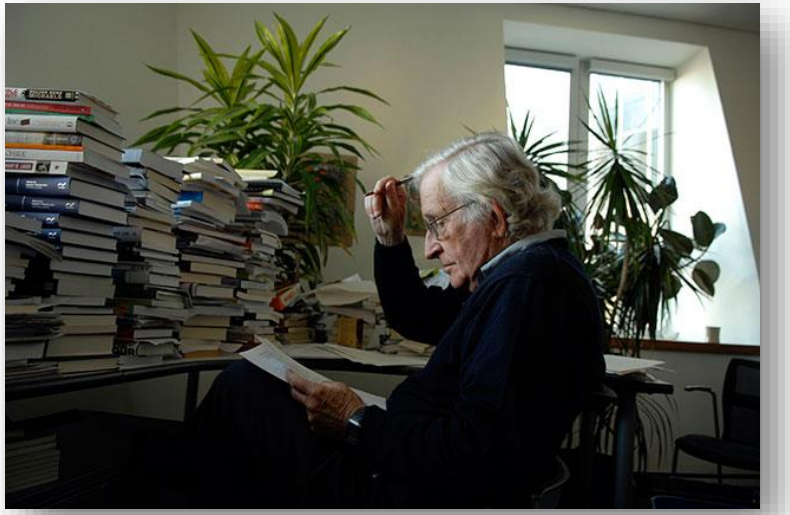
Formal languages and automata

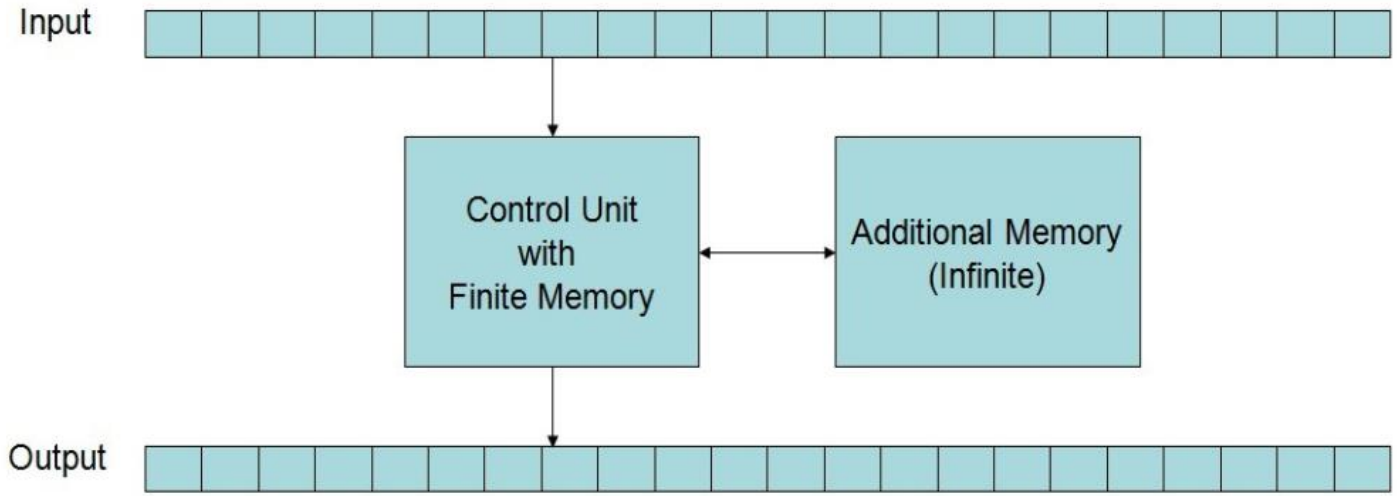
Finite Automata

https://t.me/fla_uog
mh.olyaee@gmail.com



Automata

- Abstract Models of computing devices



- Each step of operation is like:
  - If the current input symbol is X then output Y, move (left/right)

Automata

- The control unit has some finite memory and it keeps track of what step to execute next.
- Additional memory (if any) is infinite - we never run out of memory!
  - Infinite but like a stack - only the top item is accessible at a given time.
  - Infinite but like a tape, any cell is (sequentially) accessible.

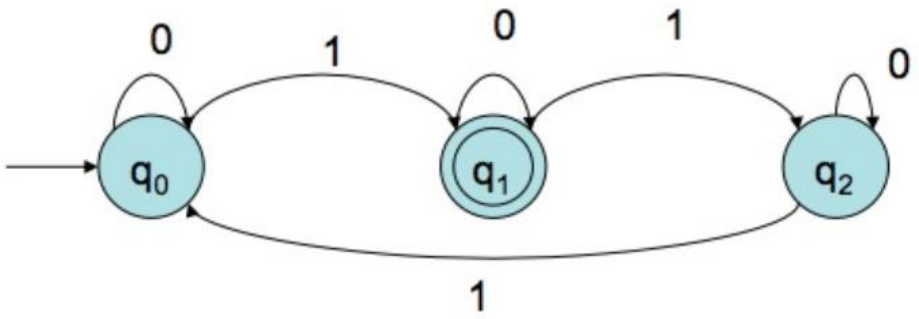
Finite Automaton (FA)

- Finite State Automata (FSA) are the simplest automata.
- Only the finite memory in the control unit is available.
- The memory can be in one of finite states at a given time – hence the name.
  - One can remember only a (fixed) finite number of properties of the past input.
  - Since input strings can be of arbitrary length, it is not possible to remember unbounded portions of the input string.
- It comes in Deterministic and Nondeterministic flavors.

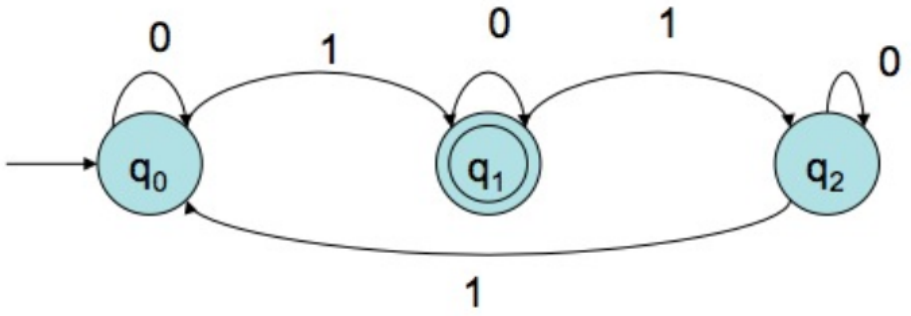
Deterministic Finite Automata (DFA)

- A DFA starts in a start state and is presented with s an input string.
- It moves from state to state, reading the input string one symbol at a time.
- What state the DFA moves next depends on
  - the current state,
  - current input symbol
- When the last input symbol is read, the DFA decides whether it should accept the input string

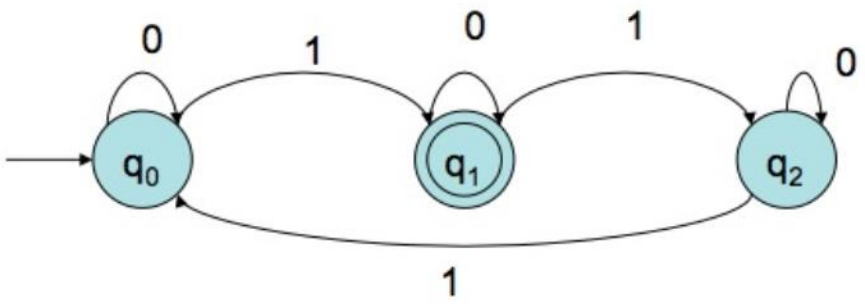
Simple example



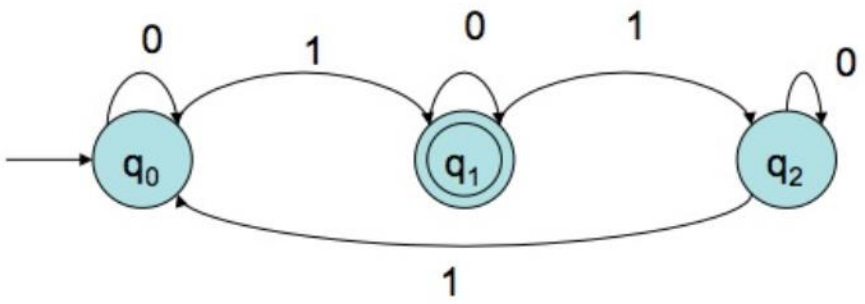
- States are shown with circles. We usually have labels on the states.
  - One designated state is the start state, (State q_0 here).
  - States with double circles denote the accepting or final states (State q_1 here)
- Directed and labeled arrows between states denote state transitions.



- This DFA stays in the same state when the next input symbol is a 0.
- In state q_0, an input of 1 moves the DFA to state q_1.
- In state q_1, an input of 1 moves the DFA to state q_2.
- In state q_2, an input of 1 moves the DFA back to state q_0.
- If the DFA is in state q_1 when the input is finished, the DFA accepts the input string.



- What kinds of strings does this DFA accept?
  - It accepts $\omega = 00010000$
  - It accepts $\omega = 00010011001$
  - It accepts $\omega = 1$
  - It rejects $\omega = 1100001$
  - It rejects $\omega = 0110000$



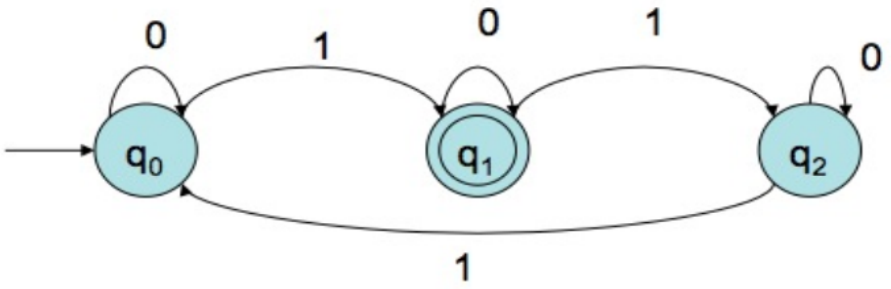
- What kinds of strings does this DFA accept?
  - It accepts $\omega = 00010000$
  - It accepts $\omega = 00010011001$
  - It accepts $\omega = 1$
  - It rejects $\omega = 1100001$
  - It rejects $\omega = 0110000$

- It accepts all strings $\omega \in \{0,1\}^*$ such that $n_1(\omega) = 1 \mod 3$

Formal Definition

- A Deterministic Finite State Acceptor (DFA) is defined as the 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$ where
  - Q is a finite set of states
  - Σ is a finite set of symbols – the alphabet
  - $\delta : Q \times \Sigma \rightarrow Q$ is the next-state function
  - $q_0 \in Q$ is the (label of the) start state
  - $F \subseteq Q$ is the set of final (accepting) states

Formal description of the example



- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{0, 1\}$
- δ :

δ	0	1
q_0	q_0	q_1
q_1	q_1	q_2
q_2	q_2	q_0

- q_0
- $F = \{q_1\}$

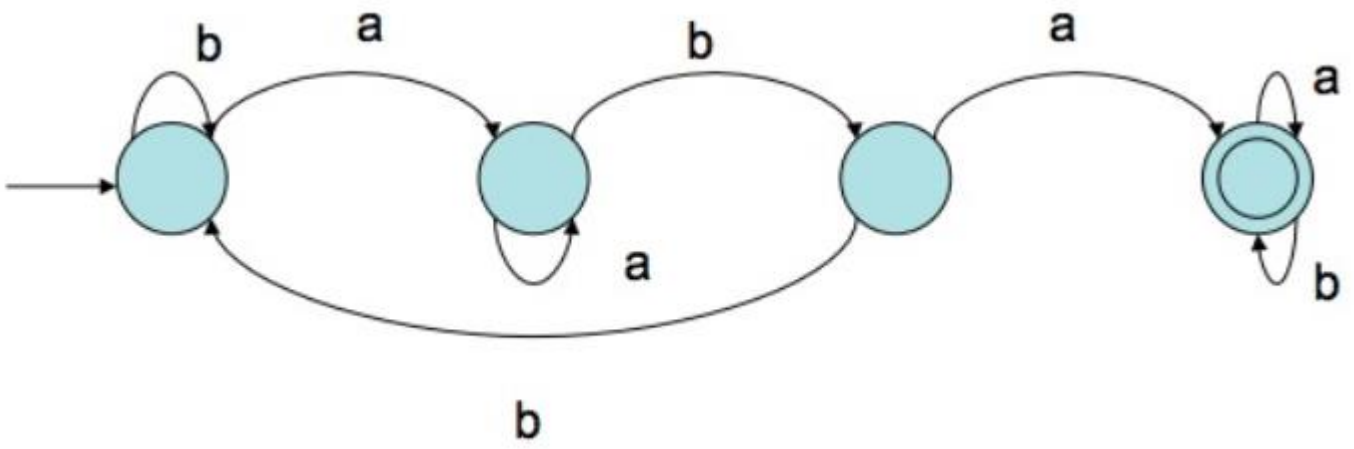
We will almost always use the graphical description for δ. The other components will always be implicit!

How DFA works?

- The DFA accepts a string $\omega = x_1 x_2 \cdots x_n$ if a sequence of states $r_0 r_1 r_2 \cdots r_n, r_i \in Q$, exists, such that
  1. $r_0 = q_0$ (Start in the initial state)
  2. $r_i = \delta(r_{i-1}, x_i)$ for $i = 1, 2, \ldots n$
     - Move from state to state.
  3. $r_n \in F$
     - End up in a final state.
- If the DFA is NOT in an accepting state when the input string is exhausted, then the string is rejected.

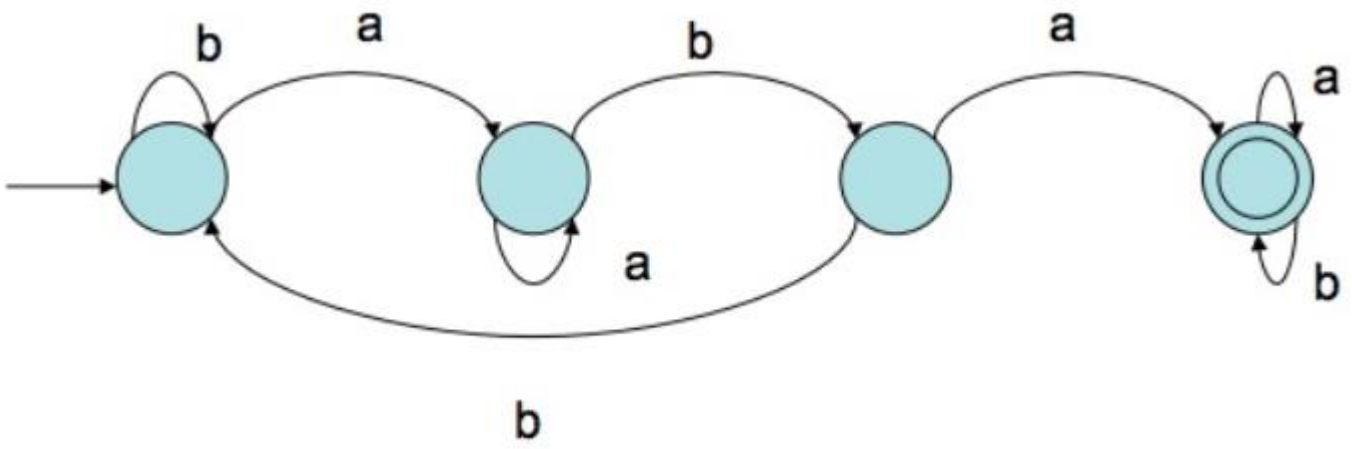
Example

- Which strings are accepted by the following DFA:



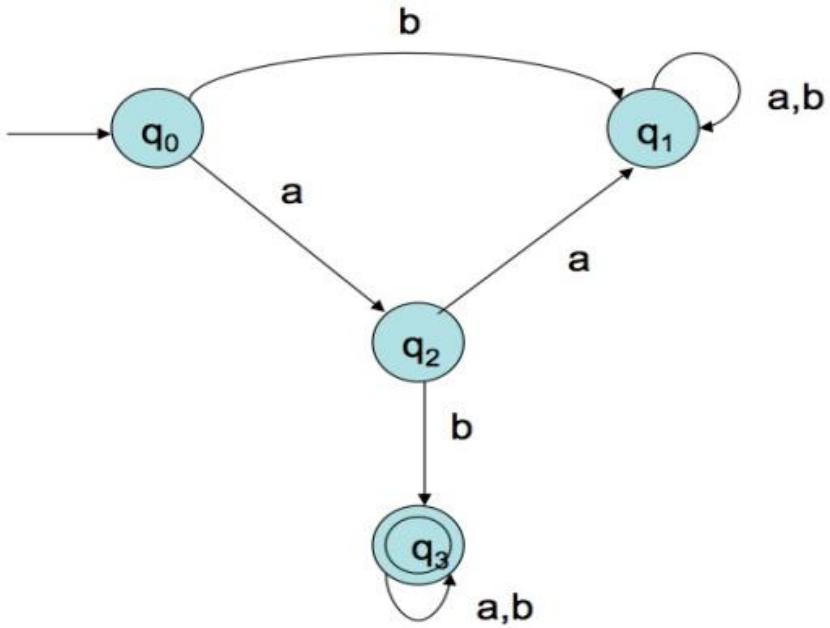
Example

- Which strings are accepted by the following DFA:

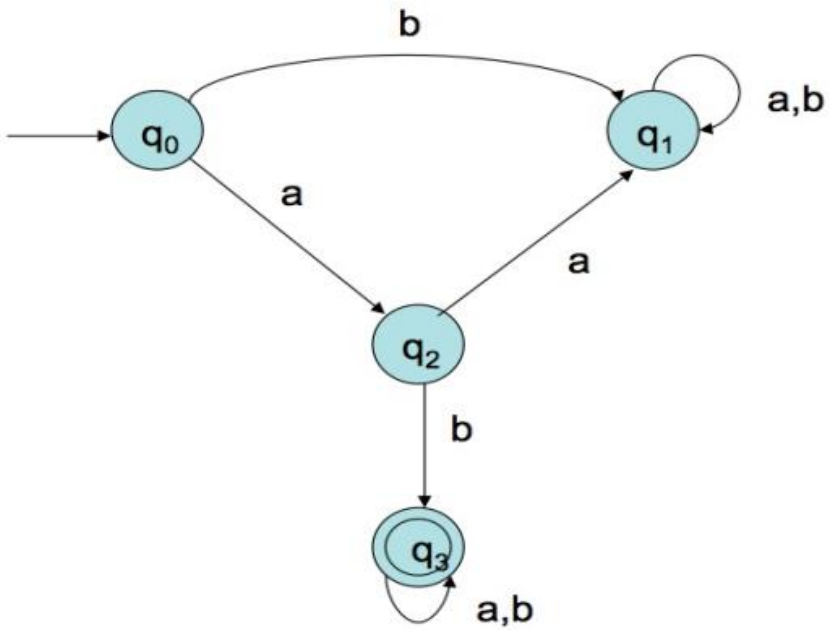


- This DFA accepts strings that have *aba* somewhere in it.
- Once the existence of *aba* is ascertained, the rest of the input is ignored!

Example

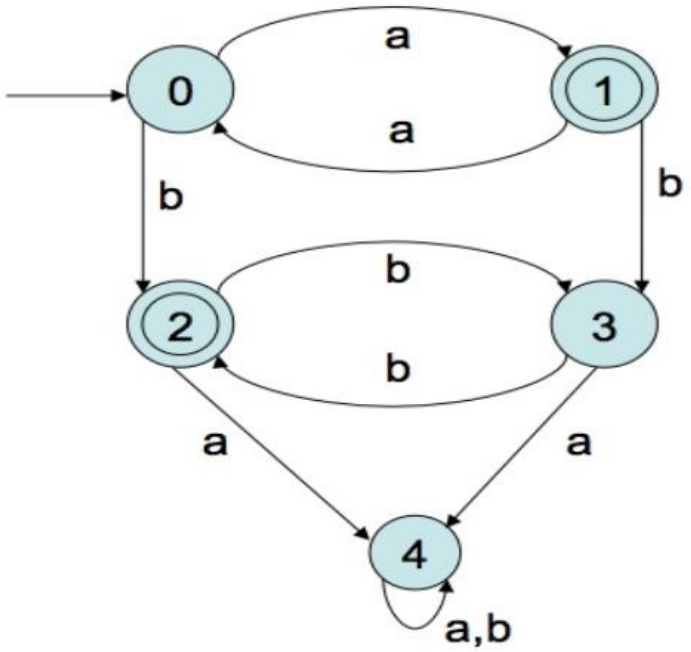


Example

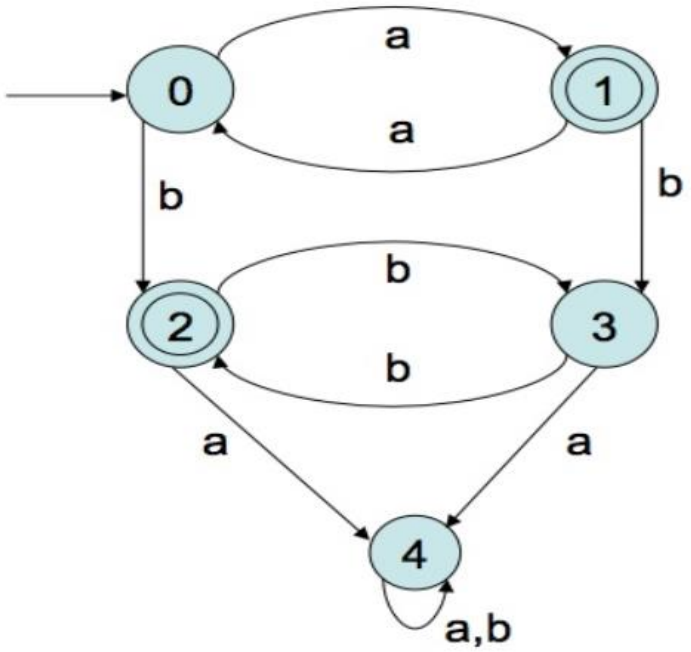


- This DFA accepts strings that start with *ab*
- Once the string starts with *ab* the rest is ignored!
- The state q_1 is known as a sink state.
  - Once a machine enters a sink state, there is no getting out! It is rejected.

Example



Example



- This DFA accepts strings of the sort $a^n b^m$ such that $n + m$ is odd.

The language accepted by DFA

- $L(M)$ denotes the language accepted by a DFA M

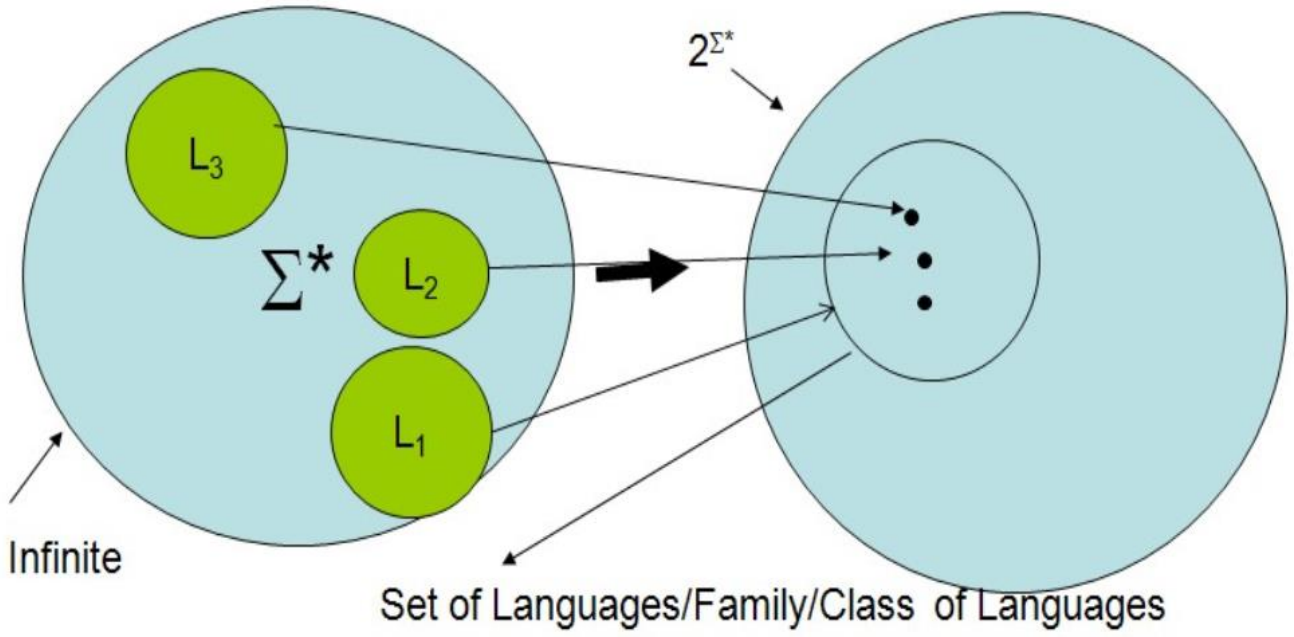
$$L(M) = \{\omega | \omega \in \Sigma^* \text{ and } \delta^*(q_0, \omega) \in F\}$$

- Similarly

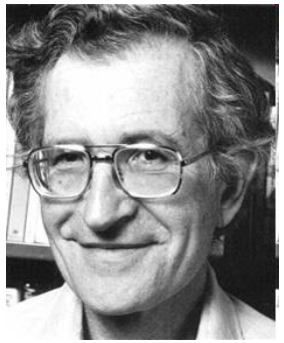
$$\overline{L(M)} = \{\omega | \omega \in \Sigma^* \text{ and } \delta^*(q_0, \omega) \notin F\}$$

Regular Languages

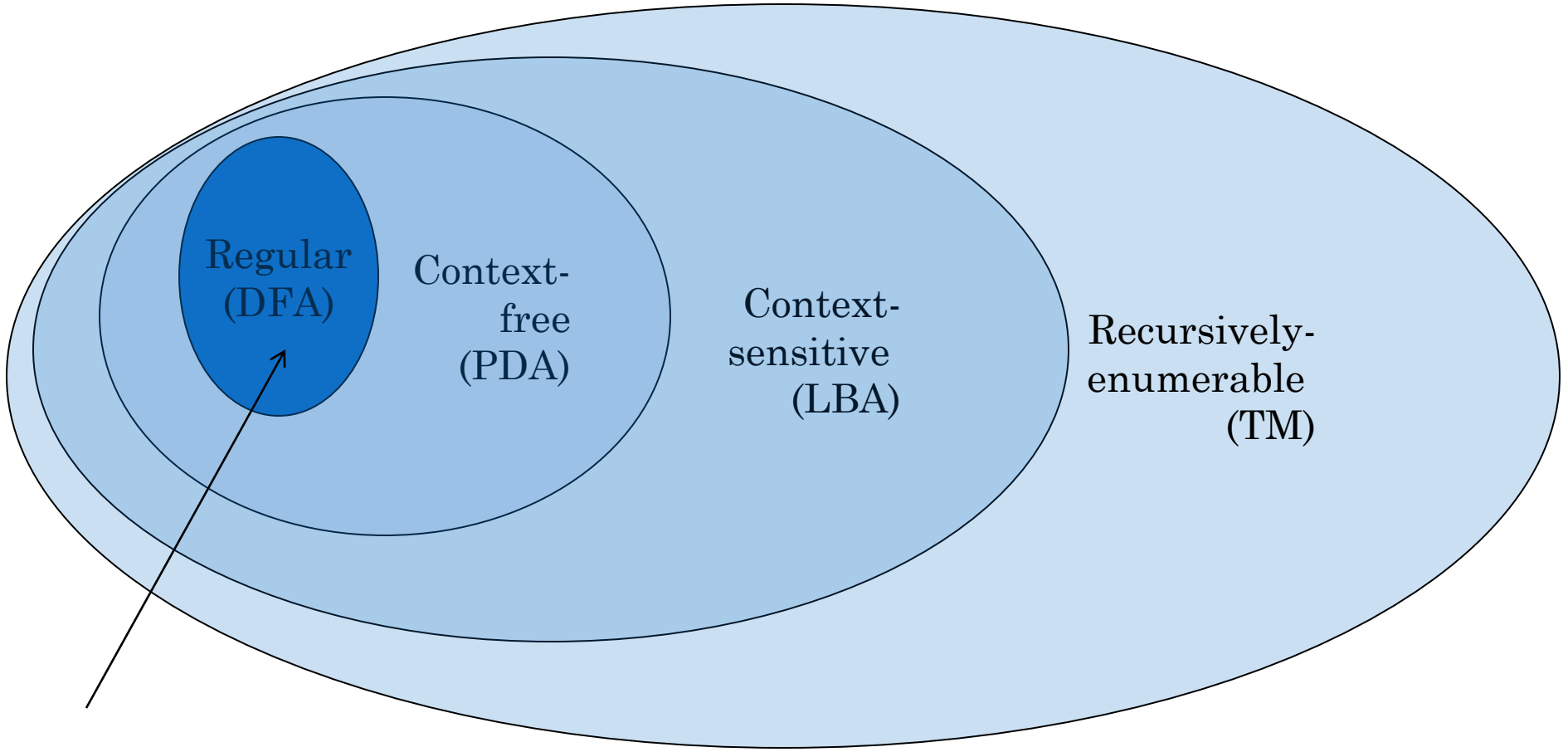
A language L is called a regular language if and only if there exists a DFA M such that $L(M) = L$.



The Chomsky Hierachy



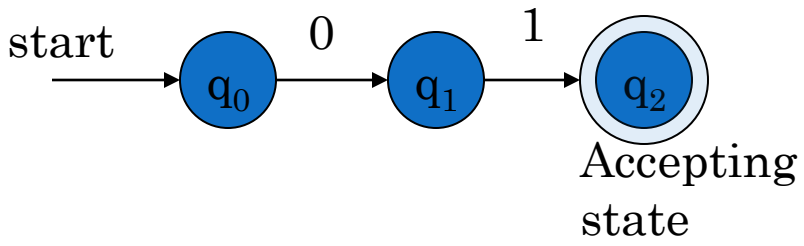
- A containment hierarchy of classes of formal languages



Example

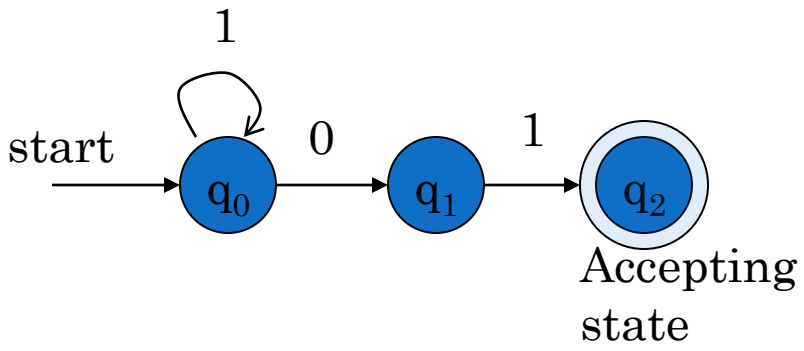
- Build a DFA for the following language:
  - L = {w | w is a binary string that contains 01 as a substring}
- Steps for building a DFA to recognize L:
  - $\sum = \{0,1\}$
  - Decide on the states: Q
  - Designate start state and final state(s)
  - δ: Decide on the transitions:
- "Final" states == same as "accepting states"
- Other states == same as "non-accepting states"

DFA for strings containing 01



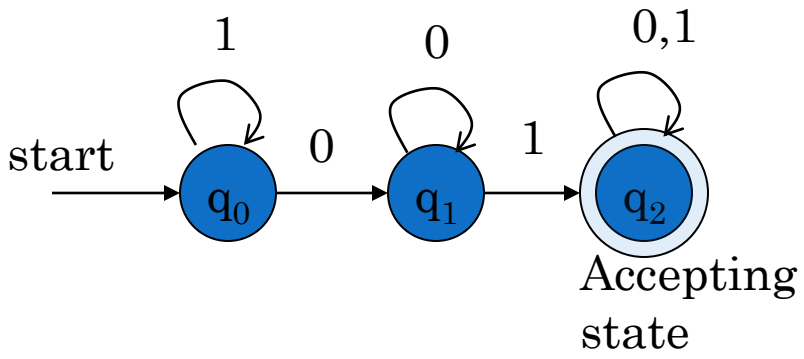
DFA for strings containing 01

- What makes this DFA deterministic?



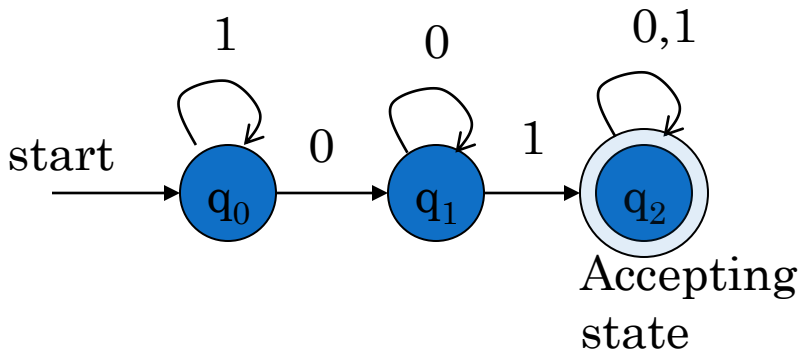
DFA for strings containing 01

- What makes this DFA deterministic?



DFA for strings containing 01

- What makes this DFA deterministic?

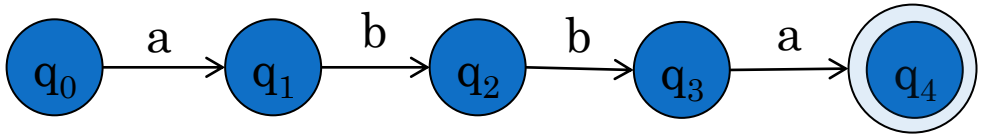


- $Q = \{q_0, q_1, q_2\}$
- $\sum = \{0, 1\}$
- start state = q_0
- $F = \{q_2\}$
- Transition table

	symbols	
δ	**0**	**1**
states → **q_0**	q_1	q_0
q_1	q_1	q_2
***q_2**	q_2	q_2

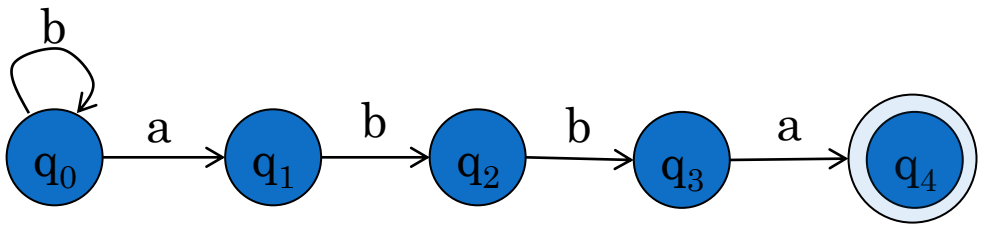
حل چند مثال

- ماشینی طراحی کنید که توالی های حاوی abba را پذیرش کند



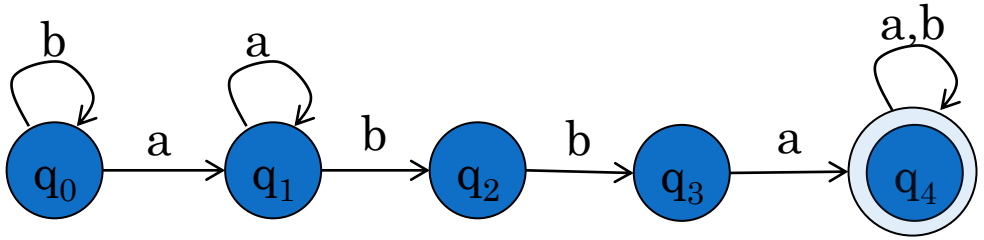
حل چند مثال

- ماشینی طراحی کنید که توالی های حاوی abba را پذیرش کند



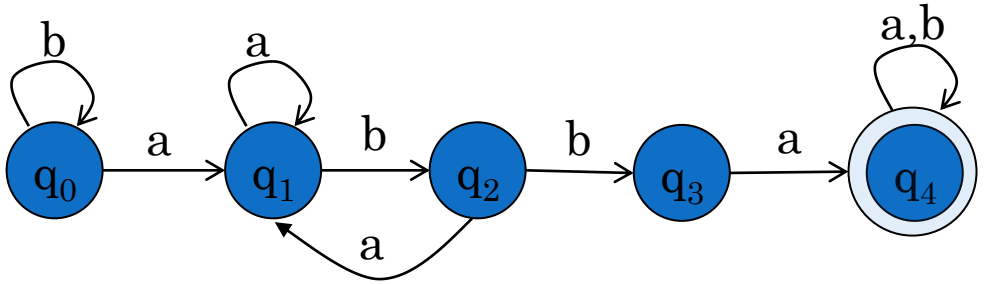
حل چند مثال

- ماشینی طراحی کنید که توالی های حاوی abba را پذیرش کند



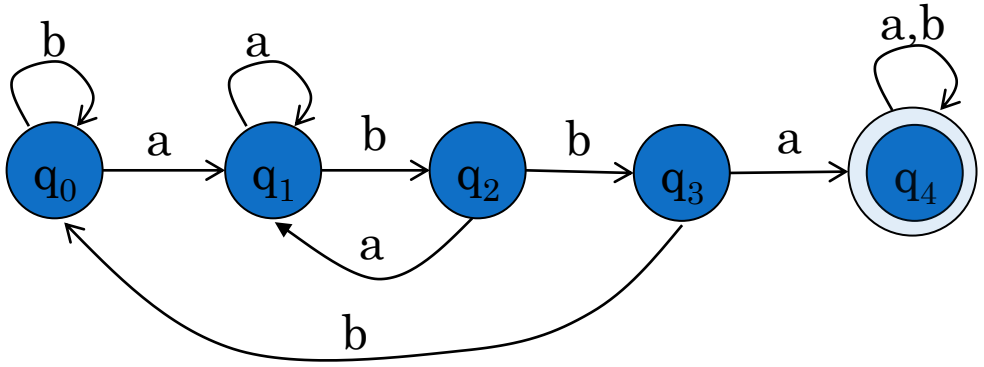
حل چند مثال

- ماشینی طراحی کنید که توالی های حاوی abba را پذیرش کند



حل چند مثال

- ماشینی طراحی کنید که توالی های حاوی abba را پذیرش کند



حل چند مثال

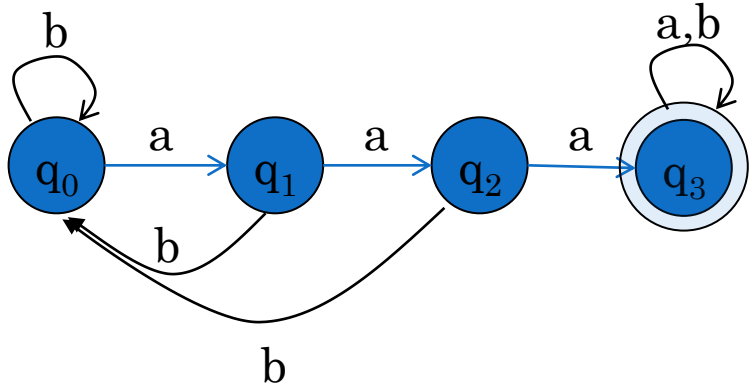
- ماشینی طراحی کنید که حاوی زیر رشته aaaنباشد

حل چند مثال

- ماشینی طراحی کنید که حاوی زیر رشته aaaنباشد
- راهنمایی: ابتدا ماشینی طراحی کنید که این توالی را بپذیر و در ادامه تمام حالات غیرپذیرش را به پذیرش تبدیل کرده و حالات پذیرش را به غیر پذیرش تغییر دهید

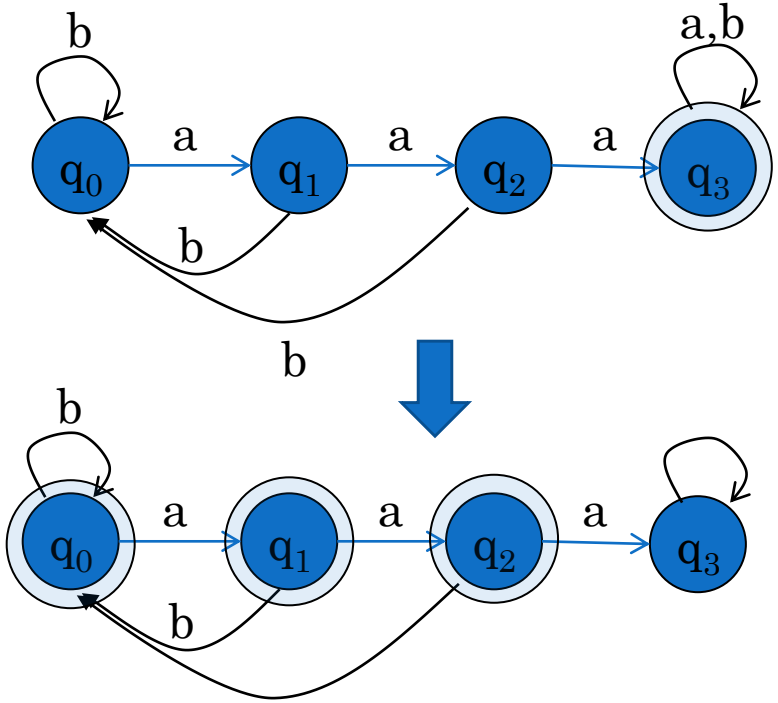
حل چند مثال

- ماشینی طراحی کنید که حاوی زیر رشته aaaنباشد
- راهنمایی: ابتدا ماشینی طراحی کنید که این توالی را بپذیر و در ادامه تمام حالات غیرپذیرش را به پذیرش تبدیل کرده و حالات پذیرش را به غیر پذیرش تغییر دهید



حل چند مثال

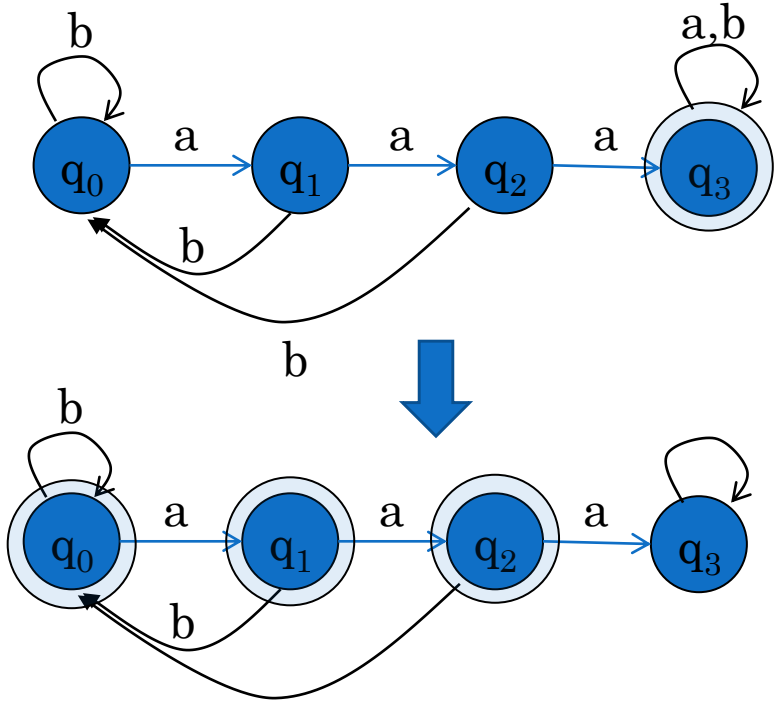
- ماشینی طراحی کنید که حاوی زیر رشته aaaنباشد
- راهنمایی: ابتدا ماشینی طراحی کنید که این توالی را بپذیر و در ادامه تمام حالات غیرپذیرش را به پذیرش تبدیل کرده و حالات پذیرش را به غیر پذیرش تغییر دهید



اگر ماشین $M(\Sigma,Q,F,q0,\delta)$ زبان L را پذیرش کند، ماشین $M'(\Sigma,Q,Q-F,q0,\delta)$ زبان $\overline{L}$ را می پذیرد

حل چند مثال

- ماشینی طراحی کنید که حاوی زیر رشته aaaنباشد
- راهنمایی: ابتدا ماشینی طراحی کنید که این توالی را بپذیر و در ادامه تمام حالات غیرپذیرش را به پذیرش تبدیل کرده و حالات پذیرش را به غیر پذیرش تغییر دهید

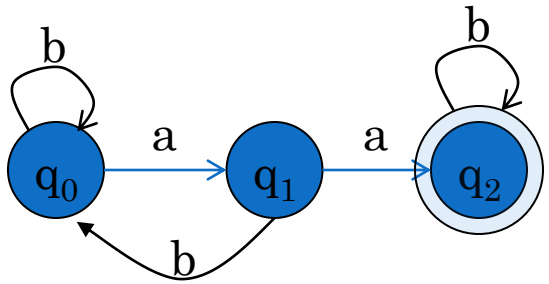


حل چند مثال

- ماشینی طراحی کنید که تنها یک aa داشته باشد

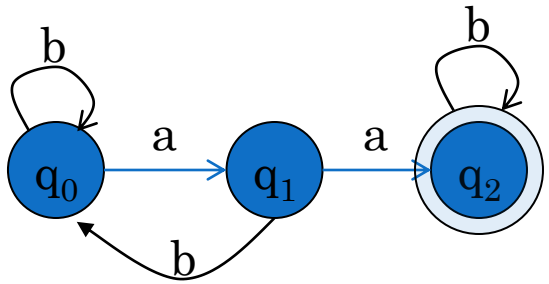
حل چند مثال

- ماشینی طراحی کنید که تنها یک aa داشته باشد



حل چند مثال

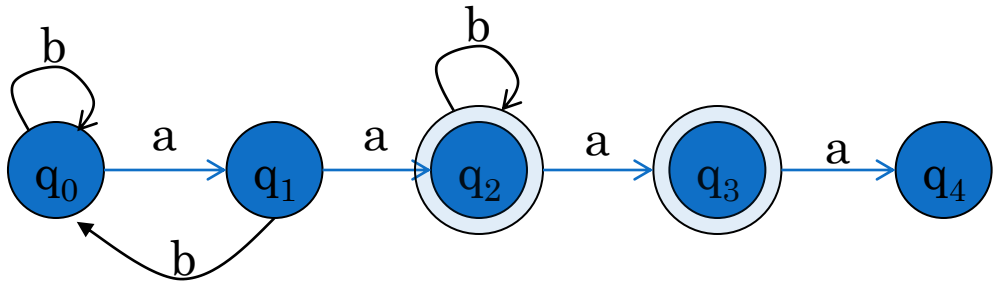
- ماشینی طراحی کنید که تنها یک aa داشته باشد



- آیا رشته ای مانند aaababaa می بایست پذیرش شود؟

حل چند مثال

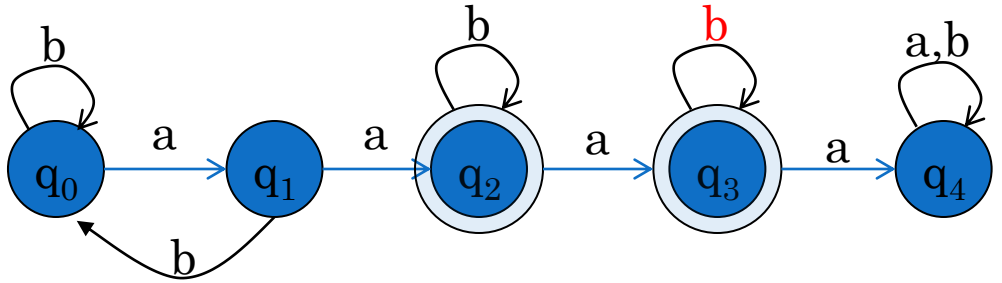
- ماشینی طراحی کنید که تنها یک aa داشته باشد



- آیا رشته ای مانند aaababaa می بایست پذیرش شود؟

حل چند مثال

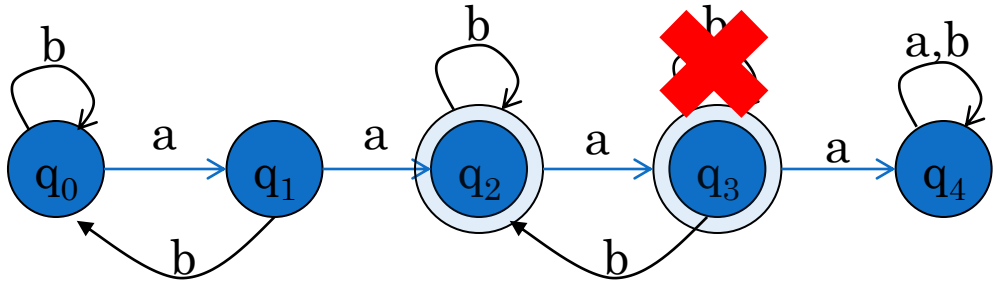
- ماشینی طراحی کنید که تنها یک aa داشته باشد



- آیا رشته ای مانند aaababaa می بایست پذیرش شود؟

حل چند مثال

- ماشینی طراحی کنید که تنها یک aa داشته باشد



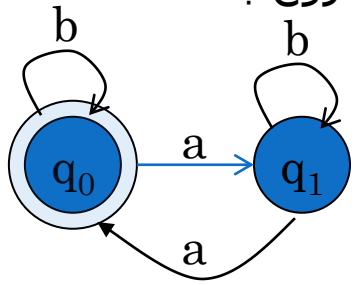
- آیا رشته ای مانند aaababaa می بایست پذیرش شود؟

حل چند مثال

- ماشینی طراحی کنید که تعداد aها زوج باشد

حل چند مثال

- ماشینی طراحی کنید که تعداد aها زوج باشد



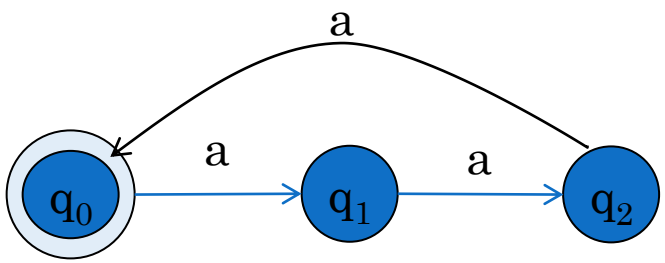
- آیا رشته تهی می بایست پذیر شود؟

حل چند مثال

- ماشینی طراحی کنید که $n_a - n_b mod 3 = 0$

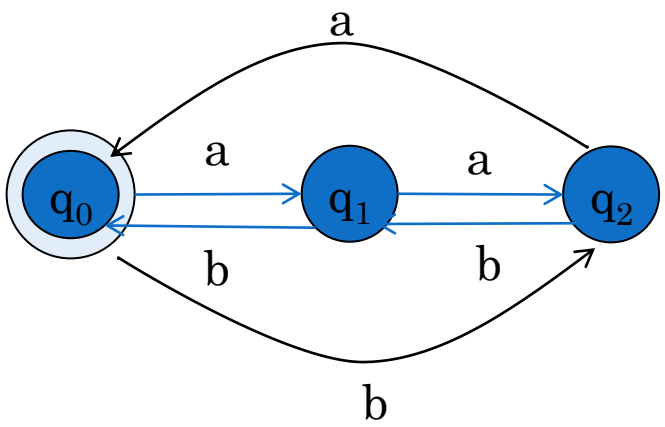
حل چند مثال

- ماشینی طراحی کنید که $n_a - n_b mod 3 = 0$



حل چند مثال

- ماشینی طراحی کنید که $n_a - n_b mod 3 = 0$



حل چند مثال

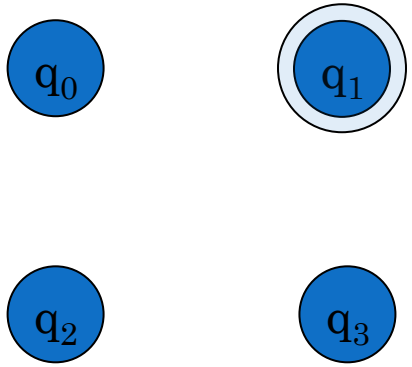
- ماشینی طراحی کنید که تعداد aها زوج و bها فرد باشد

حل چند مثال

- ماشینی طراحی کنید که تعداد aها زوج و bها فرد باشد
- چند حالت برای aها وb ها داریم؟
- Q0: تعداد هر دو زوج
- Q1: aها زوج وbها فرد
- Q2: aها فرد و bها زوج
- Q3: هر دو فرد

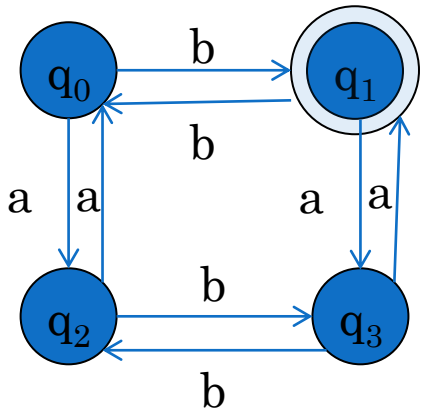
حل چند مثال

- ماشینی طراحی کنید که تعداد aها زوج و bها فرد باشد
- چند حالت برای aها وb ها داریم؟
- Q0: تعداد هر دو زوج
- Q1: aها زوج وbها فرد
- Q2: aها فرد و bها زوج
- Q3: هر دو فرد



حل چند مثال

- ماشینی طراحی کنید که تعداد aها زوج و bها فرد باشد
- چند حالت برای aها وb ها داریم؟
- Q0: تعداد هر دو زوج
- Q1: aها زوج وbها فرد
- Q2: aها فرد و bها زوج
- Q3: هر دو فرد



مثال

- Design a DFA for all strings over the alphabet $\Sigma = \{a, b\}$ that contain *aba* but not *abaa* as a substring.[1]

مثال

- Design a DFA for all strings over the alphabet $A = \{a, b, c\}$ in which no two consecutive positions are the same symbol.

Homework

- Design a DFA for the language
  $L = \{w \mid w \text{ contains at least one } 0 \text{ and at most one } 1\}$
- Design a DFA for the language
  $L = \{w \mid w \text{ does not contain } 100 \text{ as a substring}\}$