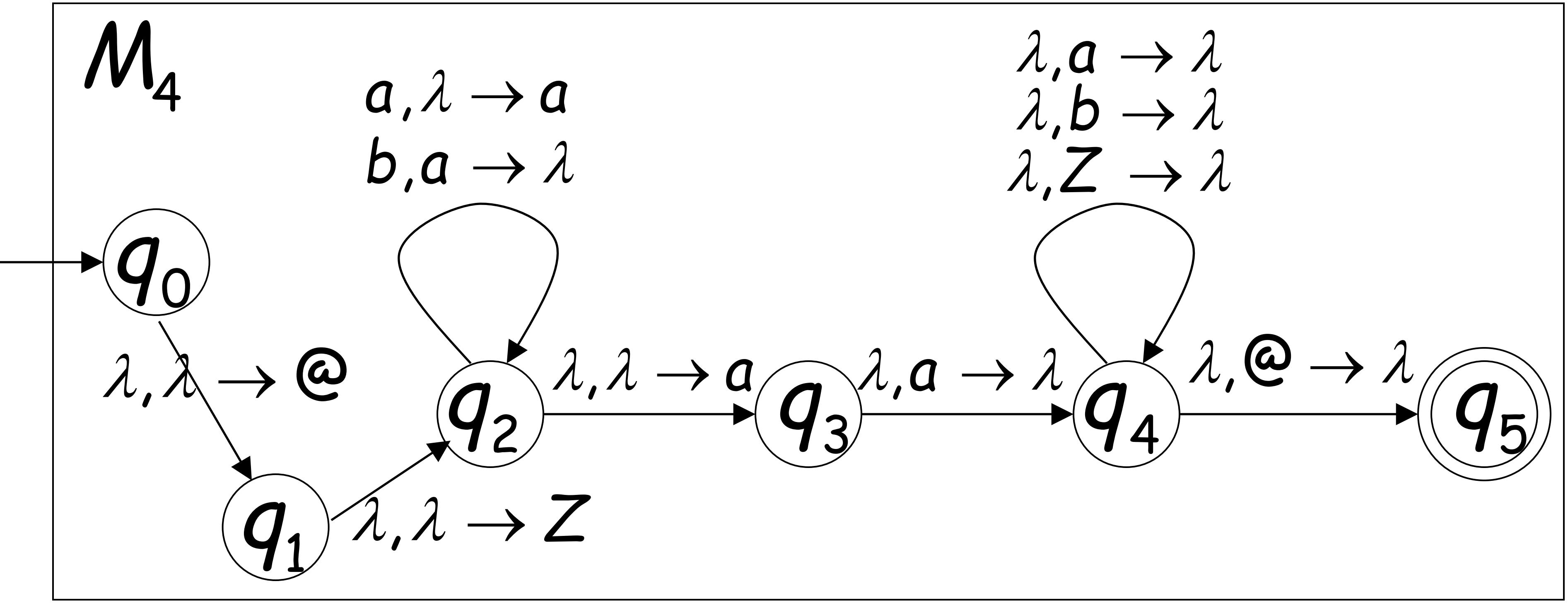
PDA M_4 è il PDA completamente modificato secondo le regole precedenti

Notiamo che il nuovo simbolo iniziale @ non è mai usato in nessuna transizione

Esempio:



M
$a, \lambda \to a$
$b, a \to \lambda$
q



M_4
q_0
$\lambda, \lambda \to @$
q_1
$\lambda, \lambda \to Z$
$a, \lambda \to a$
$b, a \to \lambda$
q_2
$\lambda, \lambda \to a$
q_3
$\lambda, a \to \lambda$
$\lambda, a \to \lambda$
$\lambda, b \to \lambda$
$\lambda, Z \to \lambda$
q_4
$\lambda, @ \to \lambda$
q_5

Costruzione della grammatica

variabili: A_{q_i, q_j}

stati del PDA

PDA

caso 1: per ogni stato

q

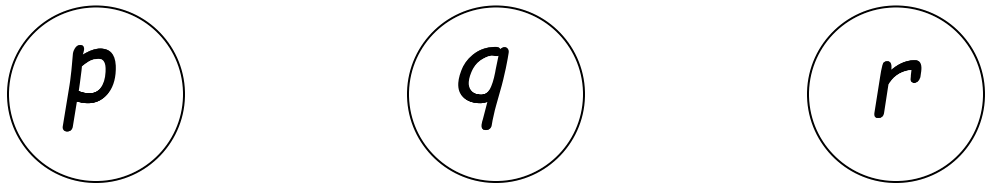
---

grammatica

$$A_{qq} \to \lambda$$

PDA

caso 2: per ogni tre stati



grammatica

$$A_{pq} \rightarrow A_{pr} A_{rq}$$

PDA

caso 3: per ogni coppia di transizioni

$$p \xrightarrow{a,\lambda \to t} r \qquad\qquad s \xrightarrow{b,t \to \lambda} q$$

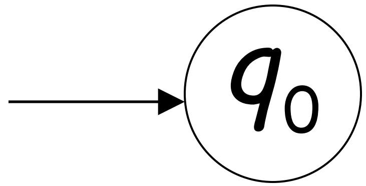
---

grammatica

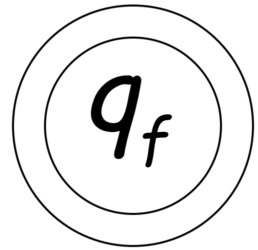
$$A_{pq} \to a A_{rs} b$$

PDA

Stato Iniziale



Stato accettazione



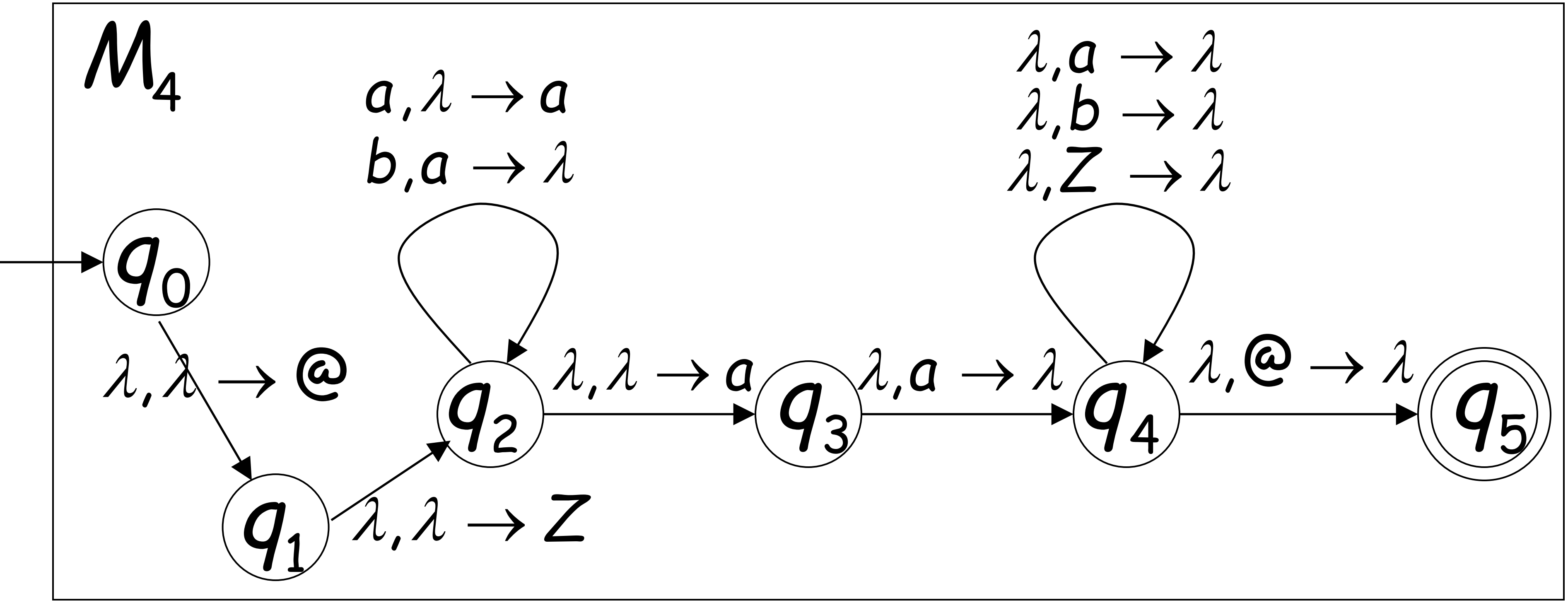
---

grammatica

Variabile Start $A_{q_0 q_f}$

Esempio:

PDA



M_4
$a, \lambda \to a$
$b, a \to \lambda$
$\lambda, a \to \lambda$
$\lambda, b \to \lambda$
$\lambda, Z \to \lambda$
q_0
$\lambda, \lambda \to @$
q_1
$\lambda, \lambda \to Z$
q_2
$\lambda, \lambda \to a$
q_3
$\lambda, a \to \lambda$
q_4
$\lambda, @ \to \lambda$
q_5

grammatica

caso 1: da stati singoli

$$A_{q_0 q_0} \to \lambda$$

$$A_{q_1 q_1} \to \lambda$$

$$A_{q_2 q_2} \to \lambda$$

$$A_{q_3 q_3} \to \lambda$$

$$A_{q_4 q_4} \to \lambda$$

$$A_{q_5 q_5} \to \lambda$$

caso 2: da triple di stati

$A_{q_0q_0} \to A_{q_0q_0}A_{q_0q_0} \mid A_{q_0q_1}A_{q_1q_0} \mid A_{q_0q_2}A_{q_2q_0} \mid A_{q_0q_3}A_{q_3q_0} \mid A_{q_0q_4}A_{q_4q_0} \mid A_{q_0q_5}A_{q_5q_0}$

$A_{q_0q_1} \to A_{q_0q_0}A_{q_0q_1} \mid A_{q_0q_1}A_{q_1q_1} \mid A_{q_0q_2}A_{q_2q_1} \mid A_{q_0q_3}A_{q_3q_1} \mid A_{q_0q_4}A_{q_4q_1} \mid A_{q_0q_5}A_{q_5q_1}$

$\vdots$

$A_{q_0q_5} \to A_{q_0q_0}A_{q_0q_5} \mid A_{q_0q_1}A_{q_1q_5} \mid A_{q_0q_2}A_{q_2q_5} \mid A_{q_0q_3}A_{q_3q_5} \mid A_{q_0q_4}A_{q_4q_5} \mid A_{q_0q_5}A_{q_5q_5}$

$\vdots$

$A_{q_5q_5} \to A_{q_5q_0}A_{q_0q_5} \mid A_{q_5q_1}A_{q_1q_5} \mid A_{q_5q_2}A_{q_2q_5} \mid A_{q_5q_3}A_{q_3q_5} \mid A_{q_5q_4}A_{q_4q_5} \mid A_{q_5q_5}A_{q_5q_5}$

Variabile Start $A_{q_0q_5}$

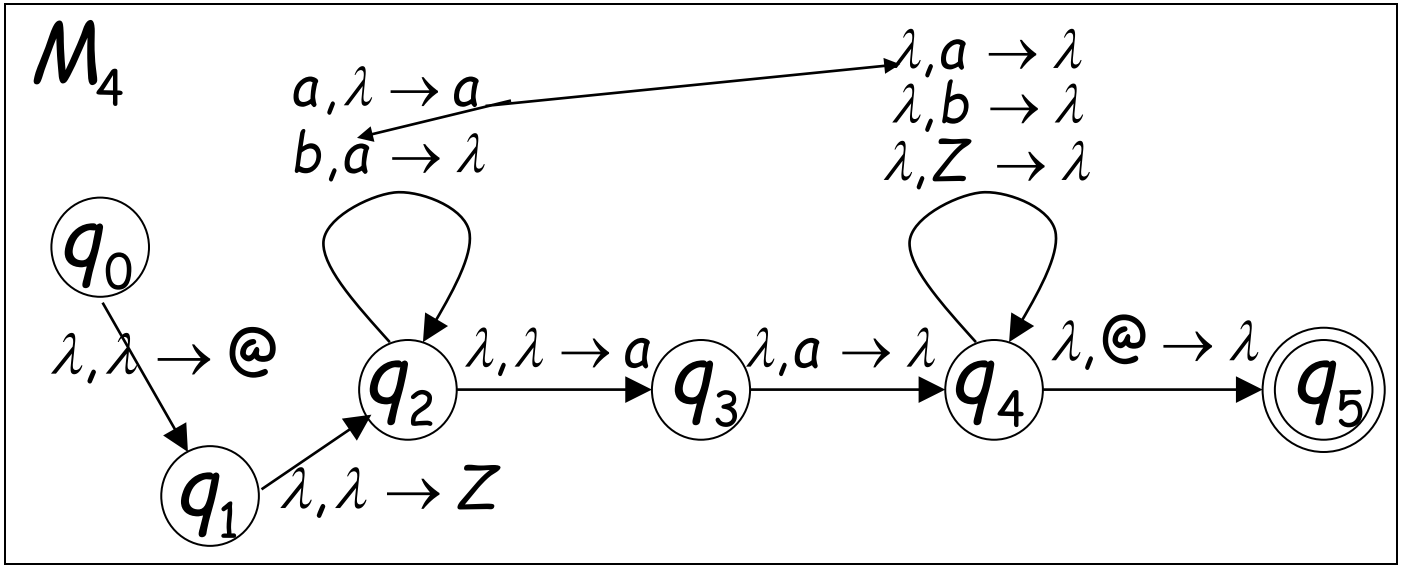
PDA

caso 3: **per ogni coppia di transizioni**

grammatica

$$A_{pq} \rightarrow a A_{rs} b$$

caso 3: da coppie di transizioni



$$A_{q_0q_5} \rightarrow A_{q_1q_4}$$

$$A_{q_1q_4} \rightarrow A_{q_2q_4}$$

$$A_{q_2q_4} \rightarrow aA_{q_2q_4}$$

$$A_{q_2q_2} \rightarrow aA_{q_2q_2}b$$

$$A_{q_2q_4} \rightarrow aA_{q_2q_3}$$

$$A_{q_2q_2} \rightarrow A_{q_3q_2}b$$

$$A_{q_2q_4} \rightarrow A_{q_3q_3}$$

$$A_{q_2q_4} \rightarrow A_{q_3q_4}$$

Supponiamo che il PDA M è stato tradotto
In una grammatica context-free G

Dobbiamo provare $L(G) = L(M)$

O in modo equivalente

$L(G) \subseteq L(M)$ $L(G) \supseteq L(M)$

$L(G) \subseteq L(M)$

Dobbiamo mostrare che se G ha una derivazione:

$$A_{q_0 q_f} \overset{*}{\Rightarrow} w \qquad \text{(stringa di terminali)}$$

Allora vi è un calcolo in M che accetta w :

$$(q_0, w, @) \overset{*}{\succ} (q_f, \lambda, @)$$

partiamo con una p e una q qualsiasi.

Se in G vi è una derivazione:

$$A_{pq} \stackrel{*}{\Rightarrow} w$$

Allora vi è un calcolo in M :

$$(p, w, \lambda) \stackrel{*}{\succ} (q, \lambda, \lambda)$$

Dopo aver provato
il passo precedente
abbiamo:

$$A_{q_0 q_f} \overset{*}{\Rightarrow} w$$

$$\Downarrow$$

$$(q_0, w, \lambda) \overset{*}{\succ} (q_f, \lambda, \lambda)$$

$$\Downarrow$$

Poichè non c'è nessuna
transizione
Con il simbolo @

$$(q_0, w, @) \overset{*}{\succ} (q_f, \lambda, @)$$

Lemma:

se $A_{pq} \overset{*}{\Rightarrow} w$ (stringa di terminali)

Allora vi è un calcolo

dallo stato p allo stato q

sulla stringa w

Che lascia lo stack vuoto:

$$(p, w, \lambda) \overset{*}{\succ} (q, \lambda, \lambda)$$

Dim intuitiva:

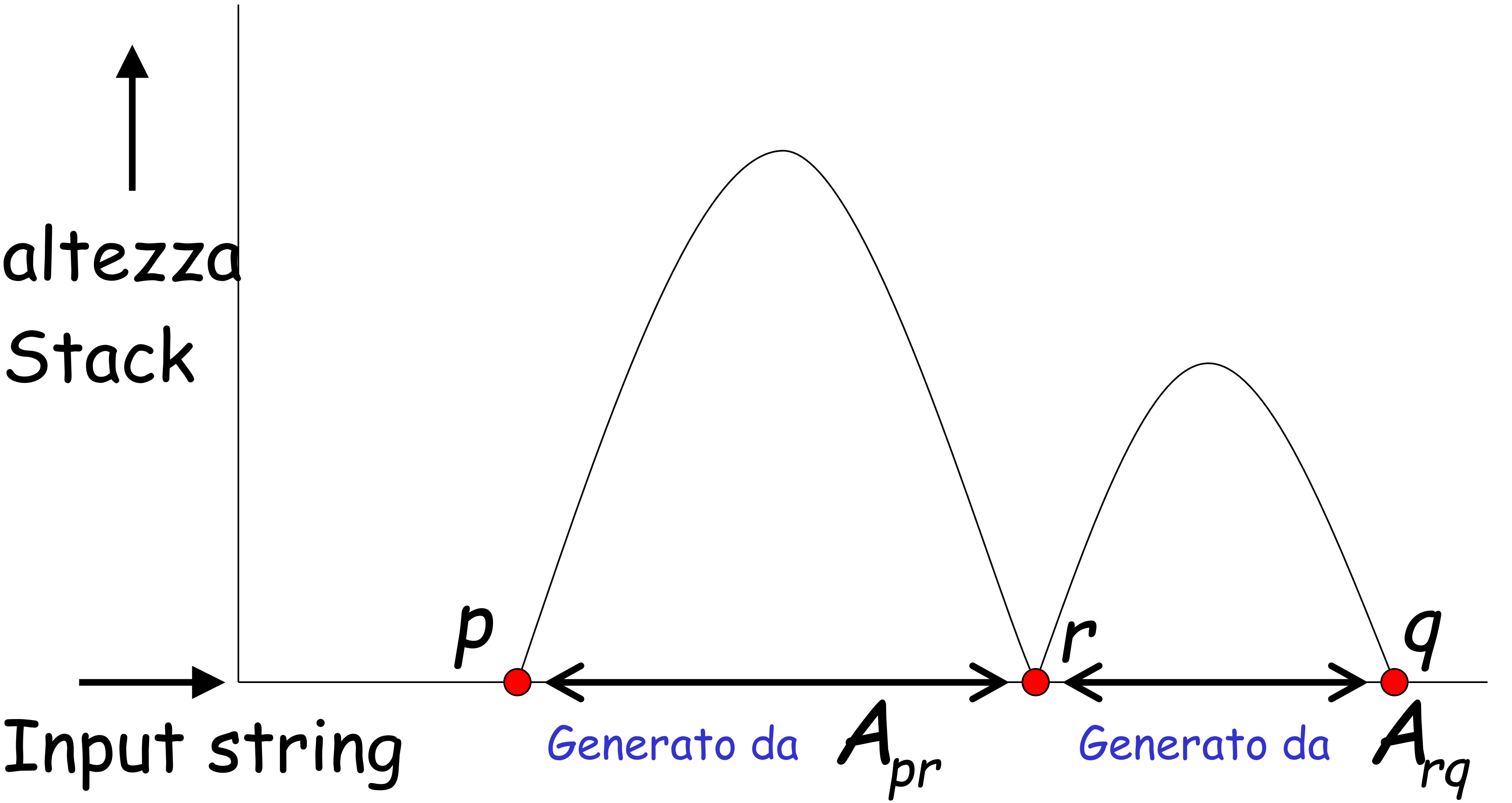
$$A_{pq} \Rightarrow \cdots \Rightarrow w$$

Case 1: $A_{pq} \Rightarrow A_{pr}A_{rq} \Rightarrow \cdots \Rightarrow w$

Case 2: $A_{pq} \Rightarrow aA_{rs}b \Rightarrow \cdots \Rightarrow w$

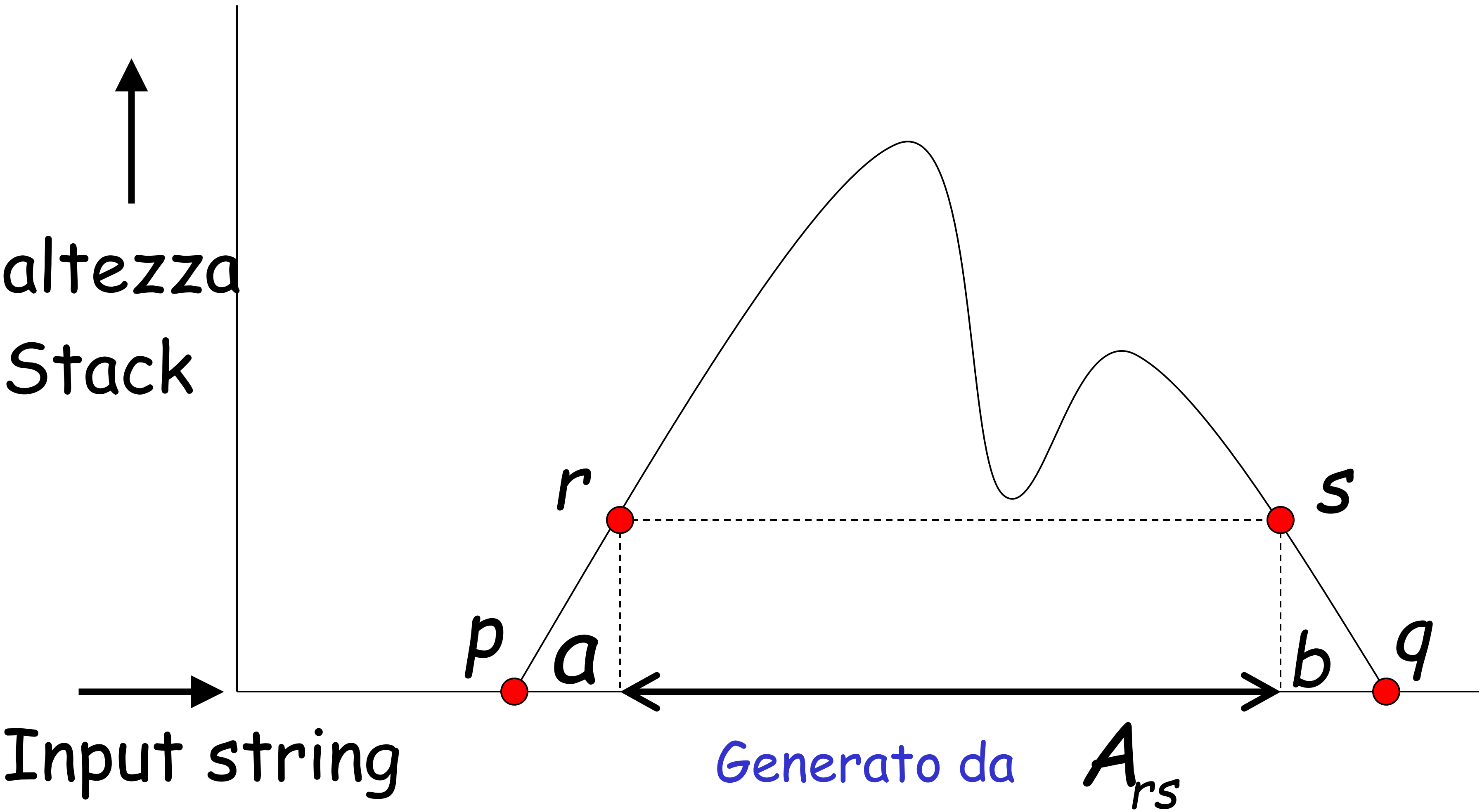
Type 2

Case 1: $A_{pq} \Rightarrow A_{pr} A_{rq} \Rightarrow \cdots \Rightarrow w$



Type 3

Case 2: $A_{pq} \Rightarrow aA_{rs}b \Rightarrow \cdots \Rightarrow w$



Formale :

Proviamo l'asserto per induzione
Sul numero di step della derivazione:

$$A_{pq} \underbrace{\Rightarrow \cdots \Rightarrow}_{\text{Numero di step}} w$$

base: $$A_{pq} \Rightarrow w$$

(Uno step di derivazione)

caso 1 produzione che deve essere usata è:

$$A_{pp} \to \lambda$$

Quindi $p = q$ e $w = \lambda$

Questo calcolo nel PDA esiste (banale):

$$(p, \lambda, \lambda) \overset{*}{\succ} (p, \lambda, \lambda)$$

Ipotesi induttiva:

$$A_{pq} \underbrace{\Rightarrow \cdots \Rightarrow}_{k} w$$

con k Step di derivazione

Quindi abbiamo, per ipotesi induttiva:

$$(p, w, \lambda) \overset{*}{\succ} (q, \lambda, \lambda)$$

Step induttivo

Data:

$$A_{pq} \underbrace{\Rightarrow \cdots \Rightarrow}_{} w$$

con $k+1$ step di derivazione

Dobbiamo provare:

$$(p, w, \lambda) \overset{*}{\succ} (q, \lambda, \lambda)$$

$$A_{pq} \underbrace{\Rightarrow \cdots \Rightarrow}_{k+1} w$$

derivazione steps

Case 1: $A_{pq} \Rightarrow A_{pr} A_{rq} \Rightarrow \cdots \Rightarrow w$

Case 2: $A_{pq} \Rightarrow a A_{rs} b \Rightarrow \cdots \Rightarrow w$

Case 1: $A_{pq} \underbrace{\Rightarrow A_{pr} A_{rq} \Rightarrow \cdots \Rightarrow}_{k+1 \text{ steps}} w$

Sia: $w = yz$

$A_{pr} \underbrace{\Rightarrow \cdots \Rightarrow}_{\text{Massimo } k \text{ steps}} y$

$A_{rq} \underbrace{\Rightarrow \cdots \Rightarrow}_{\text{Massimo } k \text{ steps}} z$

$$\underbrace{A_{pr} \Rightarrow \cdots \Rightarrow}_{\text{massimo } k \text{ steps}} y$$

Per ipotesi induttiva in PDA:

$$(p, y, \lambda) \overset{*}{\succ} (r, \lambda, \lambda)$$

$$\underbrace{A_{rq} \Rightarrow \cdots \Rightarrow}_{\text{massimo } k \text{ steps}} z$$

Per ipotesi induttiva, in PDA:

$$(r, z, \lambda) \overset{*}{\succ} (q, \lambda, \lambda)$$

$$(p,y,\lambda) \overset{*}{\succ} (r,\lambda,\lambda) \qquad (r,z,\lambda) \overset{*}{\succ} (q,\lambda,\lambda)$$

$$(p,yz,\lambda) \overset{*}{\succ} (r,z,\lambda) \overset{*}{\succ} (q,\lambda,\lambda)$$

poichè $w = yz$

$$(p,w,\lambda) \overset{*}{\succ} (q,\lambda,\lambda)$$

Case 2: $A_{pq} \underbrace{\Rightarrow aA_{rs}b \Rightarrow \cdots \Rightarrow}_{k+1 \text{ steps}} w$

Possiamo scrivere

$$w = ayb$$

$$A_{rs} \underbrace{\Rightarrow \cdots \Rightarrow}_{\text{Massimo } k \text{ steps}} y$$

$$\underbrace{A_{rs} \Rightarrow \cdots \Rightarrow}_{\text{Massimo } k \text{ steps}} y$$

Per ipotesi induttiva ,

il PDA calcola:

$$(r, y, \lambda) \overset{*}{\succ} (s, \lambda, \lambda)$$

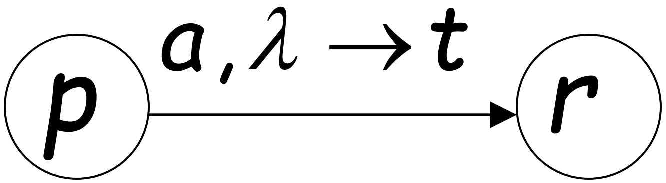
$$A_{pq} \Rightarrow aA_{rs}b \Rightarrow \cdots \Rightarrow w$$

La grammatica contiene la produzione

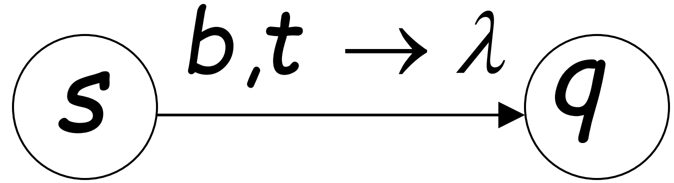
$$A_{pq} \rightarrow aA_{rs}b$$

e il PDA contiene la transizione

$$p \xrightarrow{a,\lambda \rightarrow t} r \qquad\qquad s \xrightarrow{b,t \rightarrow \lambda} q$$



$(p, ayb, \lambda) \succ (r, yb, t)$



$(s, b, t) \succ (q, \lambda, \lambda)$

sappiamo

$$(r,y,\lambda)\overset{*}{\succ}(s,\lambda,\lambda) \quad \Rightarrow \quad (r,yb,t)\overset{*}{\succ}(s,b,t)$$

Inoltre sappiamo

$$(p,ayb,\lambda)\succ(r,yb,t)$$

$$(s,b,t)\succ(q,\lambda,\lambda)$$

---

quindi:

$$(p,ayb,\lambda)\succ(r,yb,t)\overset{*}{\succ}(s,b,t)\succ(q,\lambda,\lambda)$$

$$(p, ayb, \lambda) \succ (r, yb, t) \overset{*}{\succ} (s, b, t) \succ (q, \lambda, \lambda)$$

poichè $w = ayb$

$$\Downarrow$$

$$(p, w, \lambda) \overset{*}{\succ} (q, \lambda, \lambda)$$

Fine