Unit-5: ω-regular properties

B. Srivathsan

Chennai Mathematical Institute

NPTEL-course

July - November 2015

Module 4:

Simple properties of NBA

Determinization

Product construction

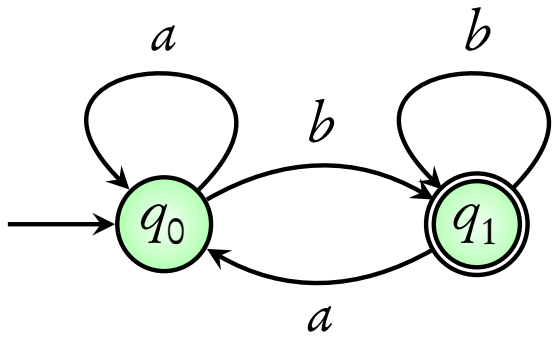
Emptiness

Complementation

Union

Deterministic Büchi Automata

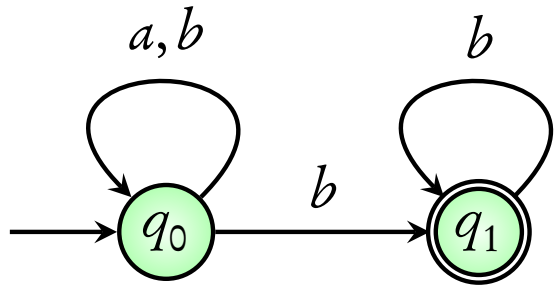
Words where b occurs infinitely often



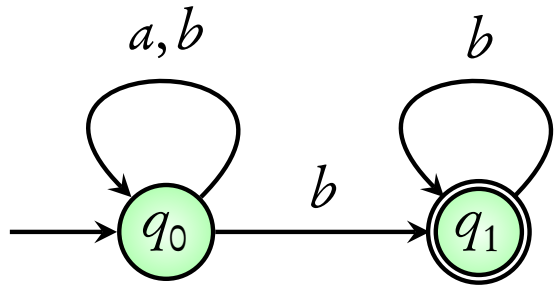
- Single initial state
- From every state - on an alphabet, there is a **unique transition**

Question: Can every NBA be converted to an **equivalent** DBA?

$(a+b)^*b^\omega$: a occurs only finitely often

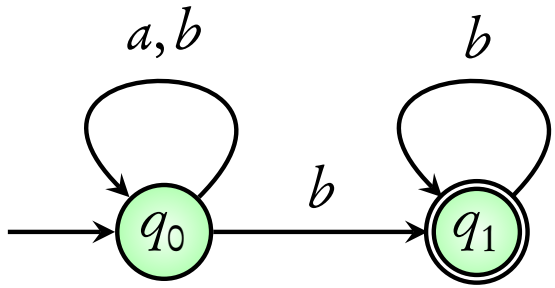


$(a+b)^*b^\omega$: a occurs only finitely often



- Automaton has to **guess** the point from where only b occurs
- A deterministic Büchi automaton cannot make this guess

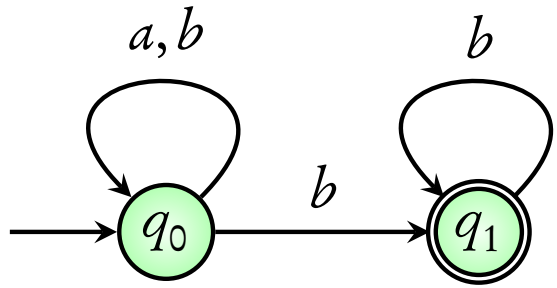
$(a+b)^*b^\omega$: a occurs only finitely often



- Automaton has to **guess** the point from where only b occurs
- A deterministic Büchi automaton cannot make this guess

The above language **cannot be** accepted by a DBA

$(a+b)^*b^\omega$: a occurs only finitely often



- Automaton has to **guess** the point from where only b occurs
- A deterministic Büchi automaton cannot make this guess

The above language **cannot be** accepted by a DBA

Theorem 4.50 (Page 190) of *Principles of Model Checking*, Baier and Katoen. MIT Press (2008)

Determinization

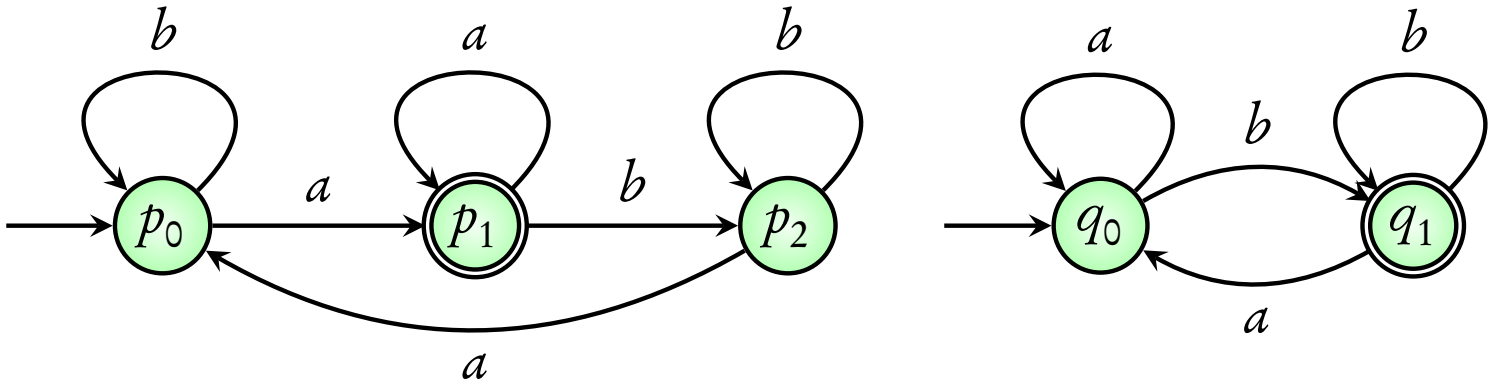
DBA less powerful than NBA

Product construction

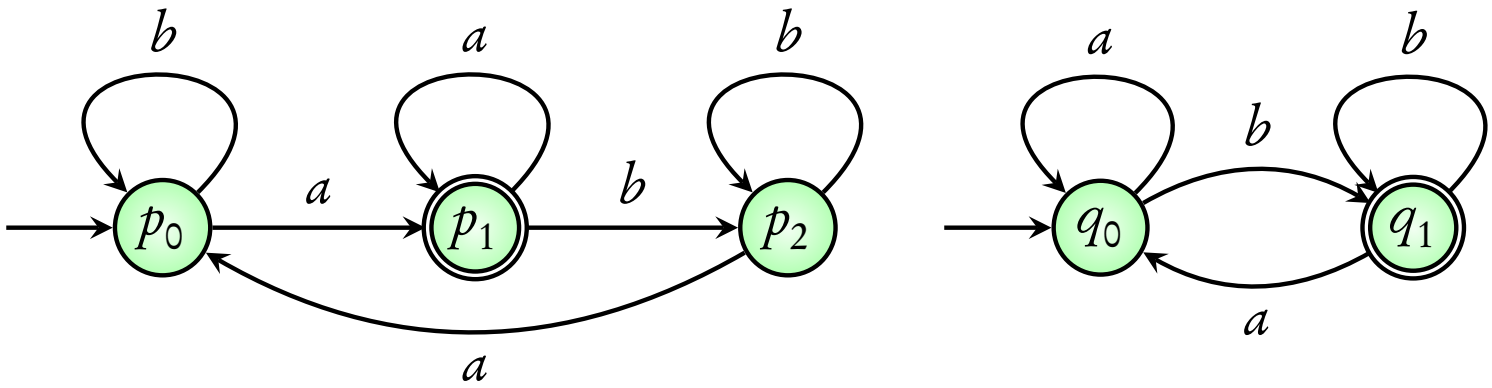
Emptiness

Complementation

Union

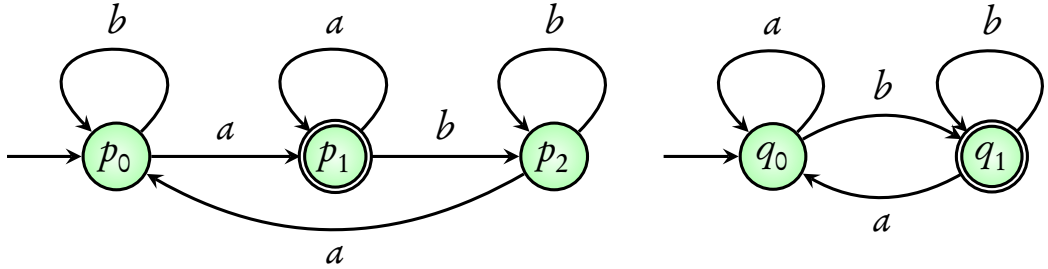


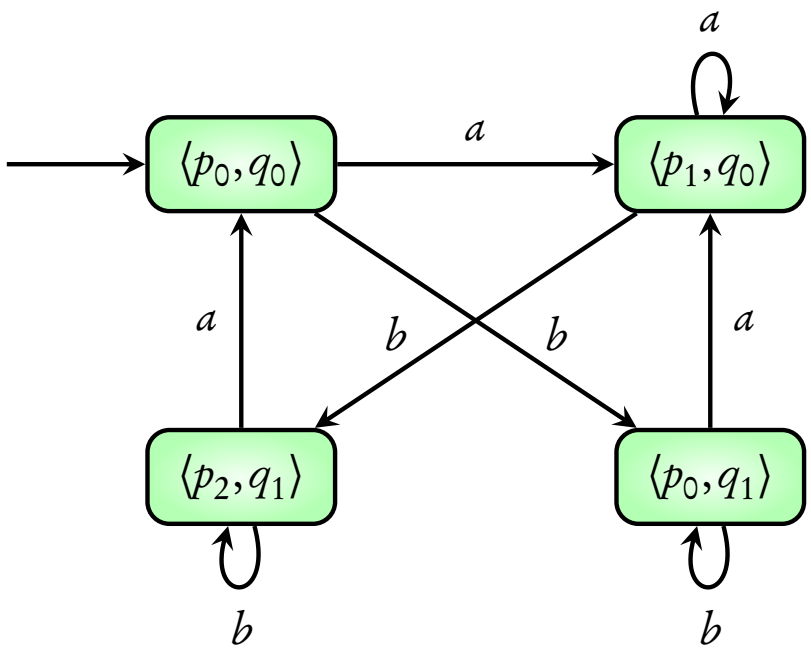
Word $(ab)^{\omega}$ is **accepted by both automata**

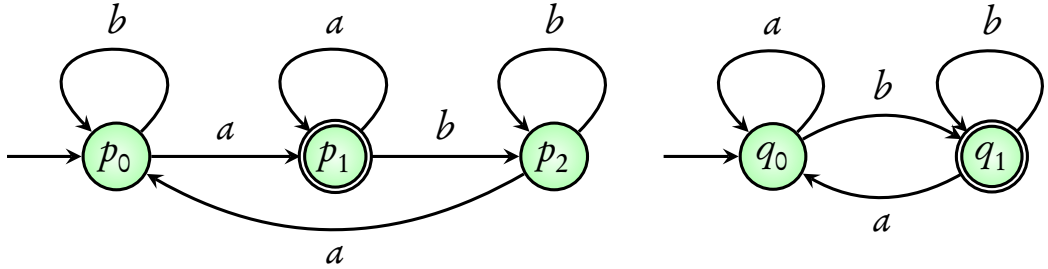


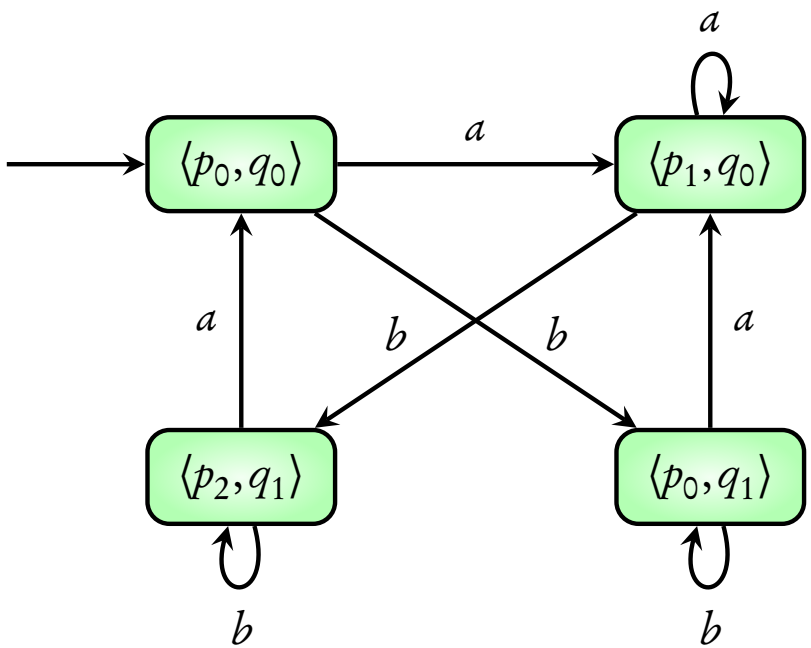
Word $(ab)^\omega$ is **accepted by both automata**

Coming next: The synchronous product construction

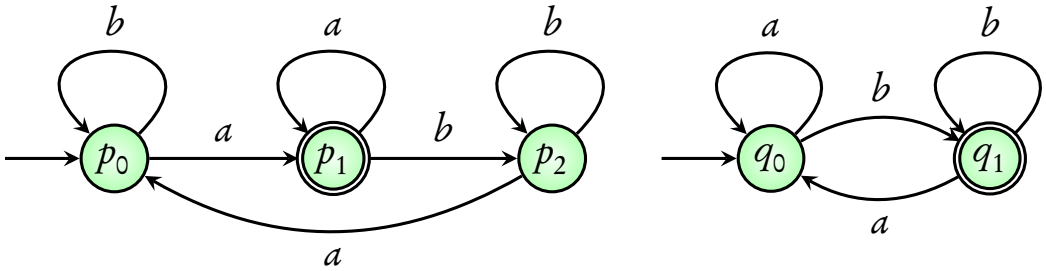

b
a
b
a
b
p_0
p_1
p_2
a
a
b
a
b
q_0
q_1

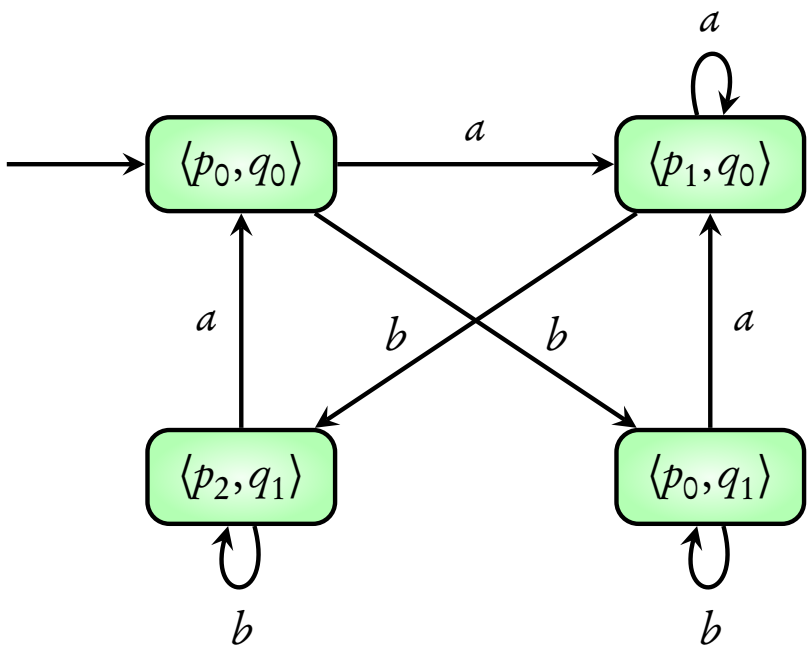

a
$\langle p_0, q_0 \rangle$
a
$\langle p_1, q_0 \rangle$
a
b
b
a
$\langle p_2, q_1 \rangle$
$\langle p_0, q_1 \rangle$
b
b





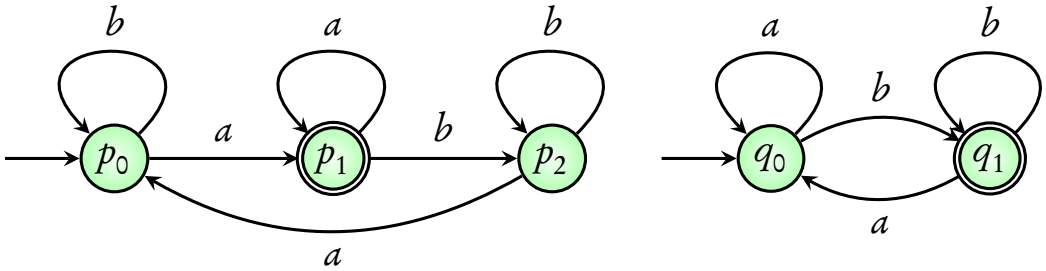
$\langle p_1, q_1 \rangle$ is not present

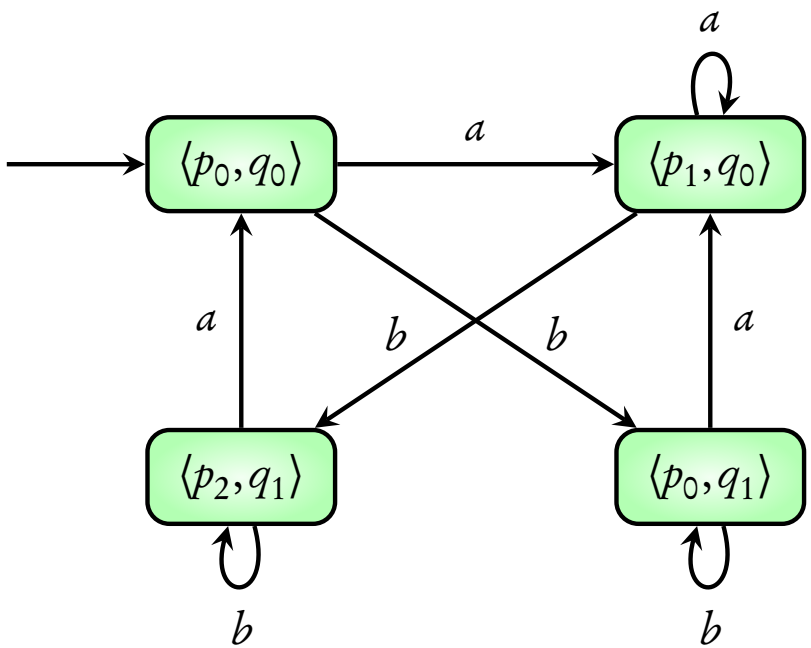




$\langle p_1, q_1 \rangle$ is not present

No accepting state!



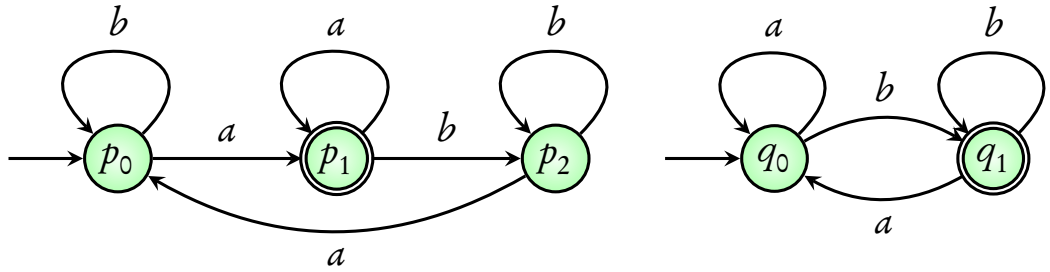


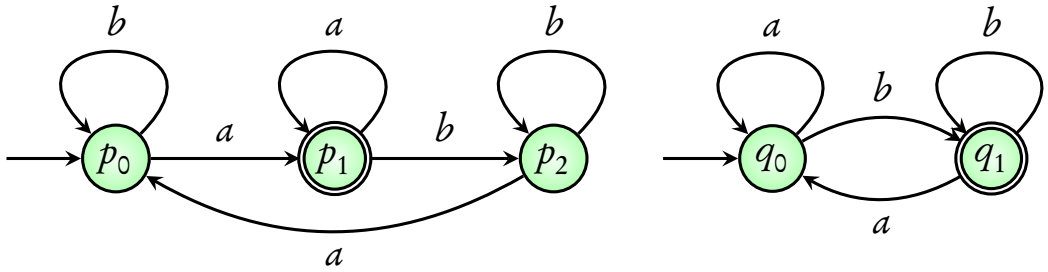
$\langle p_1, q_1 \rangle$ is not present

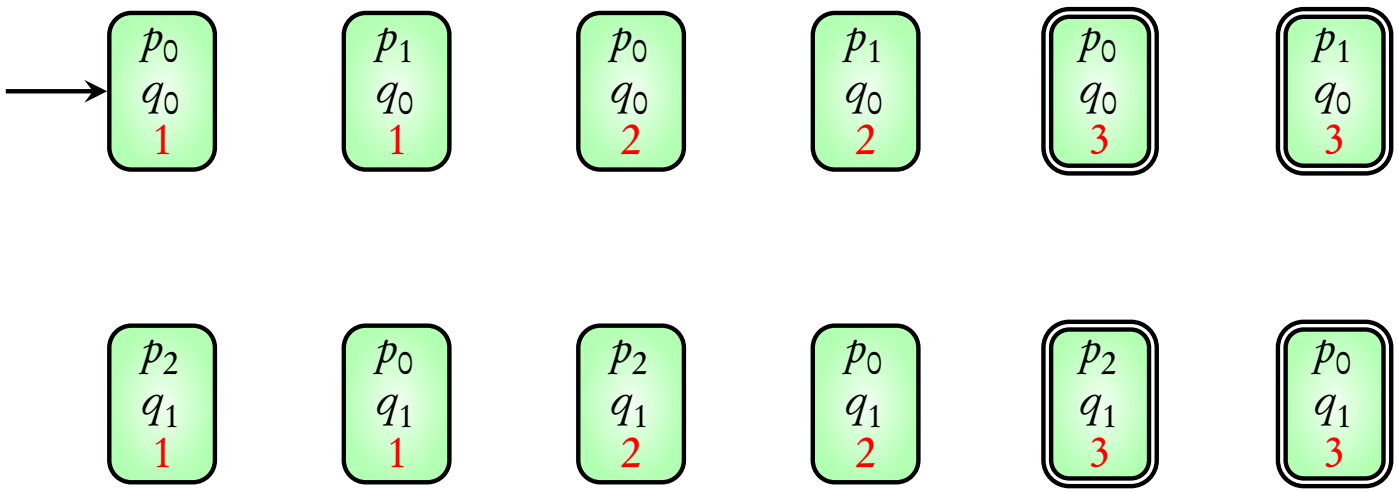
No accepting state!

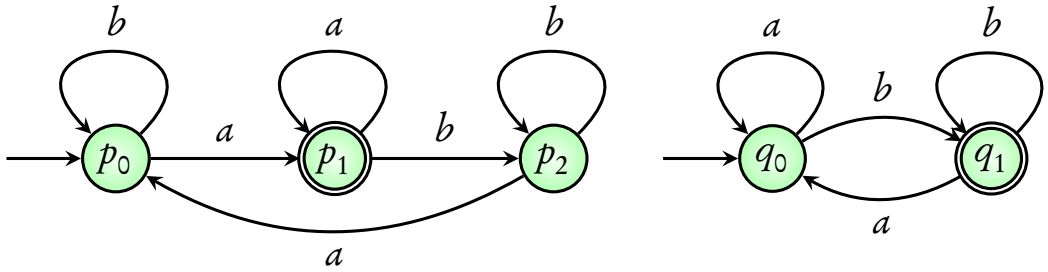
But intersection of the two automata is **not empty**

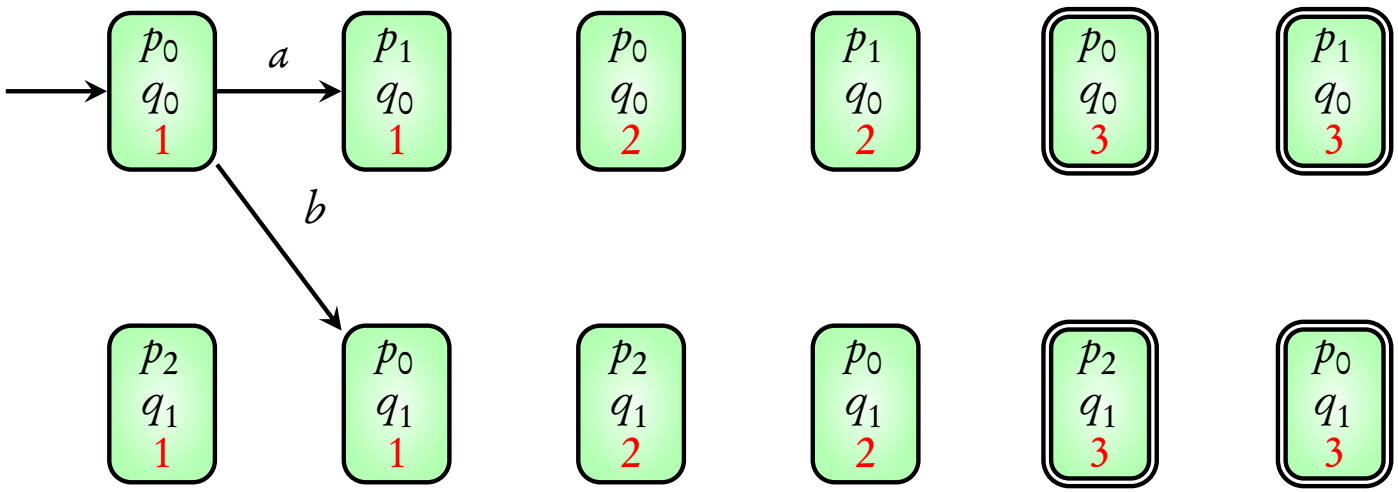
- Need to **modify** the product construction
- **Track** accepting states of **both automata**
- Ensure that **both** automata visit **accepting states infinitely often**

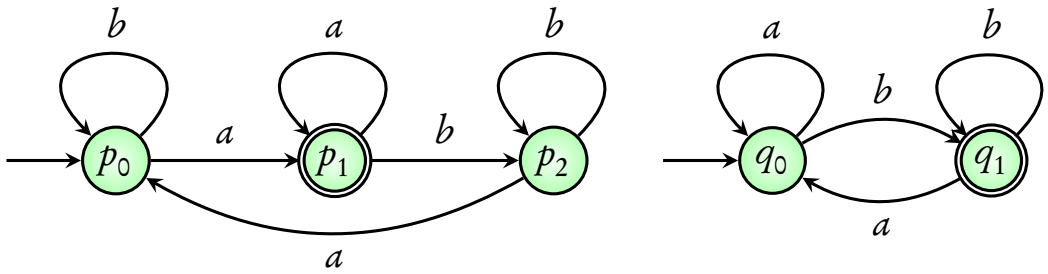

b
a
b
a
b
a
p_0
p_1
p_2
a
b
a
b
q_0
q_1
a

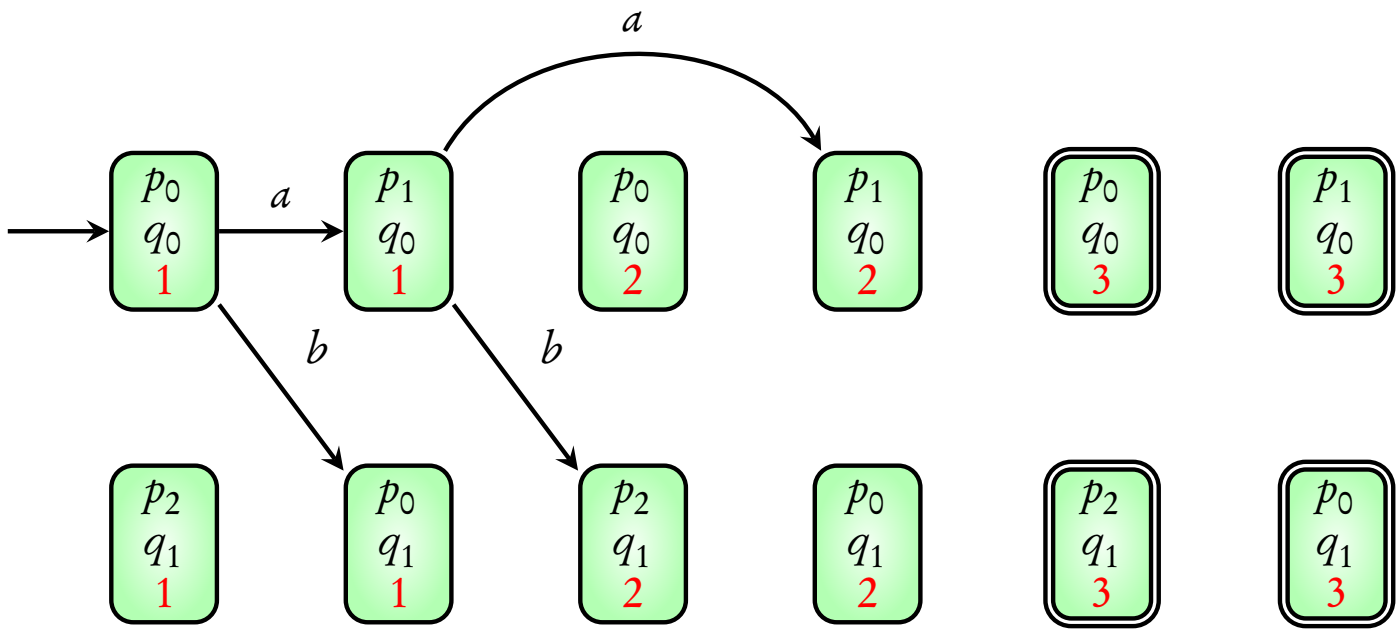

b
a
b
a
b
p0
p1
p2
a
a
b
a
b
q0
q1
a

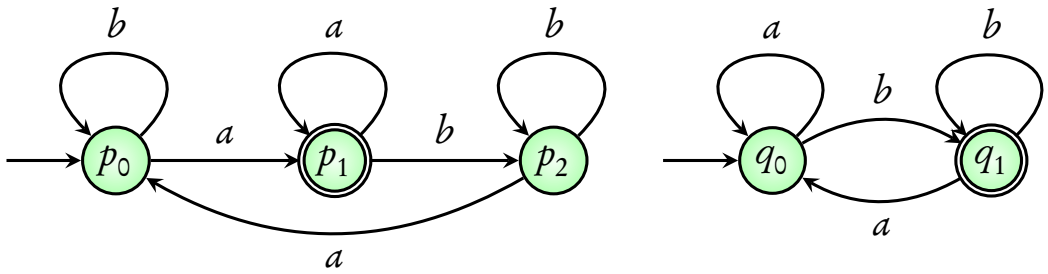

p0 q0 1
p1 q0 1
p0 q0 2
p1 q0 2
p0 q0 3
p1 q0 3
p2 q1 1
p0 q1 1
p2 q1 2
p0 q1 2
p2 q1 3
p0 q1 3

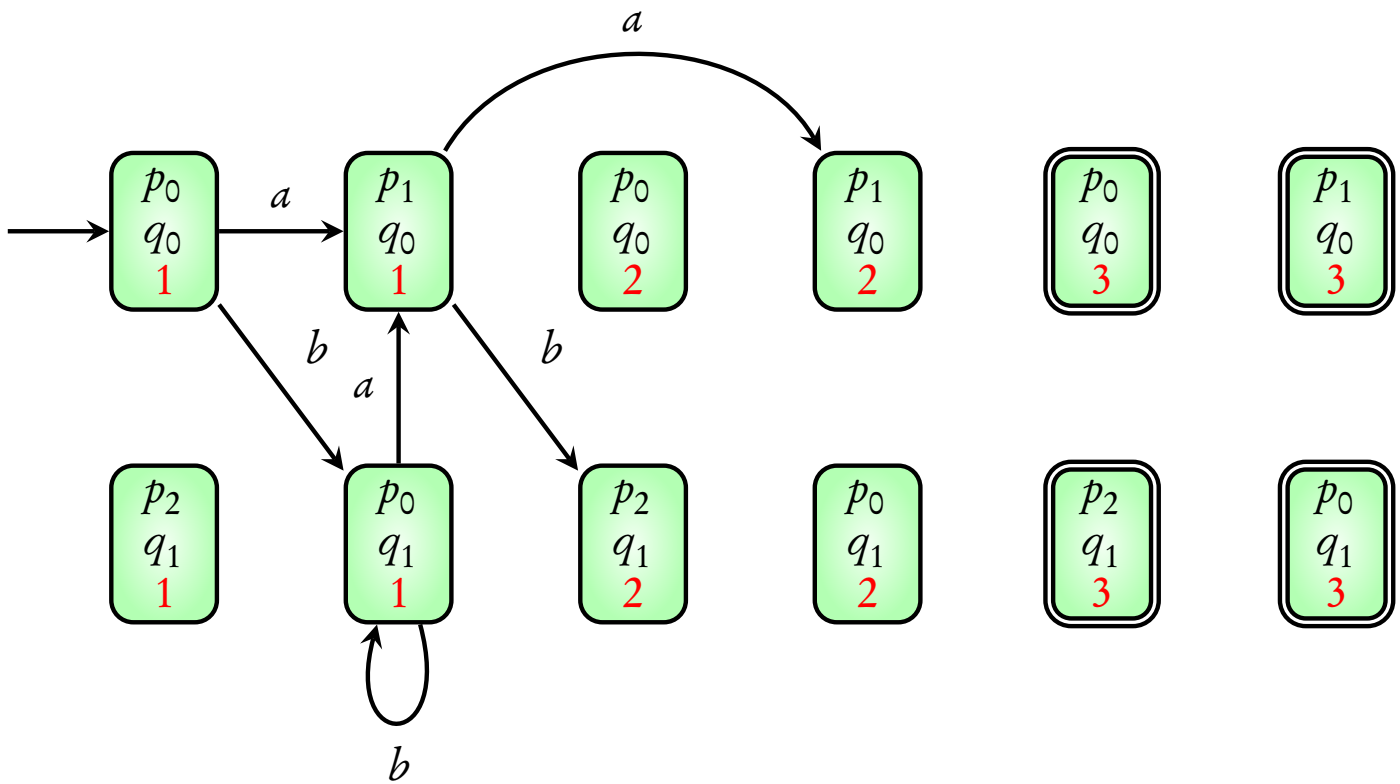

b
a
b
a
b
p0
p1
p2
a
a
b
b
q0
q1
a

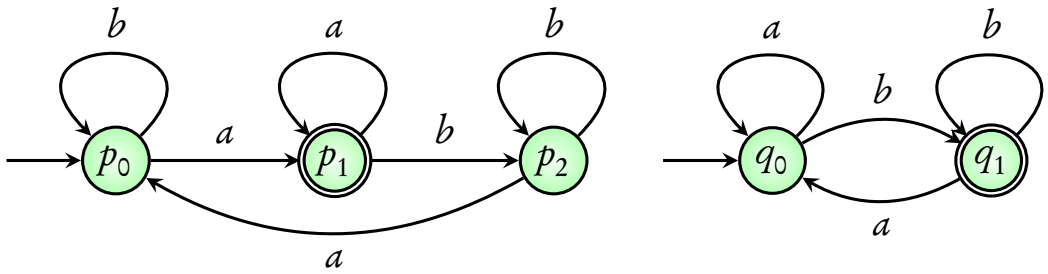

p0 q0 1
a
p1 q0 1
p0 q0 2
p1 q0 2
p0 q0 3
p1 q0 3
b
p2 q1 1
p0 q1 1
p2 q1 2
p0 q1 2
p2 q1 3
p0 q1 3

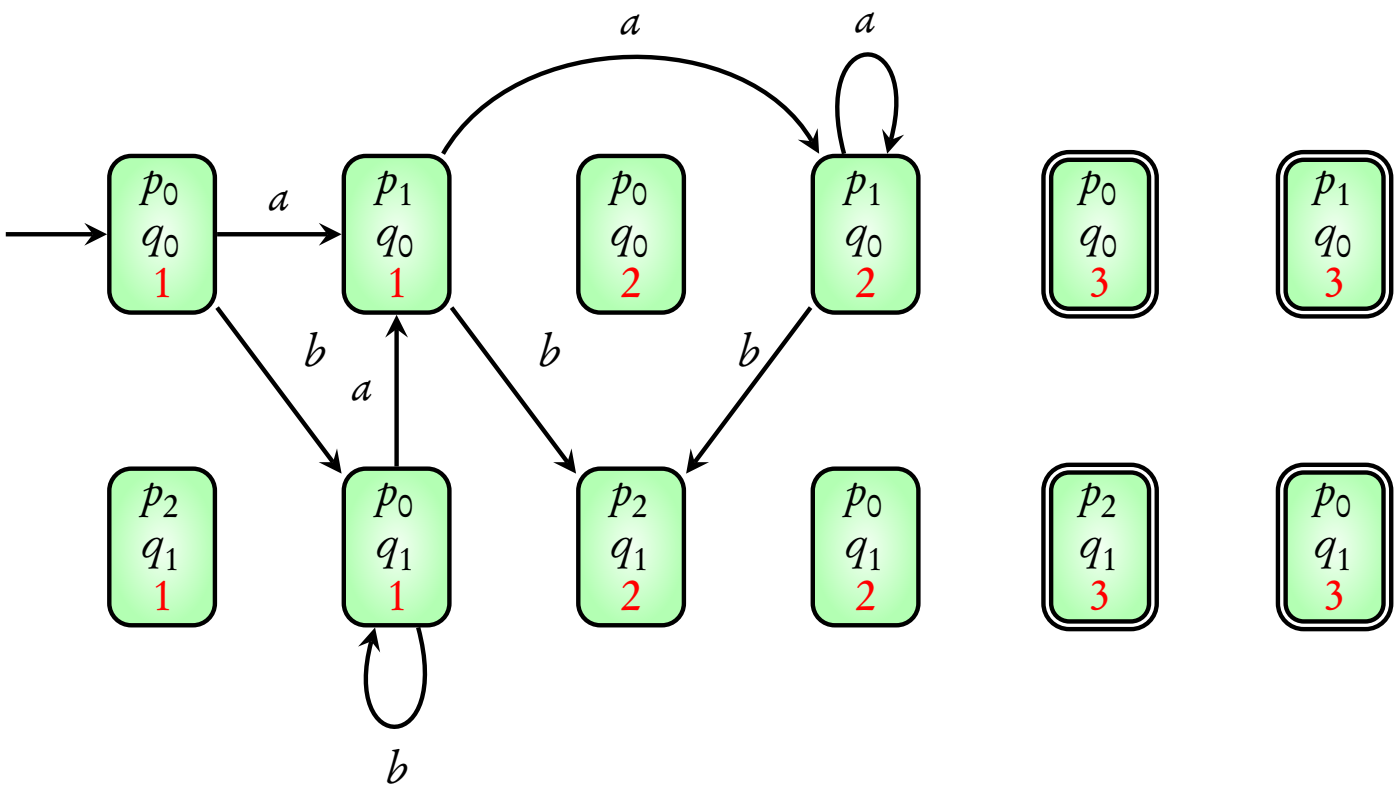

b
a
b
a
b
p0
p1
p2
a
a
b
b
q0
q1
a

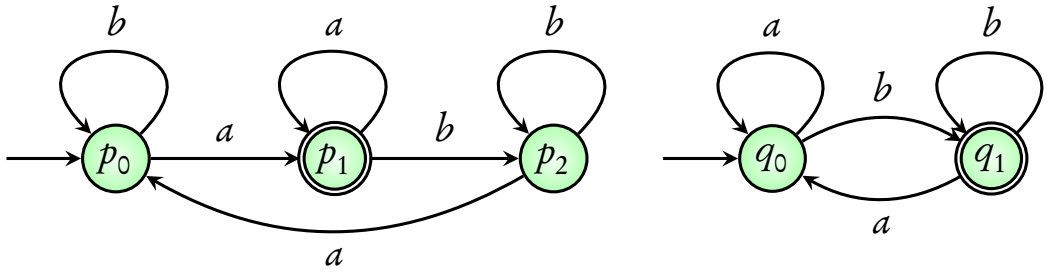

a
p0 q0 1
a
p1 q0 1
p0 q0 2
p1 q0 2
p0 q0 3
p1 q0 3
b
b
p2 q1 1
p0 q1 1
p2 q1 2
p0 q1 2
p2 q1 3
p0 q1 3

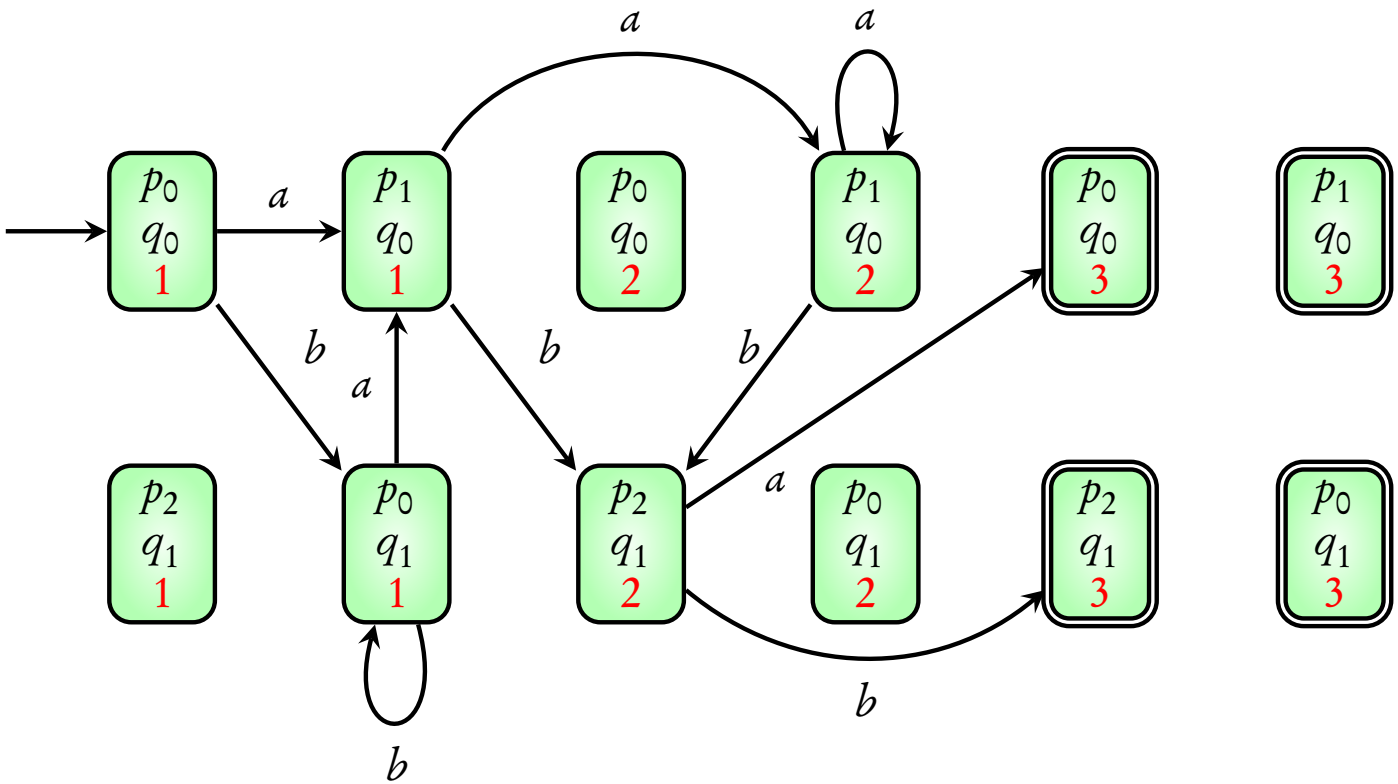

b
a
b
a
b
a
p_0
p_1
p_2
a
b
a
b
a
q_0
q_1

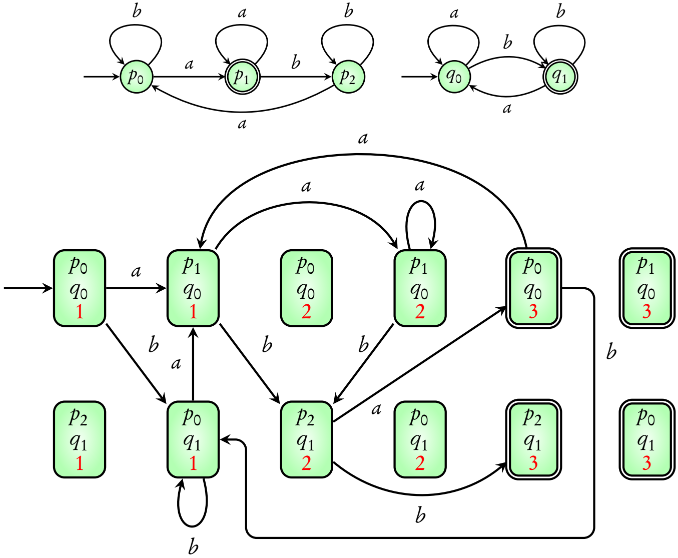

a
p_0 q_0 1
a
p_1 q_0 1
p_0 q_0 2
p_1 q_0 2
p_0 q_0 3
p_1 q_0 3
b
a
b
p_2 q_1 1
p_0 q_1 1
p_2 q_1 2
p_0 q_1 2
p_2 q_1 3
p_0 q_1 3
b


b
a
b
a
b
a
p_0
p_1
p_2
a
b
a
b
a
q_0
q_1

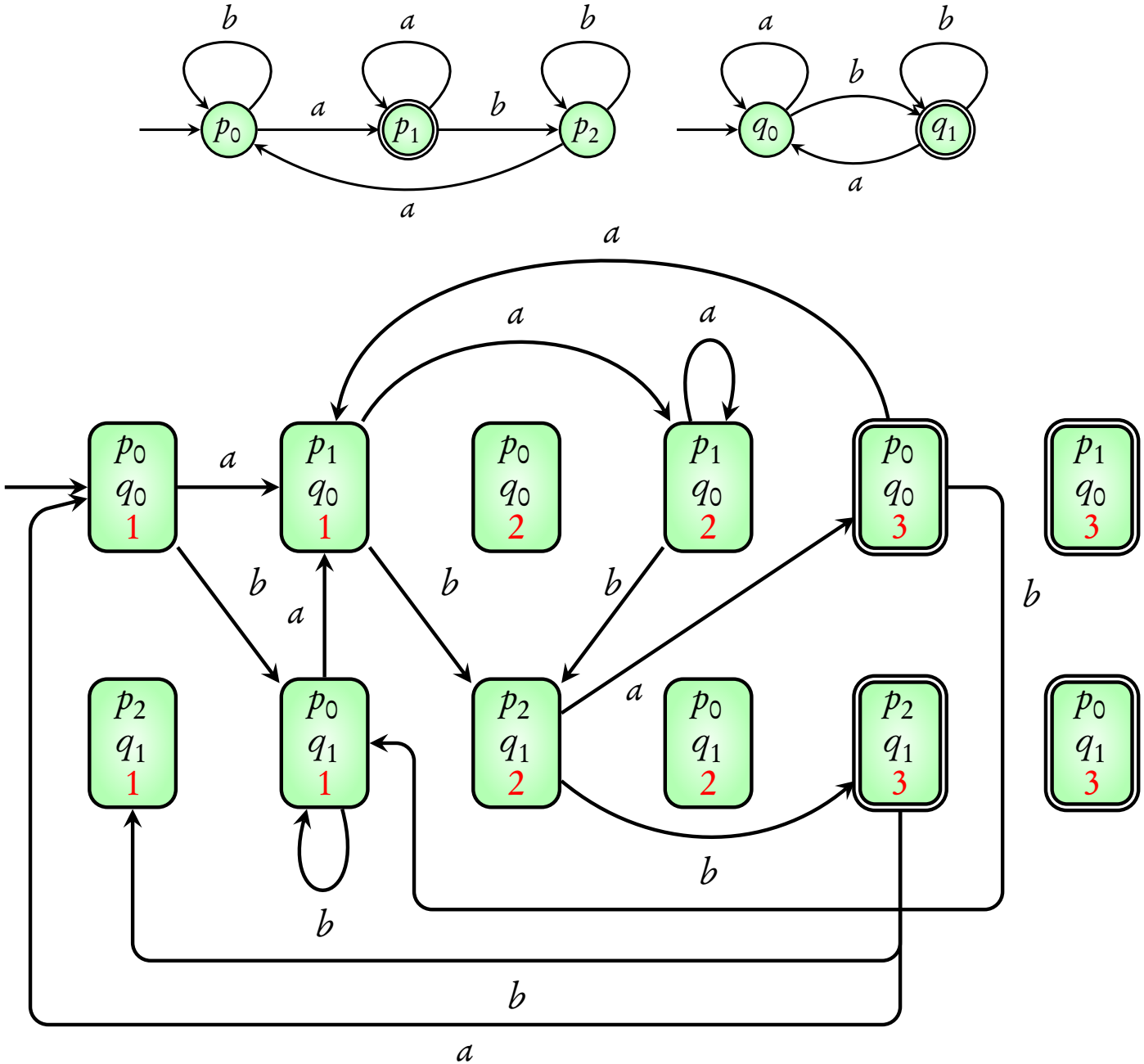

a
a
p_0 q_0 1
a
p_1 q_0 1
p_0 q_0 2
p_1 q_0 2
p_0 q_0 3
p_1 q_0 3
b
a
b
b
p_2 q_1 1
p_0 q_1 1
p_2 q_1 2
p_0 q_1 2
p_2 q_1 3
p_0 q_1 3
b

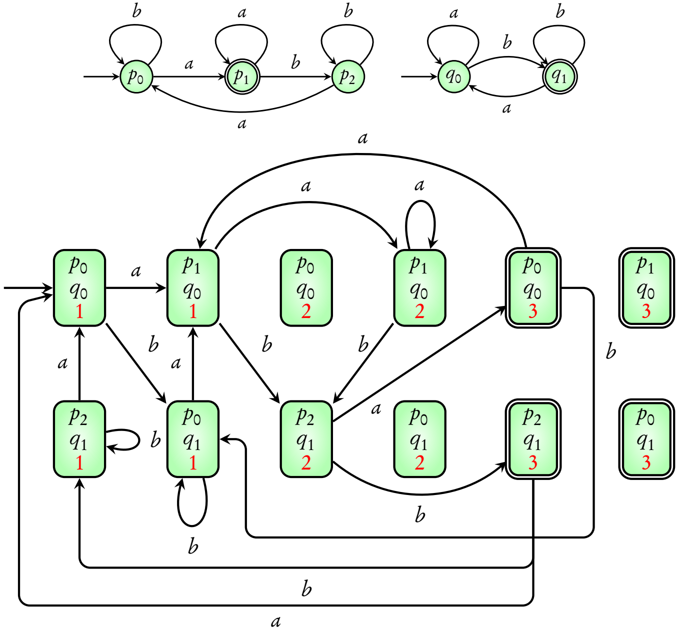

b
a
b
a
b
p0
p1
p2
a
a
b
b
a
q0
q1

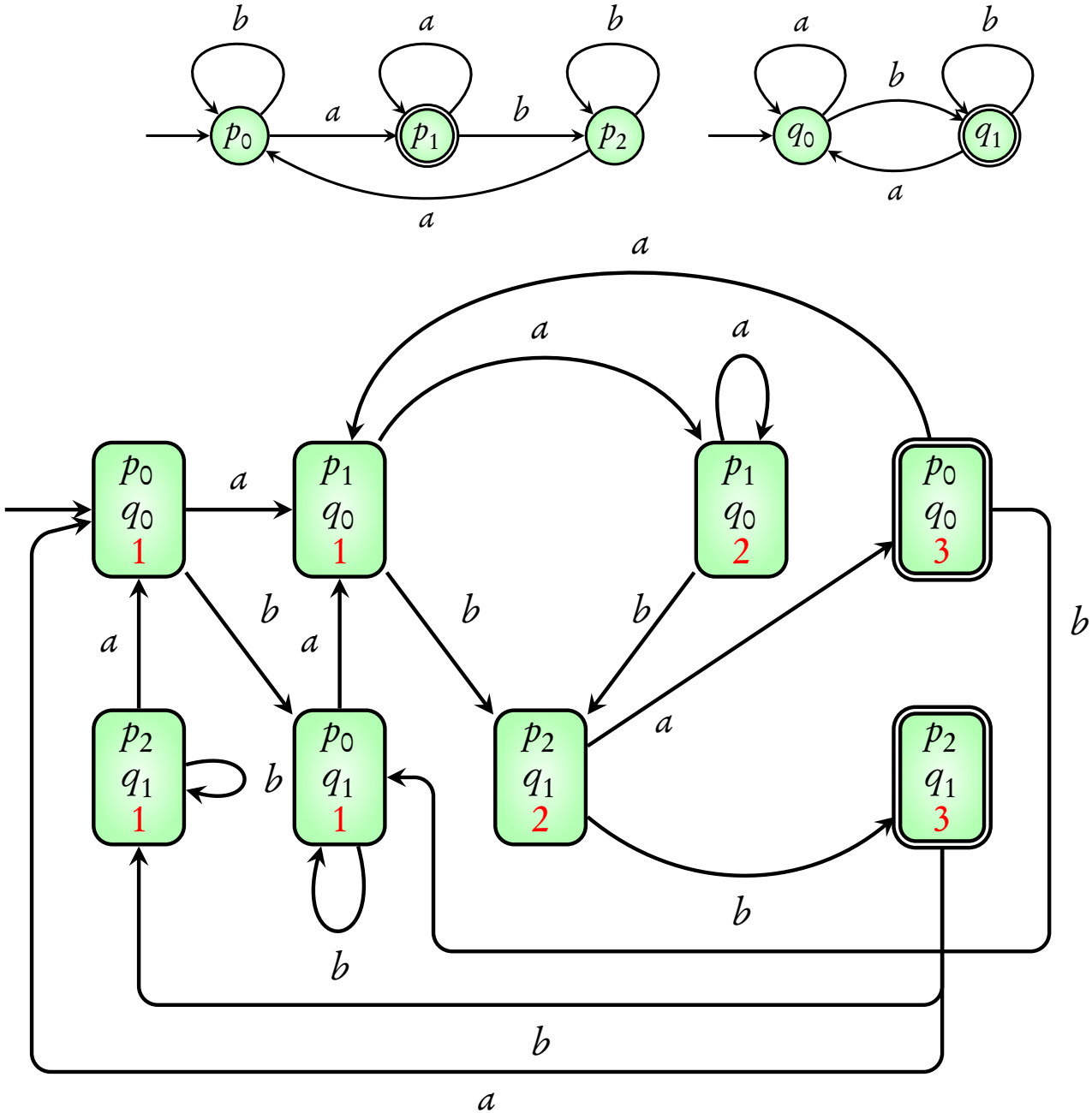

a
a
p0 q0 1
a
p1 q0 1
p0 q0 2
p1 q0 2
p0 q0 3
p1 q0 3
b
b
b
a
p2 q1 1
p0 q1 1
p2 q1 2
a
p0 q1 2
p2 q1 3
p0 q1 3
b
b

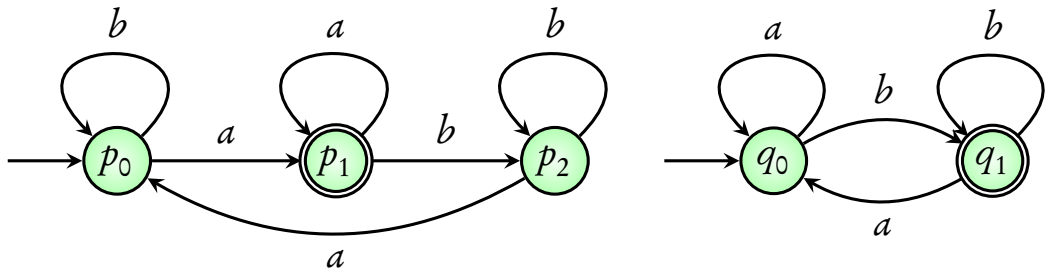


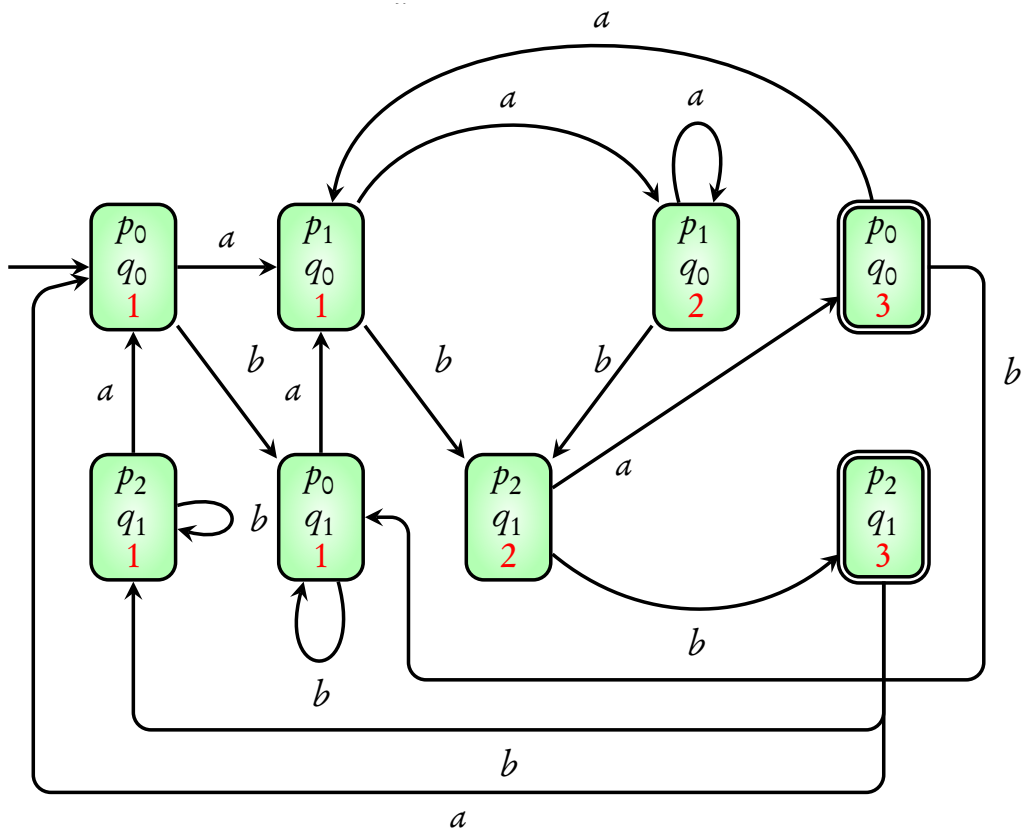
p0
p1
p2
q0
q1
a
b
p0 q0 1
p1 q0 1
p0 q0 2
p1 q0 2
p0 q0 3
p1 q0 3
p2 q1 1
p0 q1 1
p2 q1 2
p0 q1 2
p2 q1 3
p0 q1 3

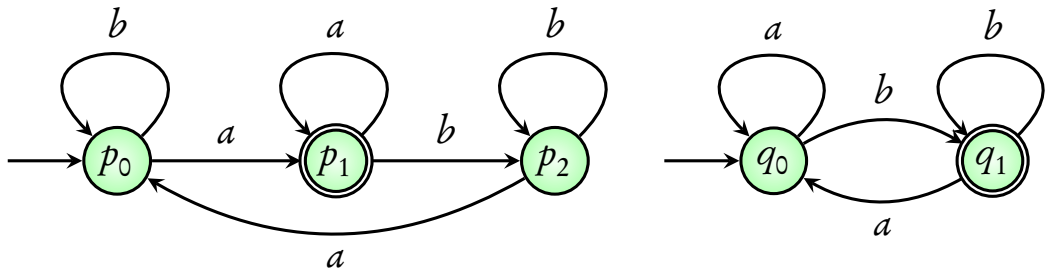

b
a
b
a
b
a
p_0
p_1
p_2
a
b
a
b
q_0
q_1
a
a
a
a
p_0 q_0 1
p_1 q_0 1
p_0 q_0 2
p_1 q_0 2
p_0 q_0 3
p_1 q_0 3
b
b
b
b
a
p_2 q_1 1
p_0 q_1 1
p_2 q_1 2
p_0 q_1 2
p_2 q_1 3
p_0 q_1 3
a
b
b
b
b
a

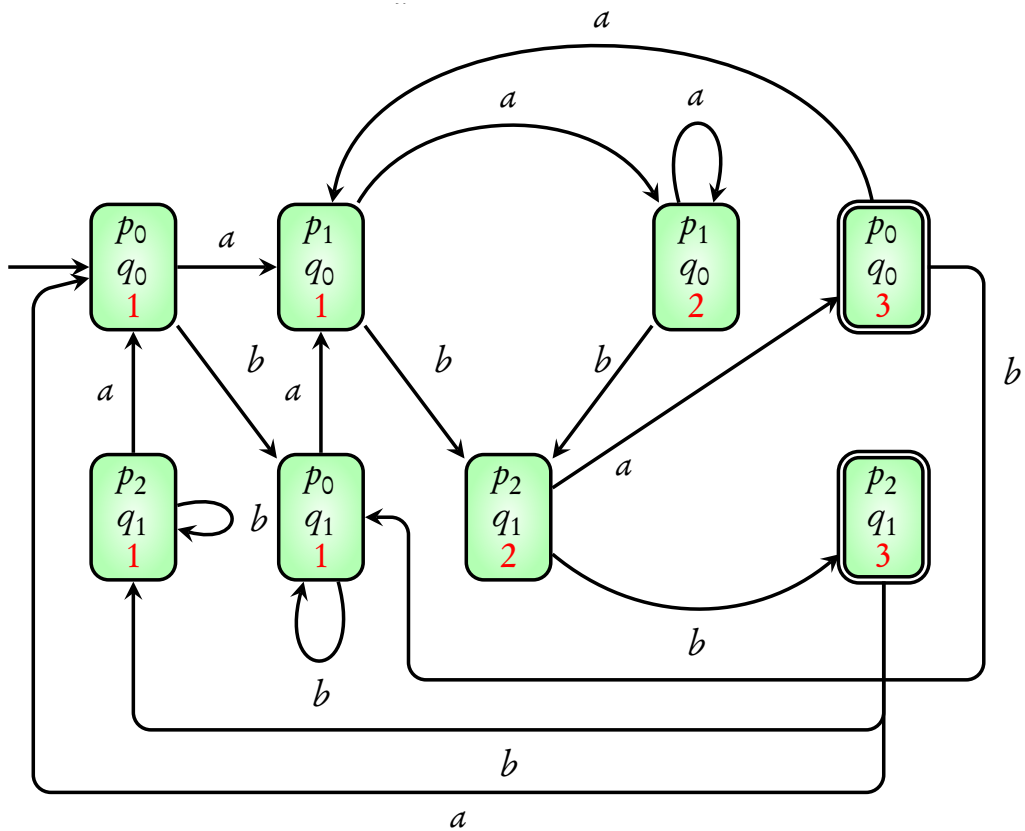

p0
p1
p2
q0
q1
p0 q0 1
p1 q0 1
p0 q0 2
p1 q0 2
p0 q0 3
p1 q0 3
p2 q1 1
p0 q1 1
p2 q1 2
p0 q1 2
p2 q1 3
p0 q1 3


b
a
b
a
b
a
p_0
p_1
p_2
a
b
a
b
q_0
q_1
a
a
a
a
p_0 q_0 1
p_1 q_0 1
p_1 q_0 2
p_0 q_0 3
b
b
b
b
a
a
p_2 q_1 1
b
p_0 q_1 1
p_2 q_1 2
a
p_2 q_1 3
b
b
b
a

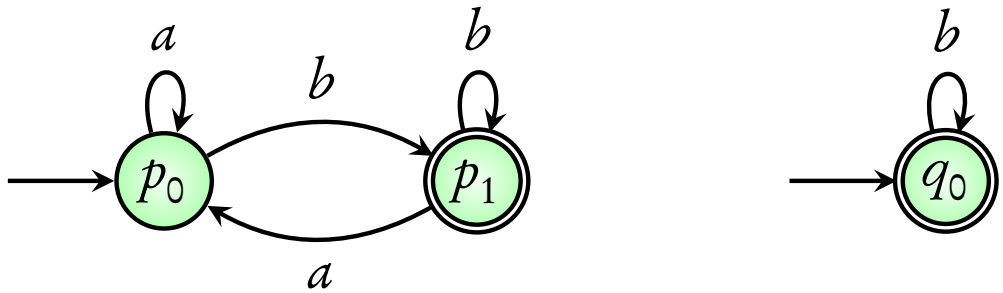

b
a
b
a
b
a
p_0
p_1
p_2
a
b
a
b
q_0
q_1
a

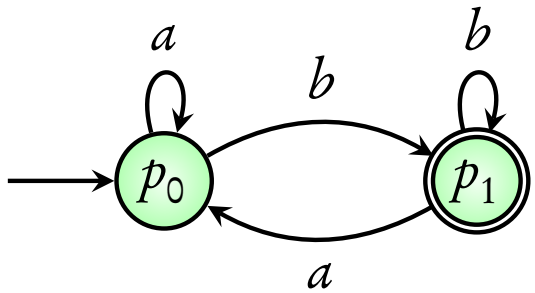

a
a
a
p_0 q_0 1
a
p_1 q_0 1
p_1 q_0 2
p_0 q_0 3
b
b
b
b
a
a
p_2 q_1 1
b
p_0 q_1 1
p_2 q_1 2
a
p_2 q_1 3
b
b
b
a

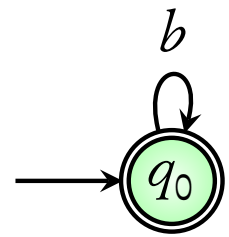


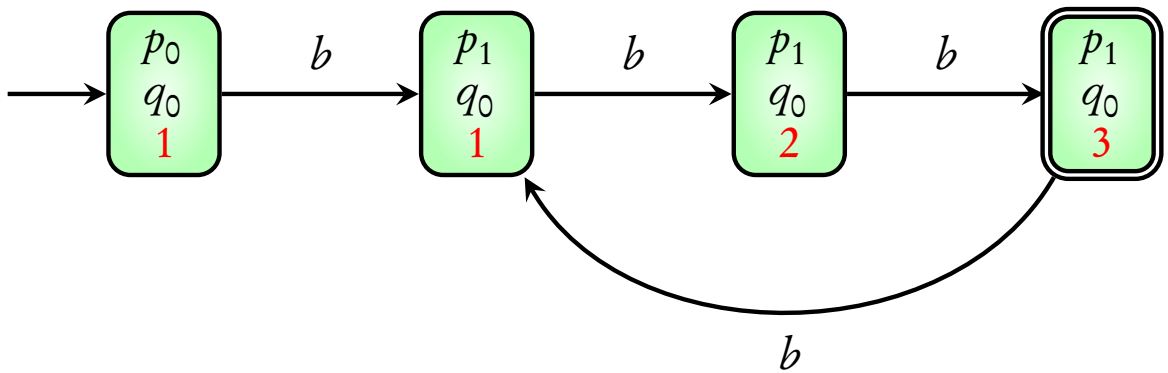


Word is accepted by product $\longleftrightarrow$ it is accepted by both component automata


a
b
b
p_0
p_1
a
b
q_0


a
b
b
p_0
p_1
a


b
q_0


p_0
q_0
1
b
p_1
q_0
1
b
p_1
q_0
2
b
p_1
q_0
3
b

Determinization

DBA less powerful than NBA

Product construction

Language intersection

Emptiness

Complementation

Union

Determinization

DBA less powerful than NBA

Product construction

Language intersection

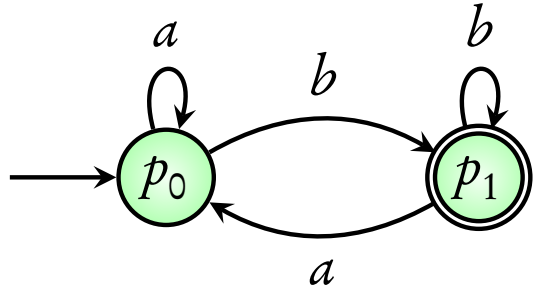
Emptiness

Next unit ...

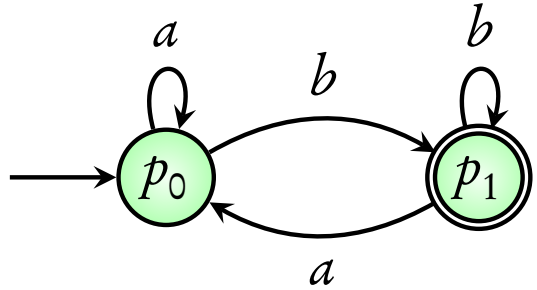
Complementation

Union

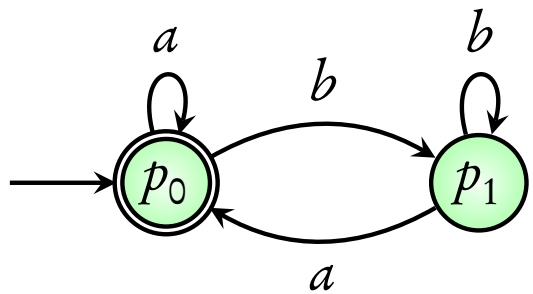
Language: b occurs infinitely often



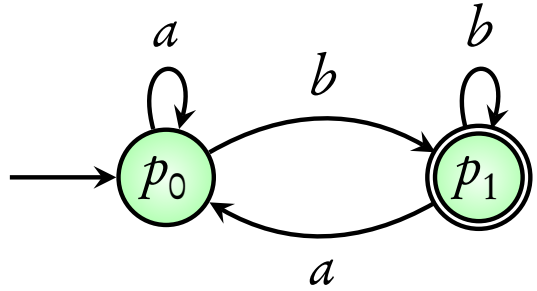
Language: b occurs infinitely often



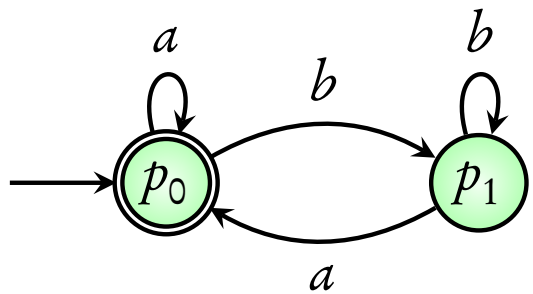
Language: a occurs infinitely often



Language: b occurs infinitely often



Language: a occurs infinitely often



Not the complement!

$(ab)^\omega$ present in both

Challenges

- Mere interchange of accepting states does not work

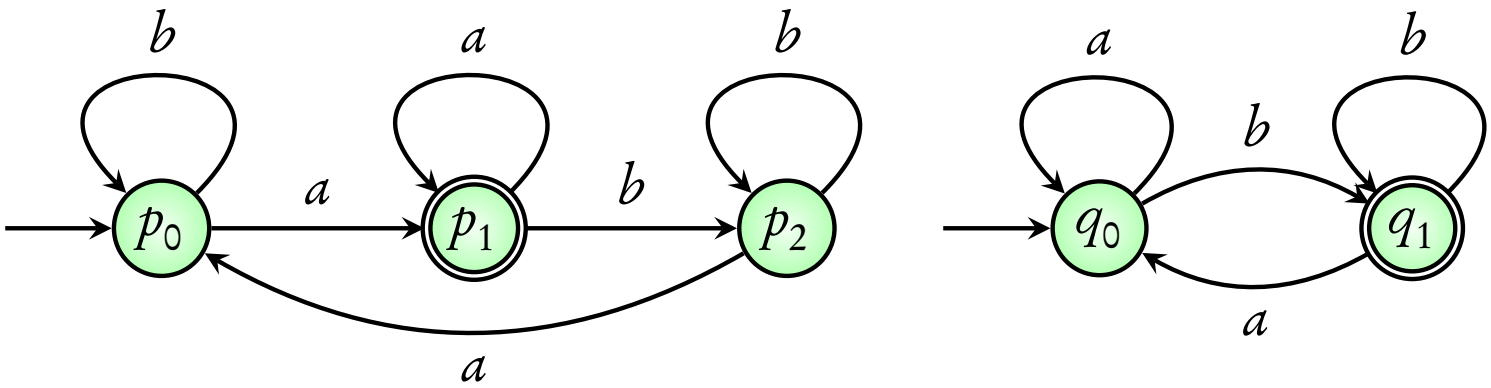
- Moreoever, NBA are more expressive than DBA

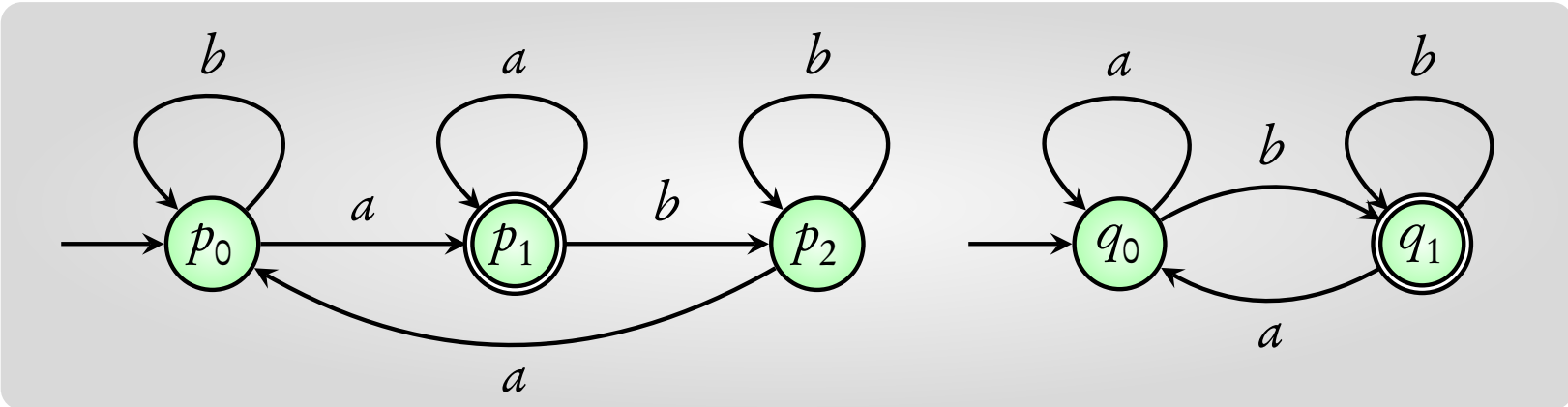
Complementation

Theorem

Given an NBA $\mathscr{A}$, there is an algorithm to compute the NBA accepting the complement language $\mathscr{L}(\mathscr{A})^c$

Proof out of scope of this course


b
a
b
a
b
p_0
p_1
p_2
a
a
b
a
b
q_0
q_1
a



For **union**, take the disjoint union of the two NBA

Determinization

DBA less powerful than NBA

Product construction

Language intersection

Emptiness

Next unit ...

Complementation

Union