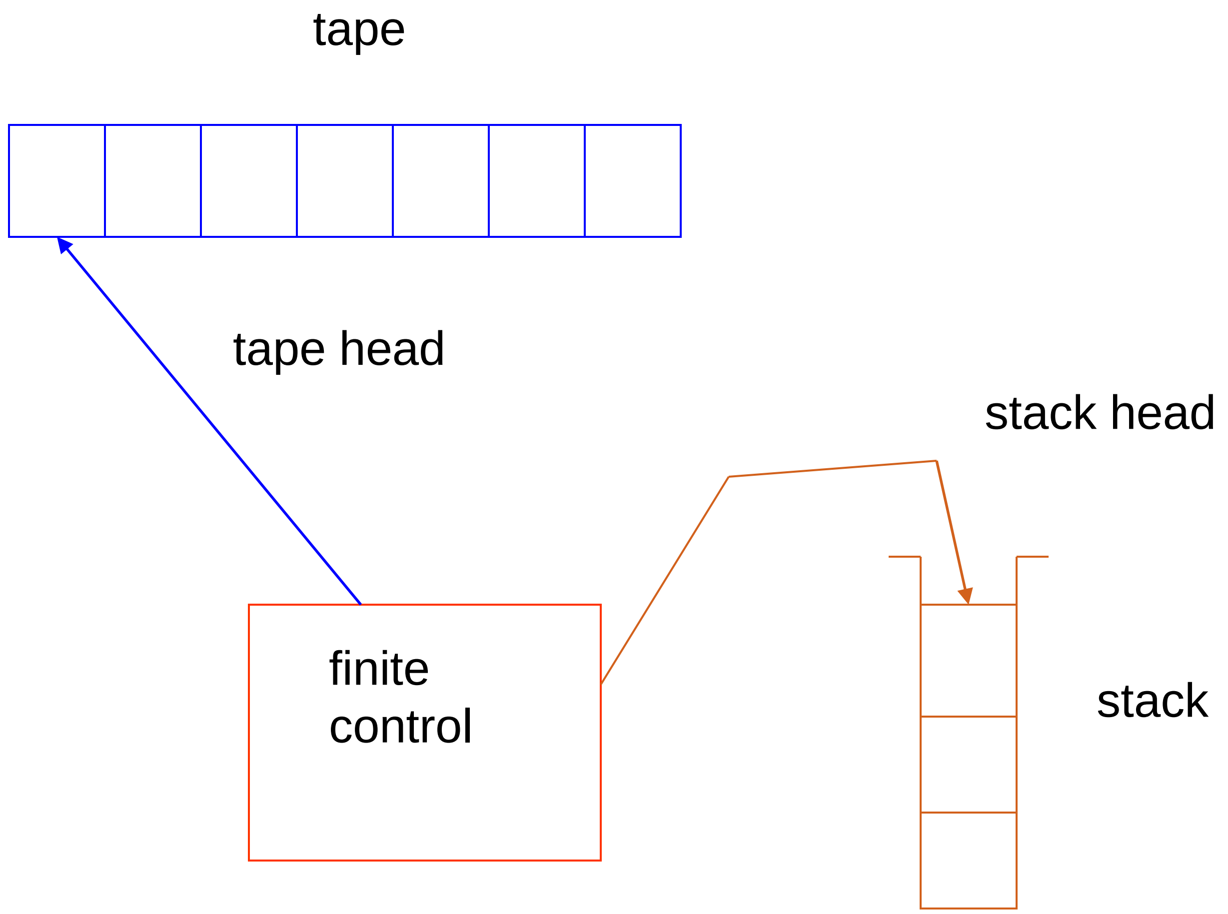
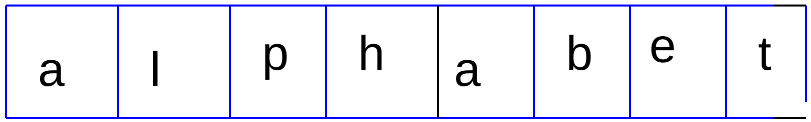
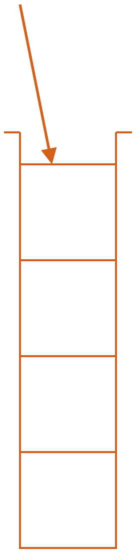
Pushdown Automata


tape
tape head
stack head
finite
control
stack



The tape is divided into finitely many cells. Each cell contains a symbol in an alphabet Σ.

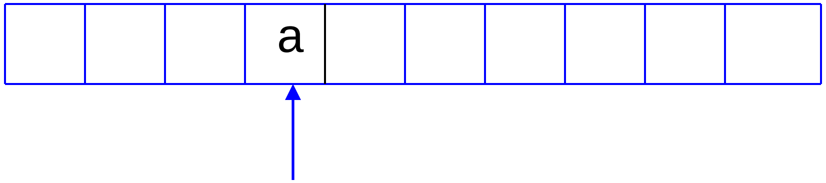


The stack head always scans the top symbol of the stack. It performs two basic operations:

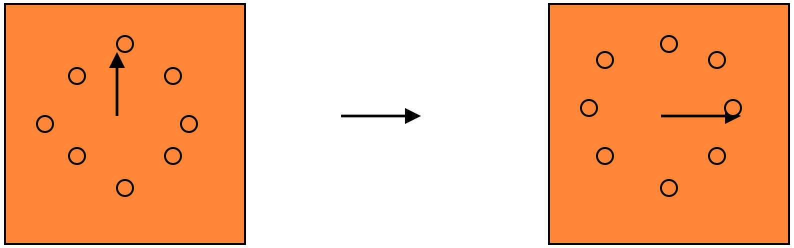
Push: add a new symbol at the top.

Pop: read and remove the top symbol.

Alphabet of stack symbols: Γ

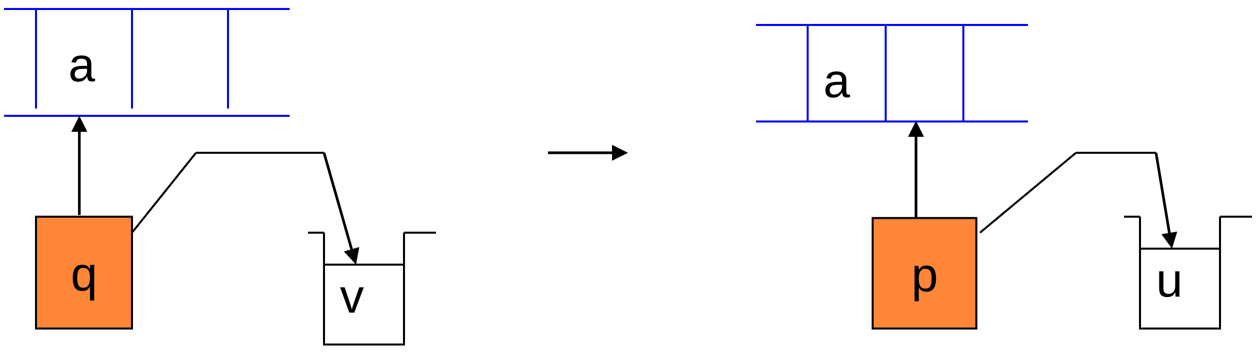


- The head scans at a cell on the tape and can *read* a symbol on the cell. In each move, the head can move to the right cell.

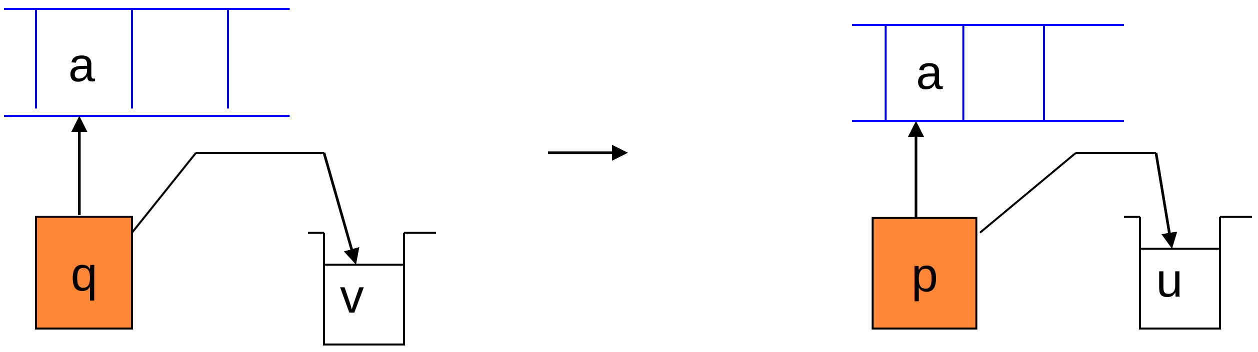


- The finite control has finitely many states which form a set Q. For each move, the state is changed according to the evaluation of a *transition function*

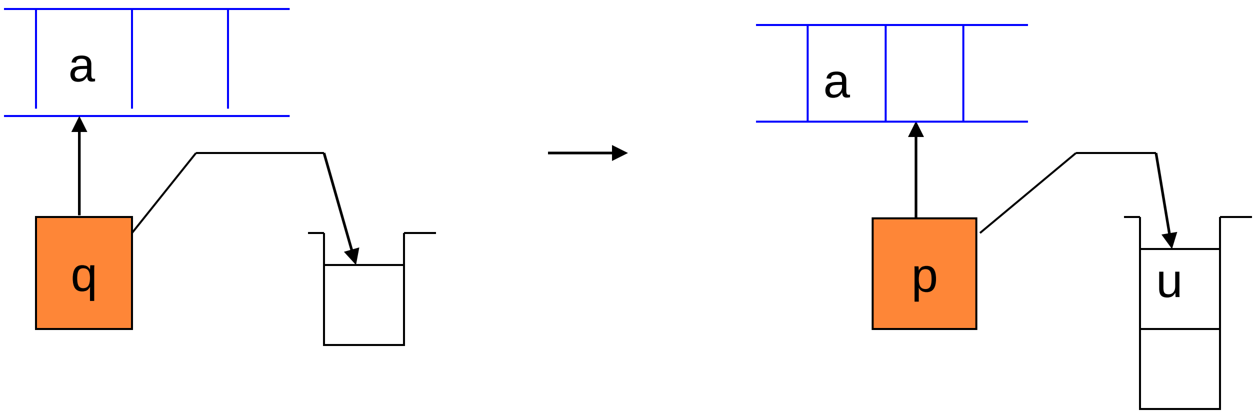
$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \rightarrow 2^{Q \times (\Gamma \cup \{\varepsilon\})}$.



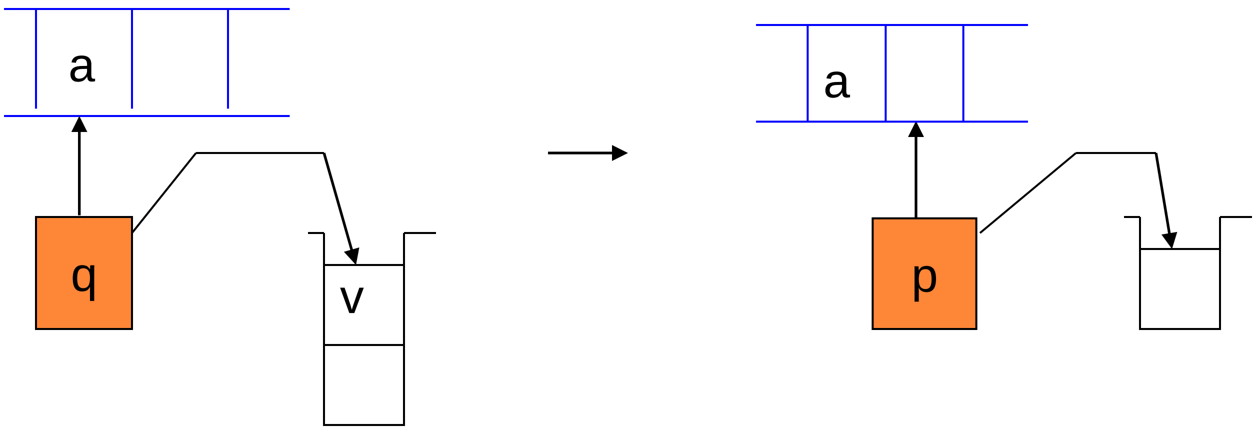
- *(p, u)* $\in$ *δ(q, a, v)* means that if the tape head reads *a,* the stack head read *v*, and the finite control is in the state *q*, then one of possible moves is that the next state is *p*, *v* is replaced by *u* at stack, and the tape head moves one cell to the right.



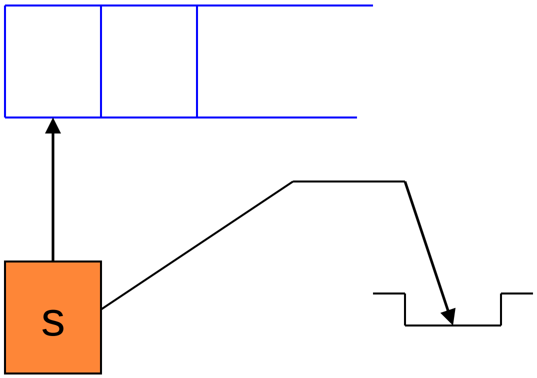
- $(p, u) \in \delta(q, \varepsilon, v)$ means that this a ε-move.



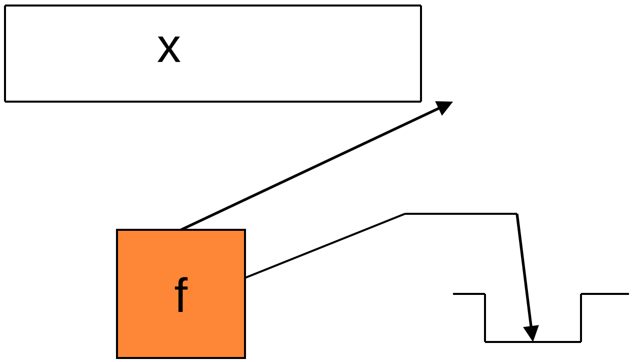
- *(p, u)* ∈ *δ(q, a, ε)* means that a push operation performs at stack.



- *(p, ε)* ∈ *δ(q, a, v)* means that a pop operation performs at stack



- There are some special states: an initial state s and a final set F of final states.
- Initially, the PDA is in the initial state s and the head scans the leftmost cell. The tape holds an input string. The stack is empty.



- When the head gets off the tape, the PDA stops. An input string *x* is accepted by the PDA if the PDA stops at a final state and the stack is empty.
- Otherwise, the input string is rejected.

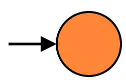
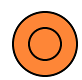
- The PDA can be represented by

$$M = (Q, \Sigma, \Gamma, \delta, s, F)$$

where Σ is the alphabet of input symbols and Γ is the alphabet of stack symbols.

- The set of all strings accepted by a PDA *M* is denoted by *L(M).* We also say that the language *L(M)* is accepted by *M*.

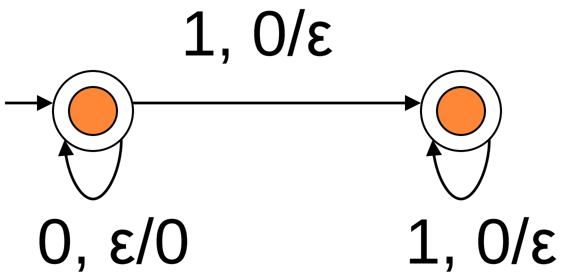
- The transition diagram of a PDA is an alternative way to represent the PDA.
- For *M = (Q, Σ, Γ, δ, s, F)*, the transition diagram of *M* is an edge-labeled digraph *G=(V, E)* satisfying the following:

$V = Q$ ($s =$ , $f =$  for $f \in F$)

$$E = \{\, q \xrightarrow{\text{a, v/u}} p \mid (p,u) \in \delta(q, a, v) \,\}.$$

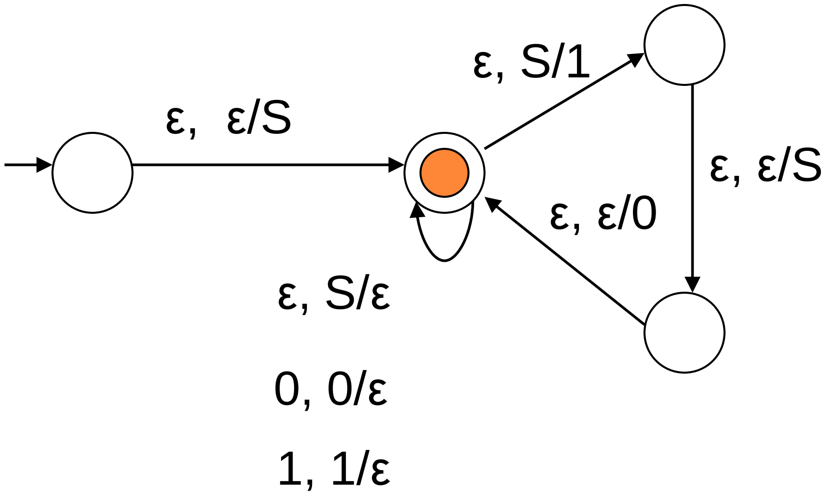
Example 1. Construct PDA to accept $L = \{0^n 1^n \mid n \geq 0\}$

Solution 1.



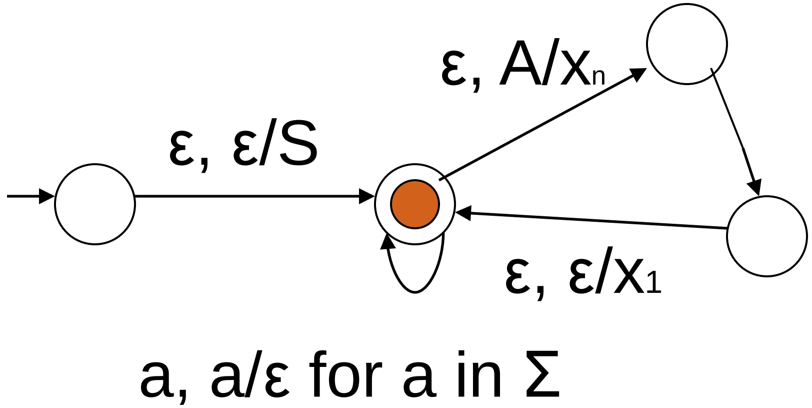
Solution 2.

Consider a CFG
$G = (\{S\}, \{0,1\}, \{S \rightarrow \varepsilon \mid 0S1\}, S)$.



Theorem Every CFL can be accepted by a PDA.

Proof. Consider a CFL L = L(G) for a CFG G = (V, Σ, R, S).



$A \rightarrow x_1\, x_2 \ldots x_n$ in R

Theorem

A language L is CFL $\Leftrightarrow$ it can be accepted by a PDA.

Sometimes, constructing the PDA is easier than constructing CFG.

Example 2

Show that $\{x \in \{a,b\} \mid \#_b(x) \leq \#_a(x) \leq 2\#_b(x)\}$ is a CFL

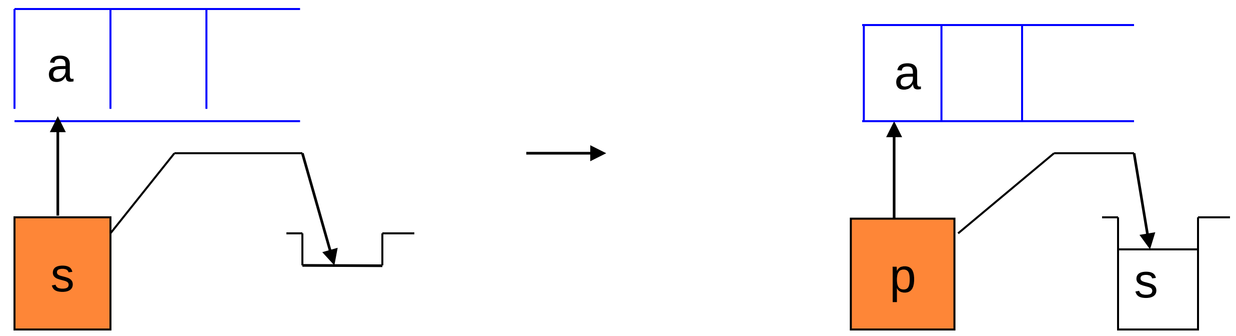
Construct a PDA or a CFG?

PDA!!!

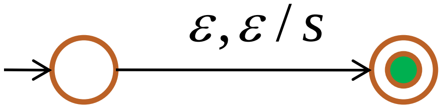
Sometimes, constructing the PDA is easier than constructing CFG.

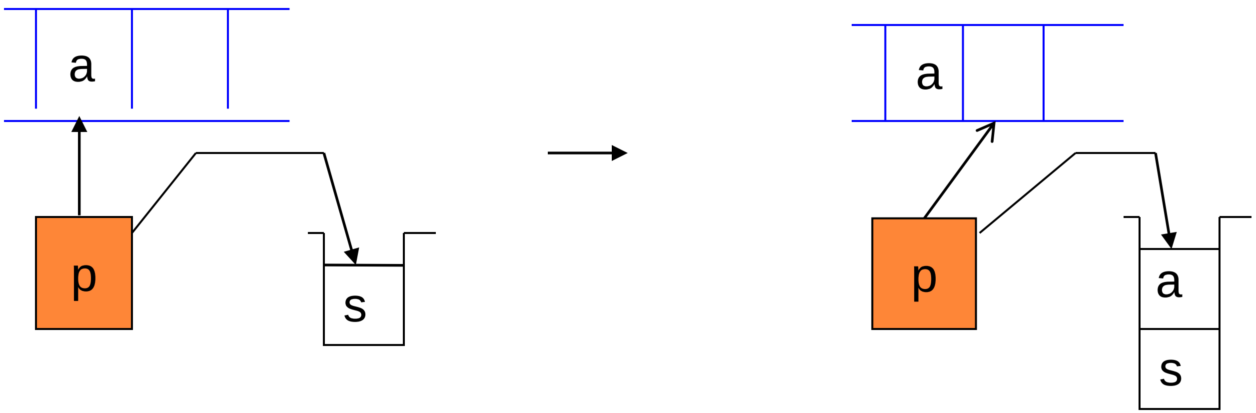
Idea

To check if $\#_b(x) \leq \#_a(x) \leq 2\#_b(x)$, we need to cancel a with b. For each b, we need to cancel sometimes one a and sometimes two a. Do we need to make a deterministic choice at each cancellation? No, the concept of nondeterministic computation solves this trouble: As long as a correct choice exists, the input string x would be accepted!



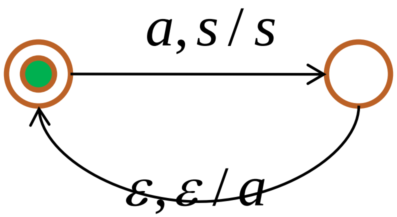
- *(p, s)* $\in$ *δ(s, ε, ε)*

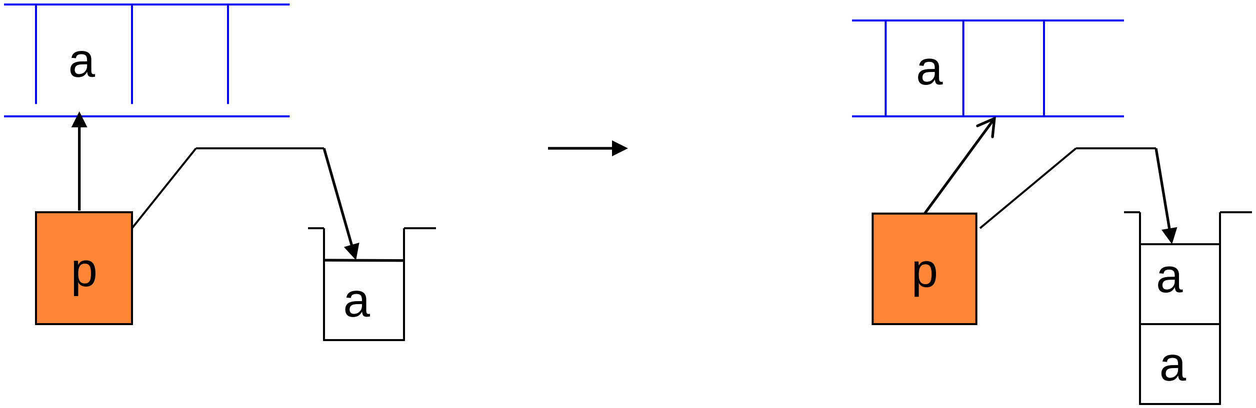




$(q, s) \in \delta(p, a, s)$

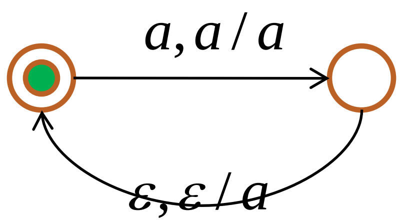
$(p, a) \in \delta(q, \varepsilon, \varepsilon)$

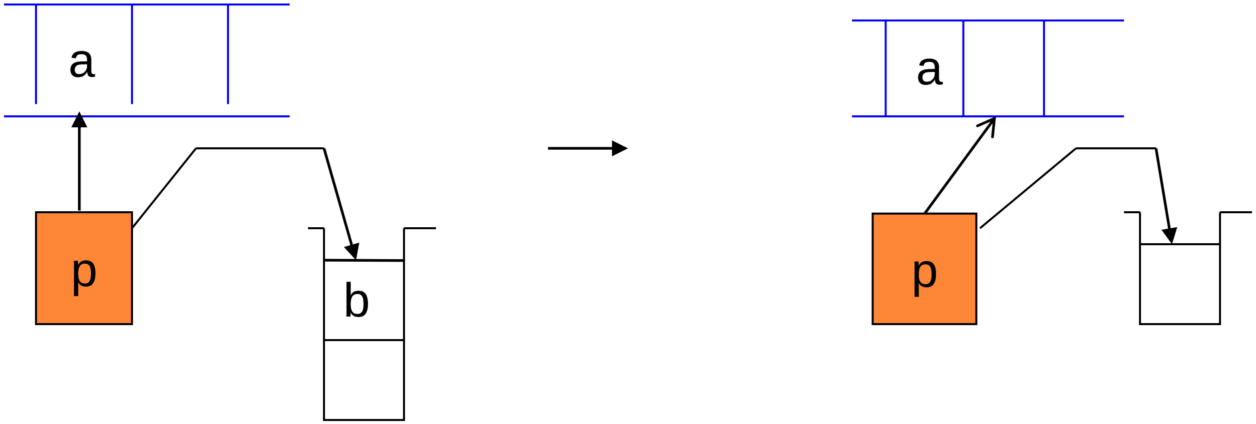




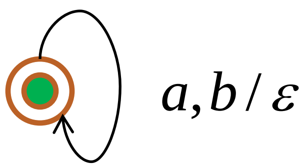
$(q1, a) \in \delta(p, a, a)$

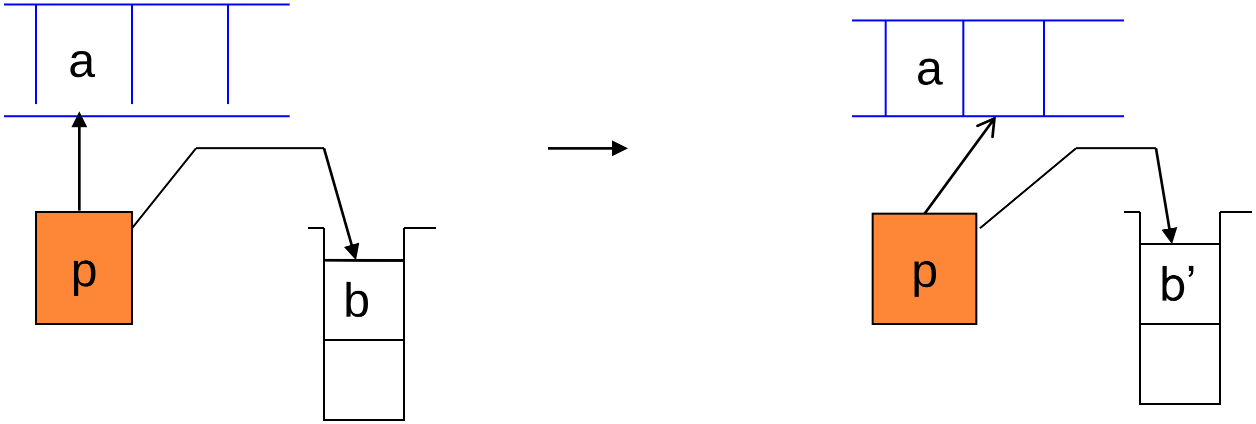
$(p, a) \in \delta(q1, \varepsilon, \varepsilon)$



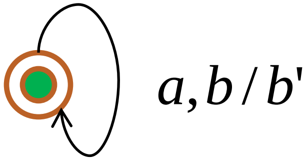


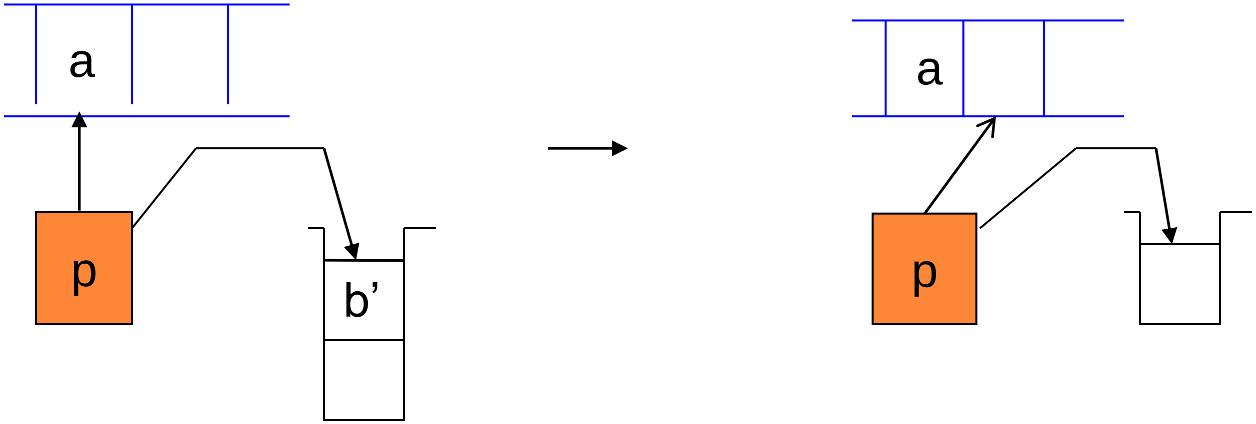
$(p, \varepsilon) \in \delta(p, a, b)$



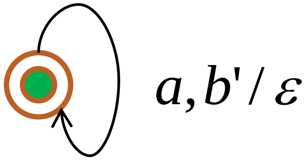


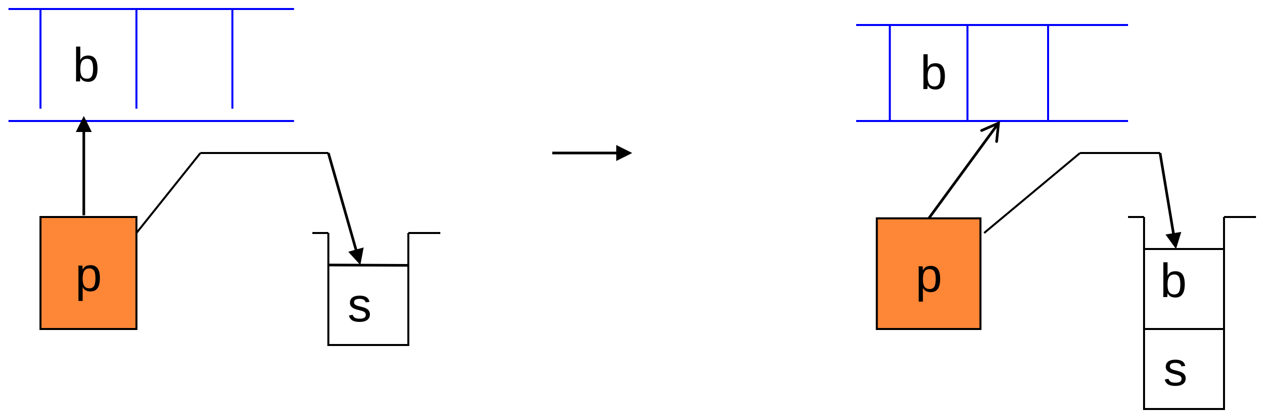
$$(p, b') \in \delta(p, a, b)$$





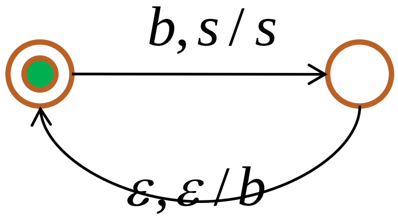
(p, ε) ∈ *δ(p, a, b')*

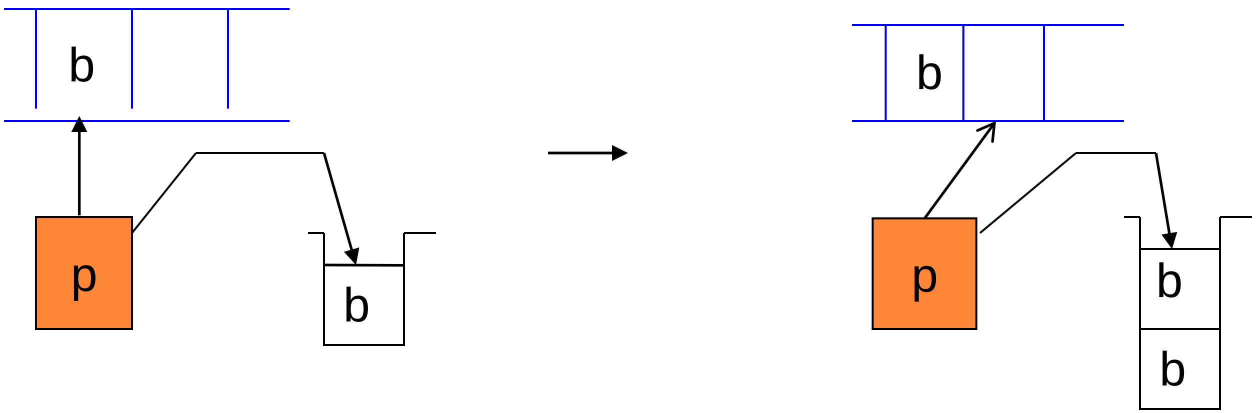




$(r1, s) \in \delta(p, b, s)$

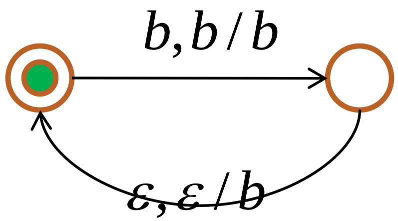
$(p, b) \in \delta(r1, \varepsilon, \varepsilon)$

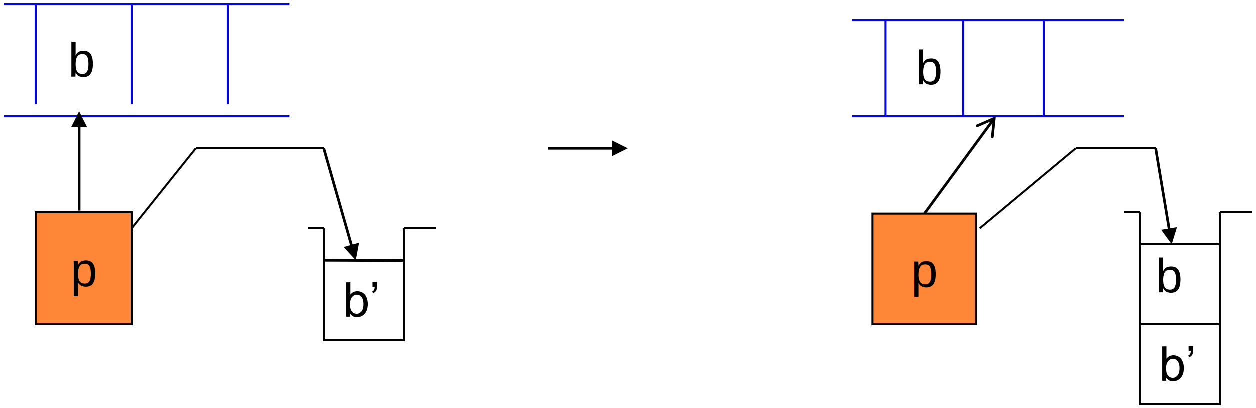




$(r2, b) \in \delta(p, b, b)$

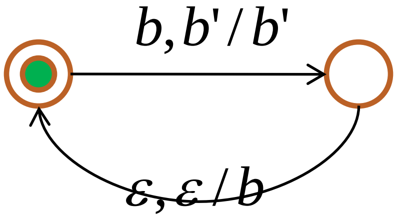
$(p, b) \in \delta(r2, \varepsilon, \varepsilon)$

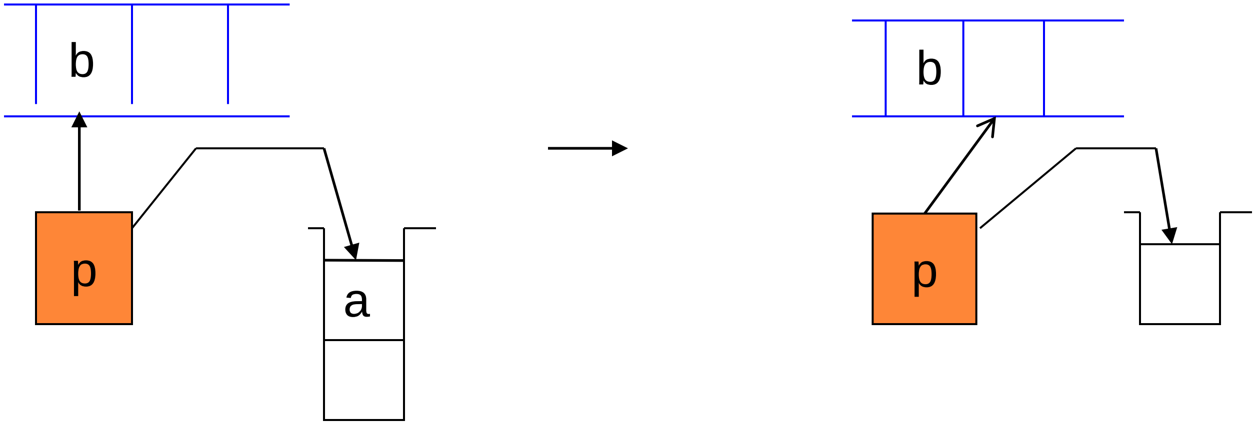




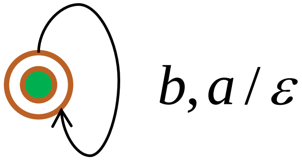
$(r3, b') \in \delta(p, b, b')$

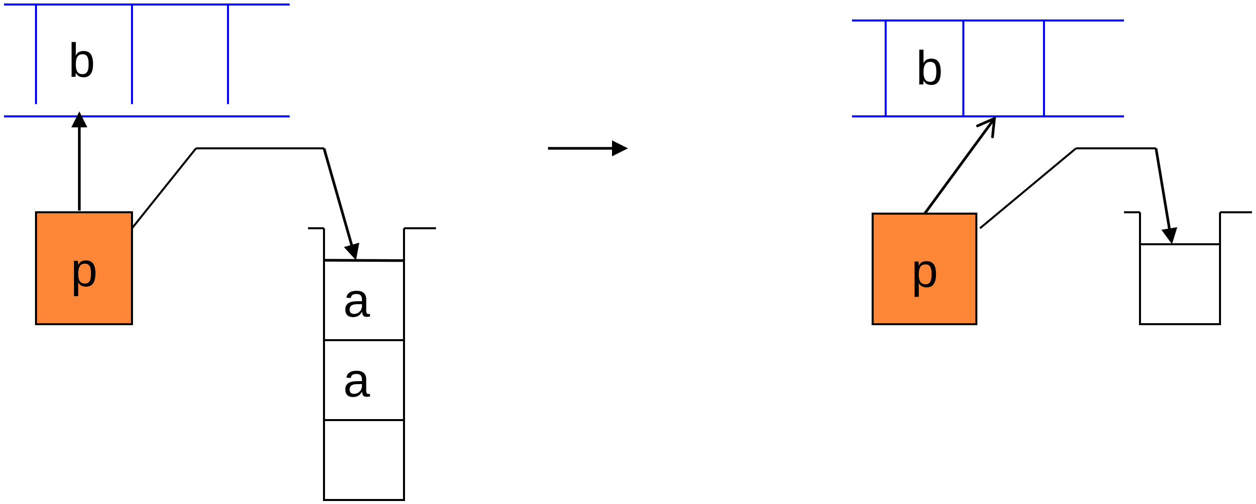
$(p, b) \in \delta(r3, \varepsilon, \varepsilon)$



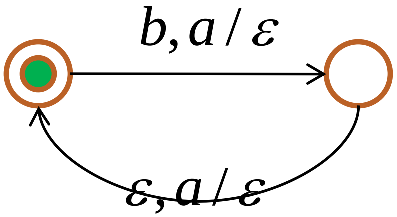


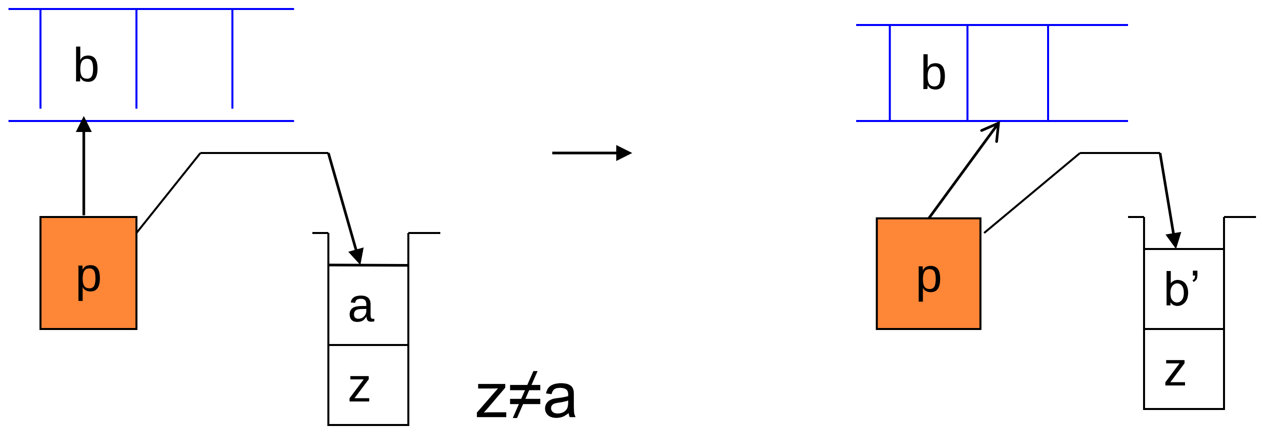
$(p, \varepsilon) \in \delta(p, b, a)$



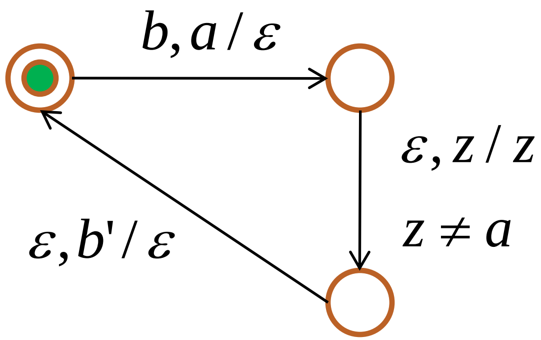


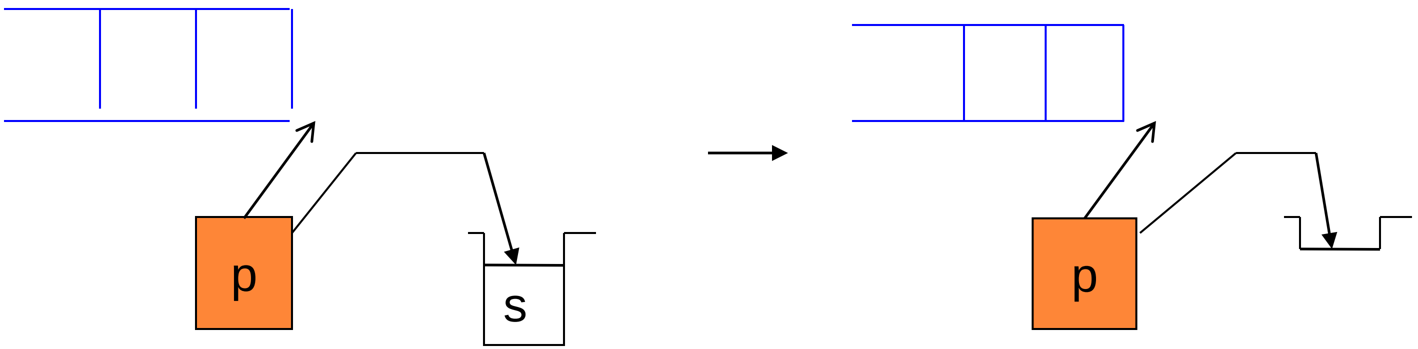
$(t, \varepsilon) \in \delta(p, b, a)$ $(p, \varepsilon) \in \delta(t, \varepsilon, a)$



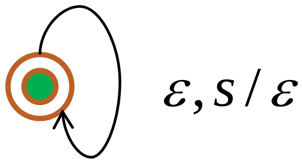


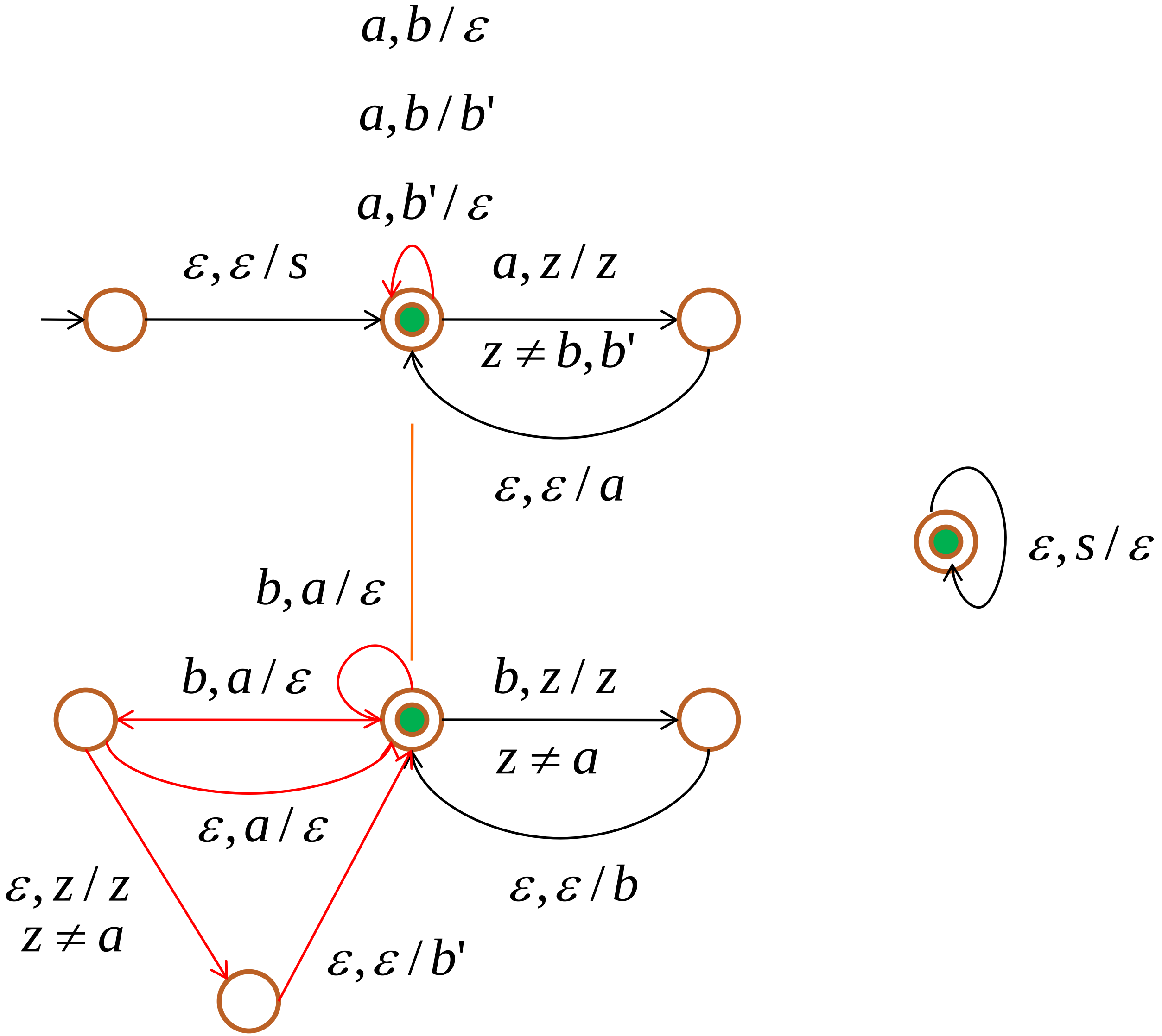
$(t, \varepsilon) \in \delta(p, b, a)$ $(u, x) \in \delta(t, \varepsilon, x)$ $(p, b') \in \delta(u, \varepsilon, \varepsilon)$





$$(p, \varepsilon) \in \delta(p, \varepsilon, s)$$




$a,b/\varepsilon$
$a,b/b'$
$a,b'/\varepsilon$
$\varepsilon,\varepsilon/s$
$a,z/z$
$z\neq b,b'$
$\varepsilon,\varepsilon/a$
$\varepsilon,s/\varepsilon$
$b,a/\varepsilon$
$b,a/\varepsilon$
$b,z/z$
$z\neq a$
$\varepsilon,a/\varepsilon$
$\varepsilon,z/z$
$z\neq a$
$\varepsilon,\varepsilon/b$
$\varepsilon,\varepsilon/b'$

Example 3

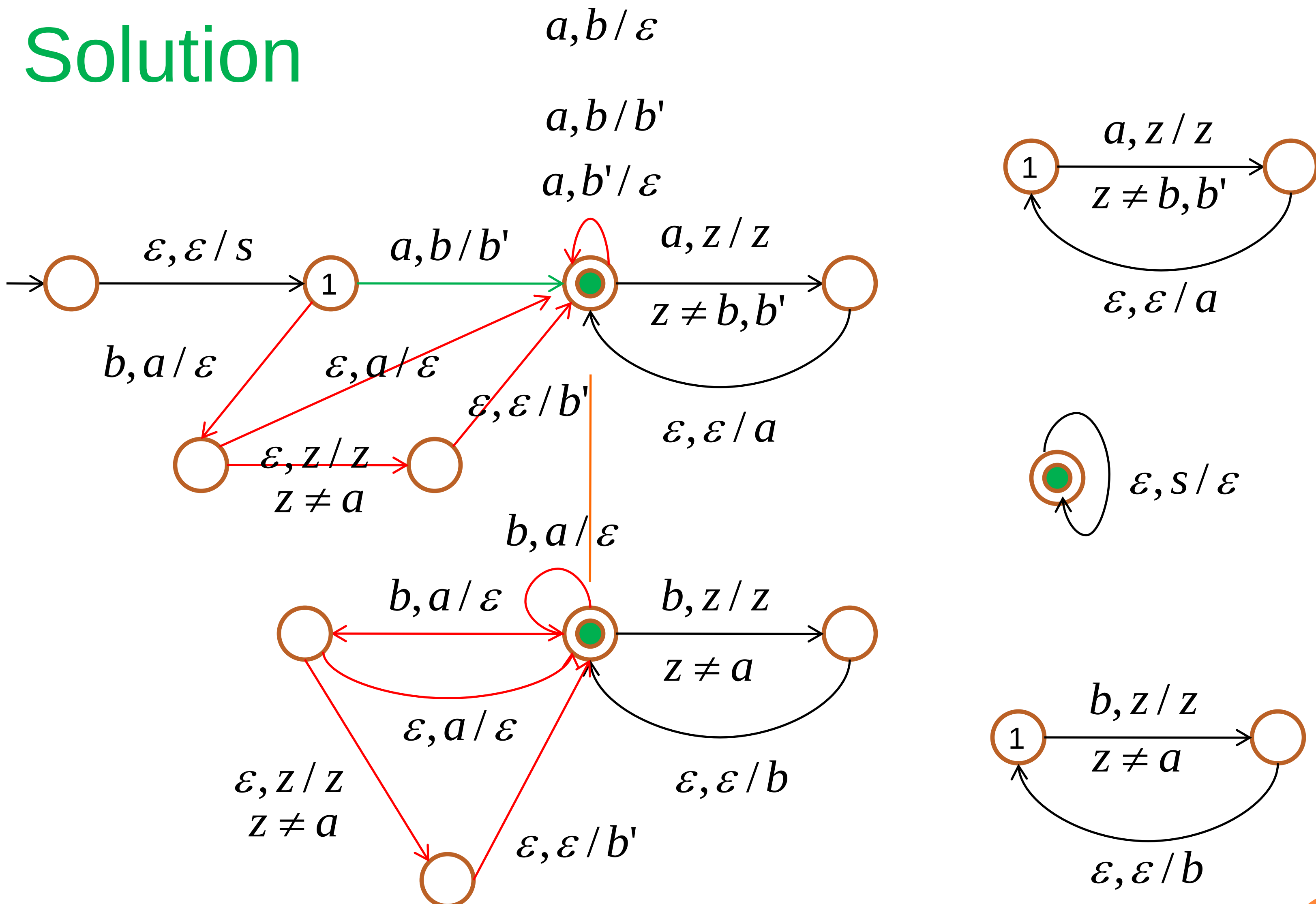
Construct PDA accepting $\{x \in \{a,b\} \mid \#_b(x) < \#_a(x) \le 2\#_b(x)\}$.

Idea

To have $\#_b(x) < \#_a(x)$, we must have a b which cancel two a's.

Let the first b do the job.

Solution



a, b / ε
a, b / b'
a, b' / ε
ε, ε / s
1
a, b / b'
a, z / z
z ≠ b, b'
b, a / ε
ε, a / ε
ε, ε / b'
ε, z / z
z ≠ a
ε, ε / a
b, a / ε
b, a / ε
b, z / z
z ≠ a
ε, a / ε
ε, ε / b
ε, z / z
z ≠ a
ε, ε / b'
1
a, z / z
z ≠ b, b'
ε, ε / a
ε, s / ε
1
b, z / z
z ≠ a
ε, ε / b

Example 4

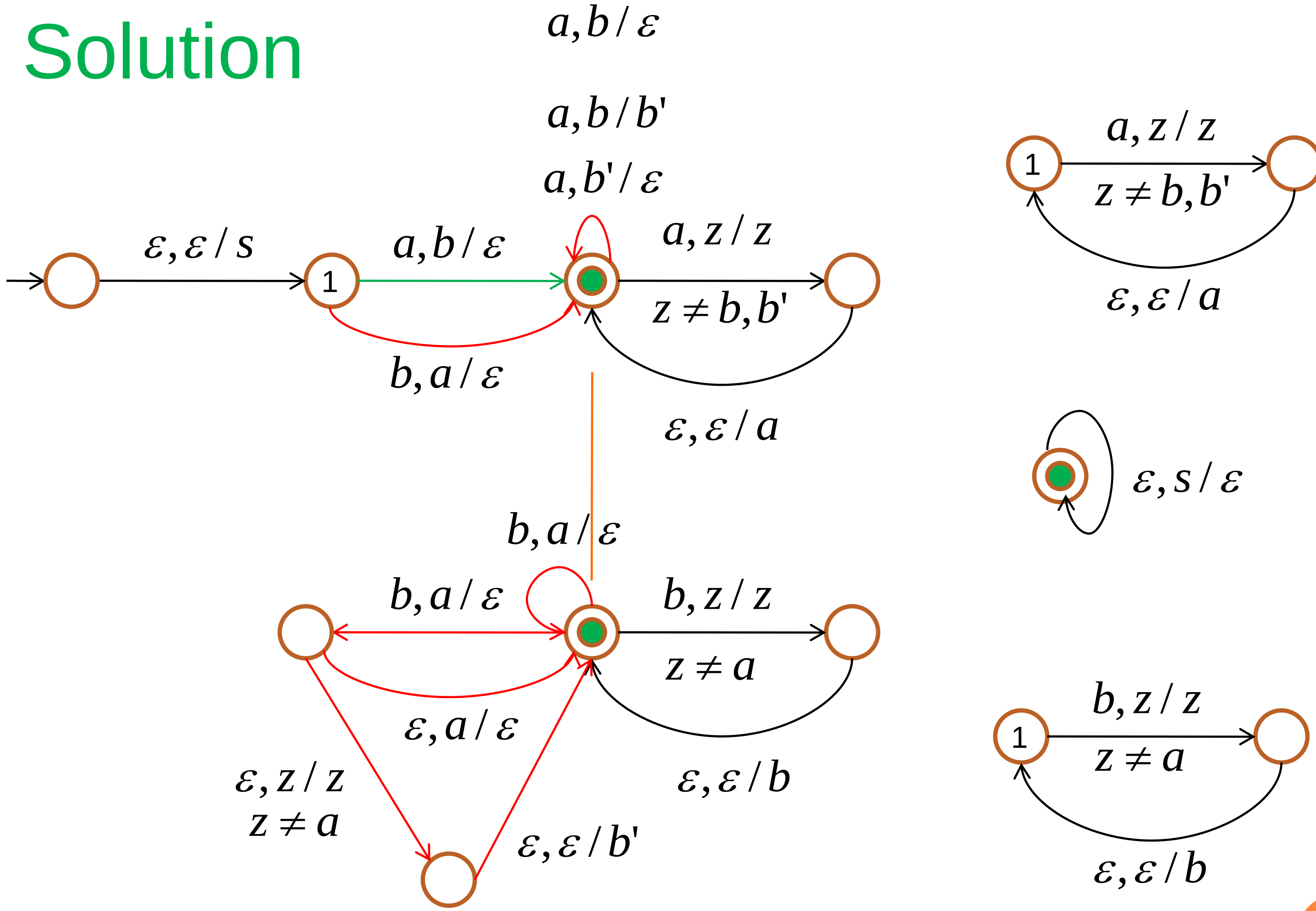
Construct PDA accepting $\{x \in \{a,b\} \mid \#_b(x) \leq \#_a(x) < 2\#_b(x)\}$.

Idea

To have $\#_a(x) < 2\#_b(x)$, we must have a b which cancel one a.

Let the first b do the job.

Solution



a,b/ε
a,b/b'
a,b'/ε
ε,ε/s
1
a,b/ε
a,z/z
z≠b,b'
b,a/ε
ε,ε/a
b,a/ε
b,a/ε
b,z/z
z≠a
ε,a/ε
ε,z/z
z≠a
ε,ε/b
ε,ε/b'
1
a,z/z
z≠b,b'
ε,ε/a
ε,s/ε
1
b,z/z
z≠a
ε,ε/b