Formal definition of NFA

We use $\sum_{\varepsilon}$ to mean $\sum \cup \{\varepsilon\}$

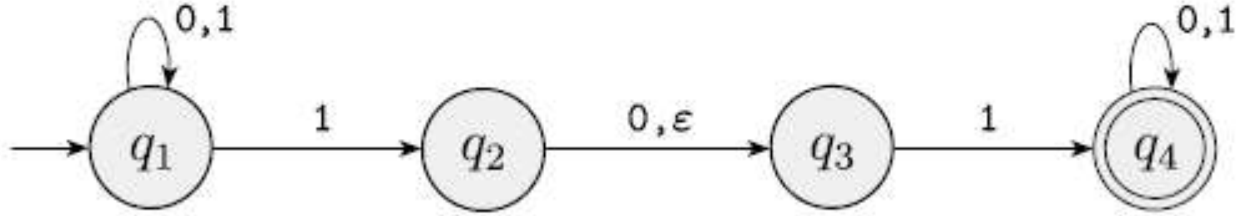
DEFINITION 1.37

A ***nondeterministic finite automaton*** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set of states,
2. Σ is a finite alphabet,
3. $\delta \colon Q \times \Sigma_{\varepsilon} \longrightarrow \mathcal{P}(Q)$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subseteq Q$ is the set of accept states.

EXAMPLE 1.38

Recall the NFA N_1:



The formal description of N_1 is $(Q, \Sigma, \delta, q_1, F)$, where

1. $Q = \{q_1, q_2, q_3, q_4\}$,
2. $\Sigma = \{0,1\}$,
3. δ is given as

	0	1	ε
q_1	$\{q_1\}$	$\{q_1, q_2\}$	$\emptyset$
q_2	$\{q_3\}$	$\emptyset$	$\{q_3\}$
q_3	$\emptyset$	$\{q_4\}$	$\emptyset$
q_4	$\{q_4\}$	$\{q_4\}$	$\emptyset$,

4. q_1 is the start state, and
5. $F = \{q_4\}$.

The formal definition of computation for an NFA is similar to that for a DFA. Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA and w a string over the alphabet Σ. Then we say that N ***accepts*** w if we can write w as $w = y_1 y_2 \cdots y_m$, where each y_i is a member of Σ_ε and a sequence of states $r_0, r_1, \ldots, r_m$ exists in Q with three conditions:

1. $r_0 = q_0$,
2. $r_{i+1} \in \delta(r_i, y_{i+1})$, for $i = 0, \ldots, m-1$, and
3. $r_m \in F$.

Equivalence of NFAs and DFAs

- We say two machines are equivalent if they recognize the same language.

THEOREM 1.39

Every nondeterministic finite automaton has an equivalent deterministic finite automaton.

Proof

- Proof by construction.
  - We build a equal DFA for the given NFA

Let $N = (Q, \Sigma, \delta, q_0, F)$ be the NFA recognizing some language A.

We construct a DFA $M = (Q', \Sigma, \delta', q_0', F')$ recognizing A.

- First, for understanding purpose, we assume that there are no edges with Ɛ transitions.

Let $N = (Q, \Sigma, \delta, q_0, F)$ be the NFA recognizing some language A. We construct a DFA $M = (Q', \Sigma, \delta', q_0', F')$ recognizing A.

1. $Q' = \mathcal{P}(Q)$.
   Every state of M is a set of states of N. Recall that $\mathcal{P}(Q)$ is the set of subsets of Q.

2. For $R \in Q'$ and $a \in \Sigma$, let $\delta'(R, a) = \{q \in Q \mid q \in \delta(r, a) \text{ for some } r \in R\}$.

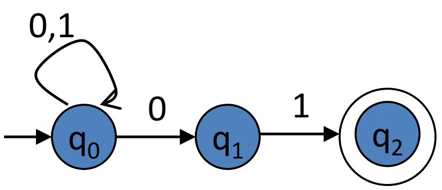
$$\delta'(R, a) = \bigcup_{r \in R} \delta(r, a).$$

3. $q_0' = \{q_0\}$.
   M starts in the state corresponding to the collection containing just the start state of N.

4. $F' = \{R \in Q' \mid R \text{ contains an accept state of } N\}$.
   The machine M accepts if one of the possible states that N could be in at this point is an accept state.

Subset construction: Example

NFA:

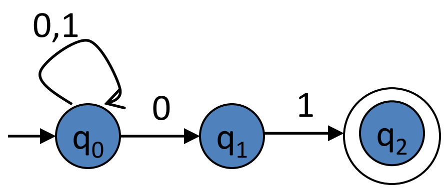


NFA to DFA construction: Example

<u>Idea:</u> To avoid enumerating all of power set, do "lazy creation of states"

- *L = {w | w ends in 01}*

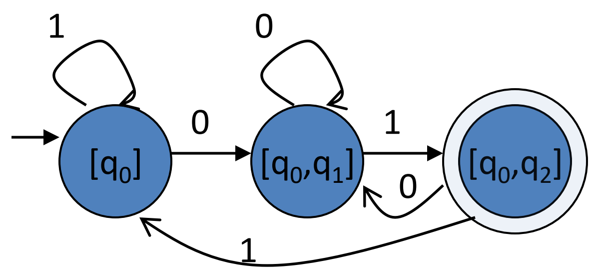
NFA:



δ_N	0	1
→q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	Ø	$\{q_2\}$
*q_2	Ø	Ø

DFA:

δ_D
~~Ø~~
→$\{q_0\}$
~~$\{q_1\}$~~
~~*$\{q_2\}$~~
$\{q_0,q_1\}$
*$\{q_0,q_2\}$
~~*$\{q_1,q_2\}$~~
~~*$\{q_0,q_1,q_2\}$~~



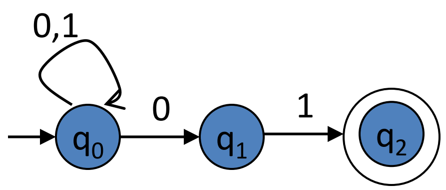
δ_D	0	1
→$[q_0]$	$[q_0,q_1]$	$[q_0]$
$[q_0,q_1]$	$[q_0,q_1]$	$[q_0,q_2]$
*$[q_0,q_2]$	$[q_0,q_1]$	$[q_0]$

0. Enumerate all possible subsets
1. Determine transitions
2. Retain only those states reachable from $\{q_0\}$

NFA to DFA: Repeating the example using *LAZY CREATION*

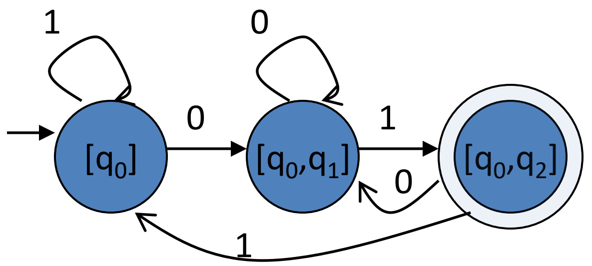
- *L = {w | w ends in 01}*

NFA:



δ_N	0	1
→ q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	Ø	$\{q_2\}$
$*q_2$	Ø	Ø

DFA:



δ_D	0	1
→ $[q_0]$	$[q_0,q_1]$	$[q_0]$

Main Idea:

Introduce states as you go (on a need basis)

Can you convert the following

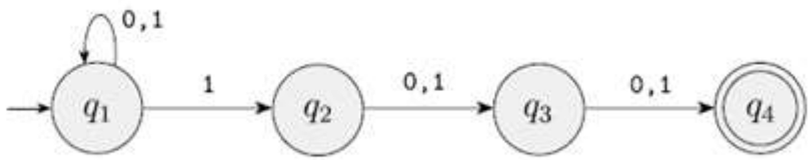


FIGURE 1.31
The NFA N_2

- What is the language accepted by this?

Now, considering Ɛ arrows

- For this purpose, we define Ɛ-CLOSURE of a set of states *R* as E(R) or ECLOSE(R)

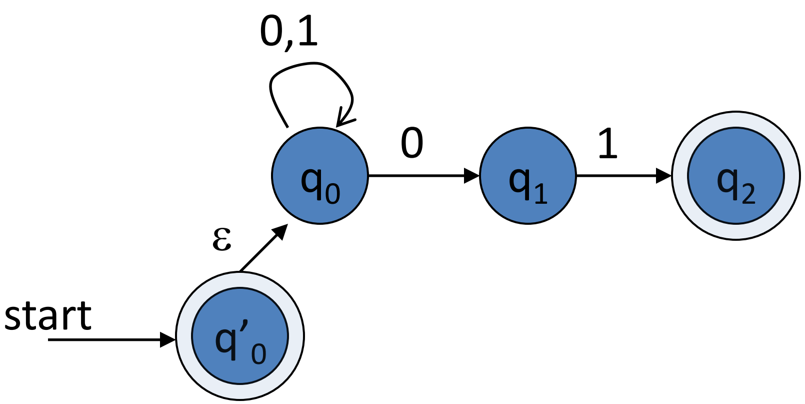
Formally, for $R \subseteq Q$ let

$E(R) = \{q |\ q$ can be reached from R by traveling along 0 or more ε arrows$\}$.

- *E(R)* is Ɛ-CLOSURE of *R*.

Example of an ε-NFA

L = {w | w is empty, or if non-empty will end in 01}



δ_E	0	1	ε	
$\rightarrow$ $*q'_0$	Ø	Ø	$\{q'_0,q_0\}$	ECLOSE(q'_0)
q_0	$\{q_0,q_1\}$	$\{q_0\}$	$\{q_0\}$	ECLOSE(q_0)
q_1	Ø	$\{q_2\}$	$\{q_1\}$	ECLOSE(q_1)
$*q_2$	Ø	Ø	$\{q_2\}$	ECLOSE(q_2)

- ε-closure of a state q, ***ECLOSE(q)***, is the set of all states (including itself) that can be *reached* from q by repeatedly making an arbitrary number of ε-transitions.

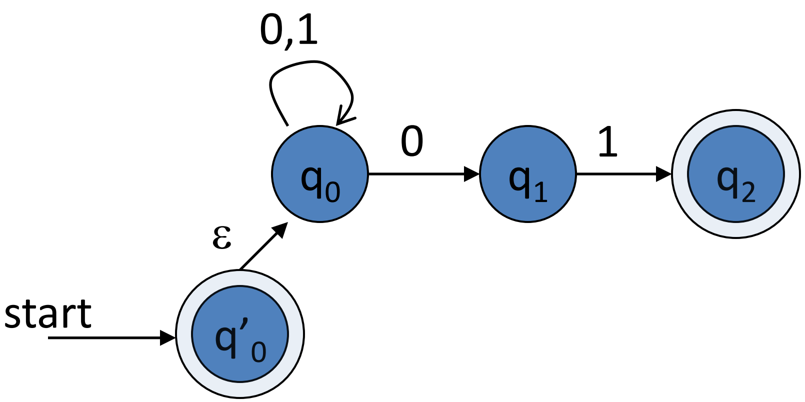
To simulate any transition:

Step 1) Go to all immediate destination states.

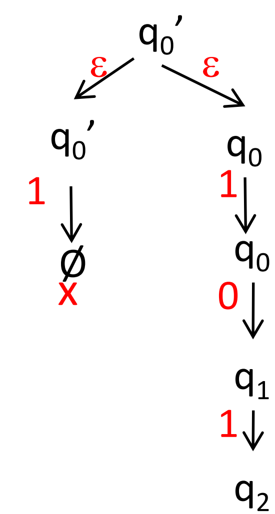
Step 2) From there go to all their ε-closure states as well.

Example of an ε-NFA

L = {w | w is empty, or if non-empty will end in 01}



Simulate for w=101:



δ_E	0	1	ε
→*q'_0	Ø	Ø	$\{q'_0, q_0\}$
q_0	$\{q_0, q_1\}$	$\{q_0\}$	$\{q_0\}$
q_1	Ø	$\{q_2\}$	$\{q_1\}$
*q_2	Ø	Ø	$\{q_2\}$

ECLOSE(q'_0)

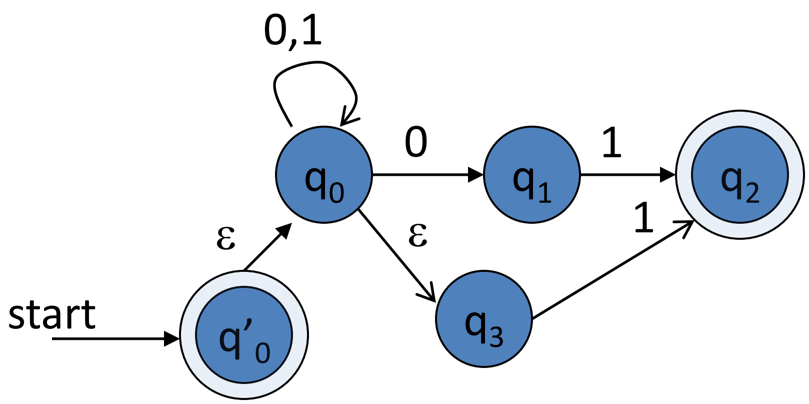
ECLOSE(q_0)

To simulate any transition:

Step 1) Go to all immediate destination states.

Step 2) From there go to all their ε-closure states as well.

Example of another ε-NFA

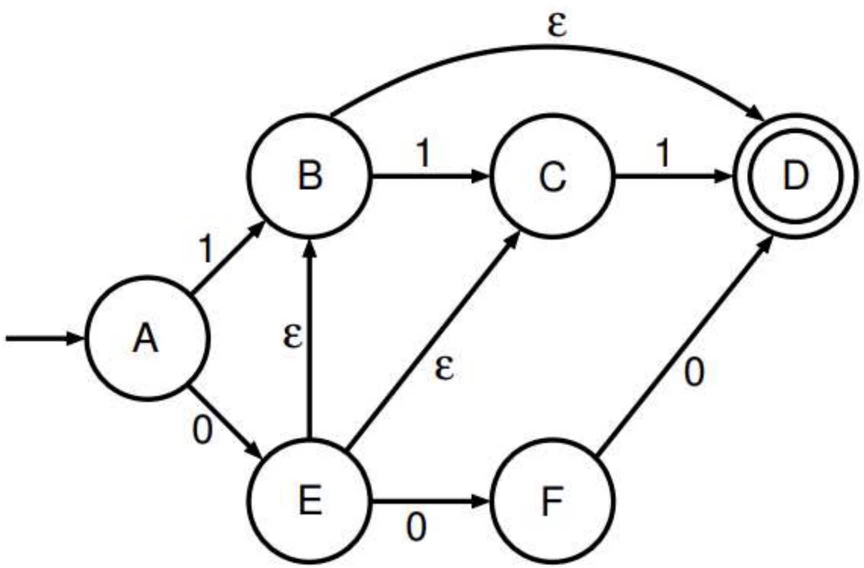


Simulate for w=101:

?

δ_E	0	1	ε
→*q'_0	Ø	Ø	$\{q'_0, q_0, q_3\}$
q_0	$\{q_0, q_1\}$	$\{q_0\}$	$\{q_0, q_3\}$
q_1	Ø	$\{q_2\}$	$\{q_1\}$
*q_2	Ø	Ø	$\{q_2\}$
q_3	Ø	$\{q_2\}$	$\{q_3\}$

Find ECLOSE for all states



Epsilon-NFA to DFA

Formal definition

- $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$ be a ε-NFA
- We define DFA, $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$
  - $Q_D = 2^{QE}$
  - $q_D = ECLOSE(q_0)$
  - F_D = sets containing at least one state from F_E

Epsilon-NFA to DFA

Computing δ_D

- $\delta_D(S, a)$ for $S \in Q_D$, $a \in \Sigma$
  - Let $S = \{ p_1, p_2, \ldots, p_n \}$
  - Compute the set of all states reachable from states in S on input a using transitions from E.

$$\{r_1, r_2, \cdots, r_m\} = \bigcup_{i=1}^{n} \delta_E(p_i, a)$$

  - $\delta_D(S, a)$ will be the union of the ε closures of the elements of $\{r_1, \ldots, r_m\}$

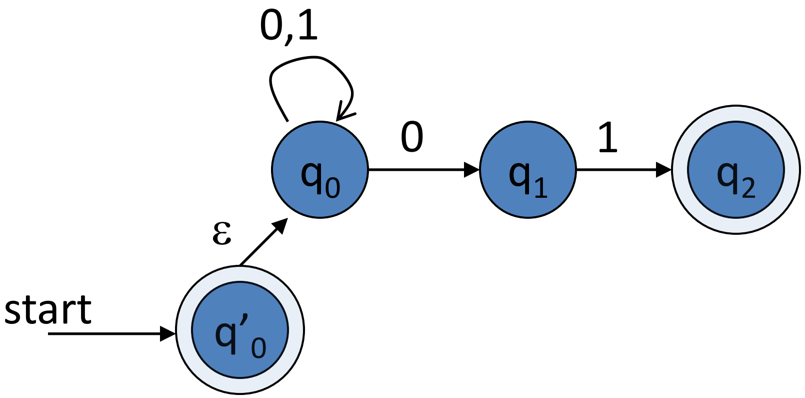
$$\delta_D(S, a) = \bigcup_{j=1}^{m} ECLOSE(r_j)$$

Epsilon-NFA to DFA

- **Step 1:** We will take the ε-closure for the starting state of NFA as a starting state of DFA.
- **Step 2:** Find the states for each input symbol that can be traversed from the present. That means the union of transition value and their closures for each state of NFA present in the current state of DFA.
- **Step 3:** If we found a new state, take it as current state and repeat step 2.
- **Step 4:** Repeat Step 2 and Step 3 until there is no new state present in the transition table of DFA.
- **Step 5:** Mark the states of DFA as a final state which contains the final state of NFA.

Example: ε-NFA → DFA

L = {w | w is empty, or if non-empty will end in 01}

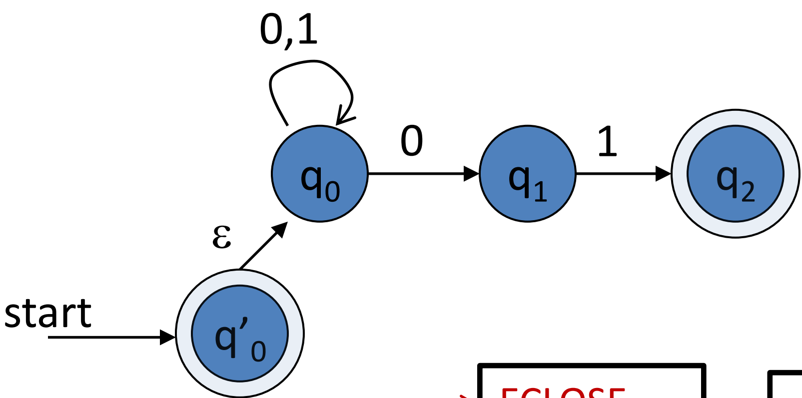


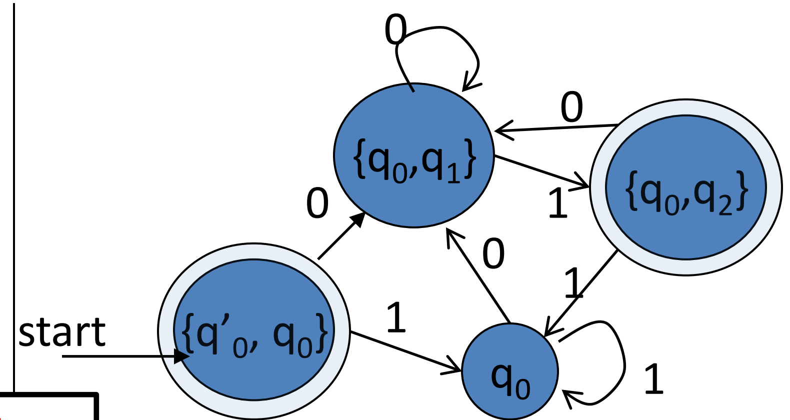
δ_E	0	1	ε
→ $*q'_0$	Ø	Ø	$\{q'_0, q_0\}$
q_0	$\{q_0, q_1\}$	$\{q_0\}$	$\{q_0\}$
q_1	Ø	$\{q_2\}$	$\{q_1\}$
$*q_2$	Ø	Ø	$\{q_2\}$

δ_D	0	1
→ $*\{q'_0, q_0\}$		
…		

Example: ε-NFA ➔ DFA

L = {w | w is empty, or if non-empty will end in 01}





ECLOSE

union

δ_E	0	1	ε
→ $*q'_0$	Ø	Ø	$\{q'_0,q_0\}$
q_0	$\{q_0,q_1\}$	$\{q_0\}$	$\{q_0\}$
q_1	Ø	$\{q_2\}$	$\{q_1\}$
$*q_2$	Ø	Ø	$\{q_2\}$

δ_D	0	1
→ $*\{q'_0,q_0\}$	$\{q_0,q_1\}$	$\{q_0\}$
$\{q_0,q_1\}$	$\{q_0,q_1\}$	$\{q_0,q_2\}$
$\{q_0\}$	$\{q_0,q_1\}$	$\{q_0\}$
$*\{q_0,q_2\}$	$\{q_0,q_1\}$	$\{q_0\}$

Example

-

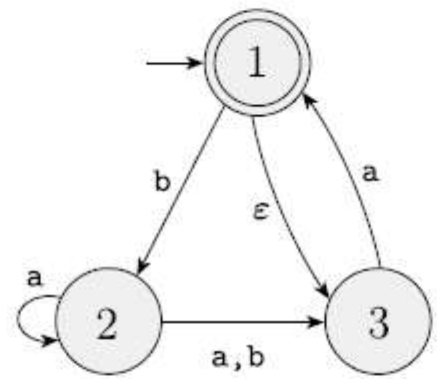
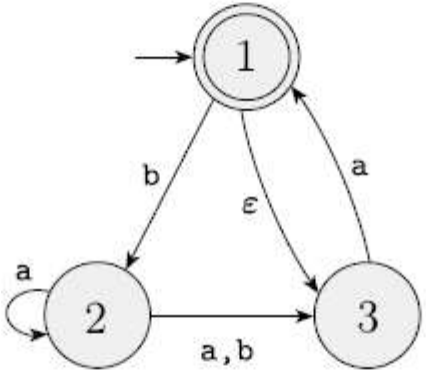
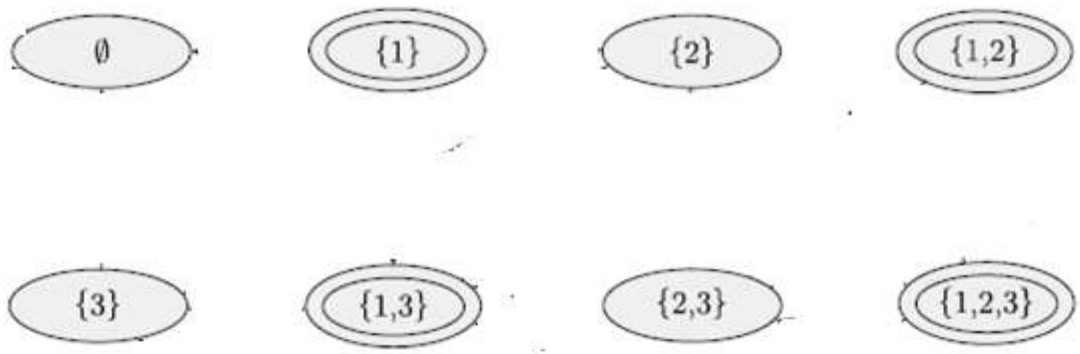


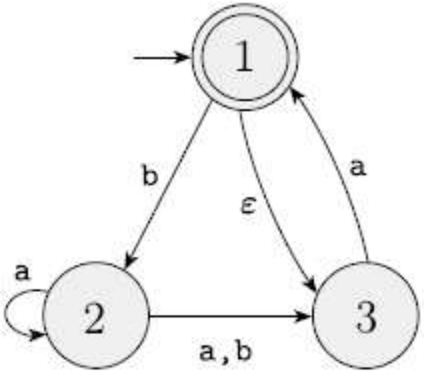
FIGURE 1.42
The NFA N_4

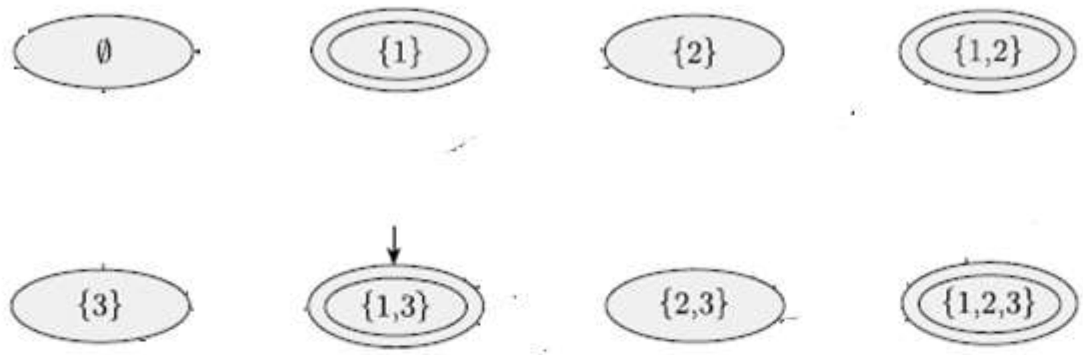




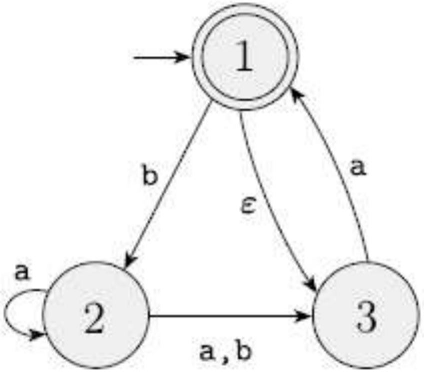
All possible states of the DFA.
(to be constructed; Final states are shown)

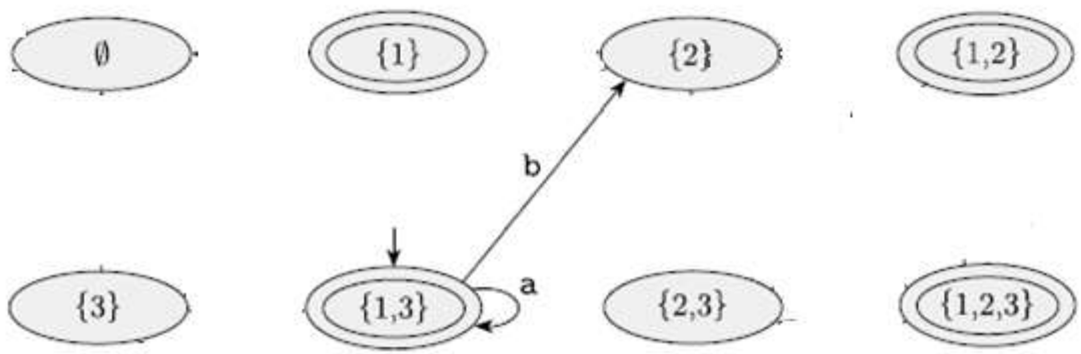
- Now we need to add edges, and
- identify the initial state.



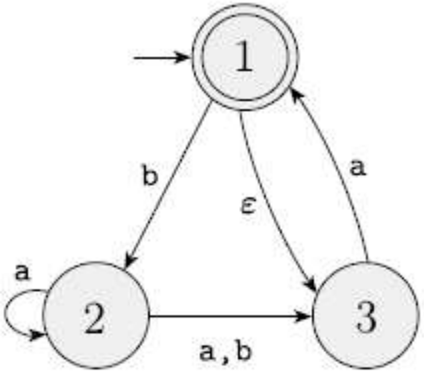


- Identify the initial state.
  - Note, it is not {1}





- Adding edges, …



After all edges …

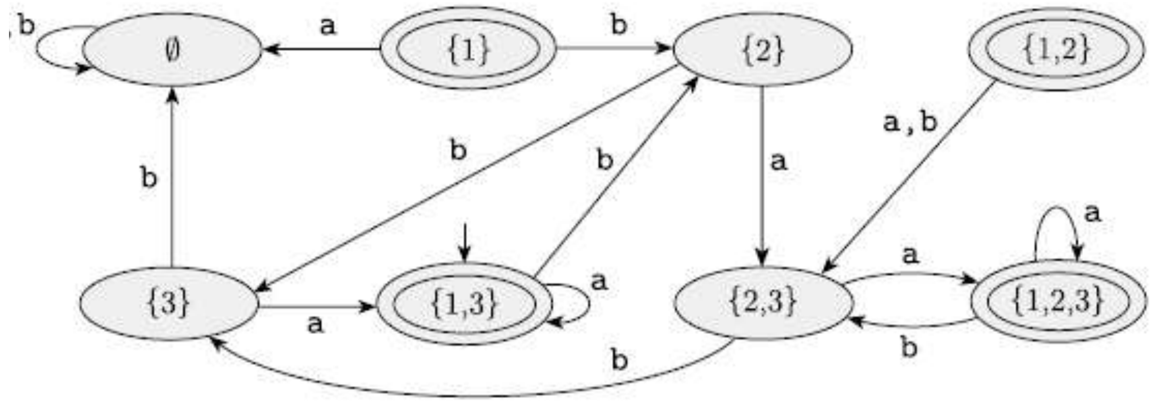


FIGURE 1.43
A DFA D that is equivalent to the NFA N_4

- But, some states are not reachable !
- Simplification can remove this.

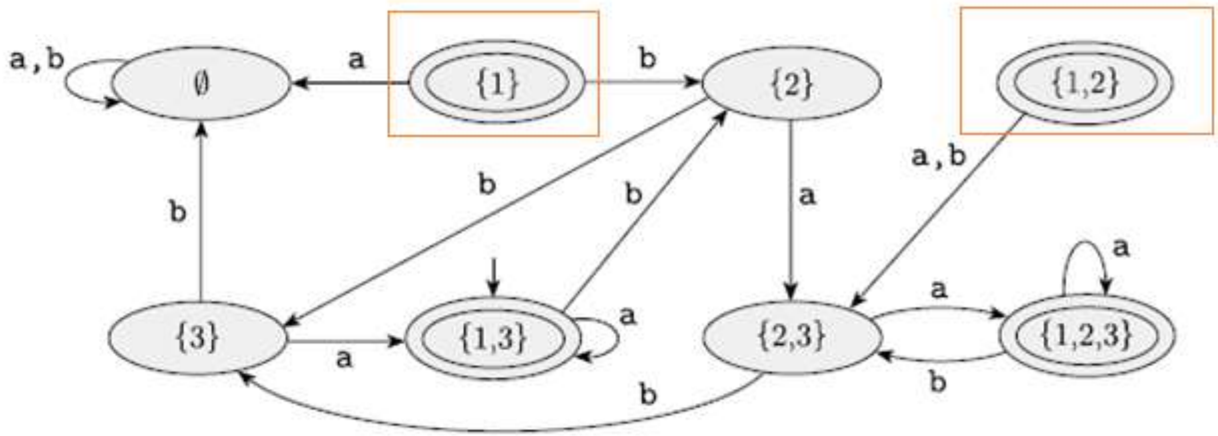
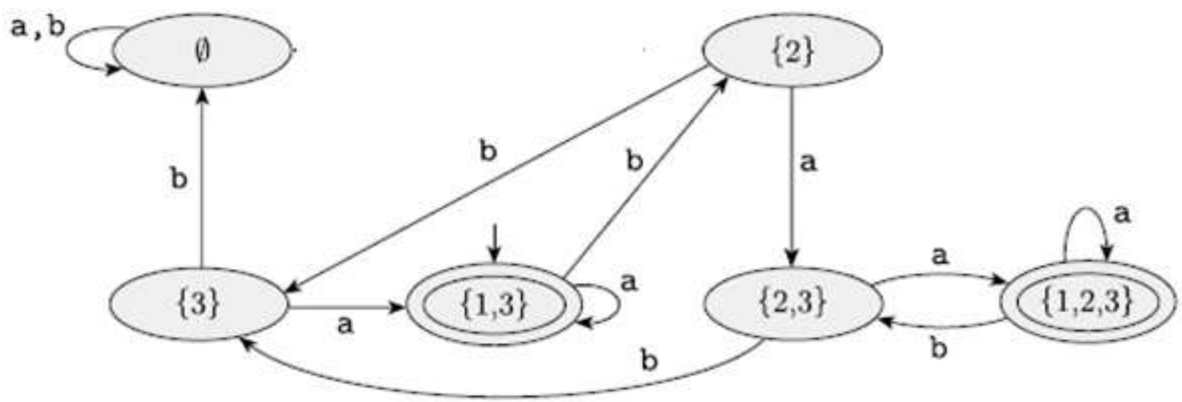
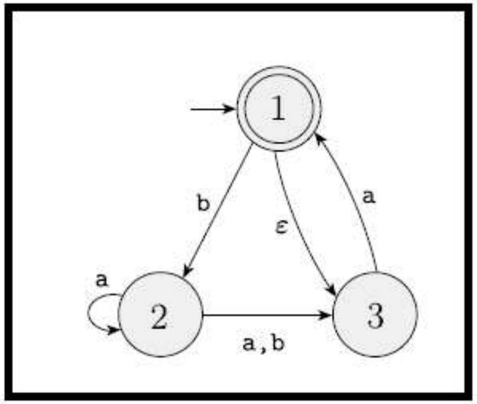


FIGURE 1.43
A DFA D that is equivalent to the NFA N_4

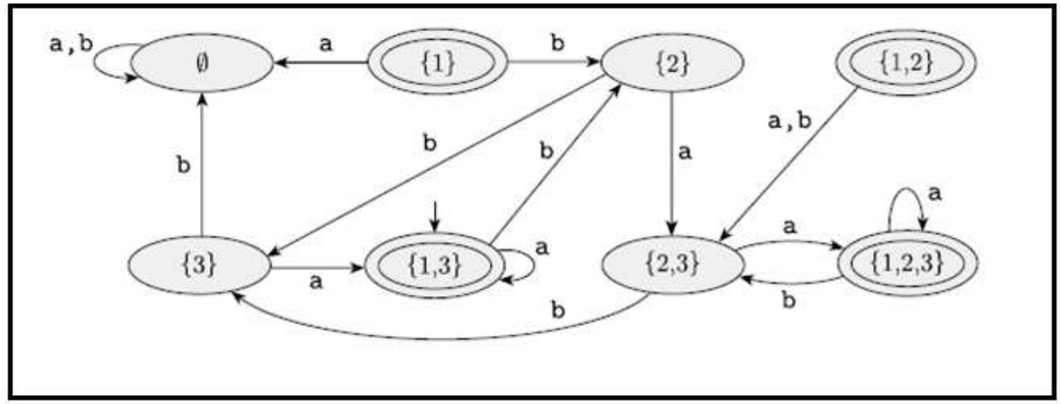


DFA D' which is equivalent to D.

Note: D' and D are different machines; but, they are equivalent.



N_4

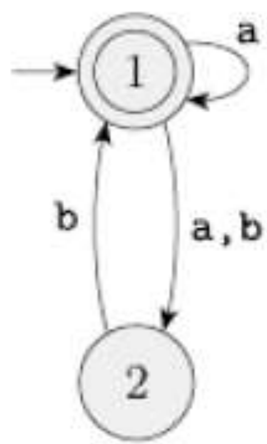


A DFA D that is equivalent to the NFA N_4

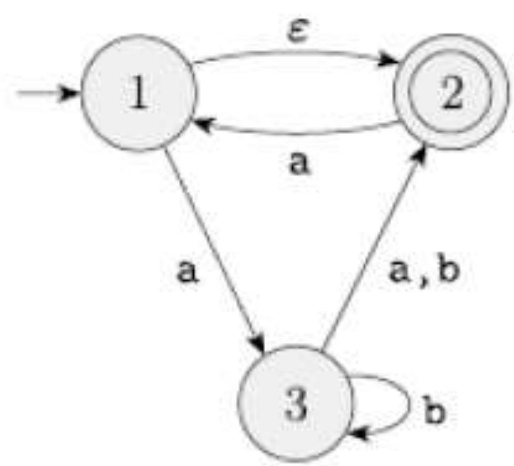
- Being in state 1 of N_4 upon reading input a the machine N_4 can be in state 1.
- Convince yourself that in the DFA D there are no mistakes.
  - From state $\{1\}$ with input a the DFA D goes to state ϕ.

Exercise

Convert the following NFAs to equivalent DFAs.



(a)



(b)

(Problem Source: Sipser's book exercise problem 1.16)

Exercise

- 1.7 Give state diagrams of NFAs with the specified number of states recognizing each of the following languages. In all parts, the alphabet is $\{0,1\}$.

  [A]a. The language $\{w|\ w \text{ ends with } 00\}$ with three states

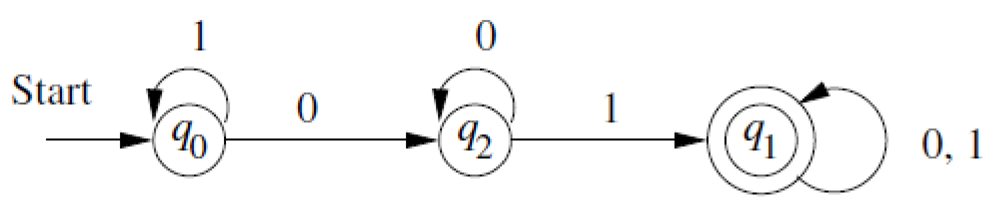
  d. The language $\{0\}$ with two states

  g. The language $\{\varepsilon\}$ with one state

  h. The language 0^* with one state

- Can you convert each of above NFAs into a corresponding DFA.

Some Notation adapted



The transition diagram for the DFA accepting all strings with a substring 01

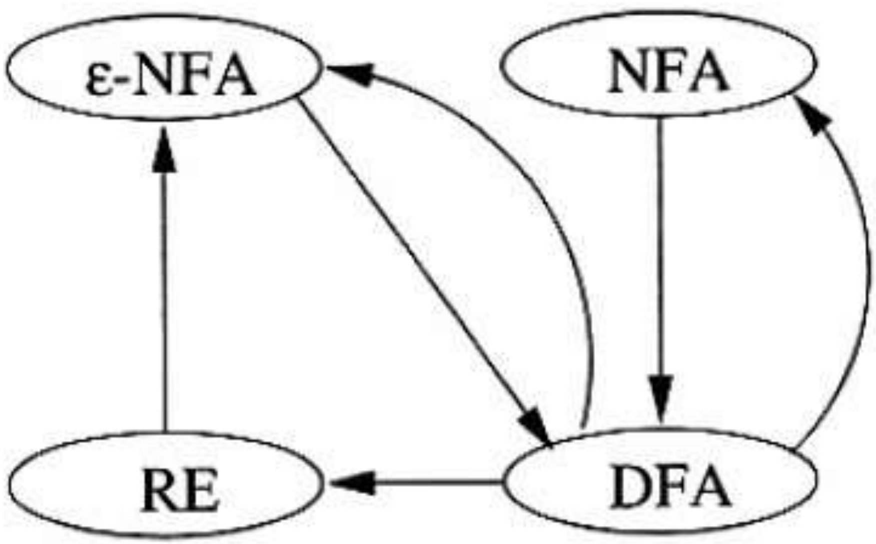
	0	1
$\rightarrow q_0$	q_2	q_0
$*q_1$	q_1	q_1
q_2	q_2	q_1

This also has all 5 components. This table is complete description of the DFA as the diagram.

Equivalency of DFA, NFA, ε-NFA

What have we shown

- For every DFA, there is an NFA that accepts the same language and visa versa
- For every DFA, there is a ε-NFA that accepts the same language, and visa versa
- Thus, for every NFA there is a ε-NFA that accepts the same language, and visa versa
- DFAs, NFAs, and ε-NFA s are equivalent!


ε-NFA
NFA
RE
DFA