Nondeterminism

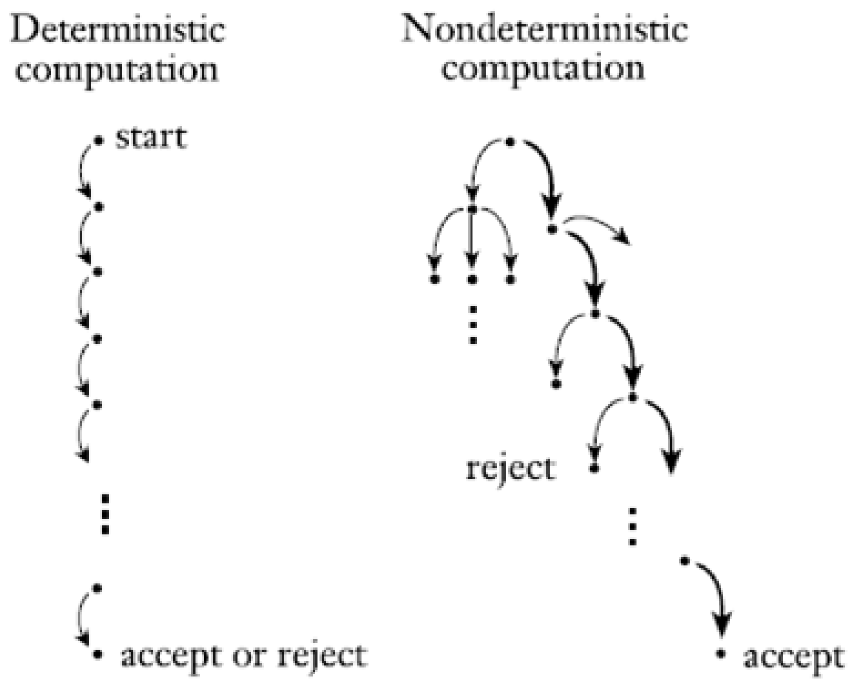
Summary

- Nondeterministic Finite Automata
- View FA as tree graph
- NFA vs DFA
- Closure under concatenation using NFA
- Closure under star using NFA

Deterministic Finite Automata

- All the previous examples are **deterministic** computation
  - At every state, there is at most 1 edge related to a particular input symbol
  - One path for each input

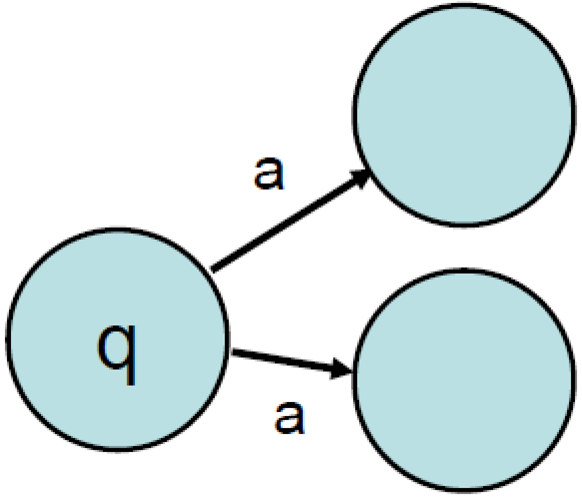
- **Nondeterministic** computation
  - Several choices may exist for the next state at any point
  - Every deterministic FA (**DFA**) is a nondeterministic FA (**NFA**)
    - Not Vice Versa



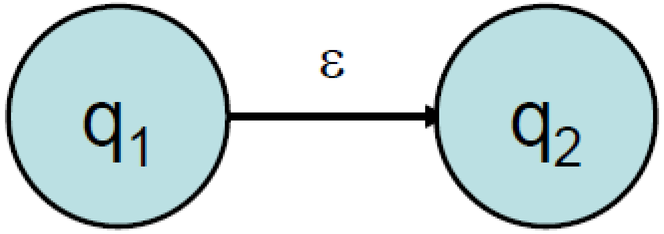
Nondeterministic Finite Automata

- DFA can be generalized by adding nondeterminism
  - Allow several alternative computations on the same input string

- Two changes:
  1. Allow **transition function**, $\delta(q,a)$, to specify more than one successor state:



  2. Add **ε-transitions** (empty strings)
     - Transitions made "for free", without "consuming" any input symbols.



How NFAs compute

- Since transitions of states is unknown, parallel processing of multiple copies of the NFA is necessary
  - Can be considered in multiple states at once at every input symbol.

- Follow allowed arrows in any possible way
  - "Consumes" the designated input symbols at after each arrow
  - New paths are followed after every split
    - All paths run in parallel
    - If there is no arrow for the next input symbol, path is terminated.

- Optionally follow any **ε-arrow** at any time, without "consuming" any input.
  - Creates another path

- Accepts a string if **some** allowed sequence of transitions on that string leads to an accepting state.

Formal Definition of an NFA

- An NFA can be formally defined as a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where:
  - Q is a finite set of states
  - Σ is a finite set (alphabet) of input symbols
  - $\delta: Q \times \Sigma_\varepsilon \rightarrow P(Q)$ is the **transition function**

The arguments are a state and either an alphabet symbol or ε. Σ_ε means $\Sigma \cup \{\varepsilon\}$.

The result is a set of states.

  - $q_0 \in Q$, is the start state
  - $F \subseteq Q$, set of accept states

- P(Q): powerset of Q
  - The set of all subsets of Q
  - Can be in multiple states at once
- How many states in P(Q)?
- Example:
  - $Q = \{a,b,c\}$
  - $P(Q) = \{ \{\}, \{a\}, \{b\}, \{c\}, \{a,b\}, \{a,c\}, \{b,c\}, \{a,b,c\} \}$

Formal Definition of Computation for NFA

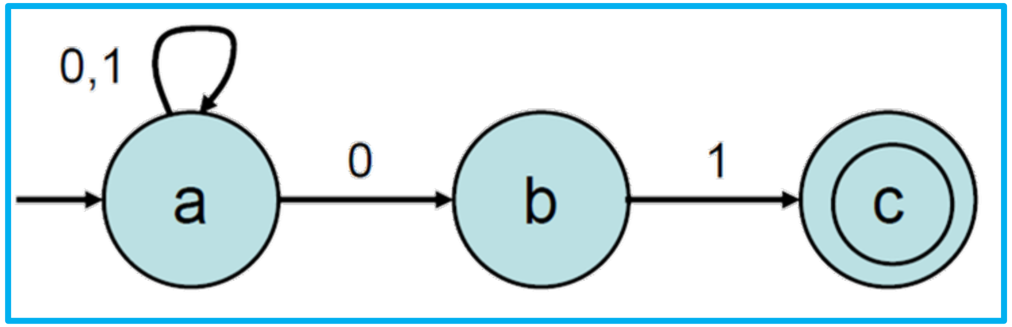
- $\delta^*(q,w)$ = States that can be reached from q by following string w

- String w is accepted if $\delta^*(q, w) \cap F \neq \emptyset$
  - F = set of accept states
  - At least one of the possible end states is an accepting state
    - Rejected otherwise

- $L(M) = \{w | w \textit{ is accepted by } M\}$
  - Language recognized by NFA M

NFA Example 1

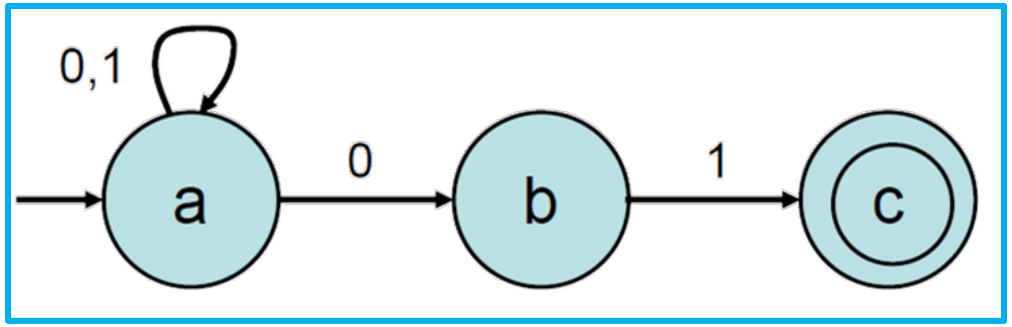
- ε is now a column
- Now being mapped to sets of states
- Example
  - $Q = \{a,b,c\}$
  - $\Sigma = \{0,1\}$
  - δ: $Q \times \Sigma_\varepsilon \rightarrow P(Q)$ is the **transition function**
  - $q_0 = a$, is the start state
  - $F = \{c\}$



	0	1	ε
a	{a,b}	{a}	∅
b	∅	{c}	∅
c	∅	∅	∅

NFA Example 1

- L(M) = {w | w ends with 01}
  - M accepts exactly the strings in this set



- Example Input String
  - Computations for input word w = 101:
    - **Many Combinations**, some listed below

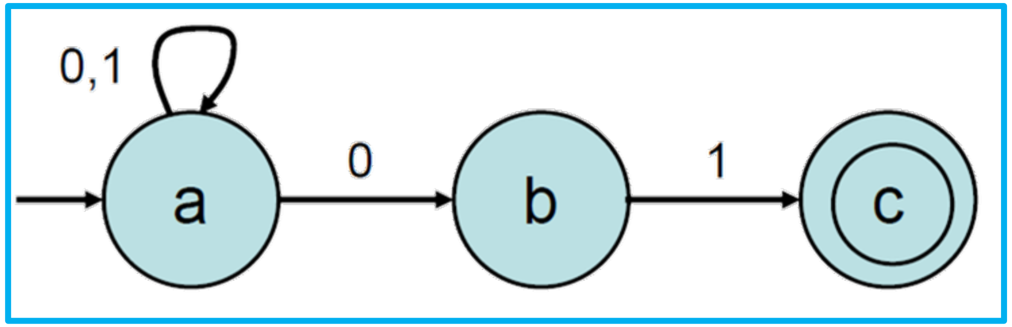
Input Word w	1	0	1
Path 1	a	a	a
Path 2	a	b	c

	0	1	ε
a	{a,b}	{a}	∅
b	∅	{c}	∅
c	∅	∅	∅

  - Since c is an accepting state, M accepts 101

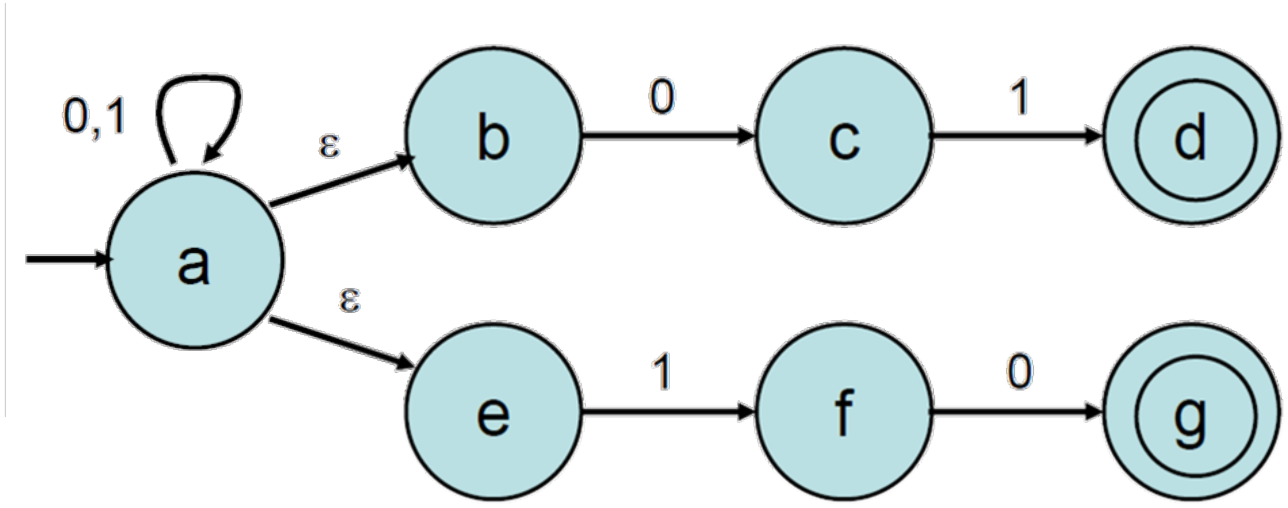
NFA Example 1

- Computations for input word w = 0010:
  - Possible states after 0 input: {a,b}
    - After another 0: {a,b}
    - After 1: {a,c}
      - After the 1 input, state is either c or a.
      - Since 0 cannot be consumed at c,
        - Path is terminated
    - After final 0: {a,b}
  - Neither a nor b are accepting states
    - M does not accept 0010
- $\{a\} \underset{\rightarrow}{0} \{a,b\} \underset{\rightarrow}{0} \{a,b\} \underset{\rightarrow}{1} \{a,c\} \underset{\rightarrow}{0} \{a,b\}$



	0	1	ε
a	{a,b}	{a}	∅
b	∅	{c}	∅
c	∅	∅	∅

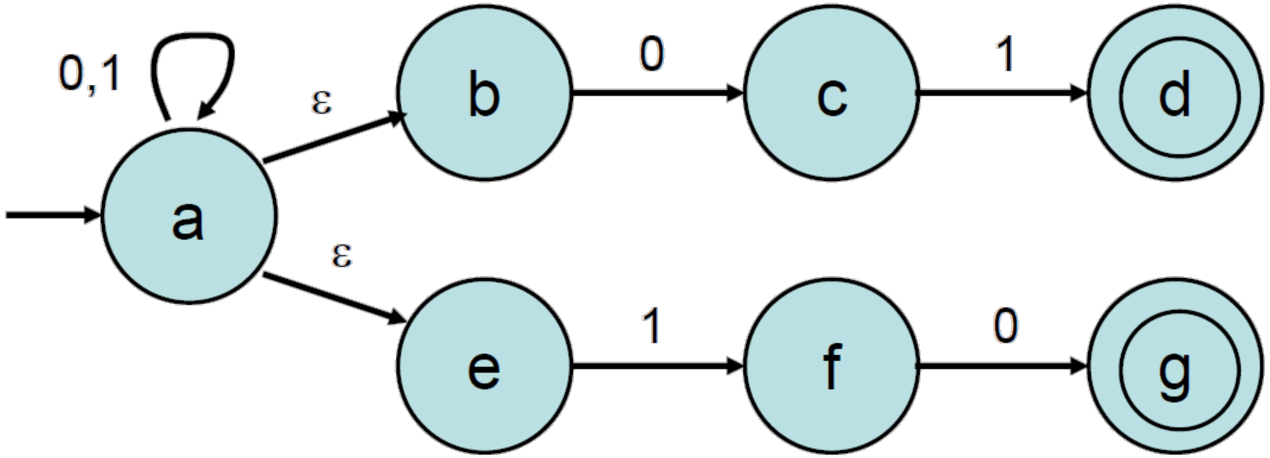
NFA Example 2



	0	1	ε
a	{a}	{a}	{b,e}
b	{c}	∅	∅
c	∅	{d}	∅
d	∅	∅	∅
e	∅	{f}	∅
f	{g}	∅	∅
g	∅	∅	∅

NFA Example 2

- L(M)={ w | w ends with 01 or 10 }
- Computations for w = 0010
  - Possible states after <u>no input</u>: {a,b,e}
  - After 0: {a,b,e,c}
  - After 0: {a,b,e,c}
  - After 1: {a,b,e,d,f}
  - After 0: {a,b,e,c,g}
  - Since g is an accepting state
    - M accepts 0010



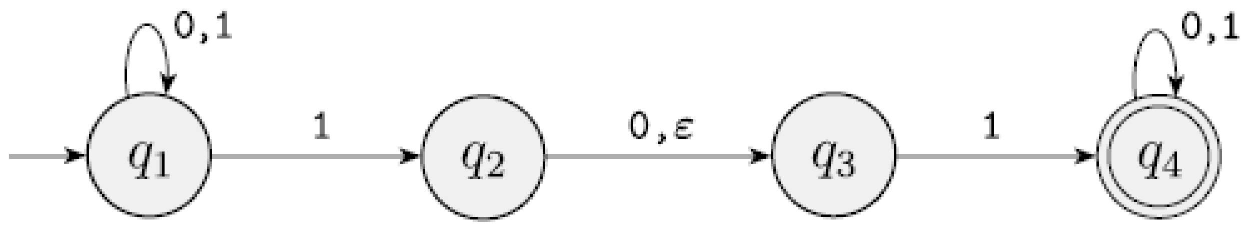
- $\{a,b,e\} \xrightarrow{0} \{a,b,e,c\} \xrightarrow{0} \{a,b,e,c\} \xrightarrow{1} \{a,b,e,d,f\} \xrightarrow{0} \{a,b,e,c,g\}$

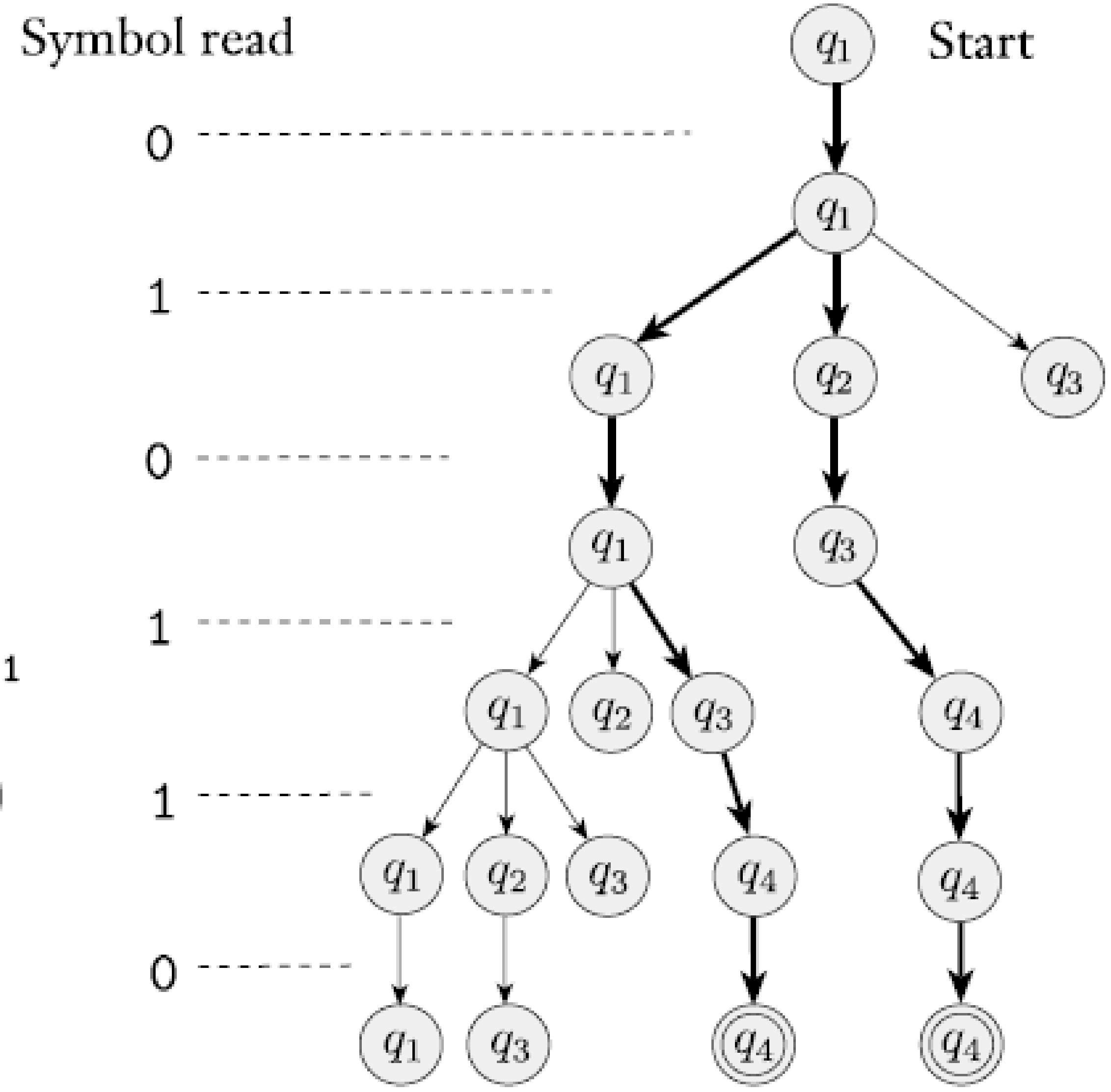
- Path to accepting state
  - $a \xrightarrow{0} a \xrightarrow{0} a \xrightarrow{\varepsilon} e \xrightarrow{1} f \xrightarrow{0} g$

Viewing Computations as a Tree

- Every input string of a NFA can be viewed as a Tree

- Sample input string: 010110





Viewing Computations as a Tree

- Input w = 01