CS420

Introduction to the Theory of Computation

Lecture 10: LDA $\iff$ CFG

Tiago Cogumbreiro

Today we will learn...

- Exercises on designing a PDA
- Convert a PDA into a CFG
- Convert a CFG into a PDA

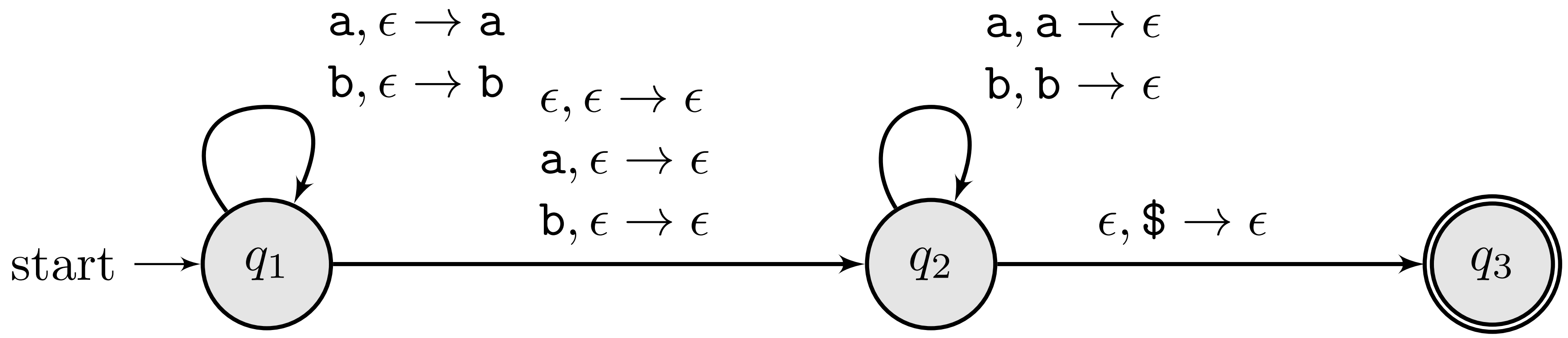
> Section 2.2
> Supplementary material: Professor David Chiang's lecture notes [1] [2]; Professor Siu On Chan slides

Exercise 1

1. aa is a palindrome
2. aba is a palindrome
3. bbb is a palindrome
4. ϵ is a palindrome
5. a is a palindrome

Give a PDA that recognizes palindromes and show it accepts aba and rejects abb

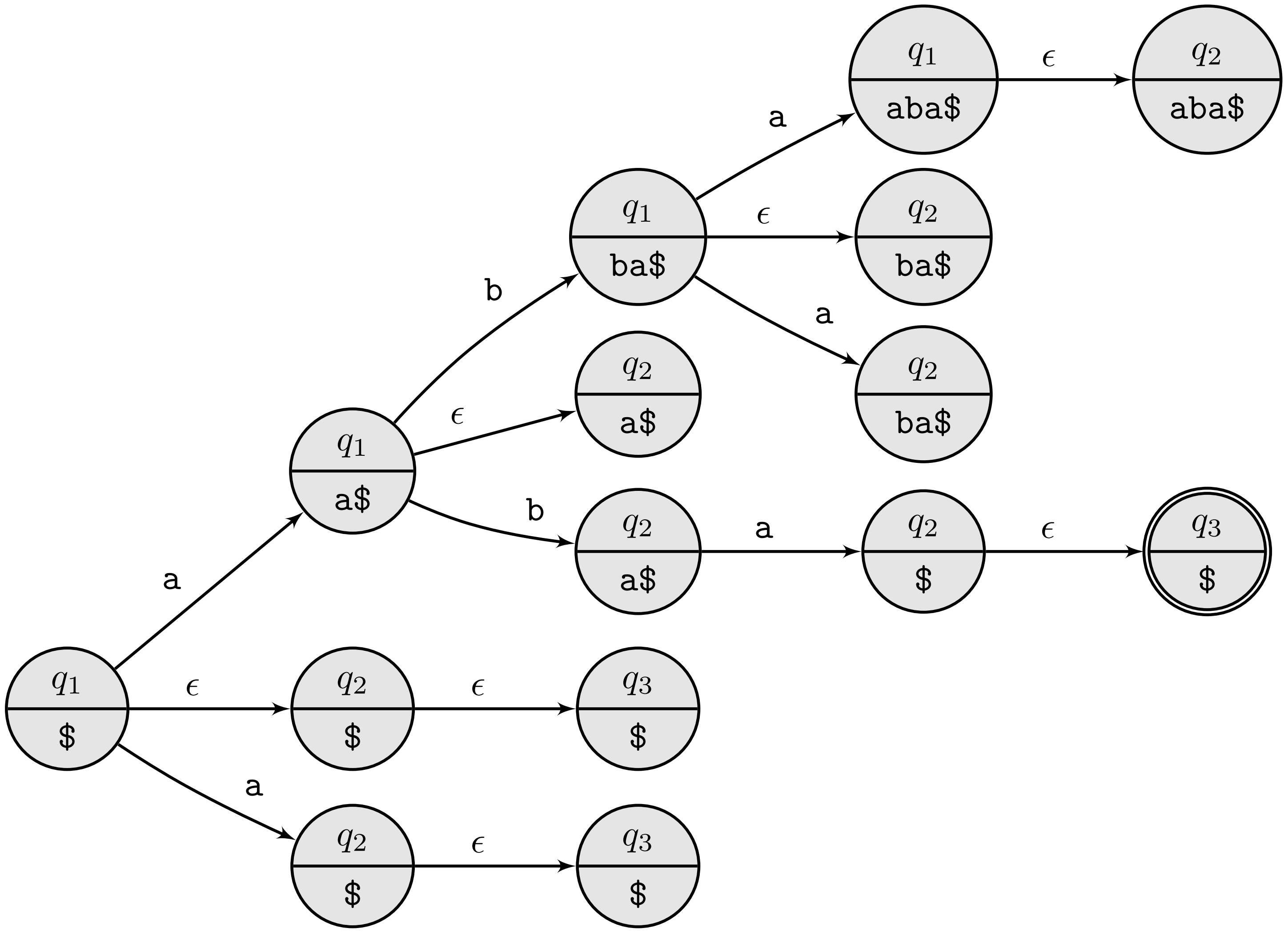
Exercise palindrome



$\mathtt{a}, \epsilon \to \mathtt{a}$
$\mathtt{b}, \epsilon \to \mathtt{b}$
$\epsilon, \epsilon \to \epsilon$
$\mathtt{a}, \epsilon \to \epsilon$
$\mathtt{b}, \epsilon \to \epsilon$
$\mathtt{a}, \mathtt{a} \to \epsilon$
$\mathtt{b}, \mathtt{b} \to \epsilon$
$\epsilon, \mathtt{\$} \to \epsilon$
start
q_1
q_2
q_3

Accepts aba

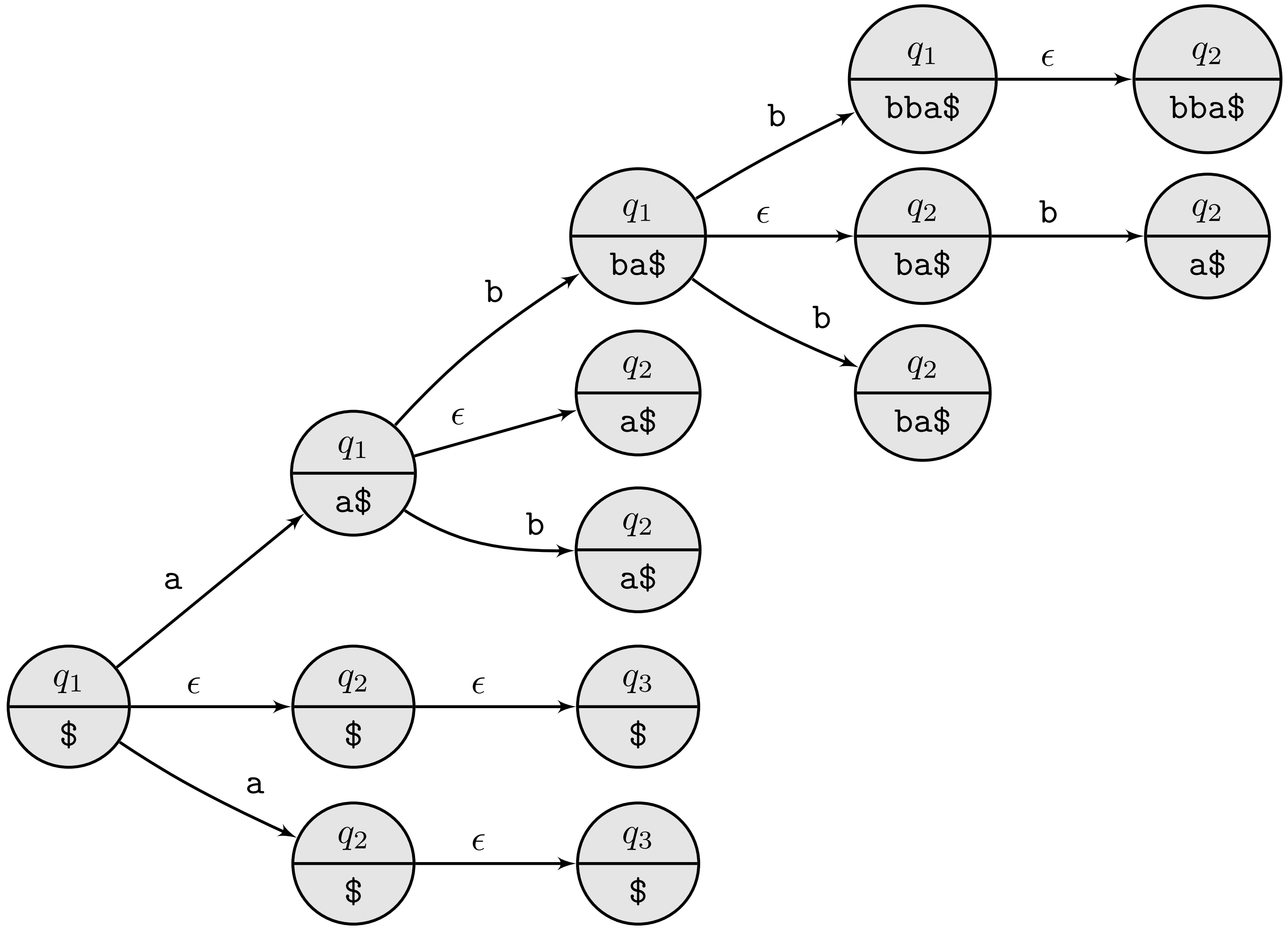
Accepts aba



q_1 / $
a
q_1 / a$
ϵ
q_2 / $
ϵ
q_3 / $
a
q_2 / $
ϵ
q_3 / $
b
q_1 / ba$
ϵ
q_2 / a$
b
q_2 / a$
a
q_2 / $
ϵ
q_3 / $
a
q_1 / aba$
ϵ
q_2 / aba$
ϵ
q_2 / ba$
a
q_2 / ba$

Rejects abb

Rejects abb

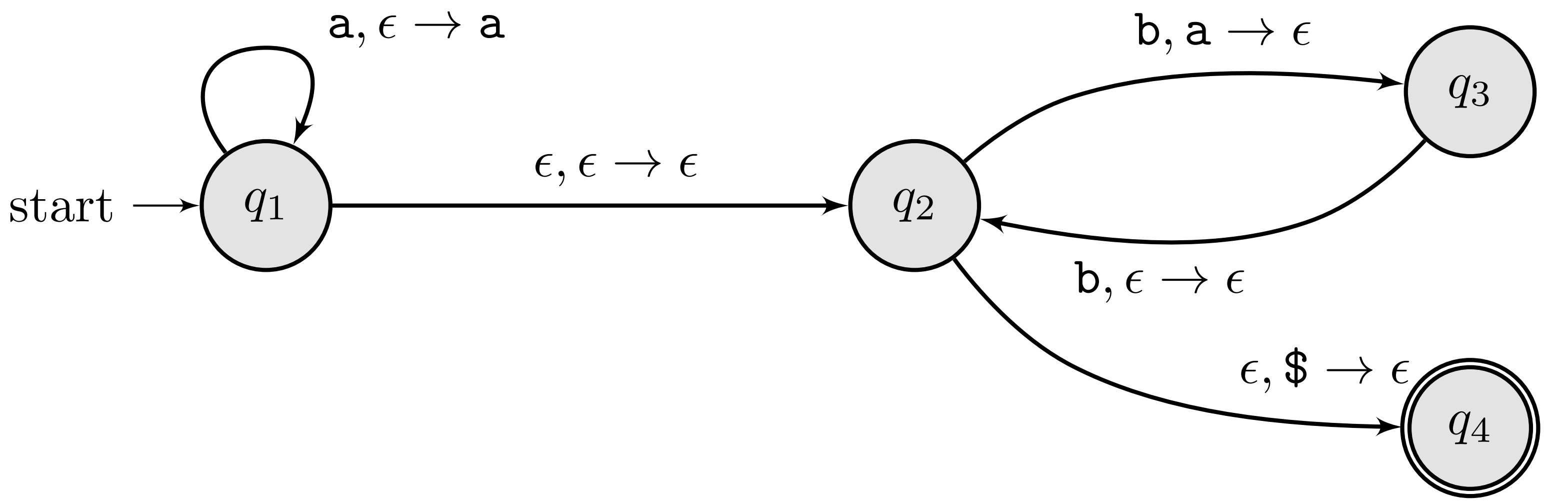

q_1 / \$
q_1 / a\$
q_1 / ba\$
q_1 / bba\$
q_2 / bba\$
q_2 / ba\$
q_2 / a\$
q_2 / ba\$
q_2 / a\$
q_2 / a\$
q_2 / \$
q_3 / \$
q_2 / \$
q_3 / \$
a
ϵ
a
b
ϵ
b
b
ϵ
b
b
ϵ
ϵ
ϵ

Exercise 2

$L_2 = \{a^n b^{2n} \mid n \geq 0\}$

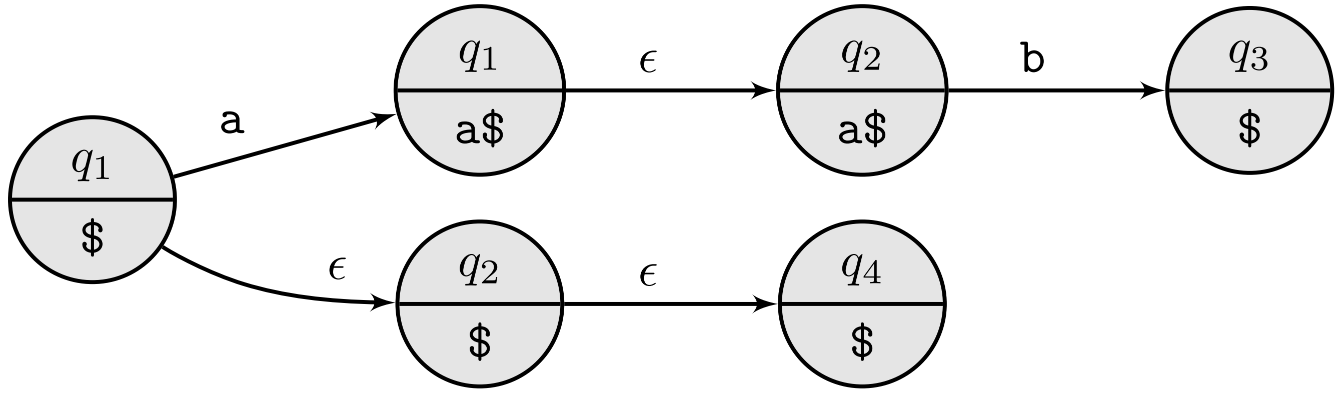
Give a PDA that recognizes L_2 and show it rejects aba and accepts abb

Exercise 2 solution


start
q_1
q_2
q_3
q_4
$\texttt{a}, \epsilon \to \texttt{a}$
$\epsilon, \epsilon \to \epsilon$
$\texttt{b}, \texttt{a} \to \epsilon$
$\texttt{b}, \epsilon \to \epsilon$
$\epsilon, \$ \to \epsilon$

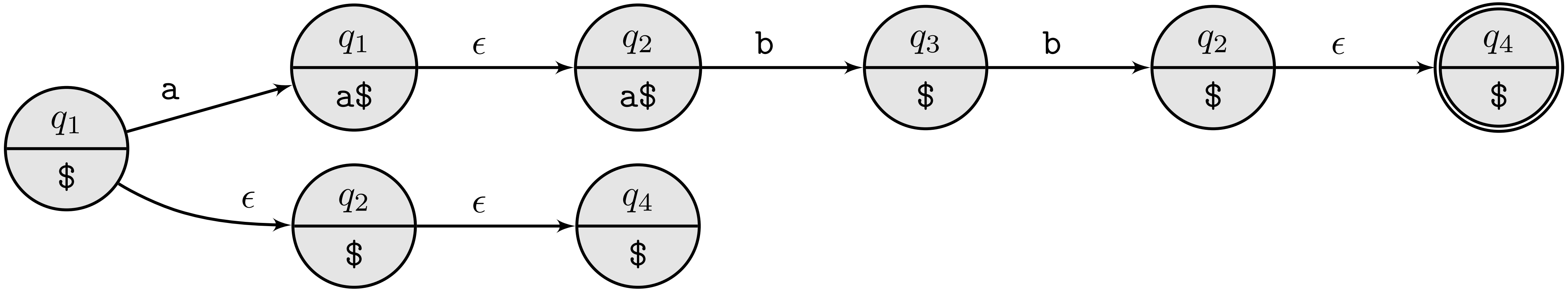
L_2 does not contain aba

L_2 does not contain aba



L_2 contains abb

L_2 contains abb



Main result

Context free languages

Theorem: Language L has a context free grammar if, and only if, L is recognized by some pushdown automaton.

Next

1. We show that from a CFG we can build an equivalent[†] PDA
2. We show that from a PDA we can build an equivalent[†] CFG

[†] Equivalence with respect to recognized languages. Let P be a PDA and C a CFG we say that P is equivalent to C (and vice versa) if, and only if, $L(P) = L(C)$

Converting a CFG into a PDA

Converting a CFG into a PDA

- (1) Initial state pushes S to the stack
- In a loop:
  - (2) Every rule $S \to w$ corresponds to poping S and pushing w (in reverse)
  - (3) Pop terminals from stack
  - (4) Empty stack means recognized

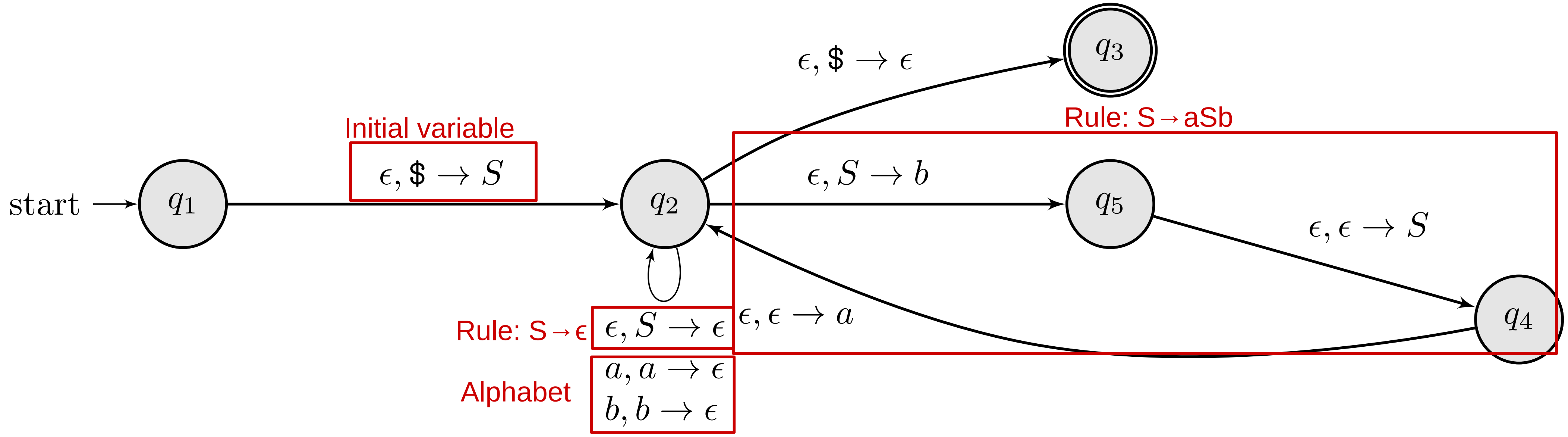
Example $L_3 = \{a^n b^n \mid n \geq 0\}$

$$\boxed{S \to aSb \mid \epsilon}$$

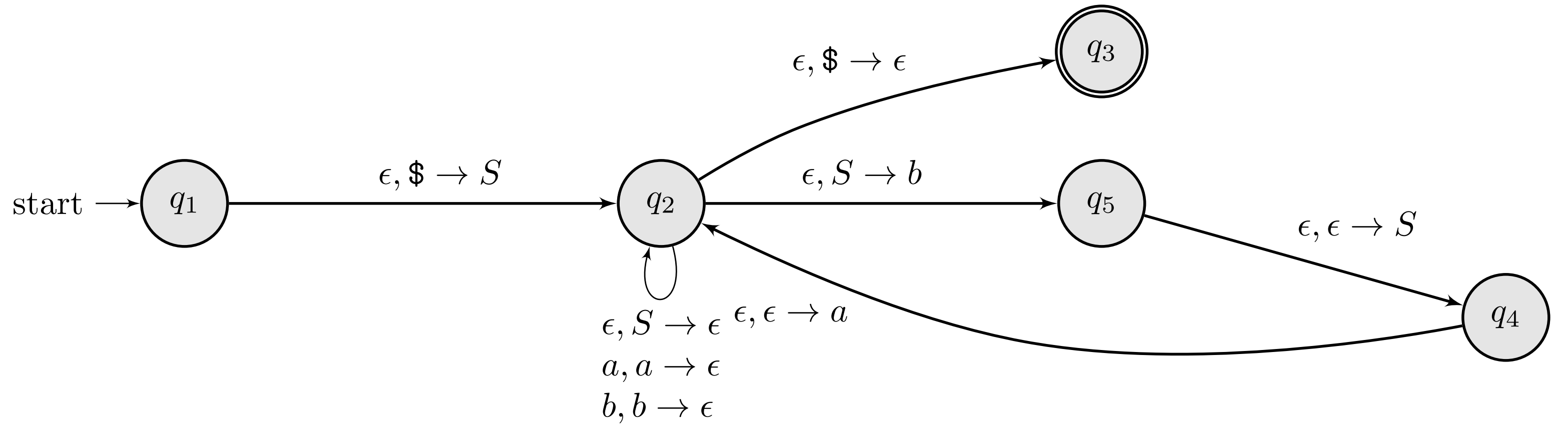
PDA operation	***Output***	***Accept?***
	\$	aabb
(1) $\epsilon, \$ \to S$	S\$	
(2) $\epsilon, S \to aSb$	aSb\$	
(3) $\epsilon, a \to \epsilon$	Sb\$	a
(2) $\epsilon, S \to aSb$	aSbb\$	a
(3) $\epsilon, a \to \epsilon$	Sbb\$	aa
(2) $\epsilon, S \to \epsilon$	bb\$	aa
(3) $\epsilon, b \to \epsilon$	b\$	aab
(3) $\epsilon, b \to \epsilon$	\$	aabb

Overview

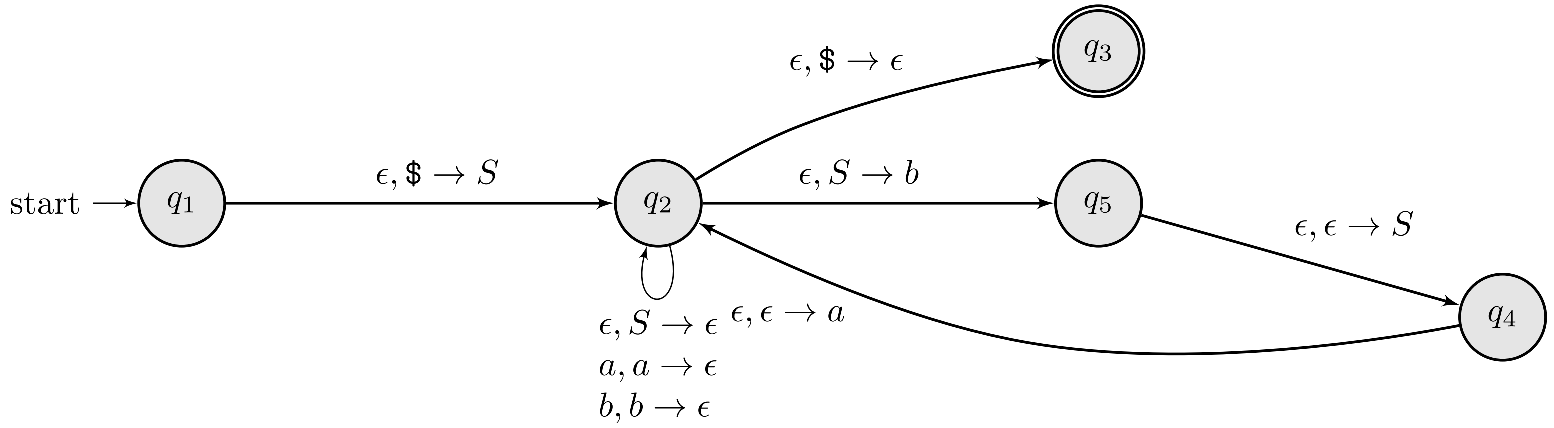
1. **Initial variable:** From the initial state q_1 push the initial variable onto the stack via ϵ and move to the loop state (q_2)
2. **Productions:** For each rule ($S \to aSb$), perform a multi-push edge via ϵ from q_2 back to q_2, by popping popping the variable of the rule S and performing a multi-push of the body aSb.
3. **Alphabet:** For each letter a of the grammar draw a self loop to q_2 that reads a and pops a from the stack
4. **Final transition:** Once the stack is empty transition to the final state q_3 via ϵ

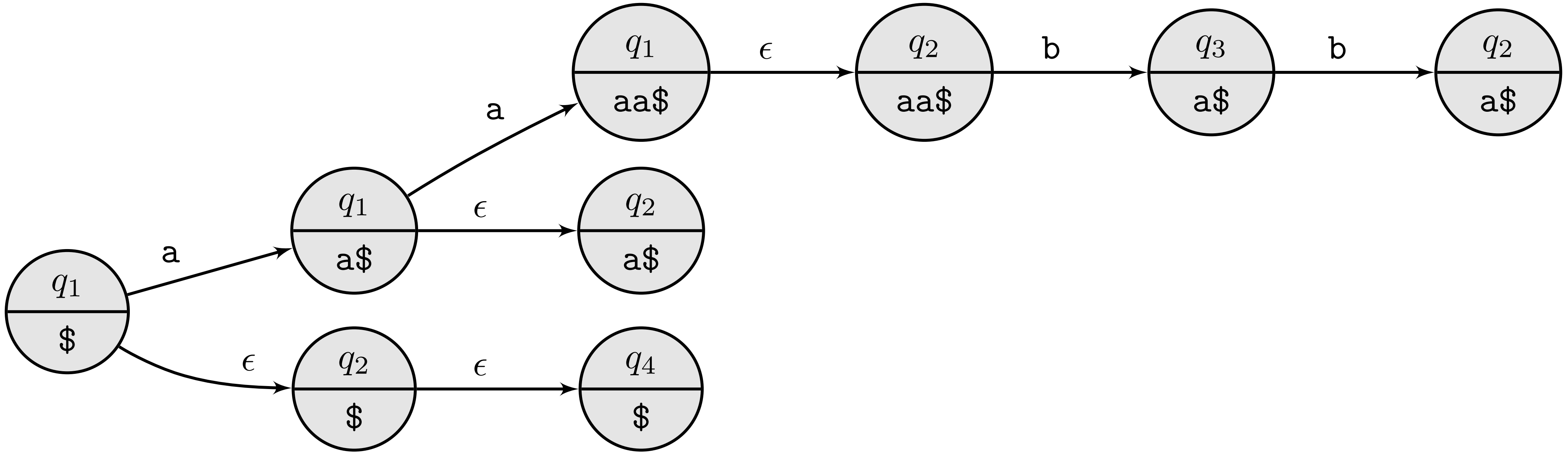


aabb is in $L_3 = \{a^n b^n \mid n \geq 0\}$, show acceptance



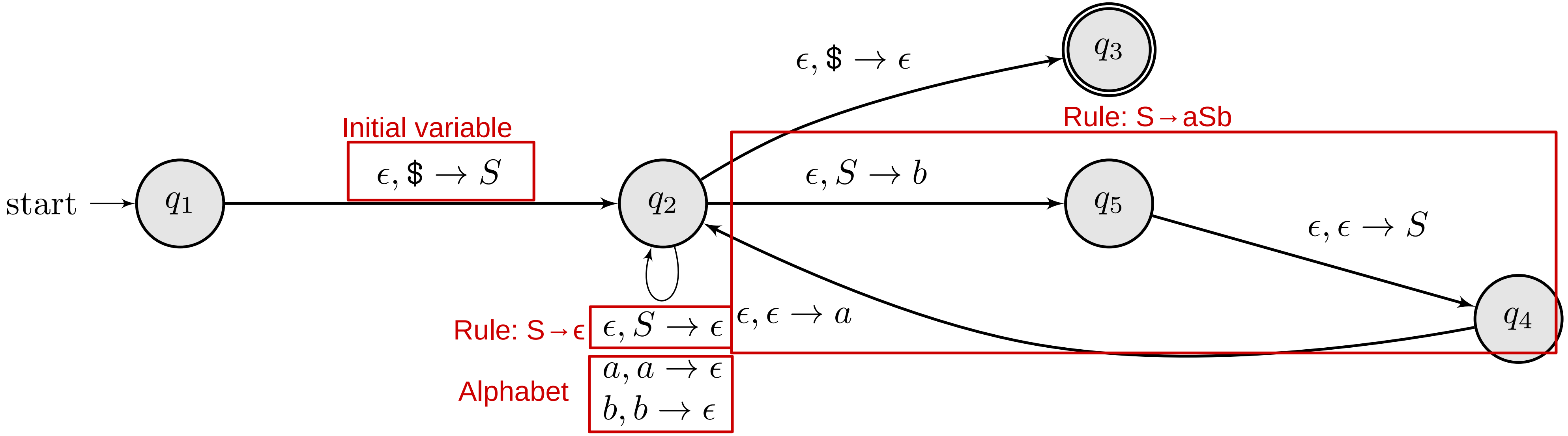
aabb is in $L_3 = \{a^n b^n \mid n \geq 0\}$, show acceptance





Overview

1. The states q_1, q_2, and q_3 are always in the converted PDA
2. The edge between q_1 and q_2 always pushes the initial variable
3. The edge between q_2 and q_3 is always $\epsilon, \$ \to \epsilon$
4. There is always a self loop for each letter in the alphabet of $a, a \to \epsilon$
5. The only difficulty is **generating the substitution rules**



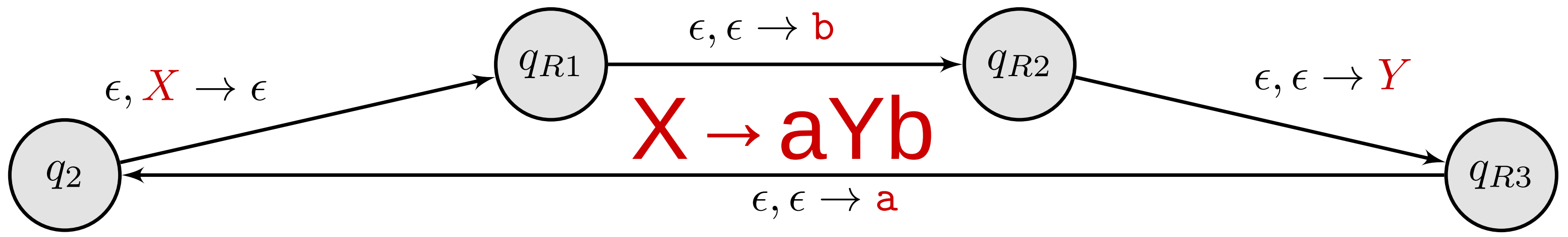
How to encode $\boldsymbol{S \to aSb}$?

(multi push)

Encoding multi-push productions

By example $X \to aYb$

1. reverse the production, example:
   $X \to aYb$ yields bYa.
2. Create one state R_i for each variable/terminal in the reversed string, each transition pushes a variable/terminal of the **reversed** string


q_2
$\epsilon, X \to \epsilon$
q_{R1}
$\epsilon, \epsilon \to$ b
q_{R2}
$\epsilon, \epsilon \to Y$
q_{R3}
$\epsilon, \epsilon \to$ a
X → aYb

Note: In the book (and in my diagrams) I merge the first two transitions. This is equivalent to the above method; you can use either, as long as you do it correctly.

Exercise 3

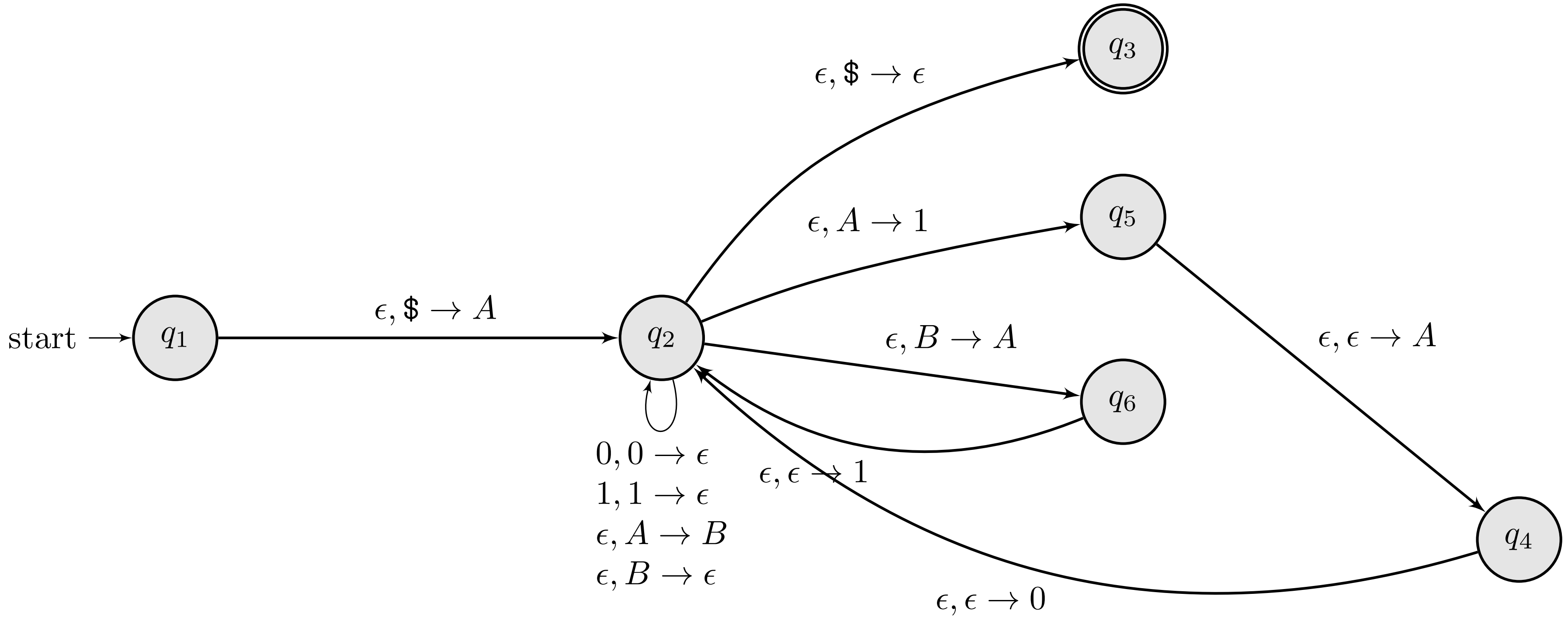
Convert the following grammar into a PDA

$$A \rightarrow 0A1 \mid B$$
$$B \rightarrow 1B \mid \epsilon$$

Exercise 3

Convert the following grammar into a PDA

$$\boxed{\begin{aligned} A &\to 0A1 \mid B \\ B &\to 1B \mid \epsilon \end{aligned}}$$



Converting a PDA into a CFG

Converting a PDA into a CFG

1. modify the PDA into a **simplified** PDA:
    - has a single accepting state
    - empties the stack before accepting
    - every transition is in one of these forms:
        - skips popping and pushes one symbol onto the stack: $\epsilon \to c$
        - pops one symbol off the stack and skips pushing: $c \to \epsilon$

Converting a PDA into a CFG

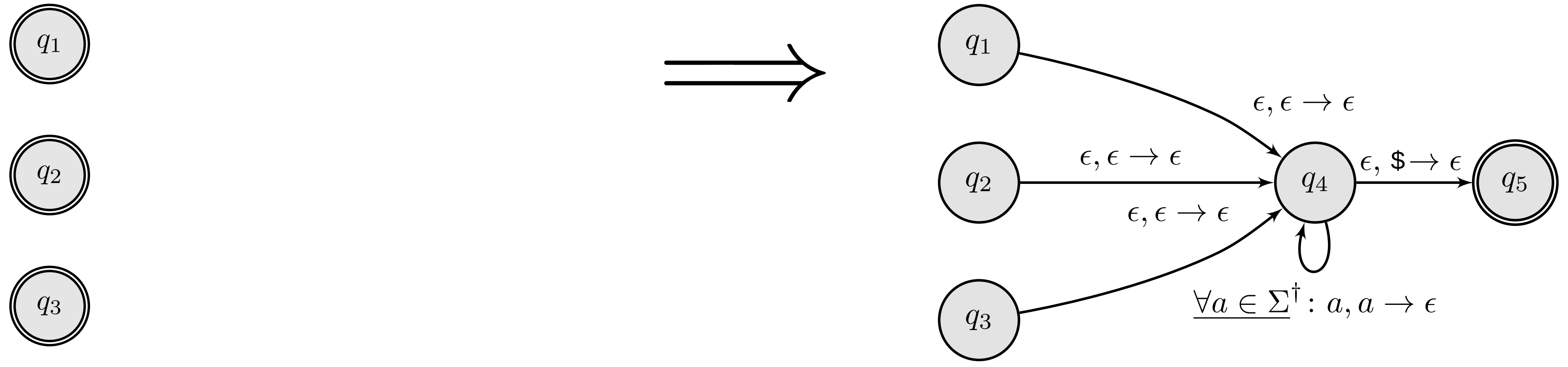
1. modify the PDA into a **simplified** PDA:
   - has a single accepting state
   - empties the stack before accepting
   - every transition is in one of these forms:
     - skips popping and pushes one symbol onto the stack: $\epsilon \to c$
     - pops one symbol off the stack and skips pushing: $c \to \epsilon$
2. given a simplified PDA build a CFG
   - $A_{qq} \to \epsilon$ if $q \in Q$
   - $A_{pq} \to A_{pr} A_{rq}$ if $p, q \in Q$
   - $A_{\underline{pq}} \to \mathtt{a} A_{rs} \mathtt{b}$ if $(r, \mathtt{u}) \in \delta(\underline{p}, \mathtt{a}, \epsilon)$ and $(\underline{q}, \epsilon) \in \delta(s, \mathtt{b}, \mathtt{u})$

Simplifying a PDA

Simplifying a PDA

Transformation 1: Has a single accepting state

Transformation 2: Empties the stack before accepting


q_1
q_2
q_3
$\Longrightarrow$
q_1
q_2
q_3
q_4
q_5
$\epsilon, \epsilon \to \epsilon$
$\epsilon, \epsilon \to \epsilon$
$\epsilon, \epsilon \to \epsilon$
$\epsilon, \$ \to \epsilon$
$\underline{\forall a \in \Sigma}^{\dagger} : a, a \to \epsilon$

† Notation $\underline{\forall a \in \Sigma}$ means that there will be one edge $a, a \to \epsilon$ per $a \in \Sigma$

Simplifying a PDA

Transformation 3

Every transition is in one of these forms:

- skips popping and pushes one symbol onto the stack: $\epsilon \to c$
- pops one symbol off the stack and skips pushing: $c \to \epsilon$

Case 1

$$\begin{tikzcd}
q_1 \arrow[r, "{a, b \to c}"] & q_2 & \Longrightarrow & q_1 \arrow[r, "{a, \epsilon \to b}"] & q_3 \arrow[r, "{\epsilon, b \to \epsilon}"] & q_2
\end{tikzcd}$$

Case 2

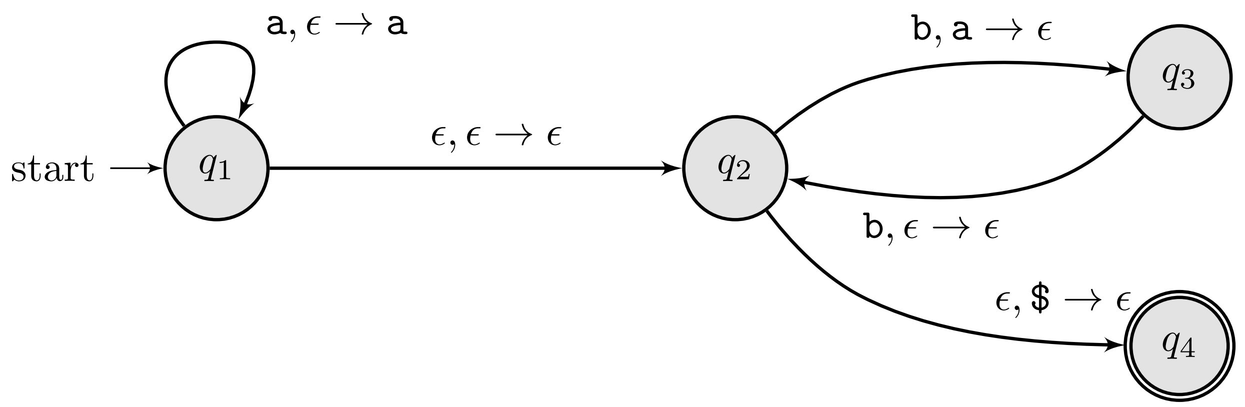
$$\begin{tikzcd}
q_1 \arrow[r, "{a, \epsilon \to \epsilon}"] & q_2 & \Longrightarrow & q_1 \arrow[r, "{a, \epsilon \to b}"] & q_3 \arrow[r, "{\epsilon, b \to \epsilon}"] & q_2
\end{tikzcd}$$

Example 4

Simplified PDA

- single accepting state
- empties the stack before accepting
- every transition is in one of these forms:
  - $\epsilon \to c$
  - $c \to \epsilon$

Is it simplified?



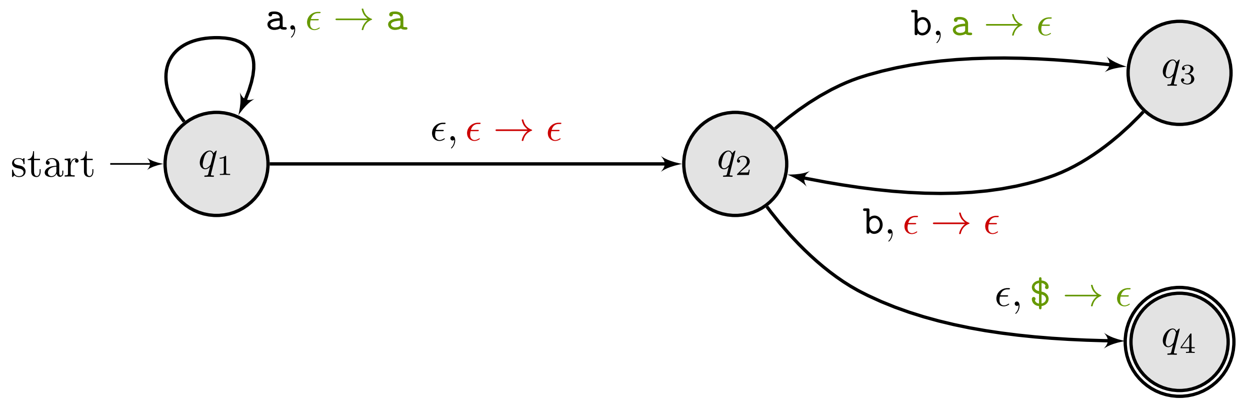
start
q_1
$\mathtt{a}, \epsilon \to \mathtt{a}$
$\epsilon, \epsilon \to \epsilon$
q_2
$\mathtt{b}, \mathtt{a} \to \epsilon$
q_3
$\mathtt{b}, \epsilon \to \epsilon$
$\epsilon, \$ \to \epsilon$
q_4

Example 4

Simplified PDA

- single accepting state
- empties the stack before accepting
- every transition is in one of these forms:
  - $\epsilon \rightarrow c$
  - $c \rightarrow \epsilon$

Is it simplified?

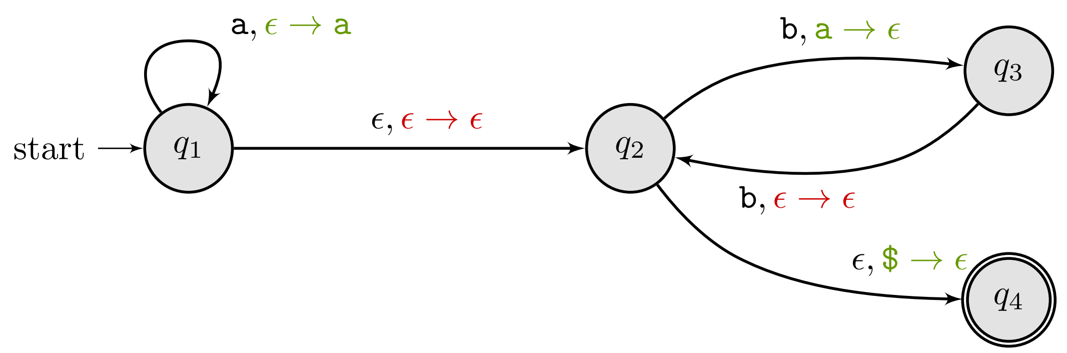


start
q_1
q_2
q_3
q_4
a, $\epsilon \rightarrow$ a
ϵ, $\epsilon \rightarrow \epsilon$
b, a $\rightarrow \epsilon$
b, $\epsilon \rightarrow \epsilon$
ϵ, \$ $\rightarrow \epsilon$

No!

Example 4

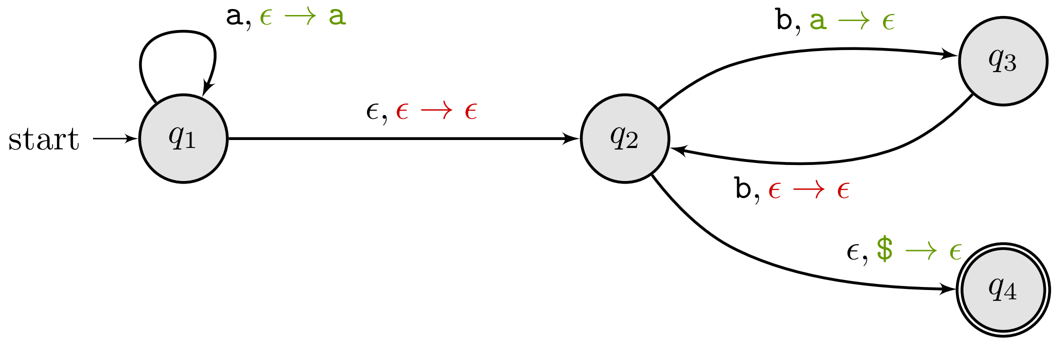
Not Simplified


a, $\epsilon \to$ a
start
q_1
$\epsilon, \epsilon \to \epsilon$
q_2
b, a $\to \epsilon$
q_3
b, $\epsilon \to \epsilon$
ϵ, \$ $\to \epsilon$
q_4

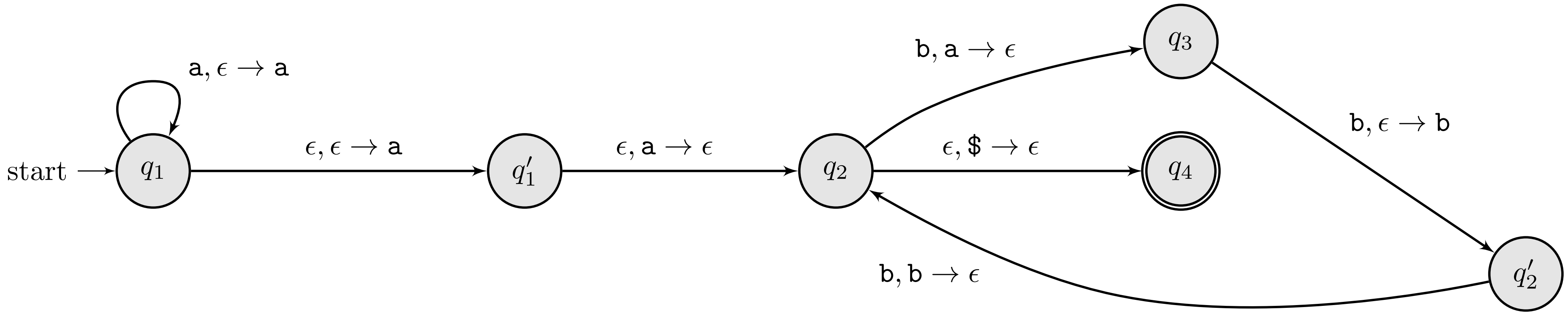
Simplified

Example 4

Not Simplified



Simplified

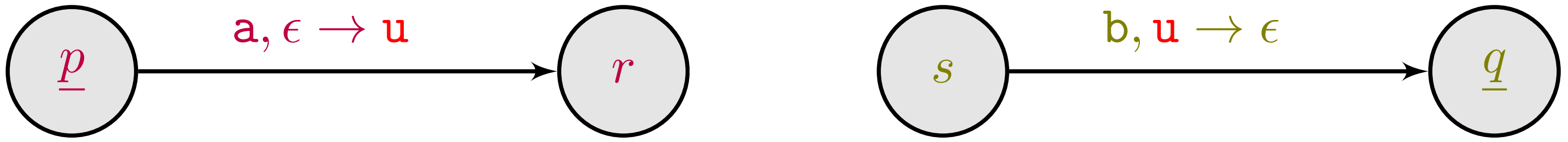


Simplified PDA to CFG

Simplified PDA to CFG

Given a simplified PDA build a CFG

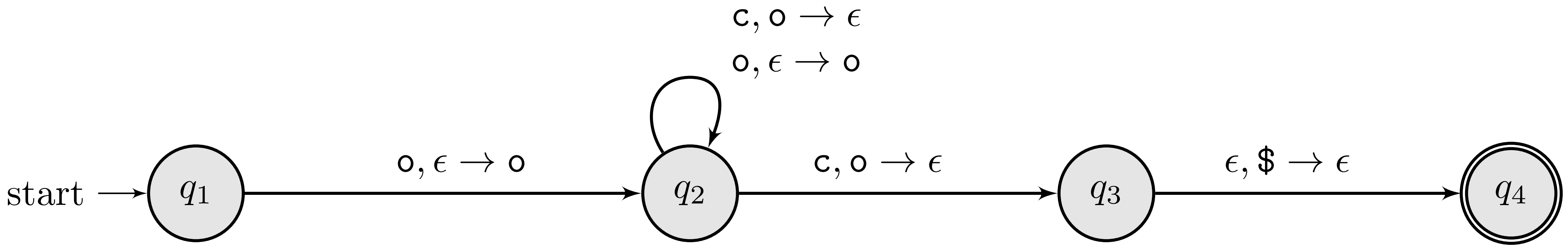
1. $A_{qq} \to \epsilon$ if $q \in Q$
2. $A_{pq} \to A_{pr} A_{rq}$ if $p, r, q \in Q$
3. $A_{\underline{pq}} \to \mathtt{a} A_{rs} \mathtt{b}$ if $(r, \mathtt{u}) \in \delta(\underline{p}, \mathtt{a}, \epsilon)$ and $(\underline{q}, \epsilon) \in \delta(s, \mathtt{b}, \mathtt{u})$


$\underline{p}$
$\mathtt{a}, \epsilon \to \mathtt{u}$
r
s
$\mathtt{b}, \mathtt{u} \to \epsilon$
$\underline{q}$

```
for p-[a, x → u]→r in transitions:      # for every transition
  if x is epsilon:
    for s-[b, _u → x]→q in transitions: # for every transition
      if _u == u and x is epsilon:
        yield A_pq → a A_rs b
```

Example 5

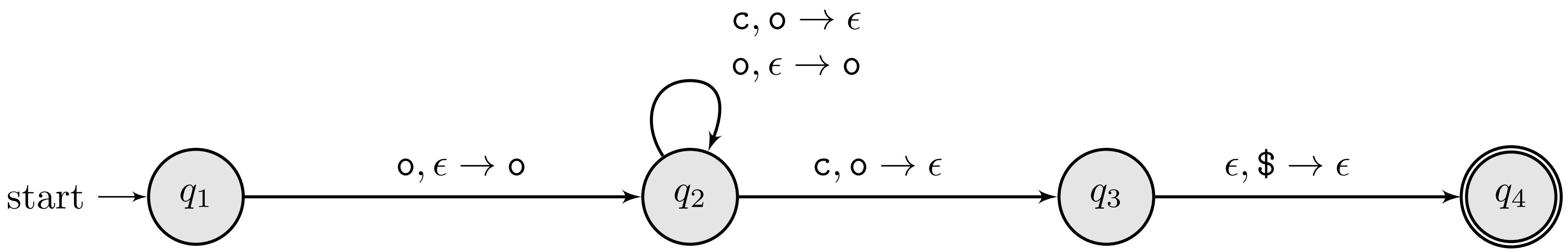
Balanced parenthesis that are wrapped inside an outermost parenthesis.


$$\begin{tikzcd}
\text{start} \arrow[r] & q_1 \arrow[r, "\mathtt{o},\epsilon \to \mathtt{o}"] & q_2 \arrow[loop above, "{\mathtt{c},\mathtt{o} \to \epsilon \atop \mathtt{o},\epsilon \to \mathtt{o}}"] \arrow[r, "\mathtt{c},\mathtt{o} \to \epsilon"] & q_3 \arrow[r, "\epsilon,\$ \to \epsilon"] & q_4
\end{tikzcd}$$

Is this PDA simplified?

Example 5

Balanced parenthesis that are wrapped inside an outermost parenthesis.

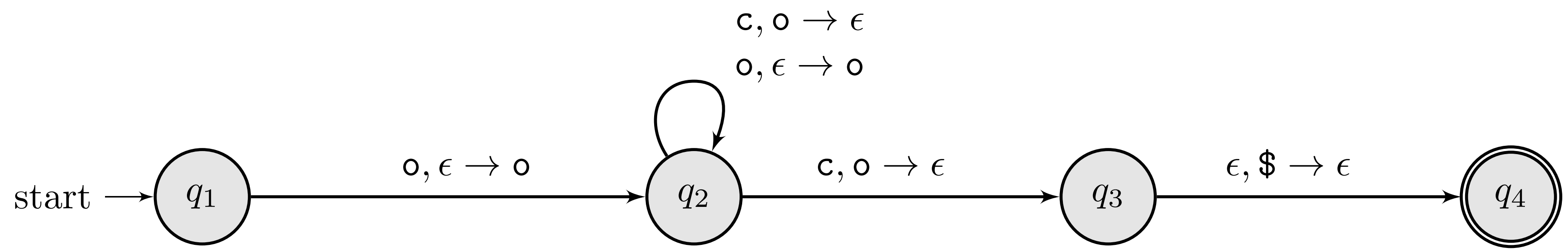


$\mathtt{c}, \mathtt{o} \to \epsilon$
$\mathtt{o}, \epsilon \to \mathtt{o}$
start
q_1
$\mathtt{o}, \epsilon \to \mathtt{o}$
q_2
$\mathtt{c}, \mathtt{o} \to \epsilon$
q_3
$\epsilon, \$ \to \epsilon$
q_4

Is this PDA simplified?

Yes!

Example 5

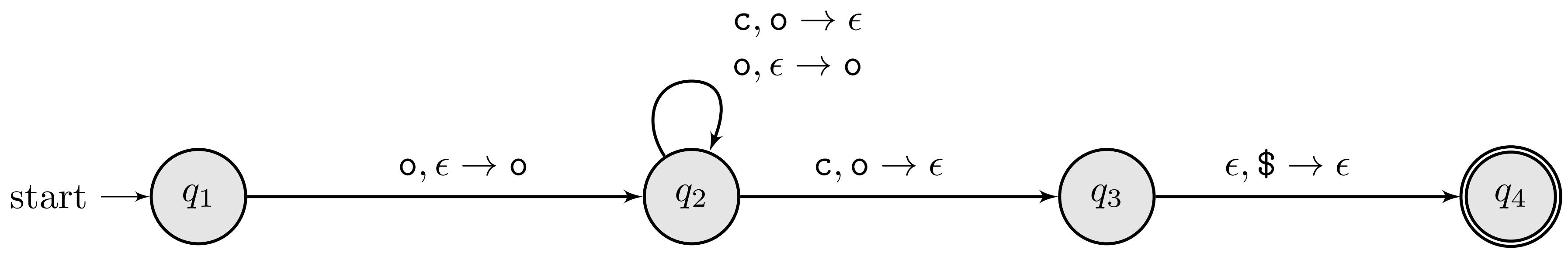


Step 1: $A_{qq} \to \epsilon$ if $q \in Q$

Step 2: $A_{pq} \to A_{pr}A_{rq}$ if $p, r, q \in Q$

$$
\begin{array}{llll}
A_{11} \to \epsilon & A_{1,3} \to A_{1,2}A_{2,3} & A_{2,3} \to A_{2,1}A_{1,3} & A_{3,2} \to A_{3,1}A_{1,2} \\
A_{22} \to \epsilon & A_{1,3} \to A_{1,4}A_{4,3} & A_{2,3} \to A_{2,4}A_{4,3} & A_{3,2} \to A_{3,4}A_{4,2} \\
A_{33} \to \epsilon & A_{1,4} \to A_{1,2}A_{2,4} & A_{2,4} \to A_{2,1}A_{1,4} & A_{3,4} \to A_{3,1}A_{1,4} \\
A_{44} \to \epsilon & A_{1,4} \to A_{1,3}A_{3,4} & A_{2,4} \to A_{2,3}A_{3,4} & A_{3,4} \to A_{3,2}A_{2,4} \\
A_{1,2} \to A_{1,3}A_{3,2} & A_{2,1} \to A_{2,3}A_{3,1} & A_{3,1} \to A_{3,2}A_{2,1} & A_{4,1} \to A_{4,2}A_{2,1} \\
A_{1,2} \to A_{1,4}A_{4,2} & A_{2,1} \to A_{2,4}A_{4,1} & A_{3,1} \to A_{3,2}A_{2,1} & A_{4,1} \to A_{4,3}A_{3,1}
\end{array}
$$

Example 5



Step 1: $A_{qq} \to \epsilon$ if $q \in Q$

Step 2: $A_{pq} \to A_{pr} A_{rq}$ if $p, r, q \in Q$

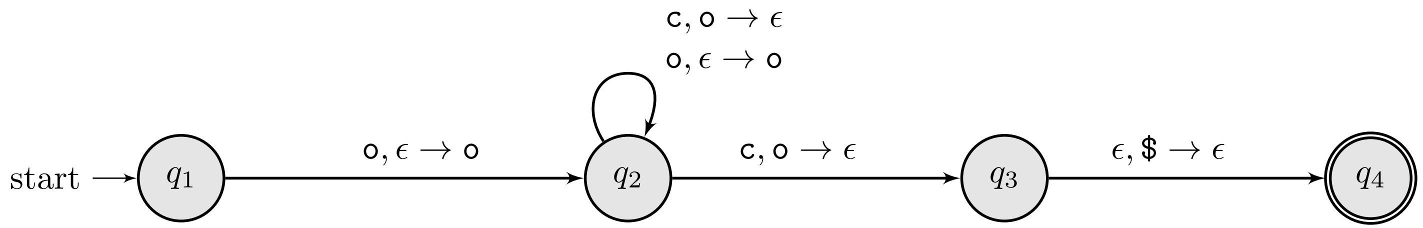
$$A_{4,2} \to A_{4,1} A_{1,2}$$

$$A_{4,2} \to A_{4,3} A_{3,2}$$

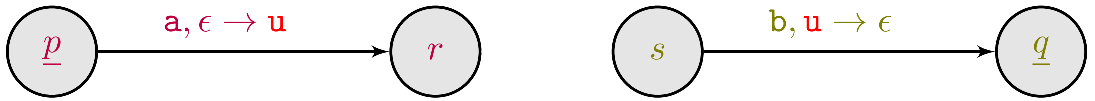
$$A_{4,3} \to A_{4,1} A_{1,3}$$

$$A_{4,3} \to A_{4,2} A_{2,3}$$

Example 5



Step 3: $A_{\underline{pq}} \to \mathtt{a}A_{rs}\mathtt{b}$ if $(r, \mathtt{u}) \in \delta(\underline{p}, \mathtt{a}, \epsilon)$ and $(\underline{q}, \epsilon) \in \delta(s, \mathtt{b}, \mathtt{u})$



Stack o

Push	Pop
`q1, read o, q2`	
`q2, read o, q2`	
	`q2, read c, q2`
	`q2, read c, q3`

New rules:

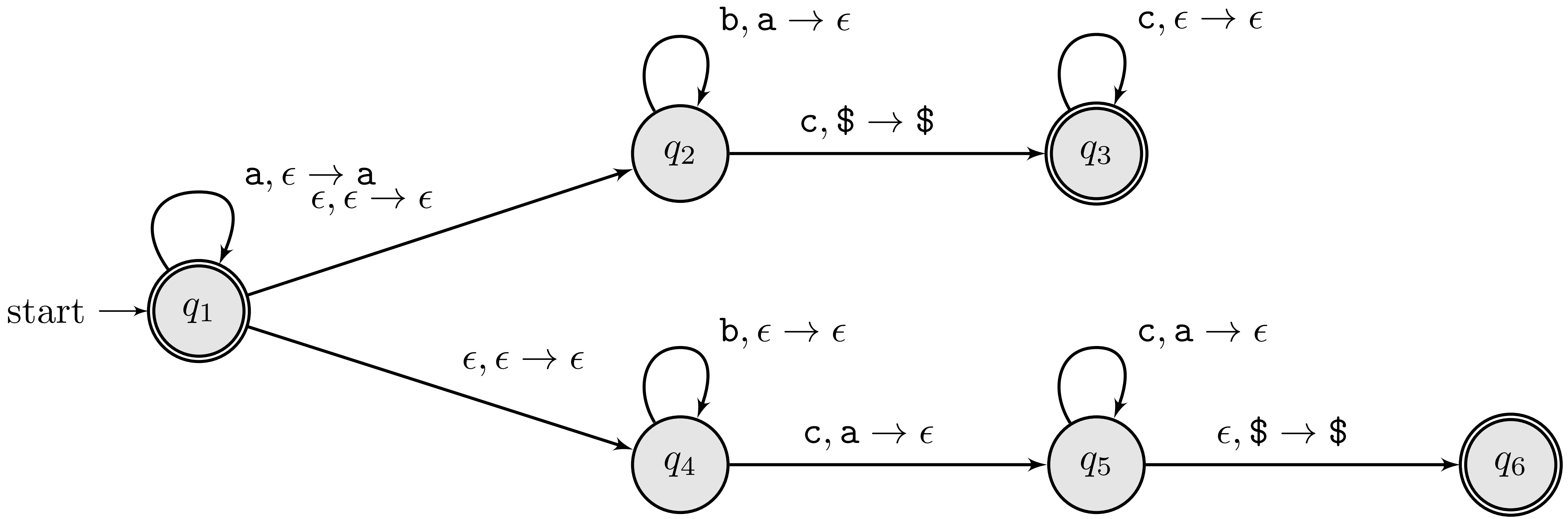
$$A_{1,2} \to oA_{22}c$$
$$A_{1,3} \to oA_{22}c$$
$$A_{2,2} \to oA_{22}c$$
$$A_{2,3} \to oA_{22}c$$

Intuition

- Create a table for each letter being pushed/poped.
- Pair each push with each pop.

Exercise 6

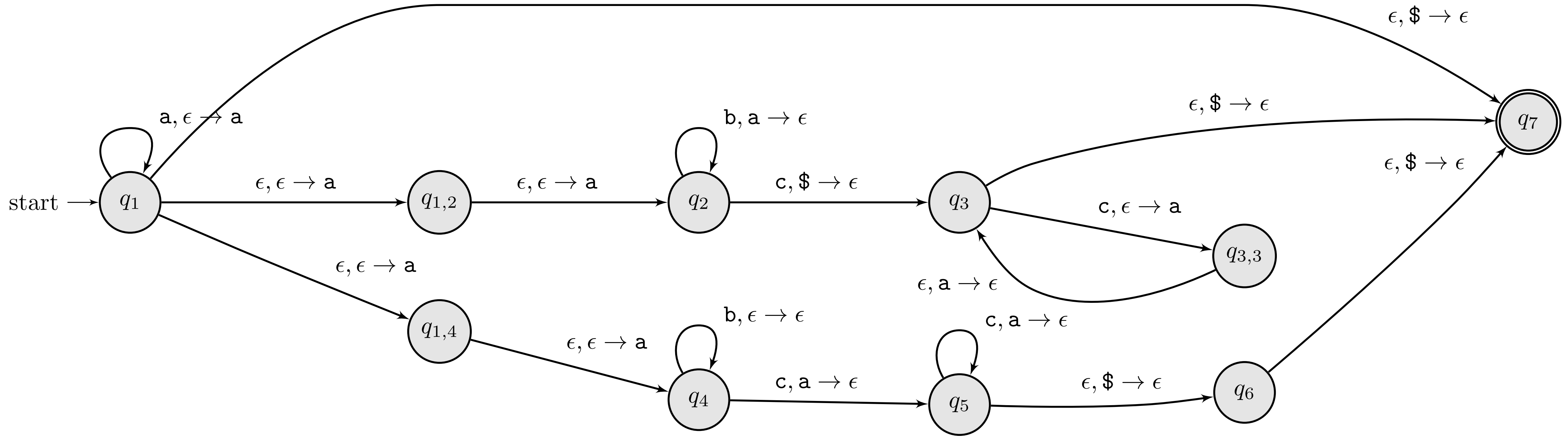
Simplify the PDA below



start
q_1
q_2
q_3
q_4
q_5
q_6
$\mathtt{a}, \epsilon \to \mathtt{a}$
$\epsilon, \epsilon \to \epsilon$
$\mathtt{b}, \mathtt{a} \to \epsilon$
$\mathtt{c}, \$ \to \$$
$\mathtt{c}, \epsilon \to \epsilon$
$\epsilon, \epsilon \to \epsilon$
$\mathtt{b}, \epsilon \to \epsilon$
$\mathtt{c}, \mathtt{a} \to \epsilon$
$\mathtt{c}, \mathtt{a} \to \epsilon$
$\epsilon, \$ \to \$$

Exercise 6

Solution


start
q_1
$q_{1,2}$
q_2
q_3
$q_{3,3}$
$q_{1,4}$
q_4
q_5
q_6
q_7
a, $\epsilon \to$ a
$\epsilon, \epsilon \to$ a
$\epsilon, \epsilon \to$ a
b, a $\to \epsilon$
c, \$ $\to \epsilon$
$\epsilon, \epsilon \to$ a
$\epsilon, \epsilon \to$ a
b, $\epsilon \to \epsilon$
c, a $\to \epsilon$
c, $\epsilon \to$ a
ϵ, a $\to \epsilon$
c, a $\to \epsilon$
ϵ, \$ $\to \epsilon$
ϵ, \$ $\to \epsilon$
ϵ, \$ $\to \epsilon$
ϵ, \$ $\to \epsilon$