King Saud University

Theory of Computation
CSC 339 – Spring 2021

Chapter-1: part2
Regular Languages

King Saud University

Department of Computer Science

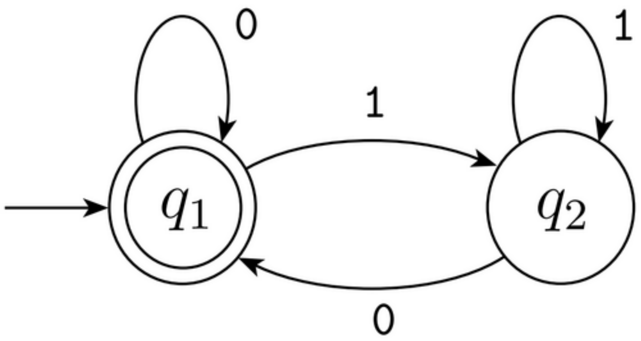
Dr. Azzam Alsudais

Outline

- Recap
- Introduction
- Nondeterministic Finite Automata (NFA)

Recap

- **Deterministic Finite Automata (DFA)**
  - **Given a word <u>w</u>, the automaton will always end up in state <u>q</u>**
  - **DFA always transition to the same state given the (q, a) ordered pair where q ∈ Q and a ∈ Σ.**



Recap

- $L = \{ab\}^+$
  - Example strings: {ab, abab, ababab, …}
- $L = \{ab\}^*$
  - Example strings: { ε , ab, abab, …}

Finite Automata: Nondeterminism

- **Nondeterministic finite automate (NFA): several choices may exist for the next state at any point.**

Finite Automata: Nondeterminism

- **Nondeterministic finite automate (NFA): several choices may exist for the next state at any point.**

- **It is a generalization of deterministic finite automata (DFA)**

Finite Automata: Nondeterminism

- Nondeterministic finite automate (NFA): several choices may exist for the next state at any point.

- It is a generalization of deterministic finite automata (DFA)

- Coming up with solutions using DFA may sometimes be extremely difficult.. So, we use NFA which is comparatively easier than DFA.

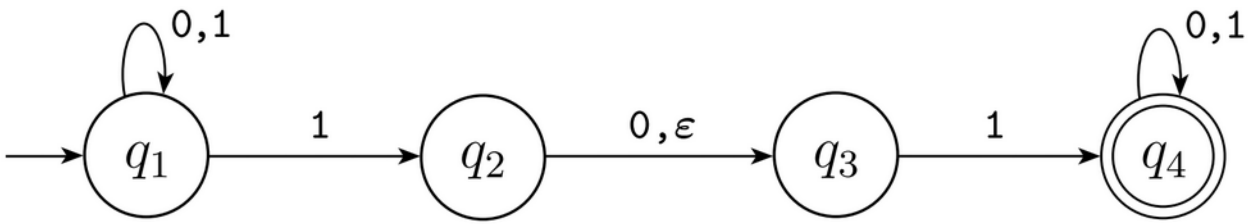
Finite Automata: Nondeterminism

- **Nondeterministic finite automate (NFA): several choices may exist for the next state at any point.**
- **It is a generalization of deterministic finite automata (DFA)**
- **Coming up with solutions using DFA may sometimes be extremely difficult.. So, we use NFA which is comparatively easier than DFA.**
- **Then, we can convert NFA to DFA**
    - **An NFA is much easier to construct than DFA.. why?**

Finite Automata: Nondeterminism

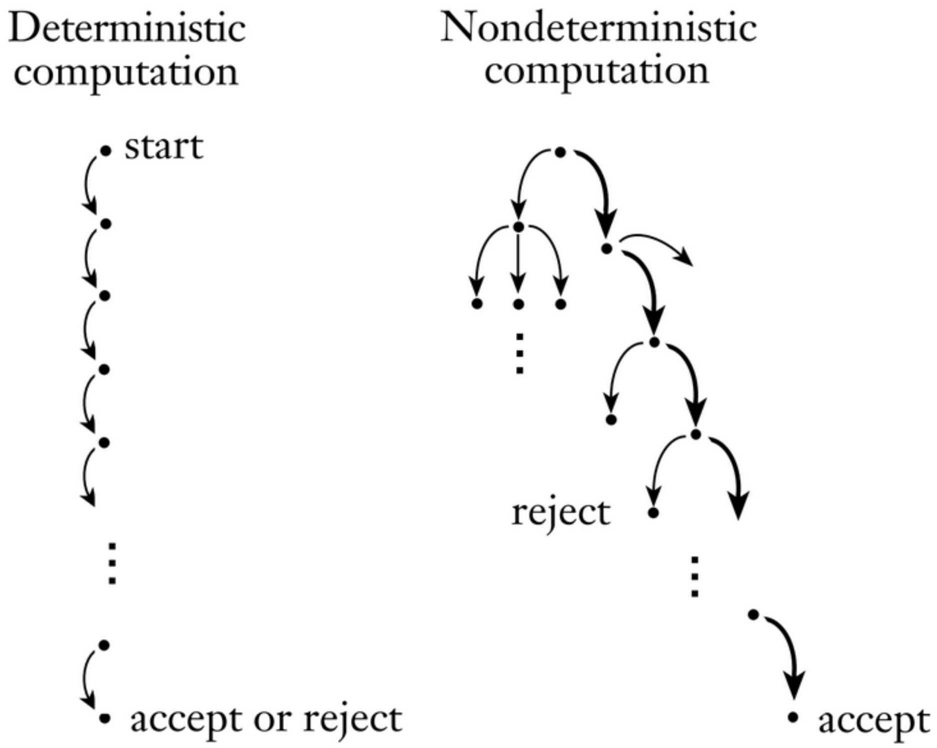
- How do we compute using NFA?
  - Think of it like a tree, where you create a new branch for each possibility
  - If one of those branches ends up in an accept state, then we say that this NFA is accepting the input string

Finite Automata: Nondeterminism

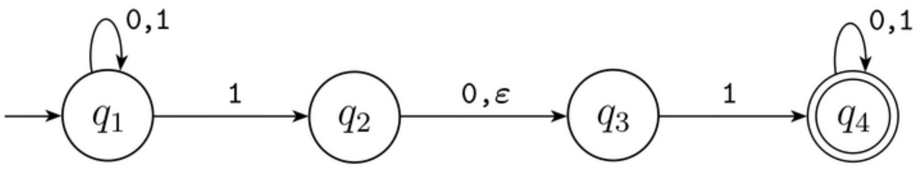


Nondeterministic finite automaton (NFA) N_1

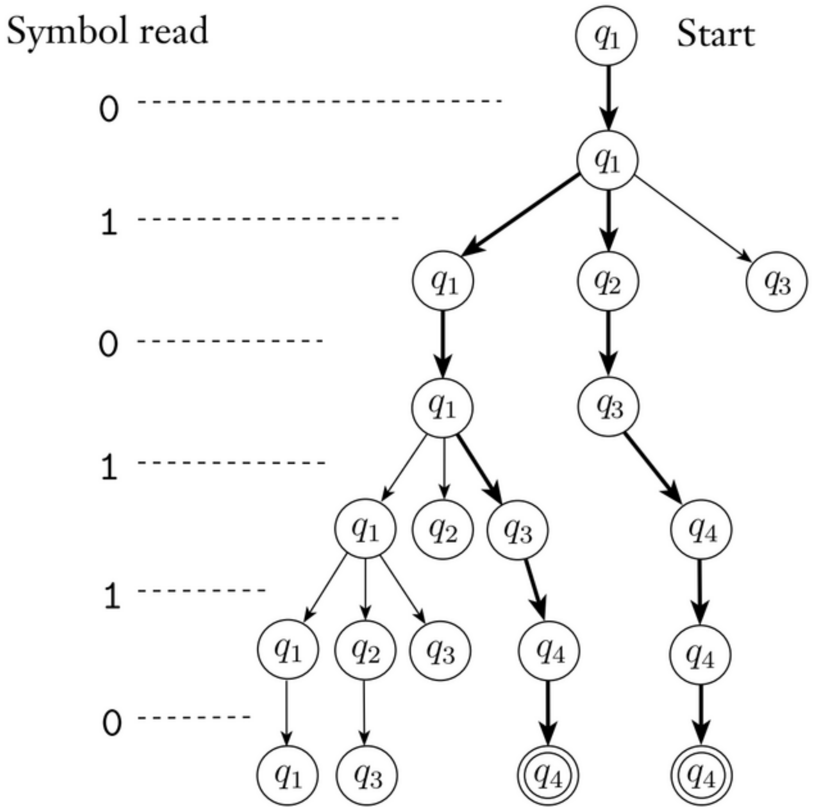
Finite Automata: Nondeterminism



Finite Automata: Nondeterminism



Input string: 010110



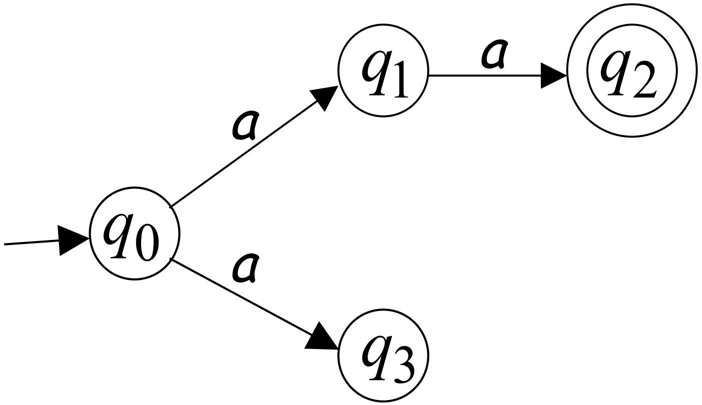
Finite Automata: Nondeterminism

- **How is NFA different from DFA?**
  - **In DFA, every state has exactly one exiting arrow for each symbol of the alphabet. In NFA, this is not necessarily the case.**
  - **In DFA, labels on transition arrows are symbols from the alphabet. In NFA, we may have an arrow for ε.**

Finite Automata: Nondeterminism

DFA	NFA
Cannot use empty string transition	Can use empty string transition
Rejects the string if it terminates in a non-accept state	Rejects the string only if all branches end up in non-accept states

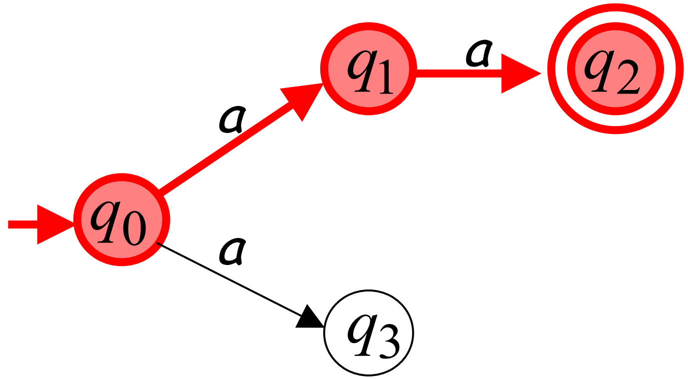
Finite Automata: Nondeterminism - Example



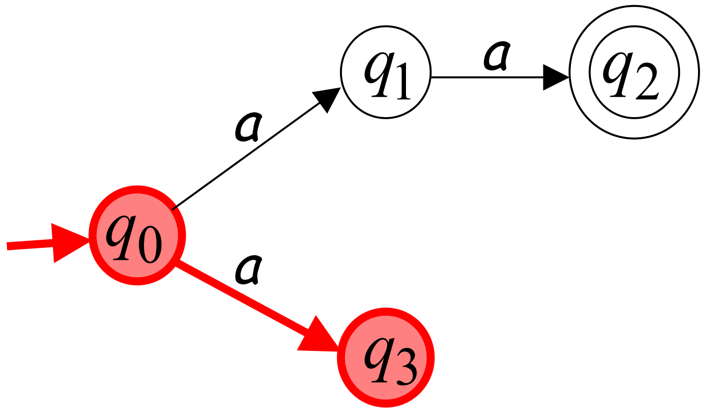
Example NFA

Finite Automata: Nondeterminism - Example

w = aa



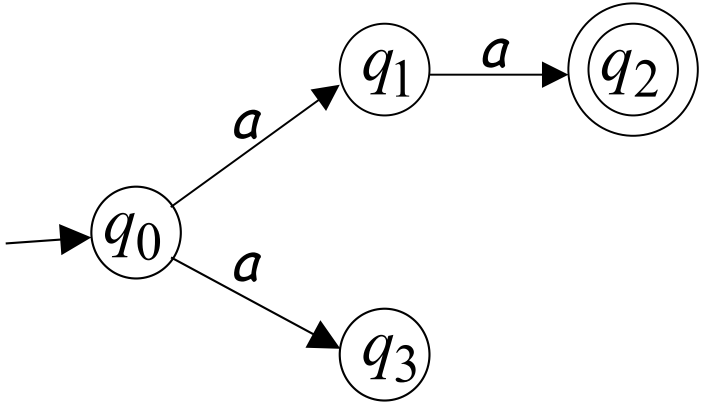
This branch accepts w



This branch rejects w

Finite Automata: Nondeterminism - Example

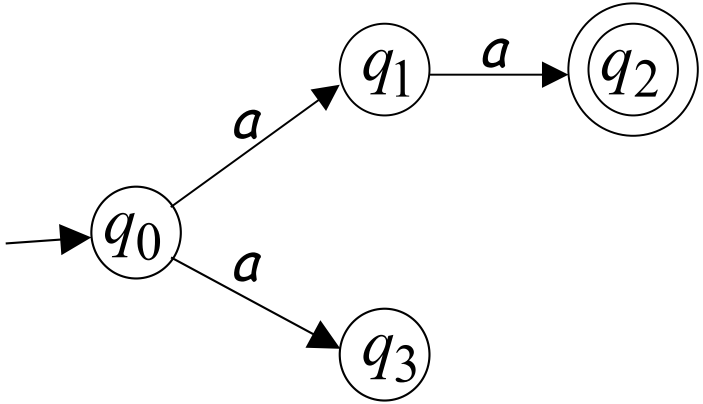
w = aaa



Will this NFA accept w?

Finite Automata: Nondeterminism - Example

What is the language that this NFA recognizes?

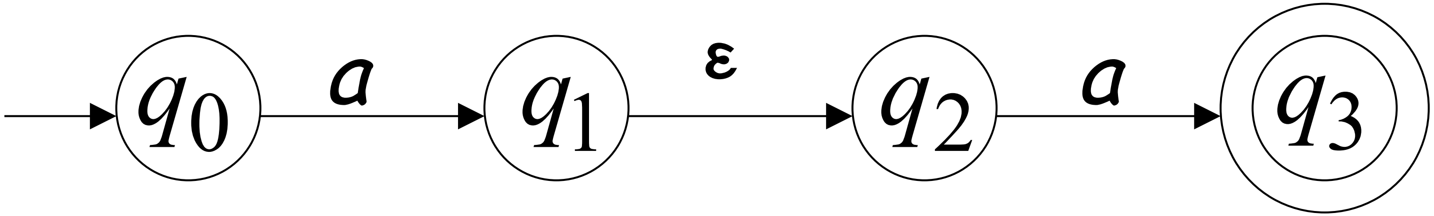


Finite Automata: Nondeterminism

- **NFA rejects a string if:**
  - **All input is consumed and the automaton ends up in a non-accept state**
  - **The input cannot be fully consumed**

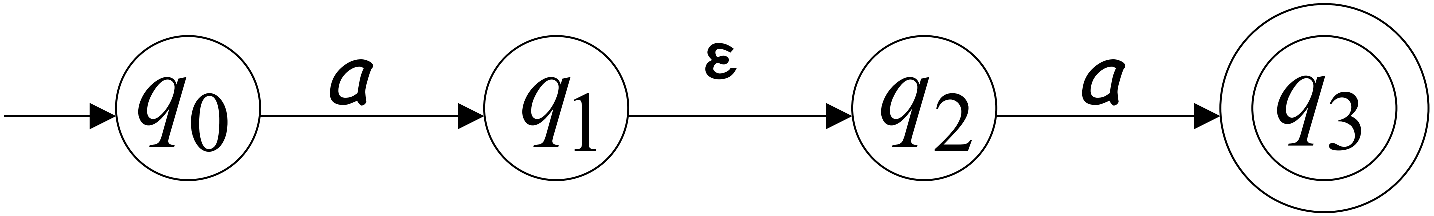
Finite Automata: Nondeterminism - Example

How can we process an empty alphabet symbol?

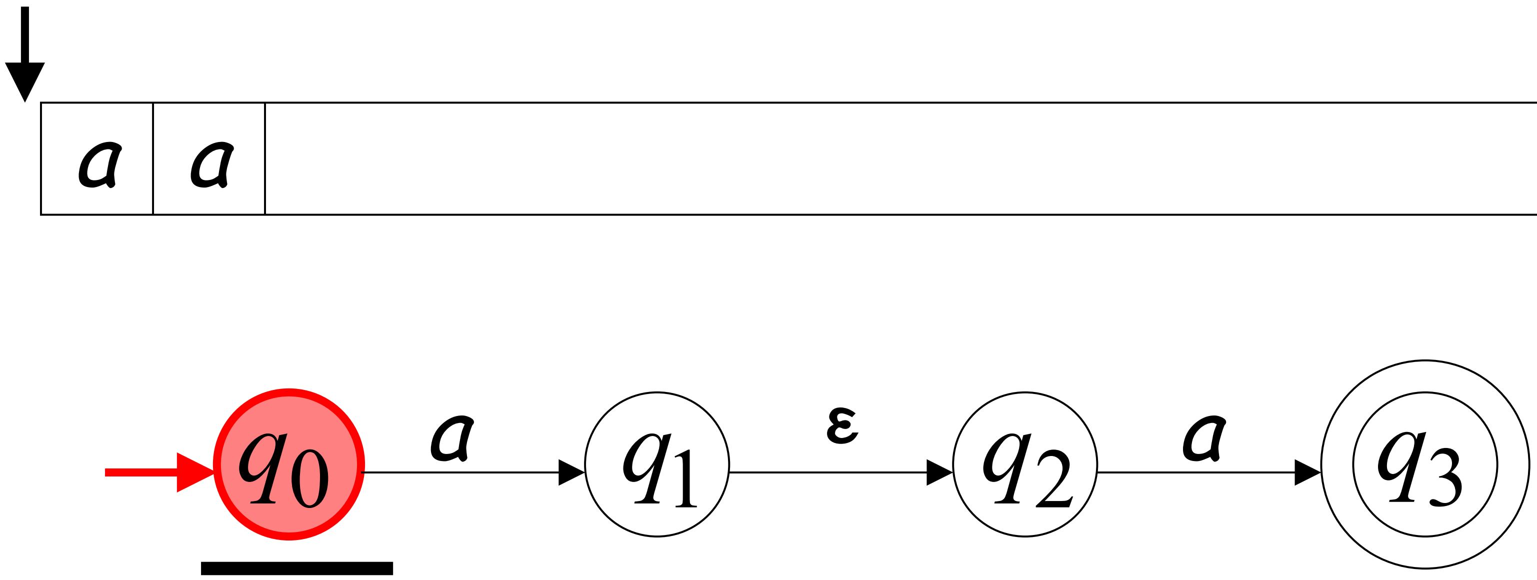


Finite Automata: Nondeterminism - Example

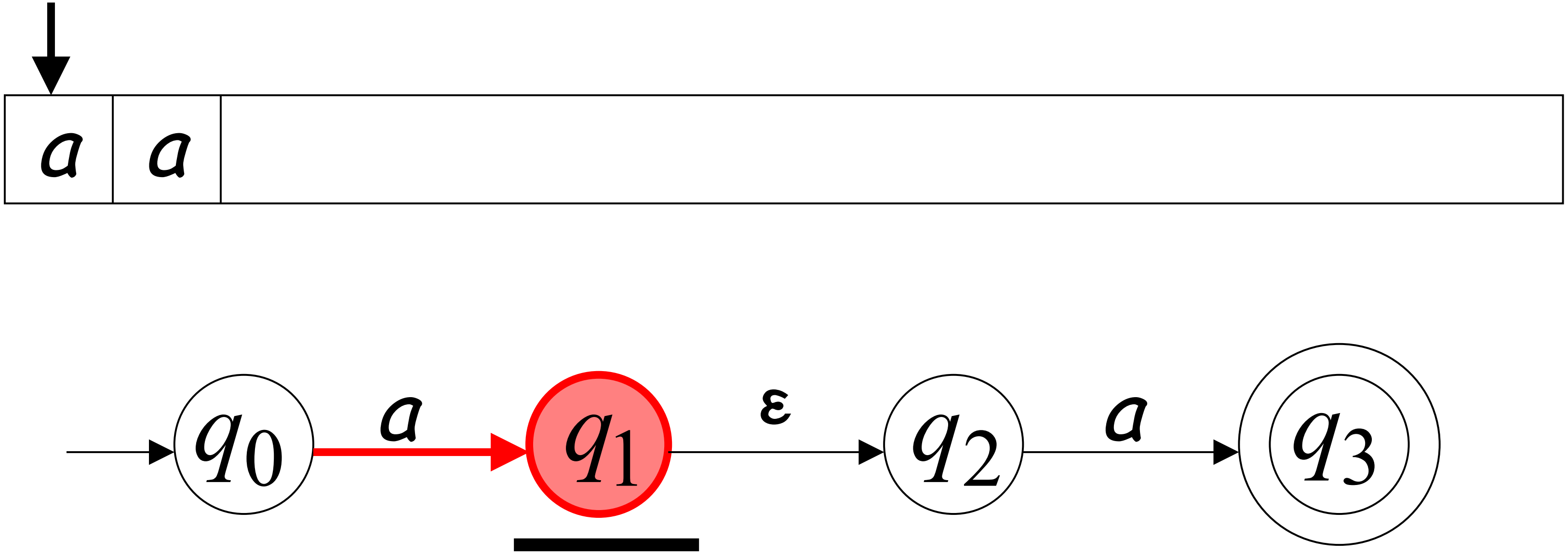
w = aa



Finite Automata: Nondeterminism - Example

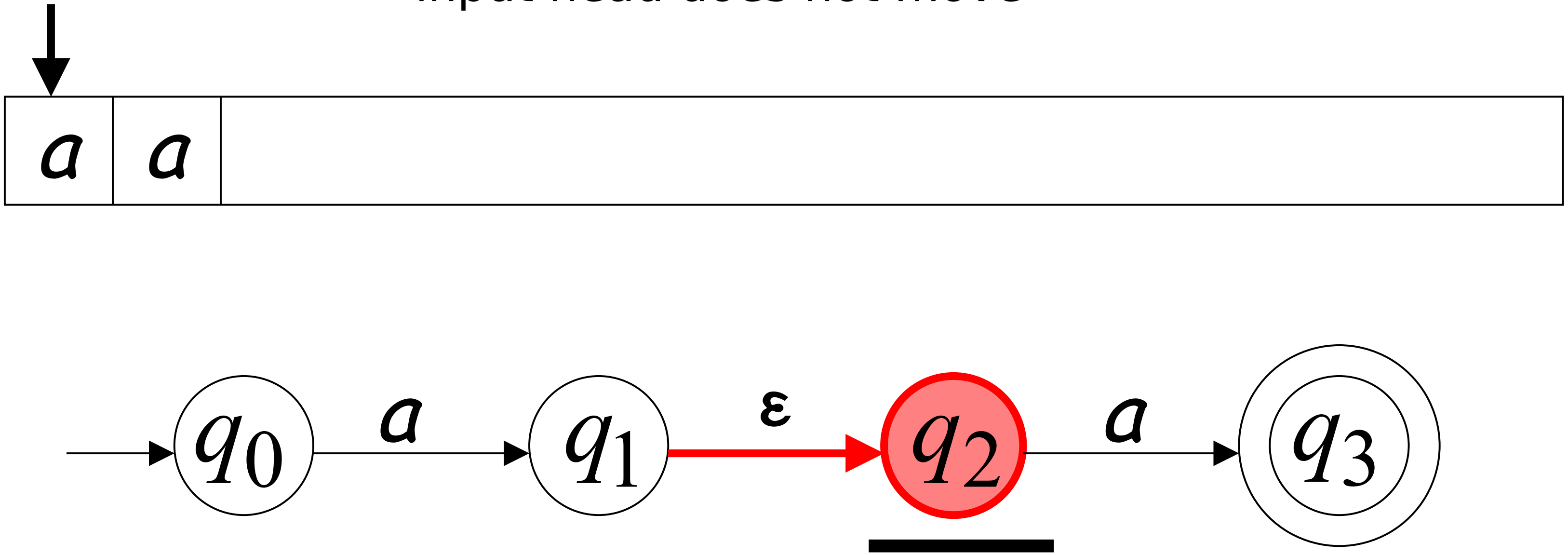


Finite Automata: Nondeterminism - Example

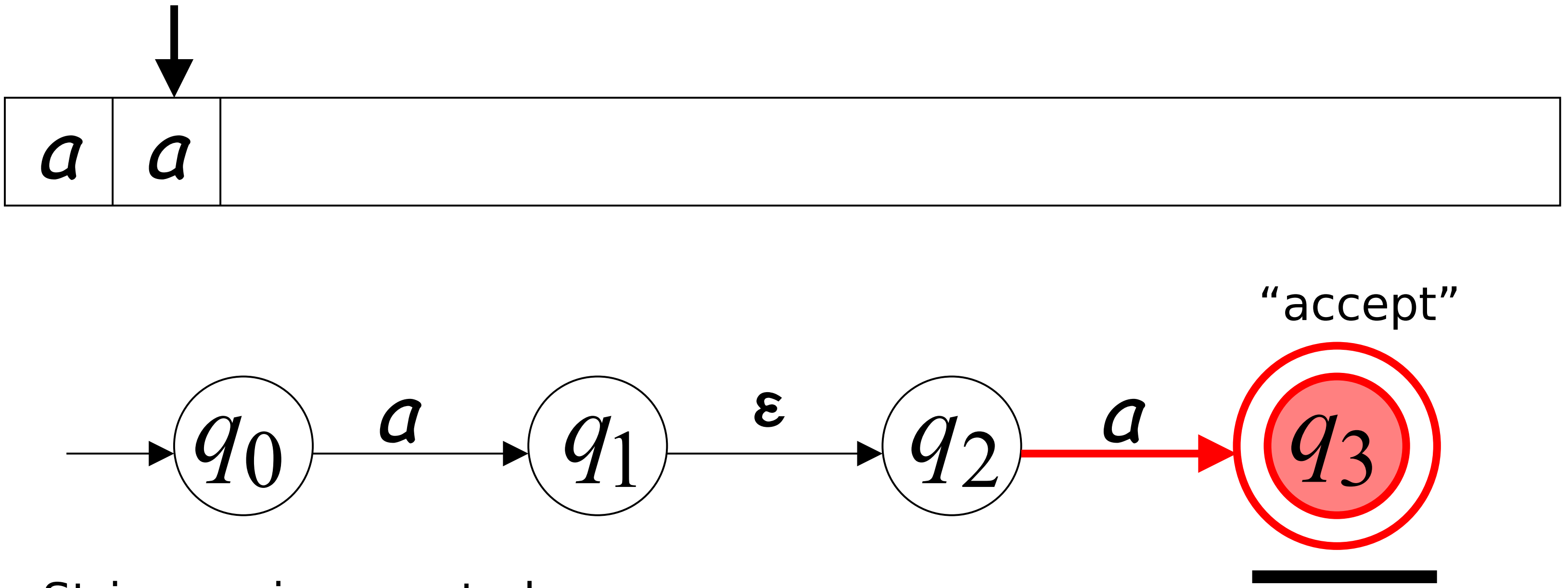


Finite Automata: Nondeterminism - Example

input head does not move



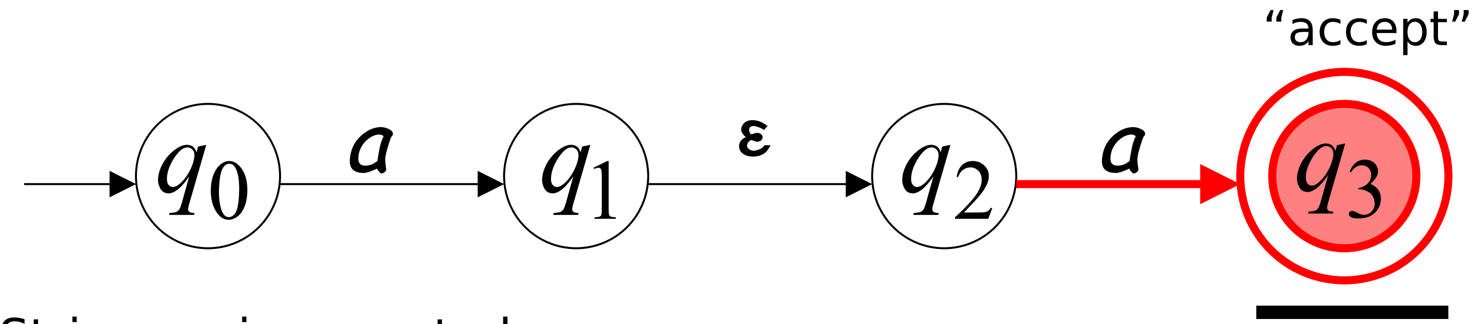
Finite Automata: Nondeterminism - Example



String ***aa*** is accepted

Finite Automata: Nondeterminism - Example

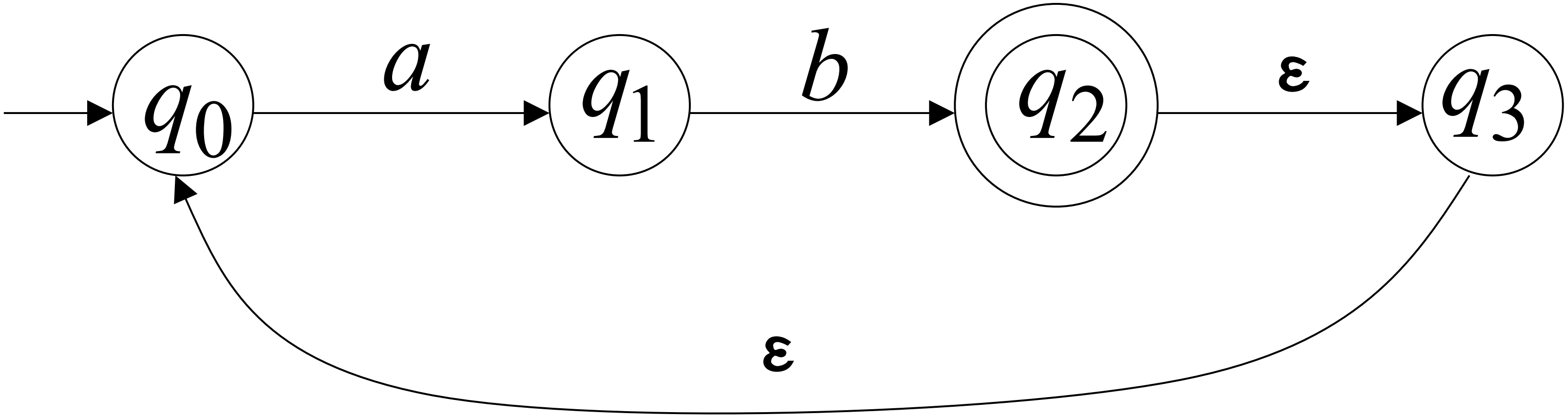
What language does this NFA recognize?



String ***aa*** is accepted

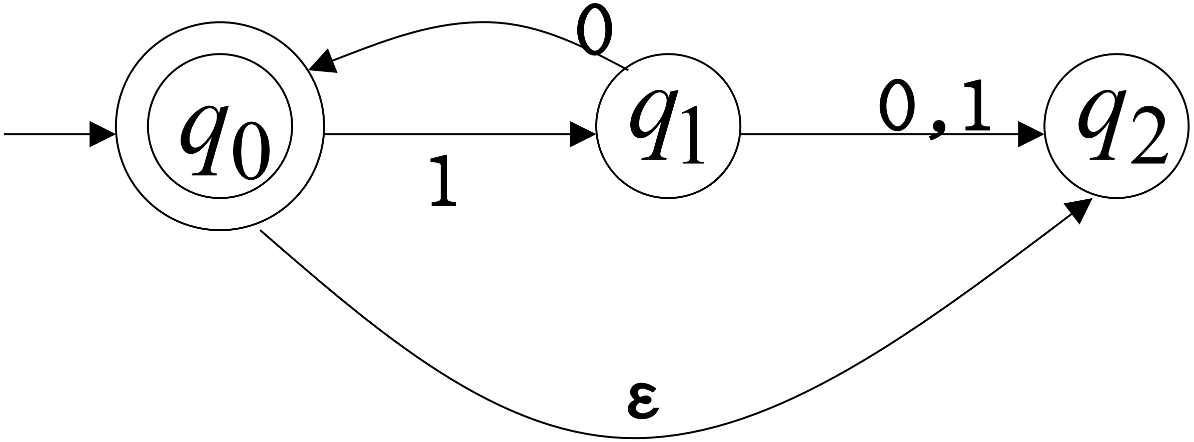
Finite Automata: Nondeterminism - Example

What language does this NFA recognize?



Finite Automata: Nondeterminism - Example

What language does this NFA recognize?



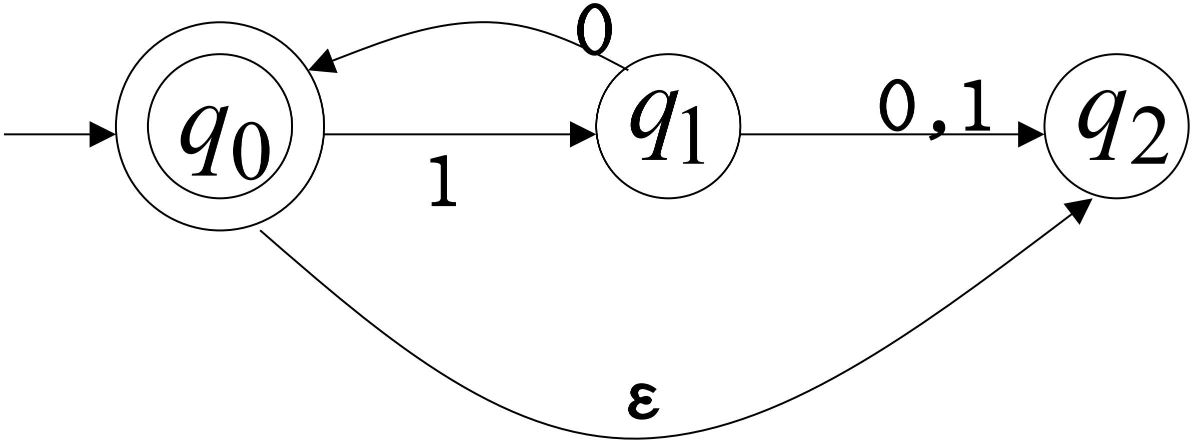
Finite Automata: Nondeterminism

- **Formal definition**
  - **Q: set of states (finite set)**
  - **Σ: input alphabet (finite set)**
  - **δ : $Q \times \Sigma_\varepsilon \rightarrow P(Q)$ transition function**
  - **$q_0 \in Q$ start state**
  - **$F \subseteq Q$: accept states**

Finite Automata: Nondeterminism - Example

- **Transition function**

$$\delta\,(q,x) = \{q_1, q_2, \ldots, q_k\}$$

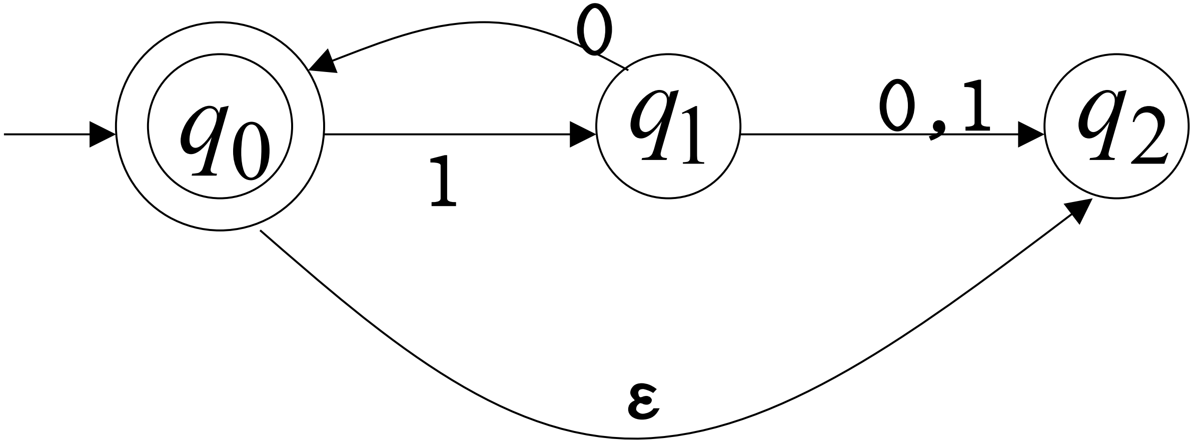


Finite Automata: Nondeterminism - Example

- **Transition function**

δ **(q,x) = {q_1, q_2, …, q_k}**

δ **(q_1, 0) = {q_0,q_2}**



Finite Automata: Nondeterminism - Example

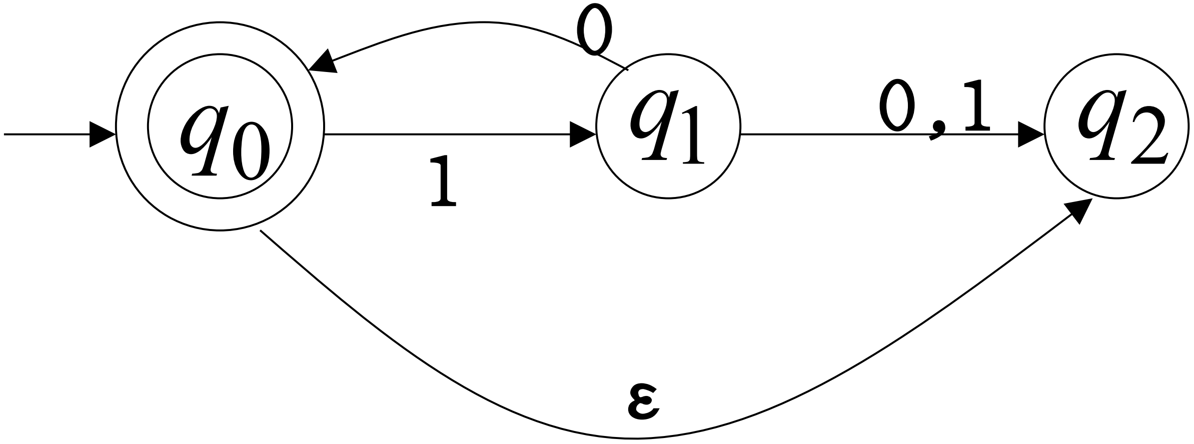
- **Transition function**

δ (q,x) = {q_1, q_2, …, q_k}

- **Extended transition function**

Applied on strings:

δ*(q,w) = {q_1, q_2, …, q_k}



Finite Automata: Nondeterminism - Example

➢**Transition function**

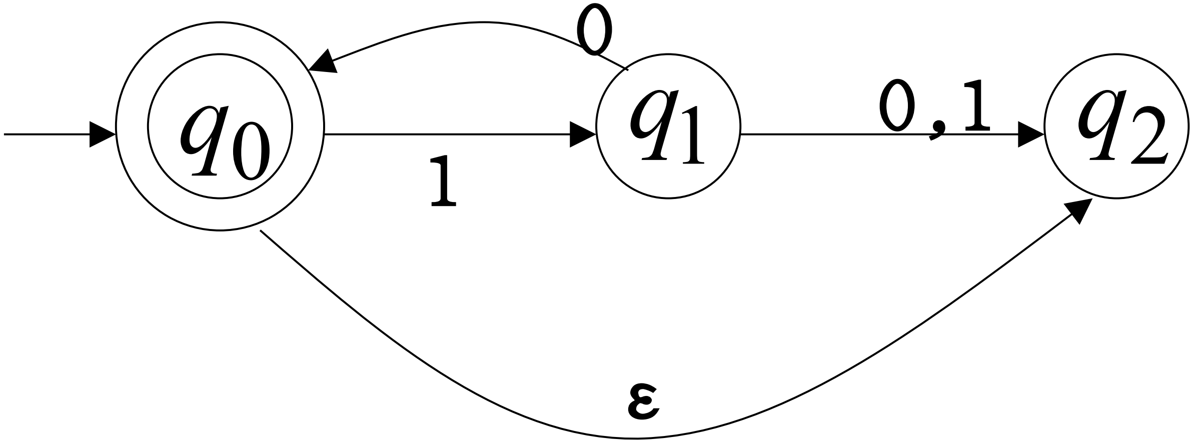
$\delta(q,x) = \{q_1, q_2, \ldots, q_k\}$

➢**Extended transition function**

Applied on strings:

$\delta^*(q,w) = \{q_1, q_2, \ldots, q_k\}$

$\delta^*(q_0, 11) = \{q_2\}$



Finite Automata: Nondeterminism

Why do we need NFA?

Finite Automata: Nondeterminism

Why do we need NFA?

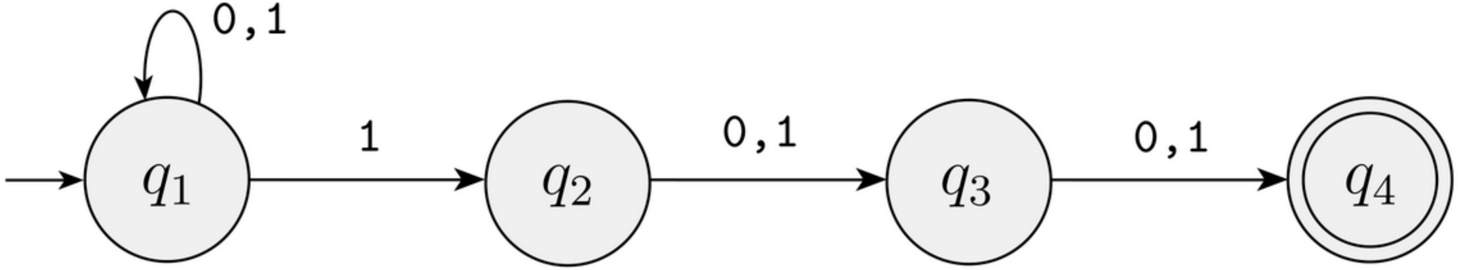
Can’t we just use DFA?

Finite Automata: Nondeterminism

Let A be the language consisting of all strings over {0,1} containing a 1 in the third position from the end (e.g., 000100 is in A but 0011 is not).

Finite Automata: Nondeterminism

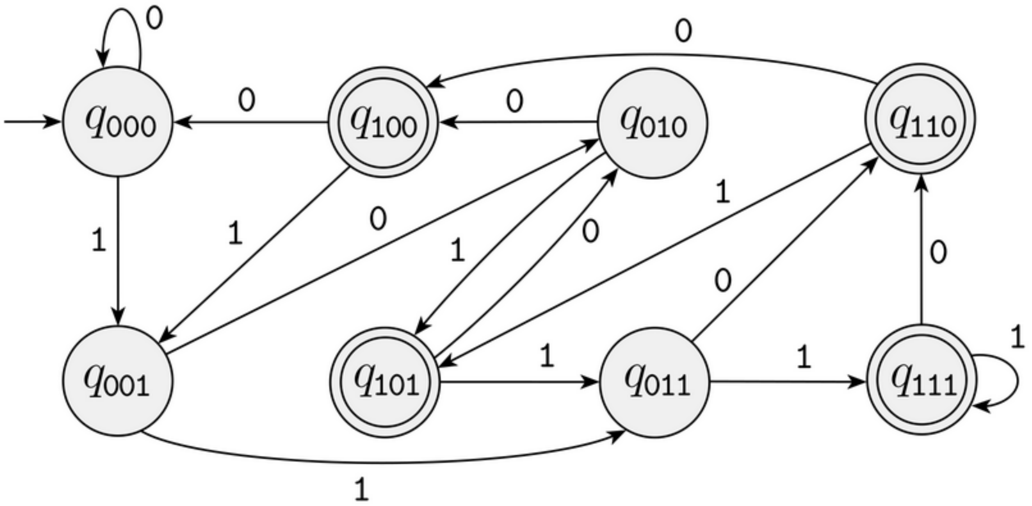
Let A be the language consisting of all strings over {0,1} containing a 1 in the third position from the end (e.g., 000100 is in A but 0011 is not).



NFA that recognizes A

Finite Automata: Nondeterminism

Let A be the language consisting of all strings over {0,1} containing a 1 in the third position from the end (e.g., 000100 is in A but 0011 is not).



DFA that recognizes A

Finite Automata: Nondeterminism

- **A language is recognized by NFA N:**
  - **$L(N) = \{w_1, w_2, w_3, \ldots, w_n\}$**

Finite Automata: Nondeterminism

- A language is recognized by NFA N:
  - $L(N) = \{w_1, w_2, w_3, \ldots, w_n\}$
  - $\delta^* (q_0, w_m) = \{q_i, \ldots, q_k, \ldots, q_j\}$

Finite Automata: Nondeterminism

- **A language is recognized by NFA N:**
  - **$L(N) = \{w_1, w_2, w_3, \ldots, w_n\}$**
  - **$\delta^*(q_0, w_m) = \{q_i, \ldots, q_k, \ldots, q_j\}$**
  - **And there is some $q_k \in F$**

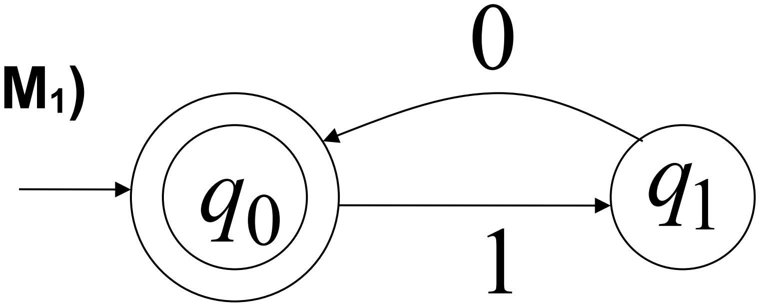
Finite Automata: Nondeterminism

- **Equivalence of machines**
  - **Machine M_1 is equivalent to machine M_2 if:**
    - **$L(M_1) = L(M_2)$**

- **Equivalence of machines**

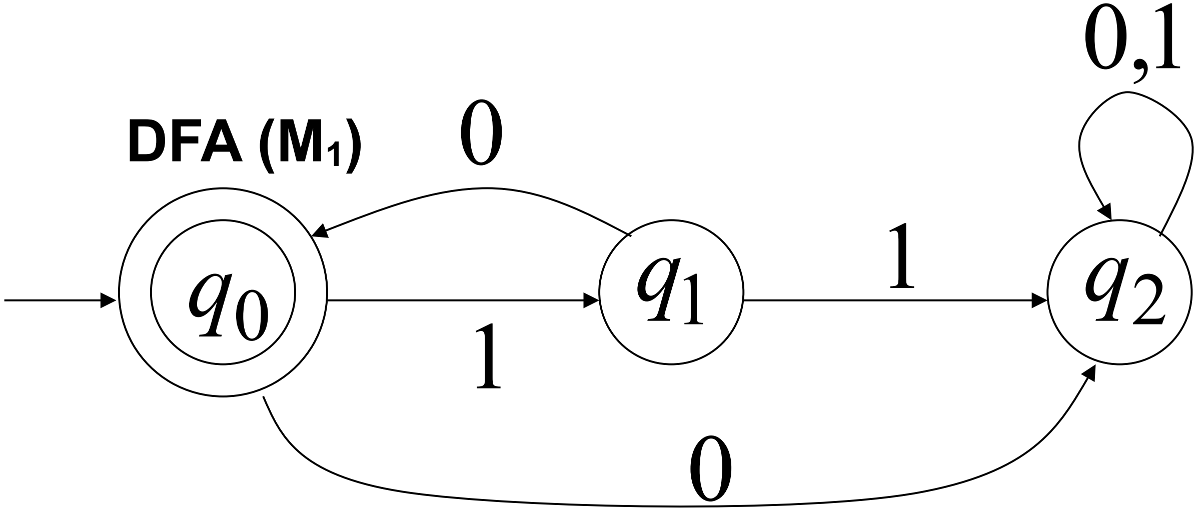
$$L(M_1) = \{10\}^*$$

NFA (M$_1$)



$$L(M_2) = \{10\}^*$$

DFA (M$_1$)



Finite Automata: Nondeterminism

$$\left\{ \begin{array}{c} \text{Languages} \\ \text{Recognized} \\ \text{By NFA} \end{array} \right\} = \left\{ \begin{array}{c} \text{Languages} \\ \text{Recognized} \\ \text{By DFA} \end{array} \right\}$$

Regular languages

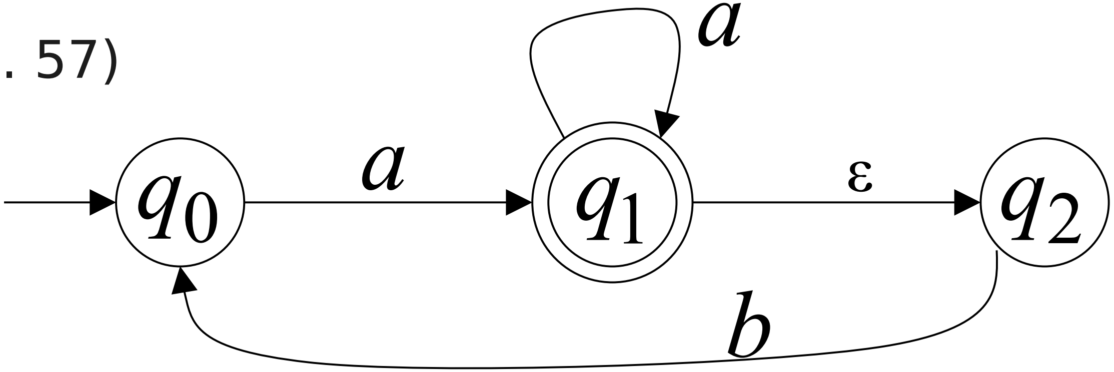
Finite Automata: Nondeterminism

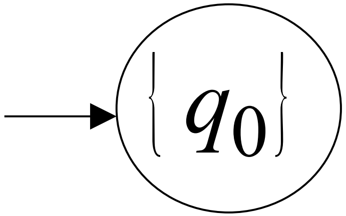
- **Converting NFA to DFA**
  - **NFA has states: q_0, q_1, …, q_n**
  - **DFA has states from the power set**

    $\varnothing$, $\{q_0\}$, $\{q_1\}$, $\{q_0,q_1\}$, $\{q_0,q_1,q_3\}$, …

Finite Automata: Nondeterminism

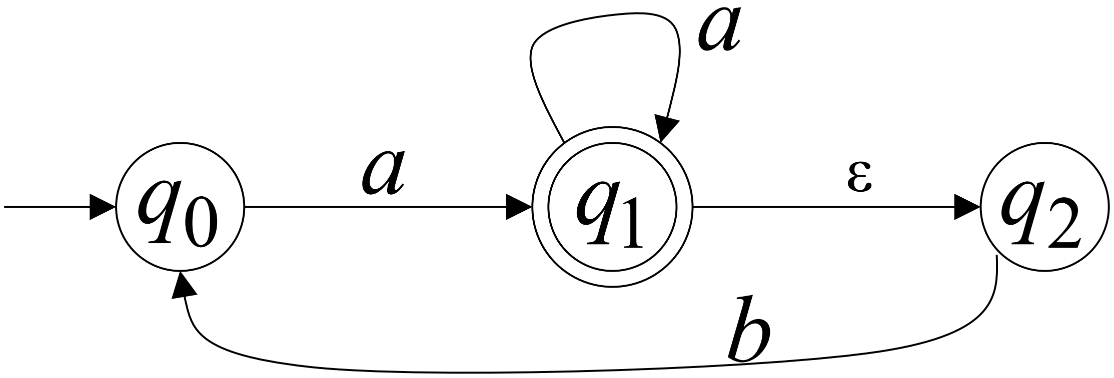
- **Converting NFA to DFA** (read p. 57)
  - **Step-1**
    - **Initial state of NFA: q_0**
    - **Initial state of DFA: $\{q_0\}$**





Finite Automata: Nondeterminism

- **Converting NFA to DFA**
  - **Step-2**
    - **Construct the DFA with states such that there is a state for each subset of NFA's states (power set)**

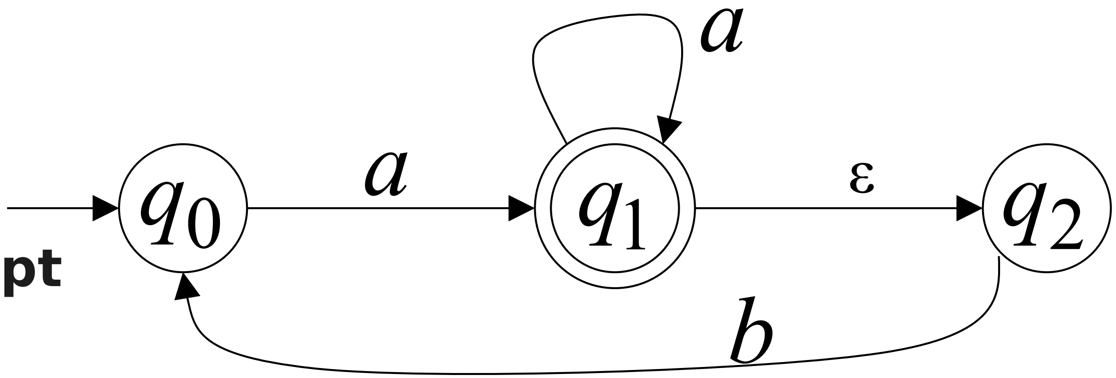


Possible states for DFA

{∅, {q0}, {q1}, {q2}, {q0, q1}, {q0, q2}, {q1,q2}, {q0, q1, q2}}

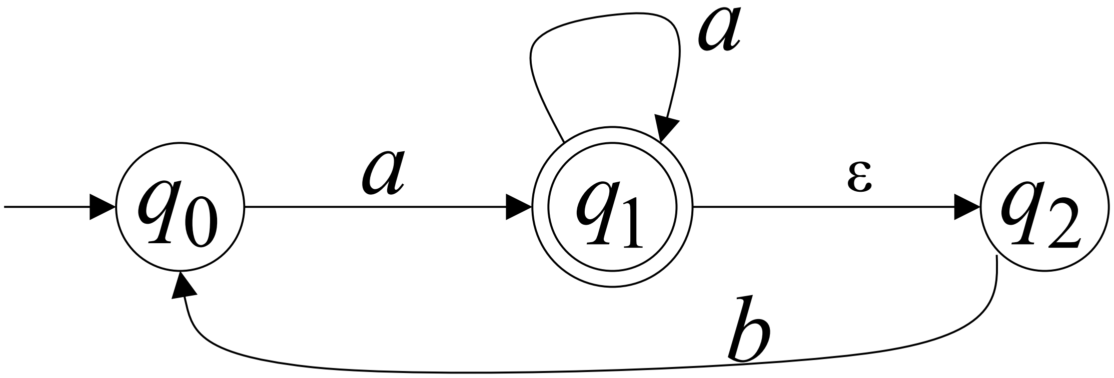
Finite Automata: Nondeterminism

- **Converting NFA to DFA**
  - **Step-3**
    - **Determine start and accept states of the DFA**
      - **Start is E({q_0}), the set of states that are reachable from q_0 traveling along ε arrows.**
      - **Accept states are those subsets that contain NFA's accept states**



Finite Automata: Nondeterminism

- **Converting NFA to DFA**
  - **Step-4**
    - **Determine DFA's transition function. Each of DFA's states goes to one place for each input symbol**



Finite Automata: Nondeterminism

- **Converting NFA to DFA**
  - **Step-5 (optional)**
    - **Simplify the DFA by removing unnecessary states**
      - **If no arrows point to some state (not the start state), we can remove them without affecting the performance of the DFA.**

Finite Automata: Nondeterminism

- **Converting NFA to DFA**
  - **Step-5 (optional)**
    - **Simplify the DFA by removing unnecessary states**
      - **If no arrows point to some state (not the start state), we can remove them without affecting the performance of the DFA.**

Read example 1.41 in the textbook

Homework

- Exercises
  - 1.7
- Reading
  - Example 1.41
  - Section 1.3