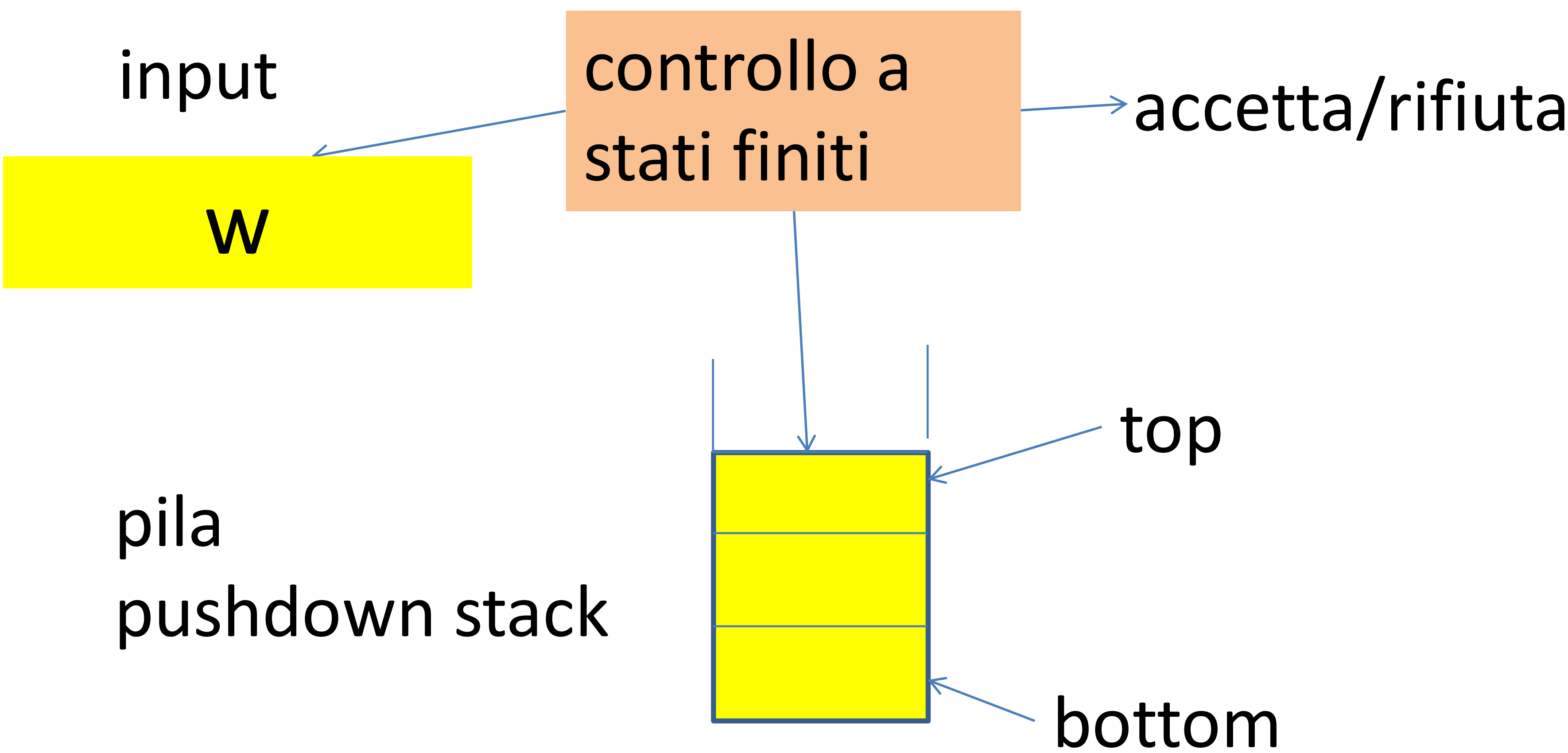
Automi a pila
PushDown Automaton (PDA)

PDA



input
controllo a stati finiti
accetta/rifiuta
w
top
pila
pushdown stack
bottom

la pila puo’ crescere arbitrariamente, ma
è usata solo con

--push: si inseriscono nuove cose in cima
alla pila
--pop: si toglie la cima della pila (se c’è)

attenzione-------> non si può ispezionare il
contenuto della pila senza buttarlo via

il controllo a stati finiti:
-legge i simboli in input da sinistra a destra
-guarda il simbolo in cima alla pila
-fa una transizione in cui può:
----cambiare stato (o no)
----consumare il simbolo di input (con ϵ) o no
--- eliminare, tenere o cambiare la cima della pila

esempio:
Lwwr={ ww^r | w in {0,1}*}, sono i palindromi pari

PDA che accetta Lwwr:
--uno stato di partenza q0 che scorre l'input e lo copia sullo stack e ad ogni passo può, sia continuare, sia invece "indovinare" (guess) di essere arrivato alla metà dell'input e quindi matcha il resto dell'input contro lo stack ➔ nondeterministico

se alla fine del match lo stack è vuoto allora OK

Definizione di PDA:
$P=(Q, \Sigma, \Gamma, \delta, q0, Z0, F)$
--Q = insieme finito di stati
-- Σ insieme finito di simboli di input
-- Γ insieme finito di alfabeto dello stack
-- δ funzione di transizione che riceve come argomenti una tripla (q,a,X) dove q è uno stato, a/ϵ è il prossimo input e X la cima della pila
δ(q,a,X) è un insieme finito di coppie (p,γ), dove p è uno stato e γ una stringa in Γ^* che rimpiazza X. Se γ è vuota allora si fa un pop, se γ=X allora lo stack non cambia.

per Lwwr={ ww^r | w in {0,1}*}

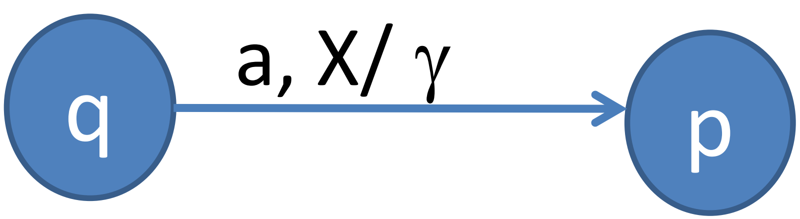
P=({q0,q1,q2},{0,1},{0,1,Z0}, δ, q0, Z0, {q2})
con δ come segue:
–– δ(q0,0,Z0)={(q0,0Z0)} e δ(q0,1,Z0)={(q0,1Z0)}

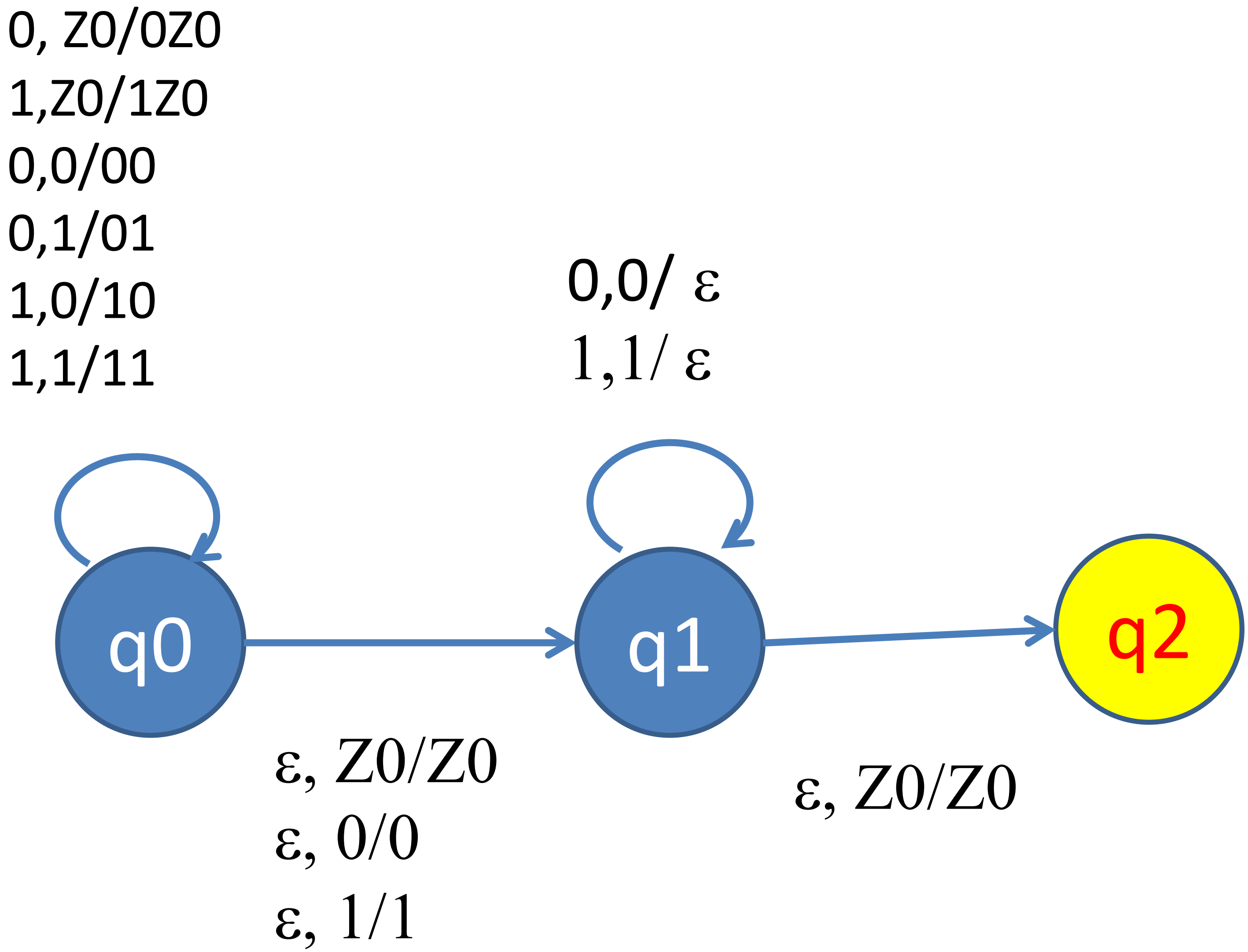
–– δ(q0,0,0)={(q0,00)}, δ(q0,1,0)={(q0,10)} ecc.
––δ(q0,ε,Z0)={(q1,Z0)}, δ(q0,ε,0)={(q1,0)} e
δ(q0,ε,1)={(q1,1)}

–– δ(q1,0,0)={(q1, ε)}, δ(q1,1,1)={(q1, ε)}
–– δ(q1, ε, Z0)={(q2, Z0)},

notazione grafica per PDA

-nodi corrispondono agli stati
-si distingue lo stato iniziale con una freccia entrante dal nulla
-gli archi corrispondono alle transizioni e hanno etichette che rappresentano cosa succede su input e stack:
se $\delta(q,a,X)$ contiene (p, γ) allora:




0, Z0/0Z0
1,Z0/1Z0
0,0/00
0,1/01
1,0/10
1,1/11
0,0/ ε
1,1/ ε
q0
q1
q2
ε, Z0/Z0
ε, 0/0
ε, 1/1
ε, Z0/Z0

Descrizioni istantanee (ID)

supponiamo che $\delta(q,a,X)$ contenga (p,γ), allora

$(q,aw,X\alpha) \vdash (p,w,\gamma\alpha)$

come al solito rappresentiamo la chiusura di $\vdash$ come $\vdash^*$

calcolo del PDA di ww^r

intuizione: posso aggiungere stringhe non usate all'input e allo stack, mantenendo la computazione (context-freeness)

Teorema 6.5

dato un PDA P, se $(q,x,\alpha) \vdash^* (p,y,\beta)$, allora per ogni stringa w e γ è vero che

$$(q,xw,\alpha\gamma) \vdash^* (p,yw,\beta\gamma)$$

l'inverso è falso

però vale per l'input:

Teorema 6.6

se $(q,xw,\alpha) \vdash^* (p, yw,\beta)$

allora è vero che

$(q,x,\alpha) \vdash^* (p, y,\beta)$

Dimostrazione: non si può rigenerare l'input, quindi w non ha influenza

Modalità di accettazione:

--<u>per stato finale</u>:

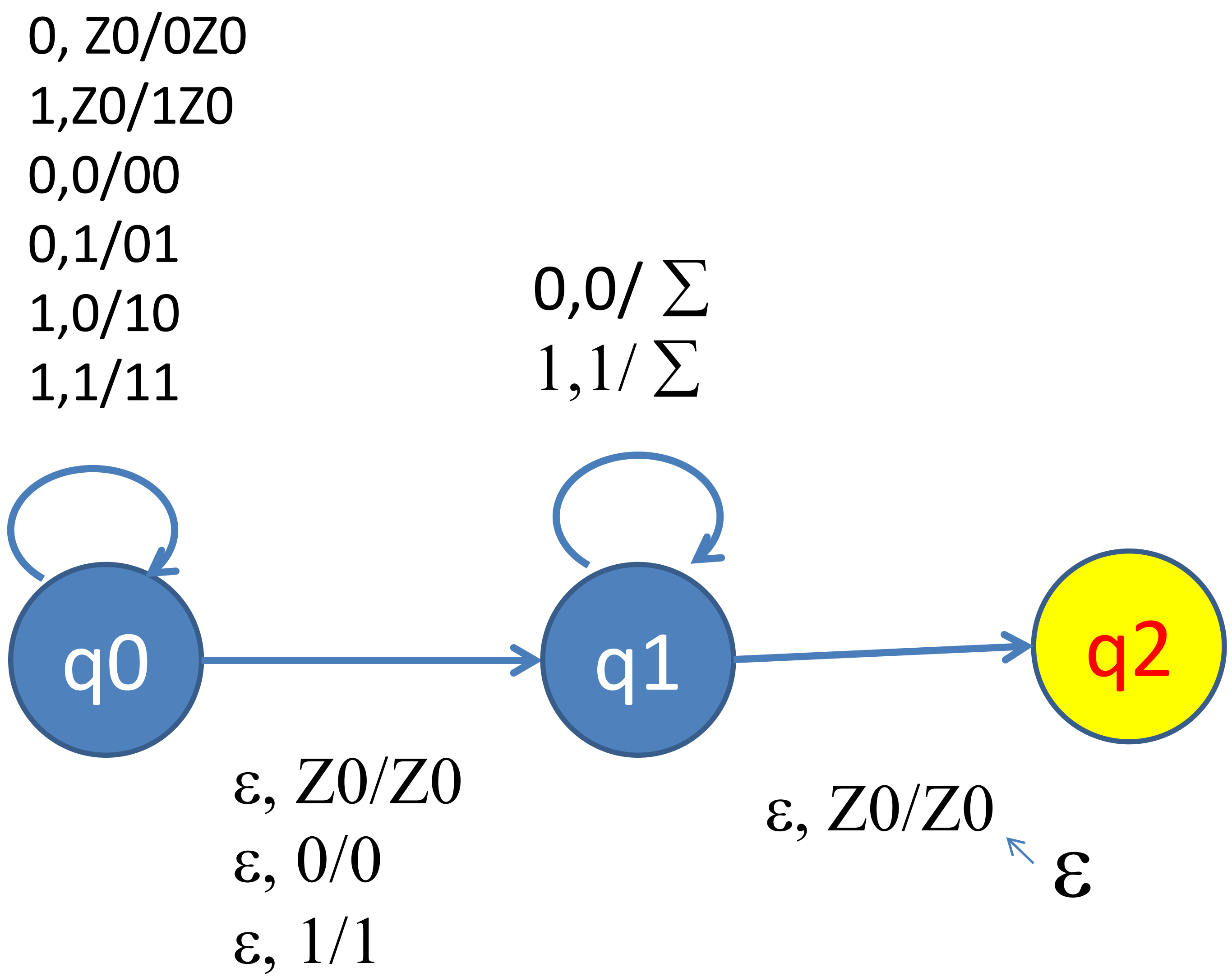
Dato P, L(P) = { w | w in $\Sigma*$, $(q0,w,Z0) \vdash (qf,\varepsilon,\alpha)$, con qf stato finale }

--<u>con stack vuoto</u>

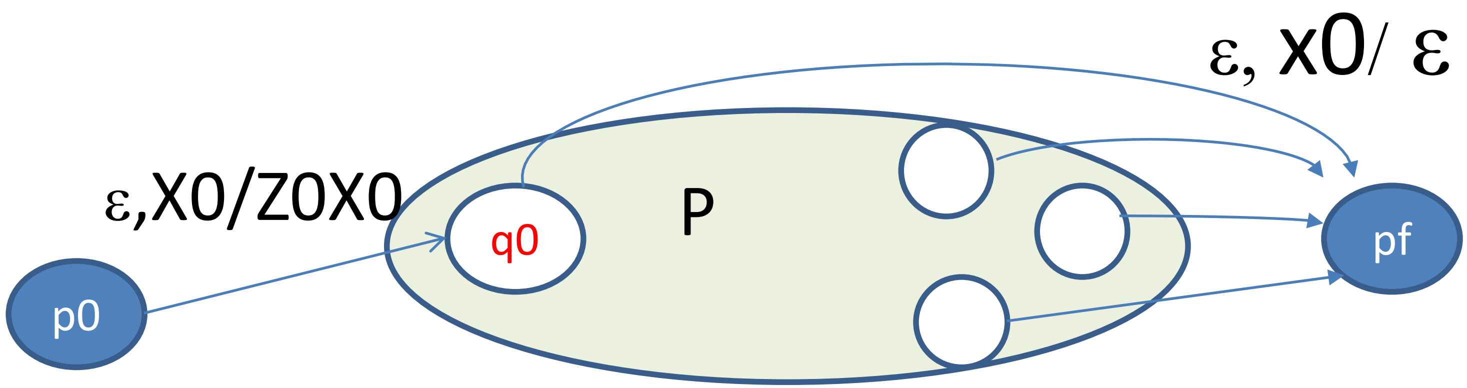
N(P)={w | w in $\Sigma*$, $(q0,w,Z0) \vdash* (q,\varepsilon,\varepsilon)$}

per un dato PDA P, L(P) e N(P) possono essere diversi

Il PDA che accetta ww^R, i palindromi di lunghezza pari, ha $N(P)= \emptyset$, ma è facile modificarlo



è sempre possibile passare da un PDA P che accetta in uno dei due modi ad PDA P' che accetta nell'altro modo e accetta lo stesso linguaggio di P.
da P che accetta per stack vuoto a P' che accetta per stato finale



Dimostrazione:

w è il $N(P)$ sse è in $L(P')$

(=>)

esiste $(q0,w,Z0) \vdash^* (q, \varepsilon, \varepsilon)$ per il Teorema 6.5,

$(q0,w,Z0X0) \vdash^* (q, \varepsilon, X0)$

e per costruzione esiste

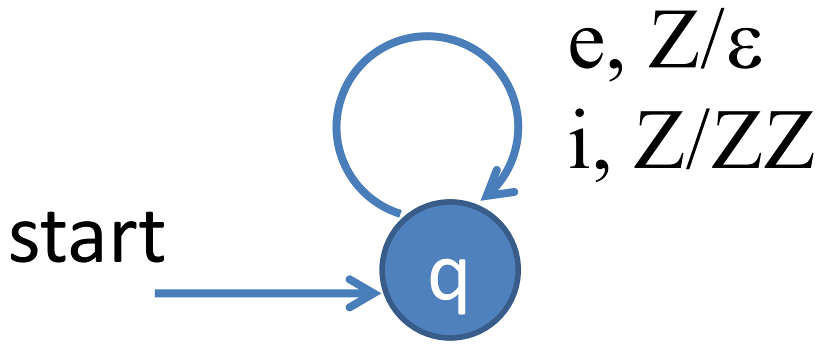
$(q, \varepsilon, X0) \vdash (qf, \varepsilon, \varepsilon)$

infine esiste: $(p0,w,X0) \vdash (q0,w,Z0X0)$

(<=) per costruzione una computazione di P' è:

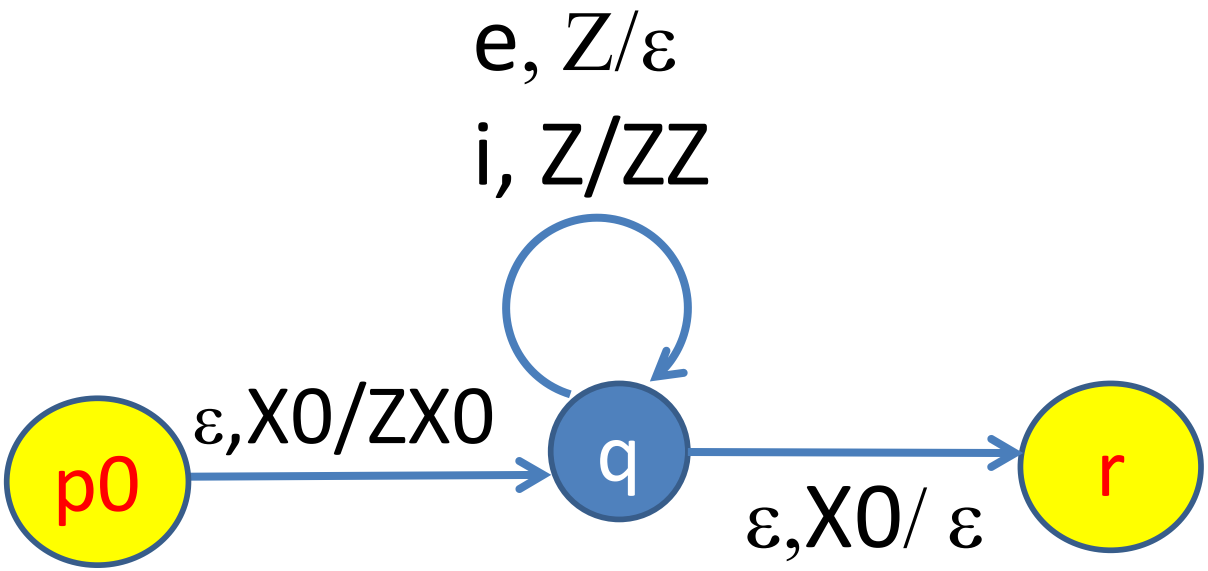
$(p0,w,X0) \vdash (q0,w,Z0X0) \vdash^* (q,\varepsilon,X0) \vdash (pf, \varepsilon, \varepsilon)$

dove questa è una computazione di P che accetta w esempio: un PDA che accetta per stack vuoto le stringhe w in {i,e}* tali che in tutti prefissi propri di w il n. di i >= n. di e, ma la condizione è falsa per w

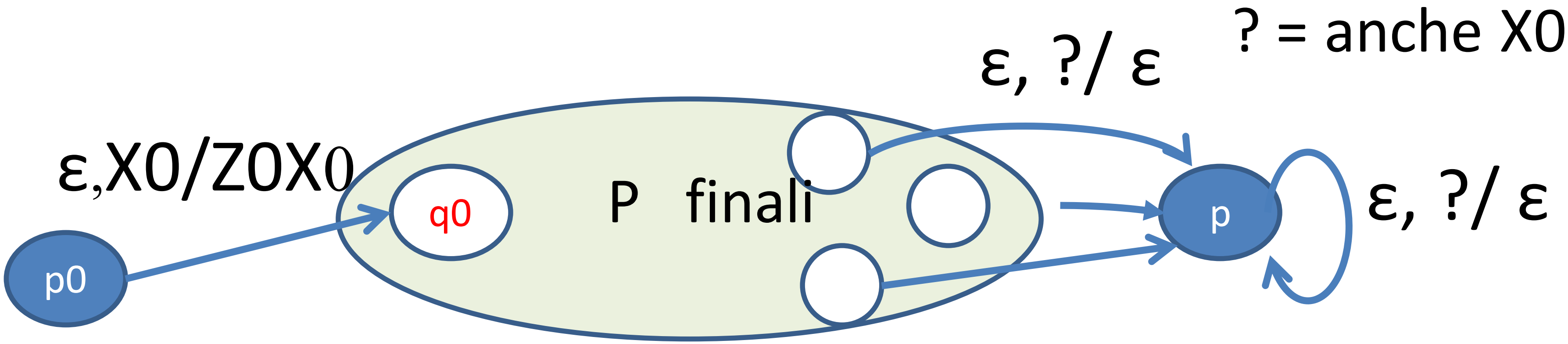


per semplicità Z0 -> Z

PDA che accetta per stato finale



Da P che accetta per stato finale a P' che accetta per stack vuoto



X0 previene che P svuoti lo stack accettando inavvertitamente, visto che P non ha mosse per X0

(=>) se (q0,w,Z0)|-*(qf, ε,α) allora
(p0,w,X0)|-(q0,w,Z0X0)|-*(qf, ε,αX0)|-*(p, ε, ε)
(<=) è l'inversa