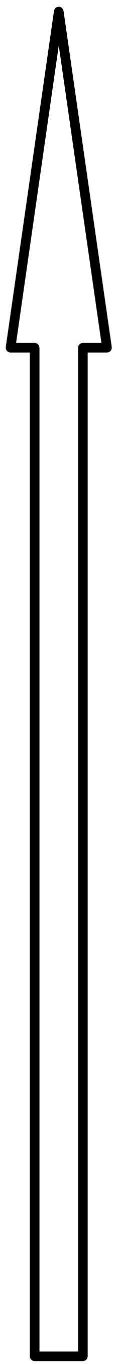
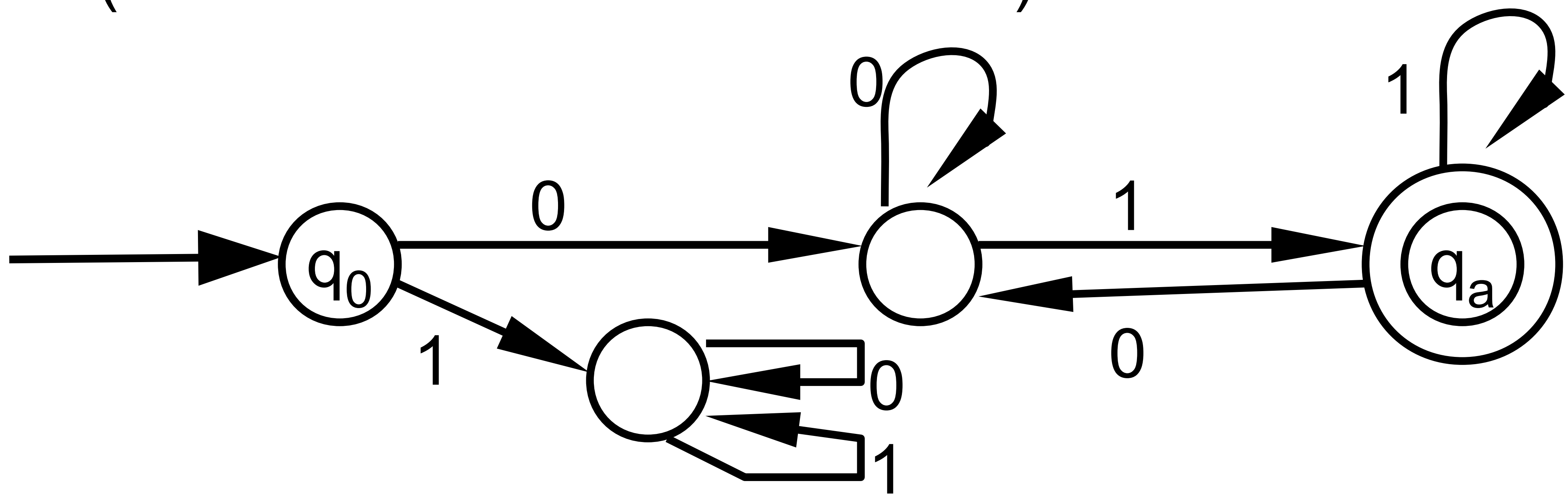
Big picture

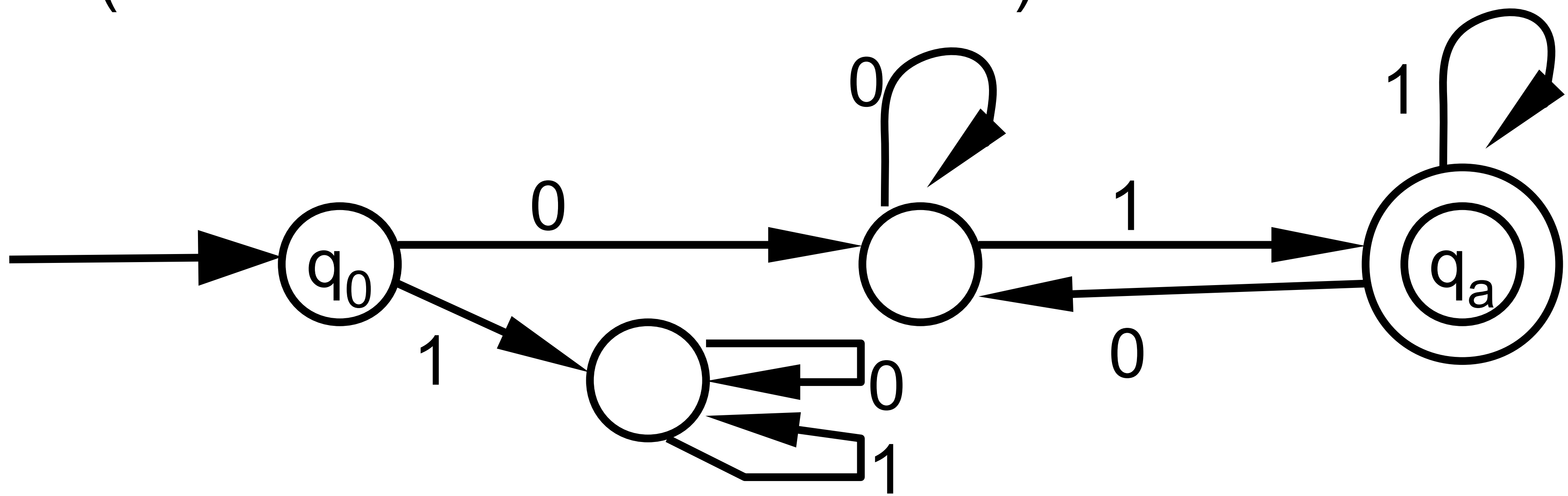


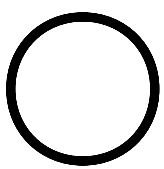
- All languages
- Decidable
  - Turing machines
- NP
- P
- Context-free
  - Context-free grammars, push-down automata
- Regular
  - Automata, non-deterministic automata, regular expressions

DFA (Deterministic Finite Automata)



DFA (Deterministic Finite Automata)

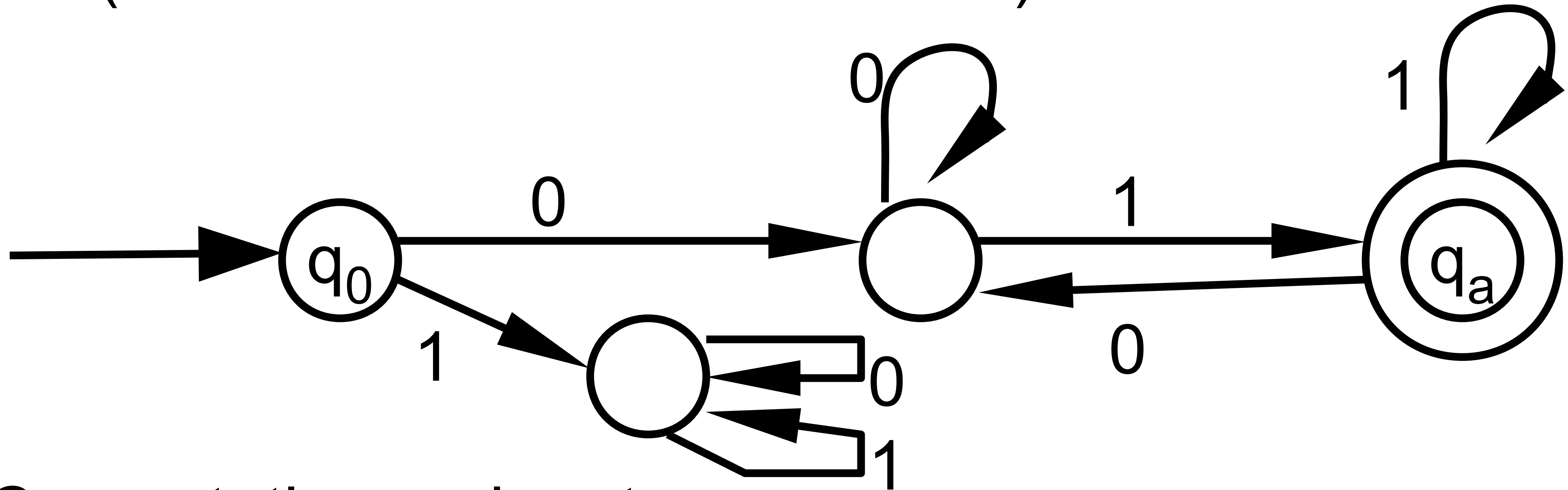


- States  , this DFA has 4 states

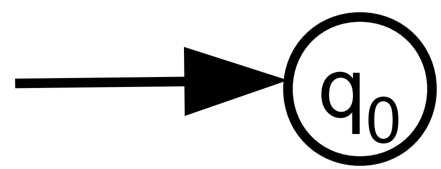
- Transitions 

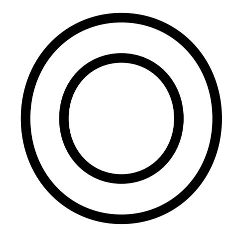
  labelled with elements of the alphabet $\Sigma = \{0,1\}$

DFA (Deterministic Finite Automata)



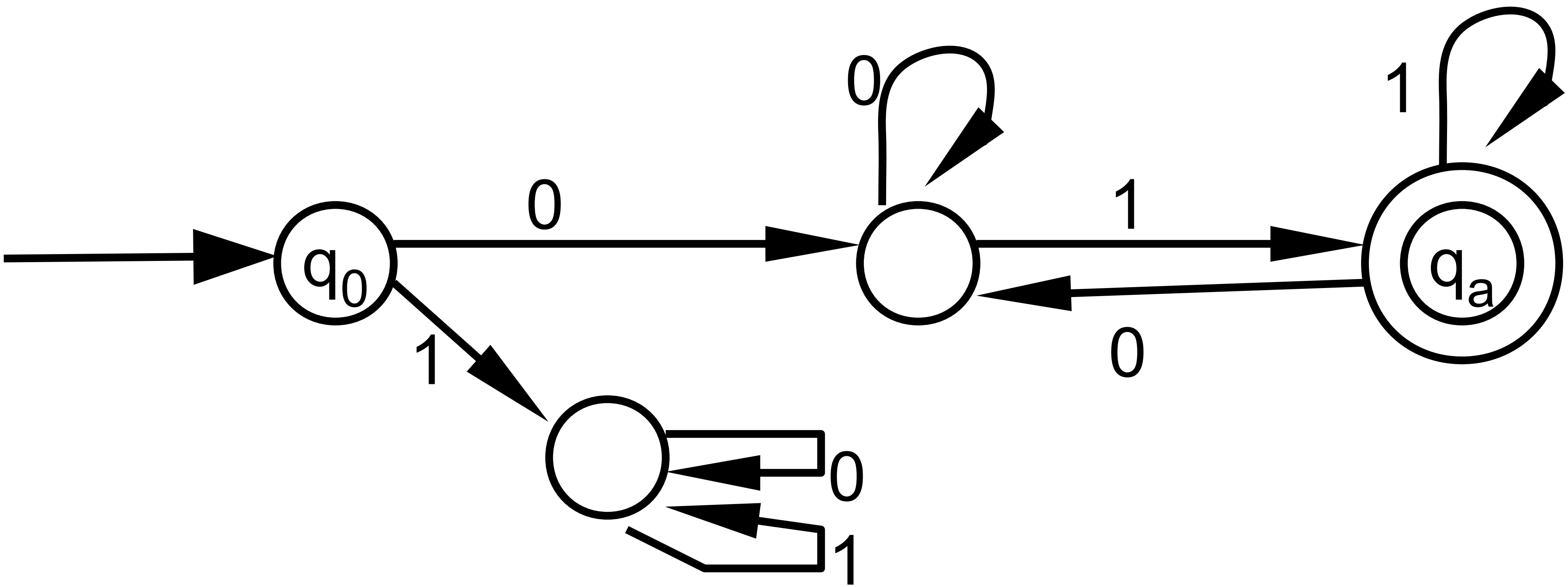
Computation on input w:

- Begin in start state 

- Read input string in a one-way fashion
- Follow the arrows matching input symbols
- When input ends: ACCEPT if in accept state 

  REJECT if not

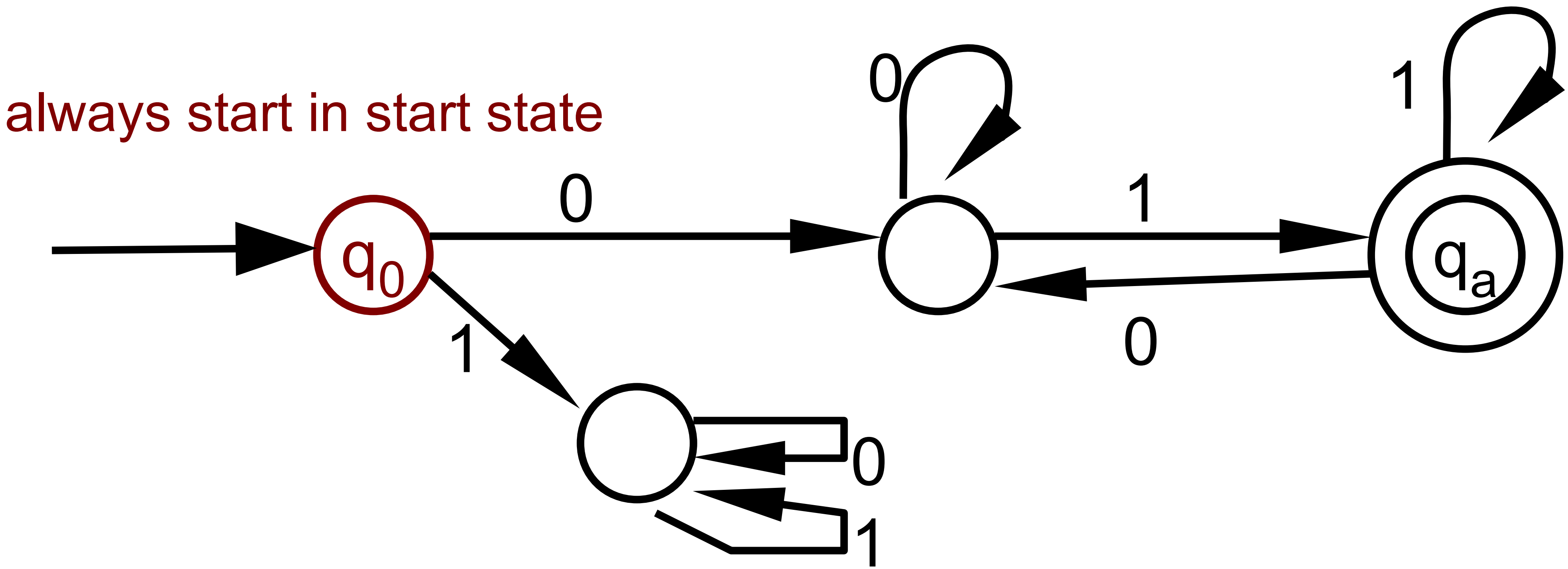
DFA (Deterministic Finite Automata)



Example: Input string

w = 0011

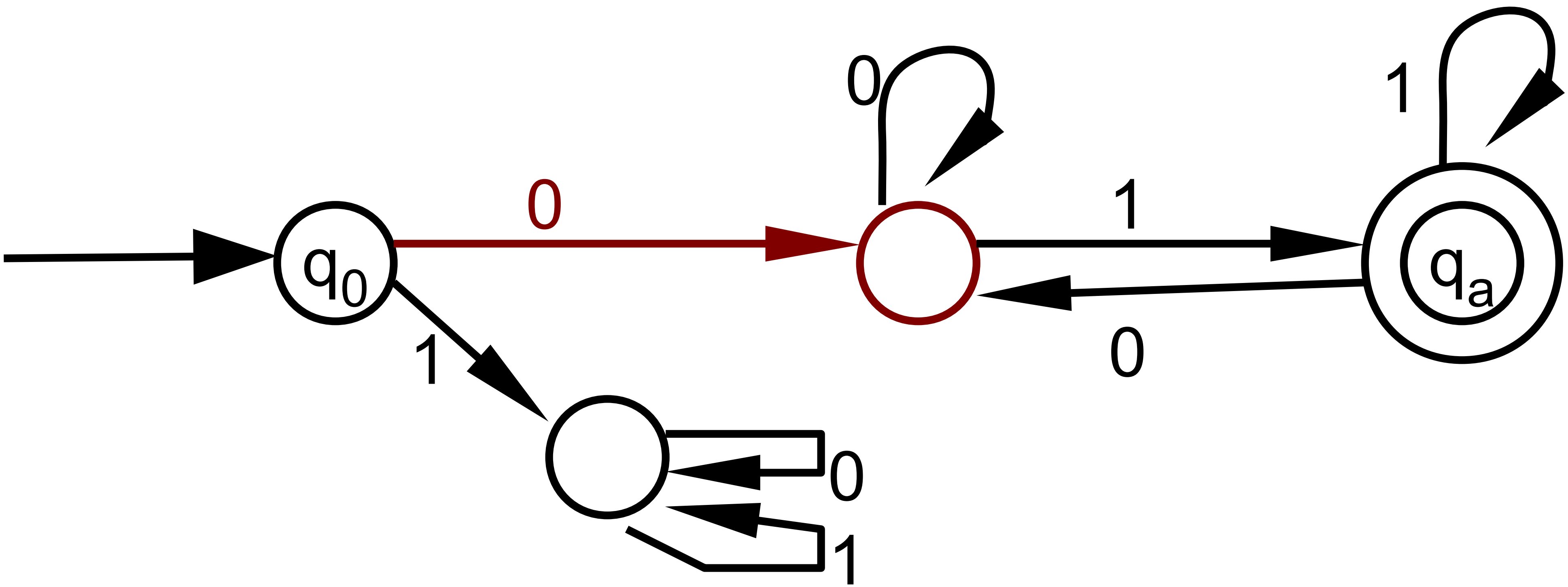
DFA (Deterministic Finite Automata)



Example: Input string

w = 0011

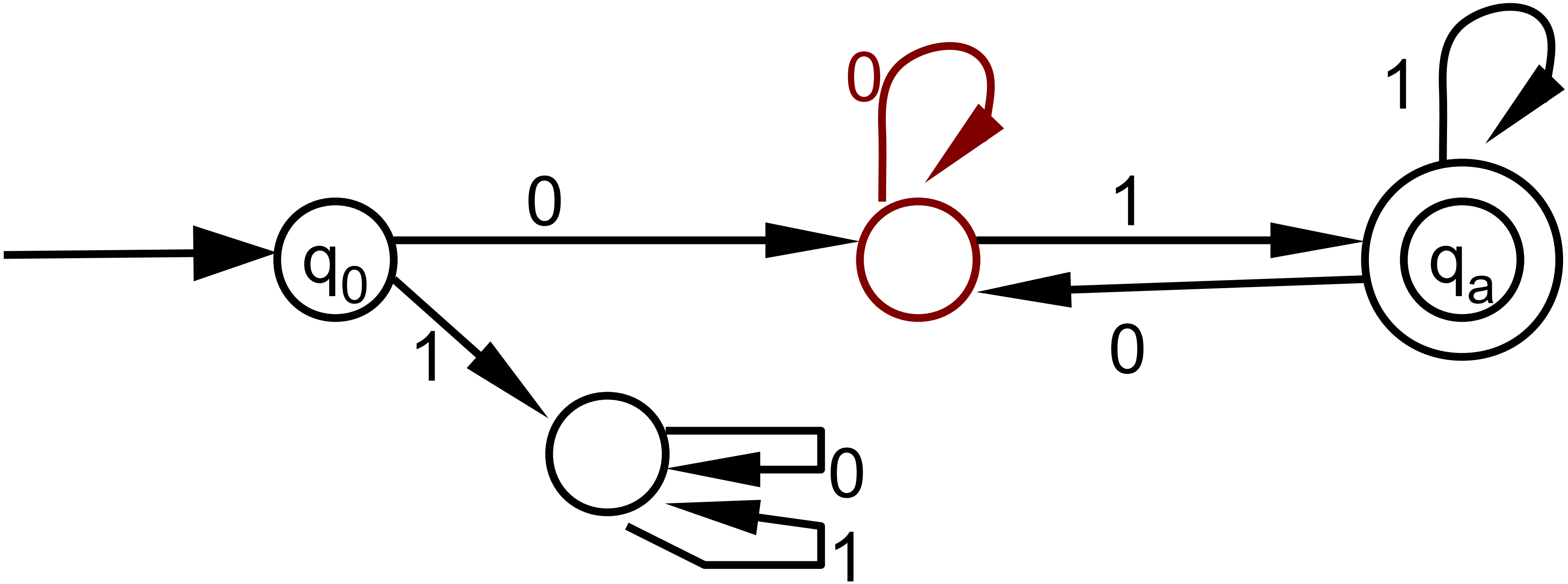
DFA (Deterministic Finite Automata)



Example: Input string

w = **0**011

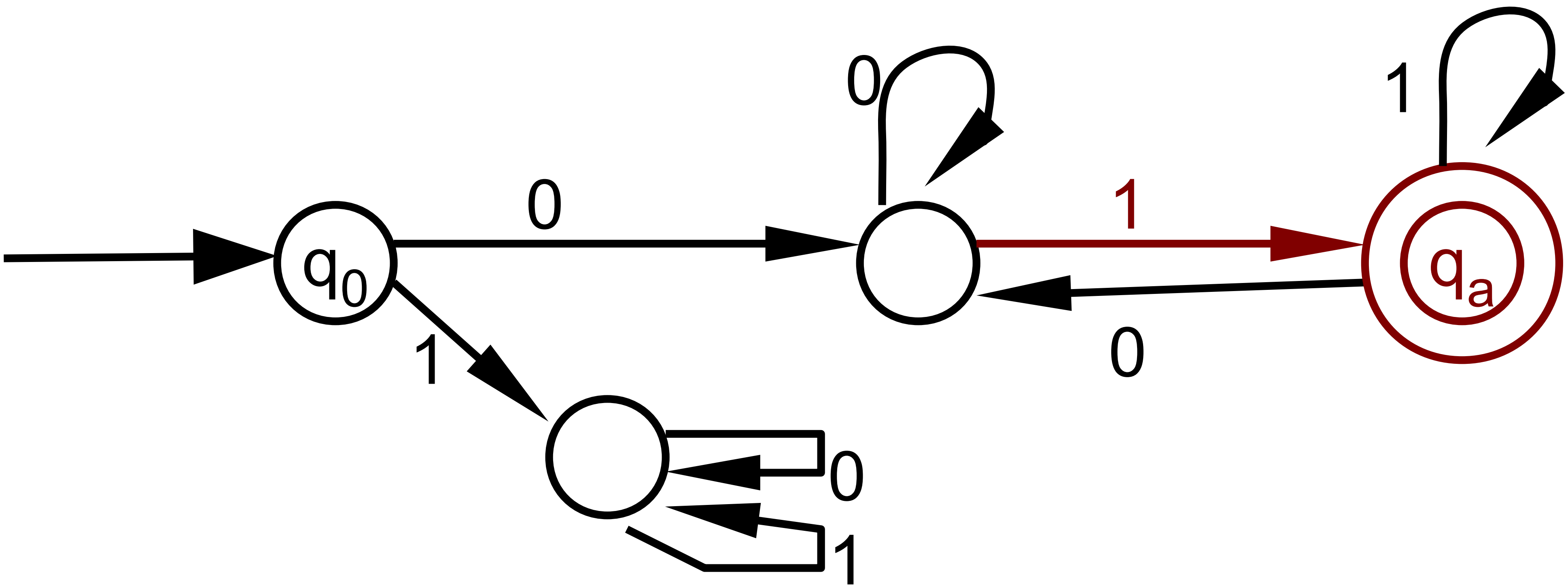
DFA (Deterministic Finite Automata)



Example: Input string

w = 0**0**11

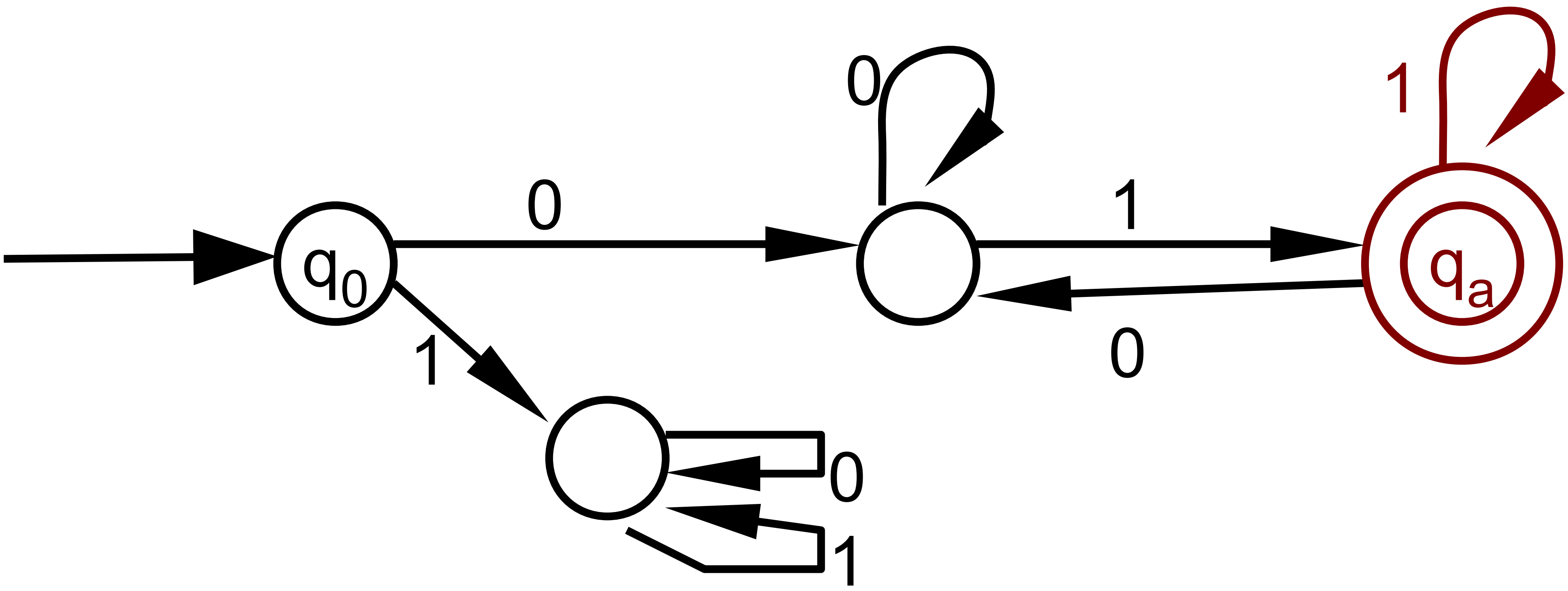
DFA (Deterministic Finite Automata)



Example: Input string

w = 00**1**1

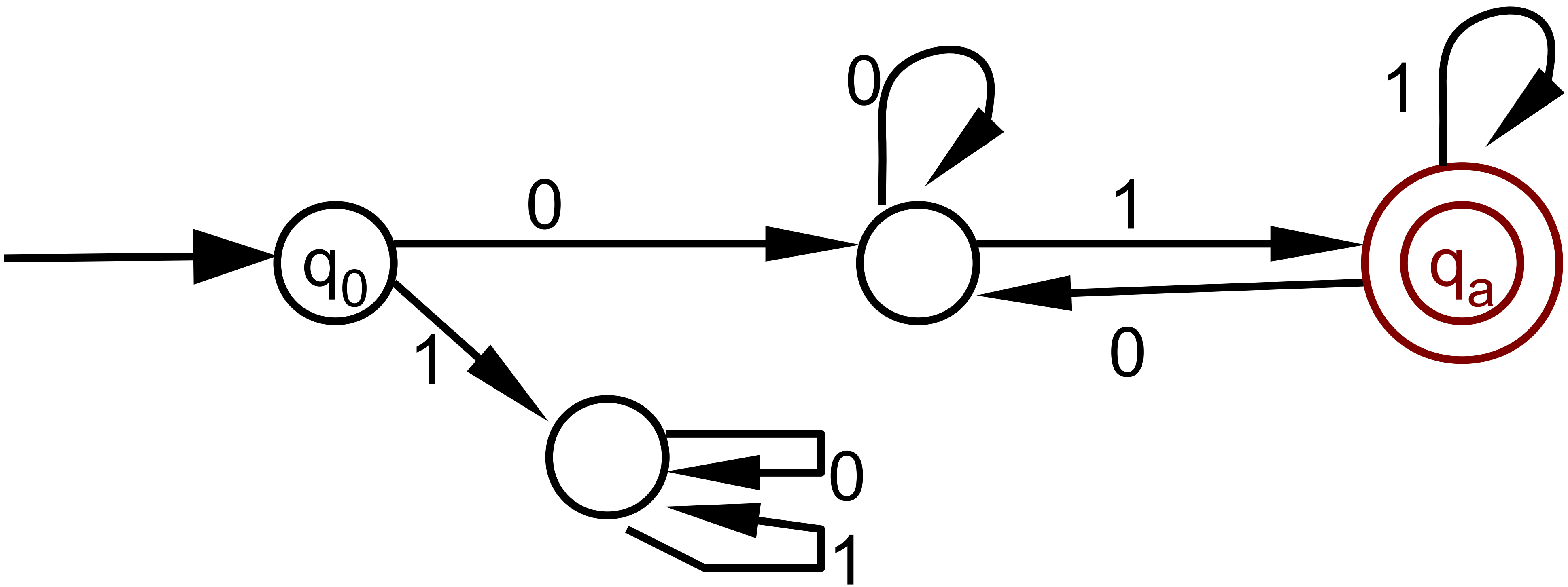
DFA (Deterministic Finite Automata)



Example: Input string

w = 0011

DFA (Deterministic Finite Automata)



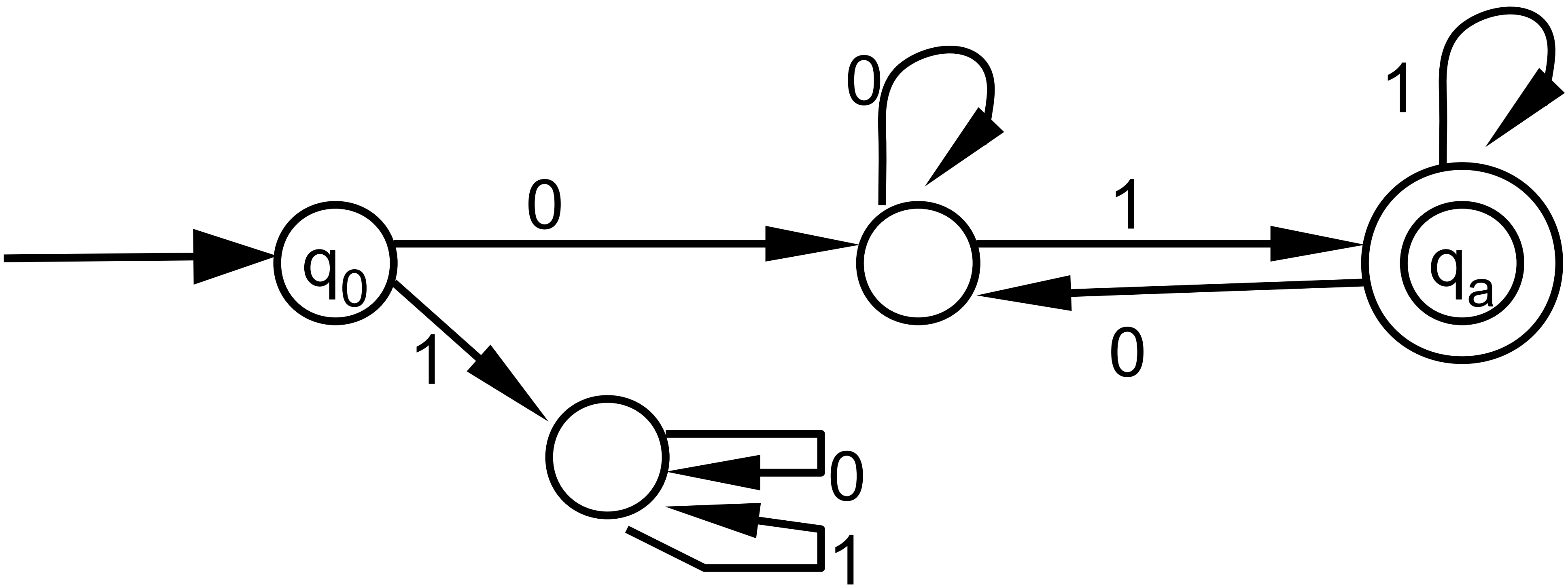
Example: Input string

w = 0011 **ACCEPT**

because end in

accept state

DFA (Deterministic Finite Automata)

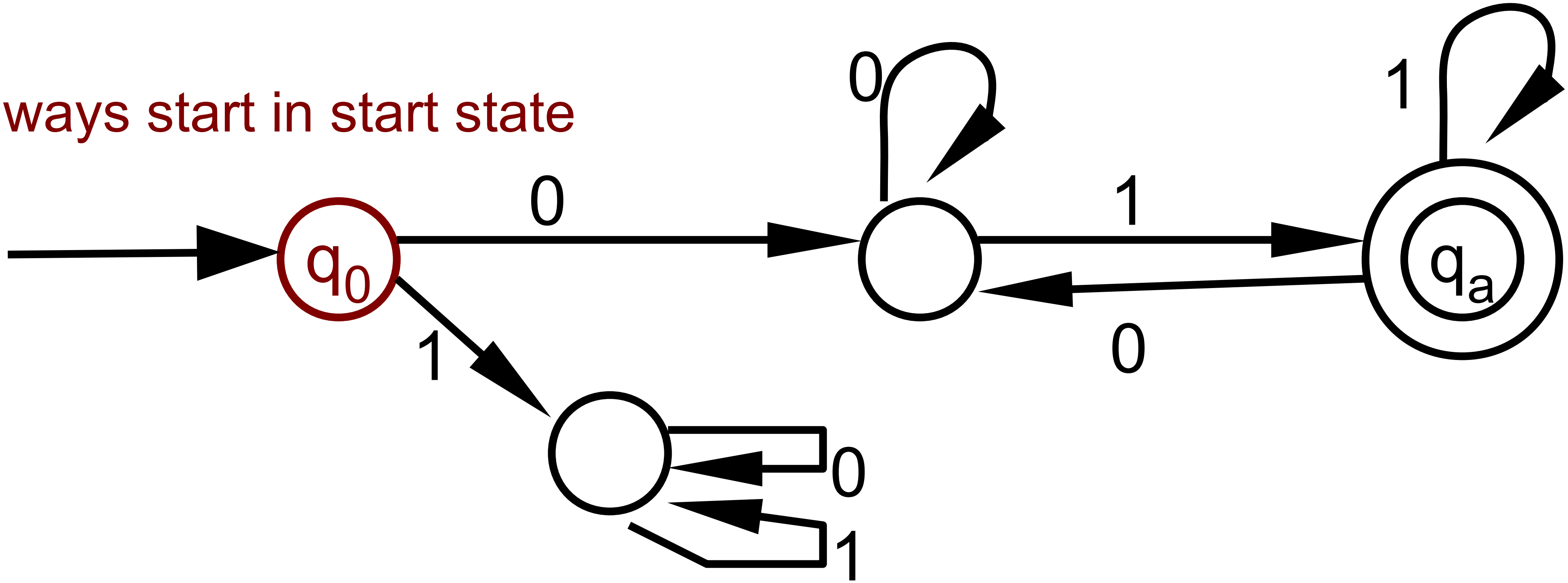


Example: Input string

w = 010

DFA (Deterministic Finite Automata)

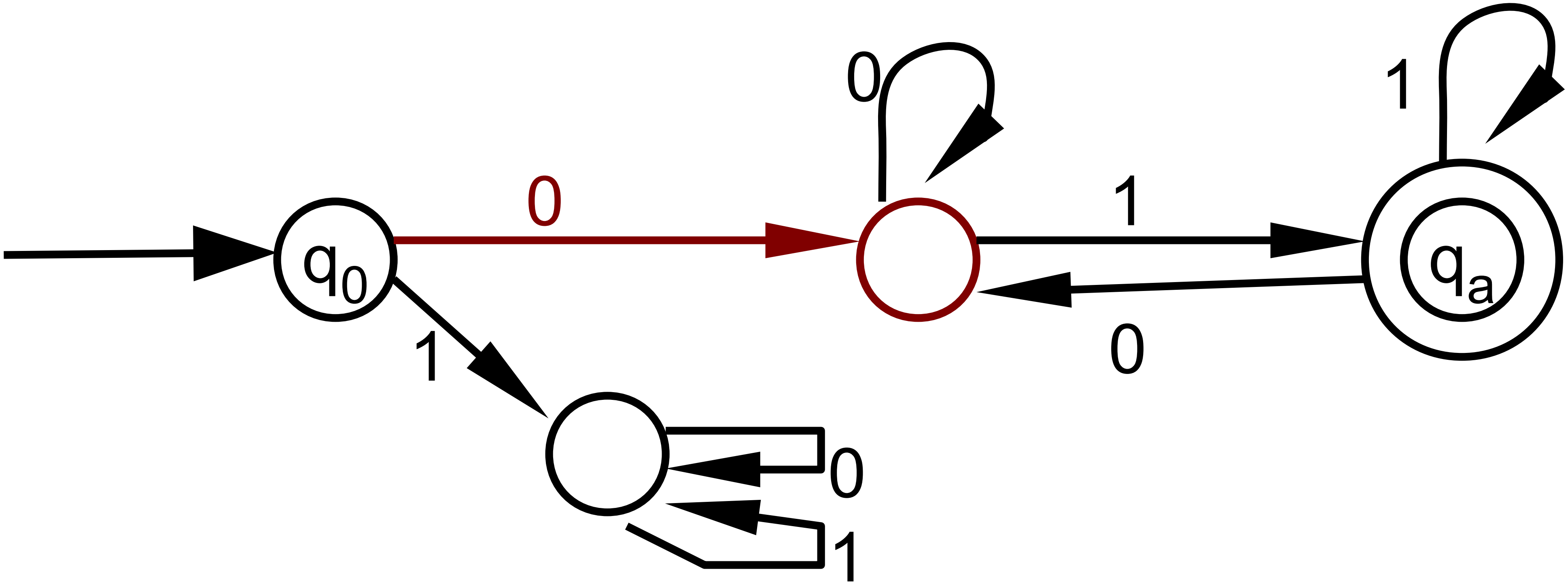
always start in start state



Example: Input string

w = 010

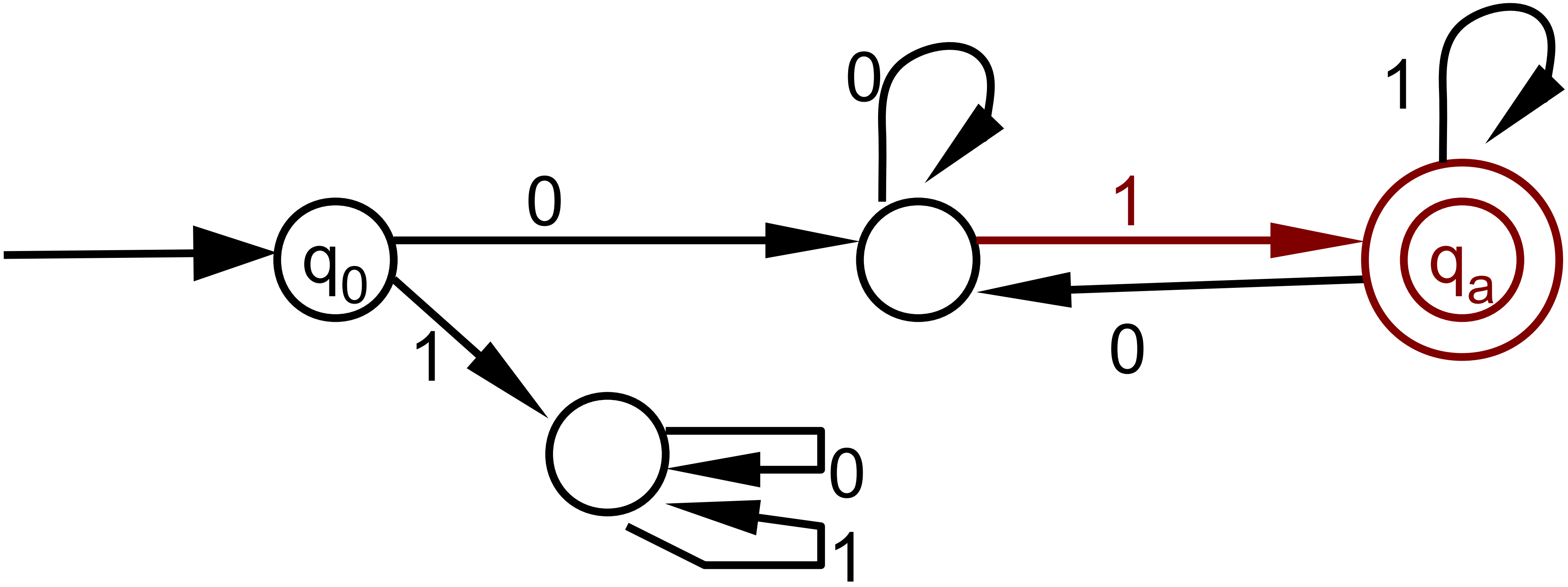
DFA (Deterministic Finite Automata)



Example: Input string

w = **0**10

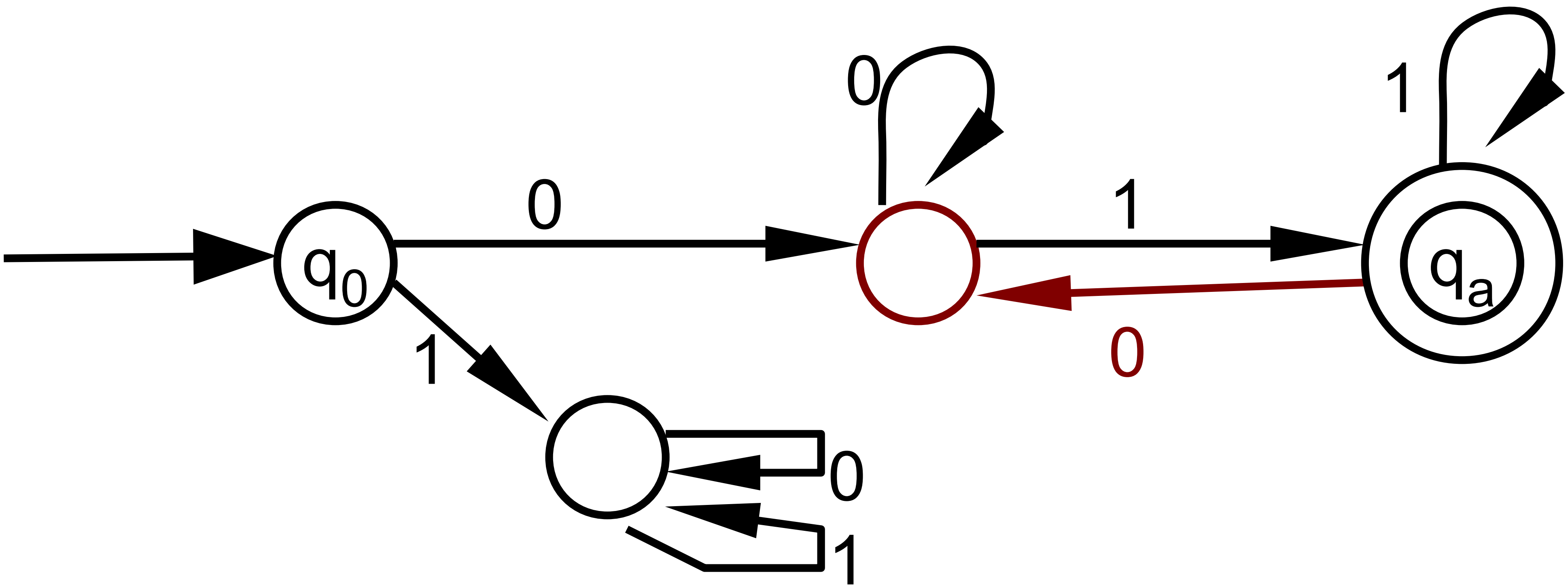
DFA (Deterministic Finite Automata)



Example: Input string

w = 01**0**

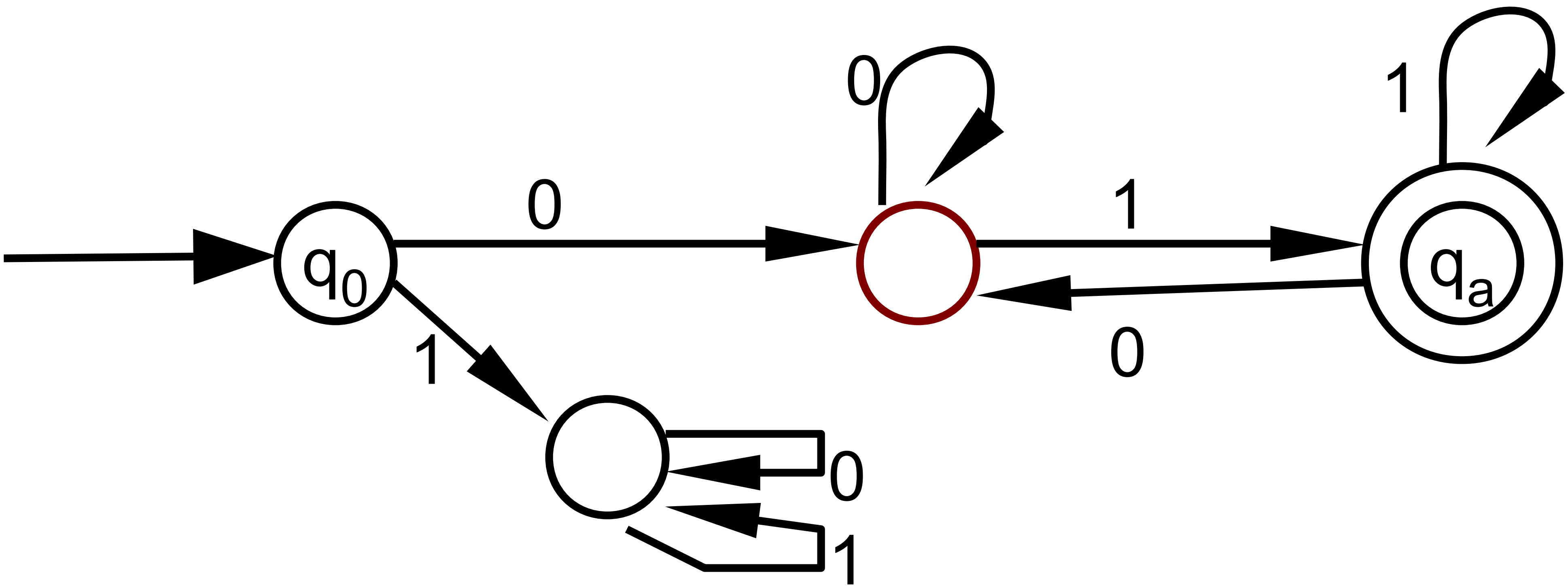
DFA (Deterministic Finite Automata)



Example: Input string

w = 01**0**

DFA (Deterministic Finite Automata)

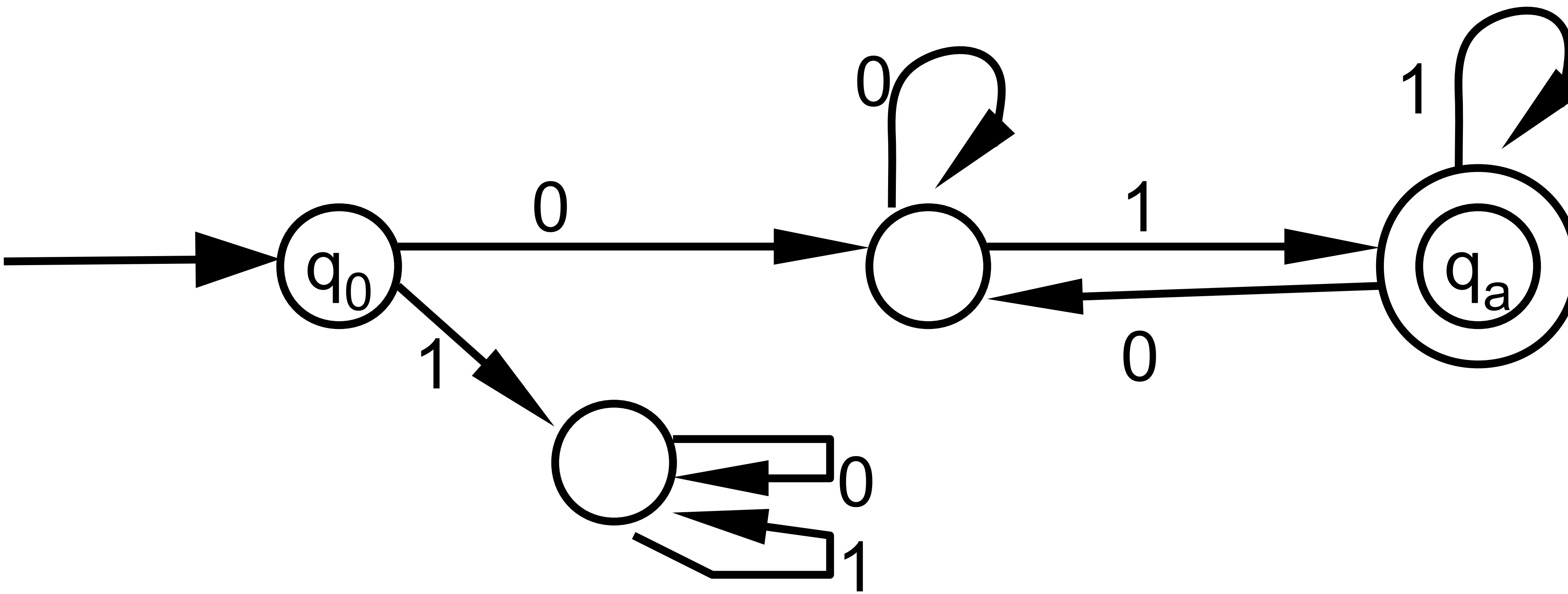


Example: Input string

w = 010 **REJECT**

because does not end in accept state

DFA (Deterministic Finite Automata)



Example: Input string w = 01 ACCEPT

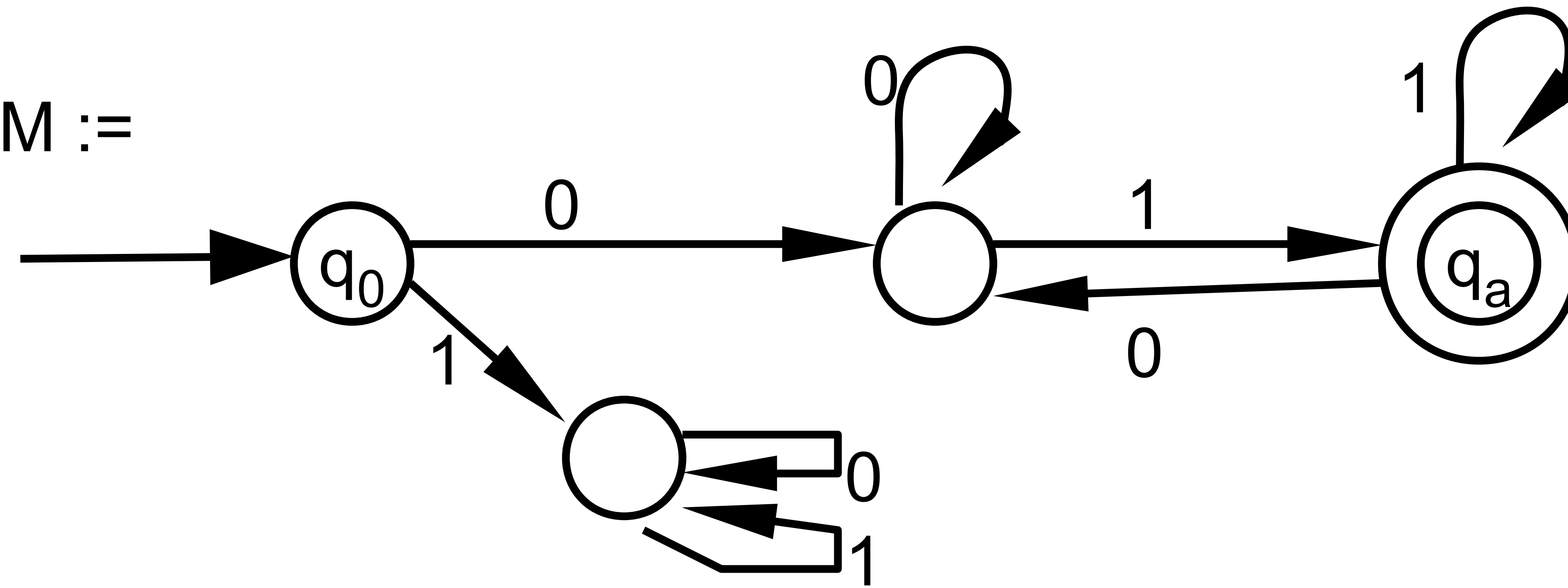
w = 010 REJECT

w = 0011 ACCEPT

w = 00110 REJECT

DFA (Deterministic Finite Automata)

M :=



M recognizes language

L(M) = { w : w starts with 0 and ends with 1 }

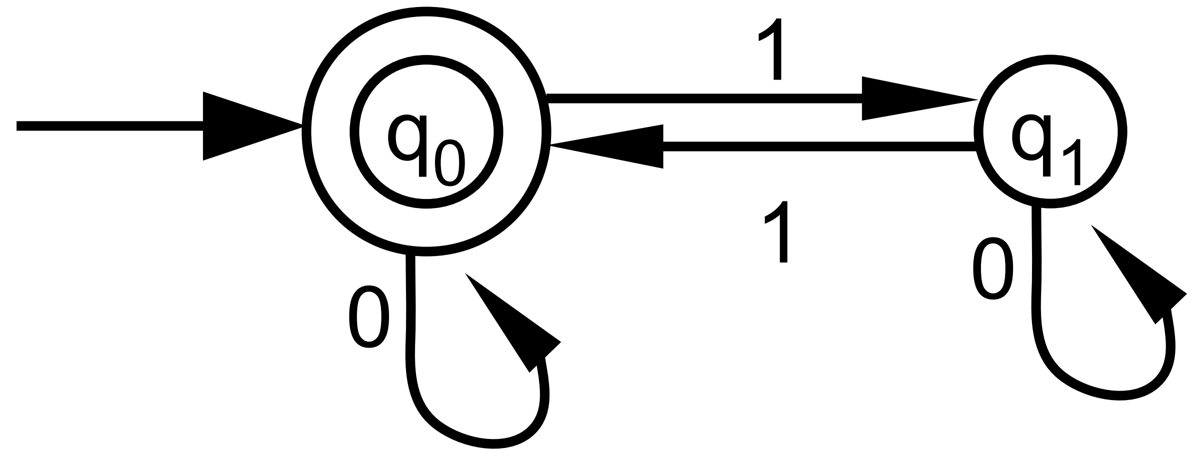
L(M) is the language of strings causing M to accept

Example: 0101 is an element of L(M), 0101 $\in$ L(M)

Example

$\Sigma = \{0,1\}$

M :=

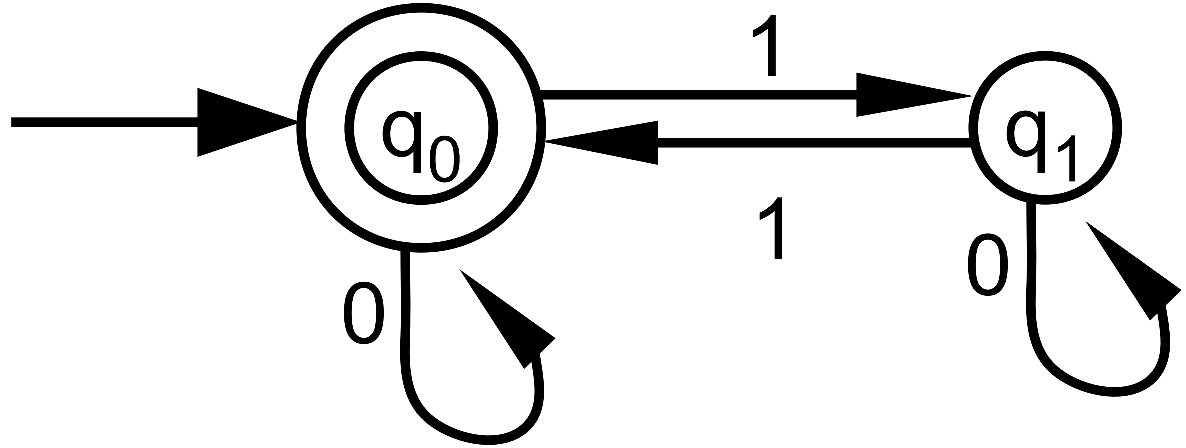


- 00 causes M to accept, so 00 is in L(M) $00 \in L(M)$
- 01 does not cause M to accept, so 01 not in L(M),

$01 \notin L(M)$

- 0101 $\in L(M)$
- 01101100 $\in L(M)$
- 011010 $\notin L(M)$

Example

M := 

$\Sigma = \{0,1\}$

$L(M) = \{w : w$ has an even number of 1 $\}$

Note: If there is no 1, then there are zero 1,

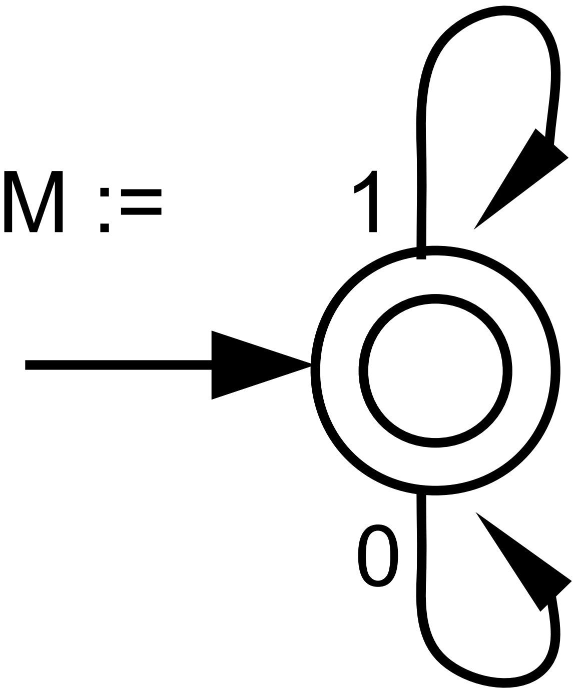
zero is an even number, so M should accept.

Indeed 0000000 $\in L(M)$

Example

M :=

$\Sigma = \{0,1\}$

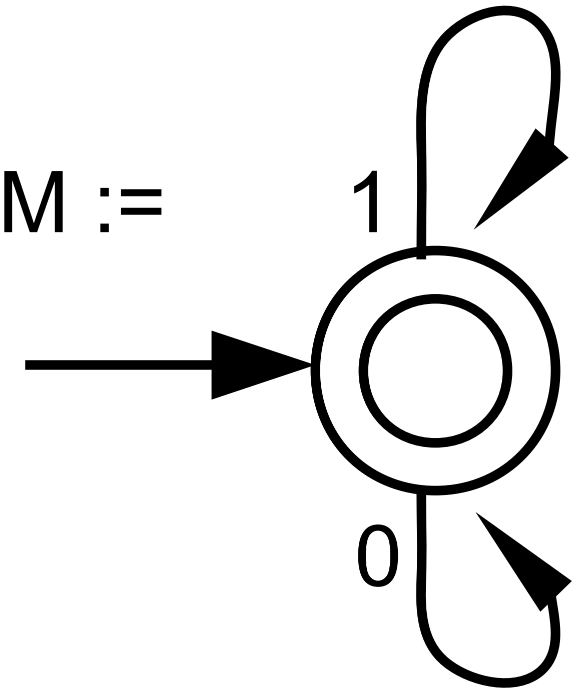


- L(M) = ?

Example

M :=

$\Sigma = \{0,1\}$



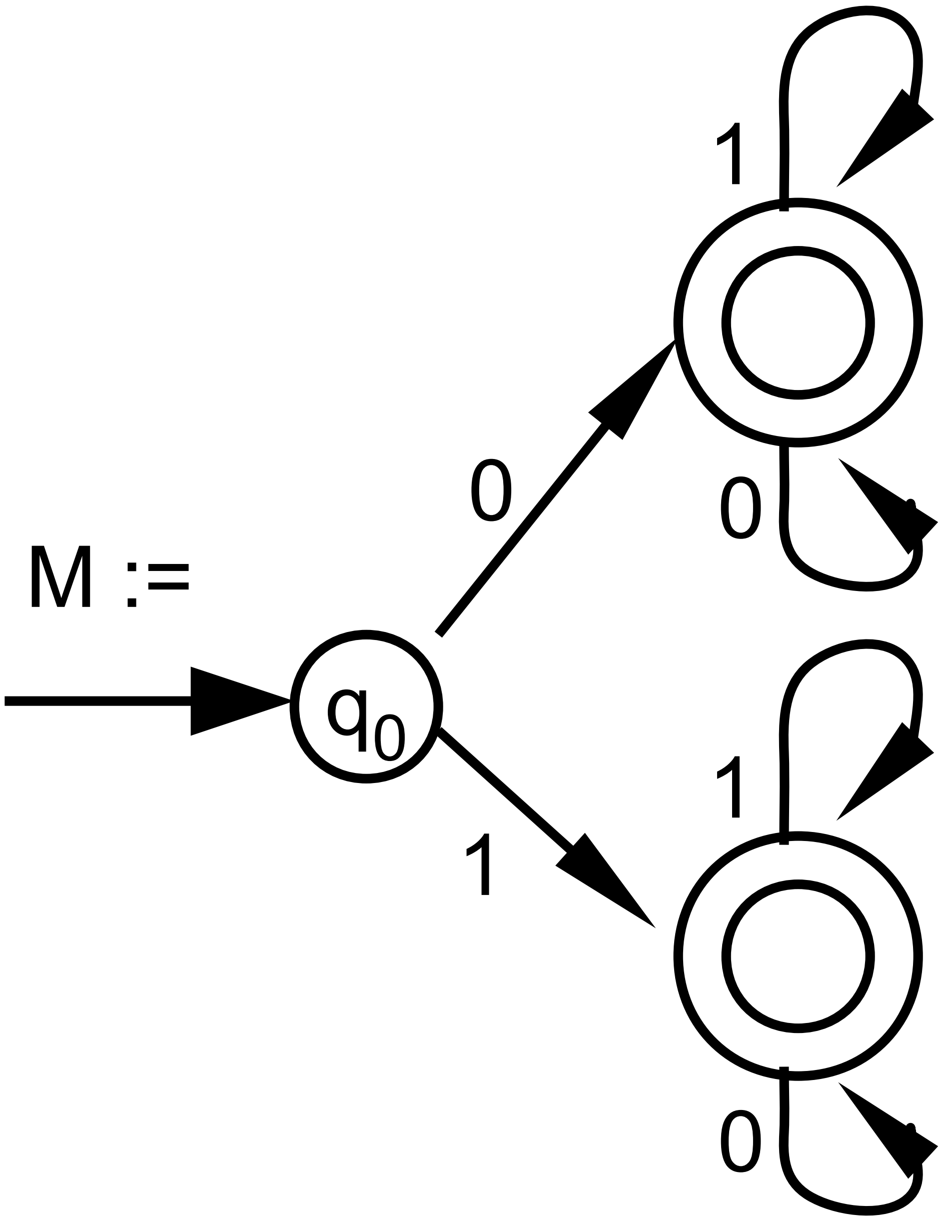
- L(M) = every possible string over {0,1}

  $= \{0,1\}^*$

Example

$\Sigma = \{0,1\}$

M :=

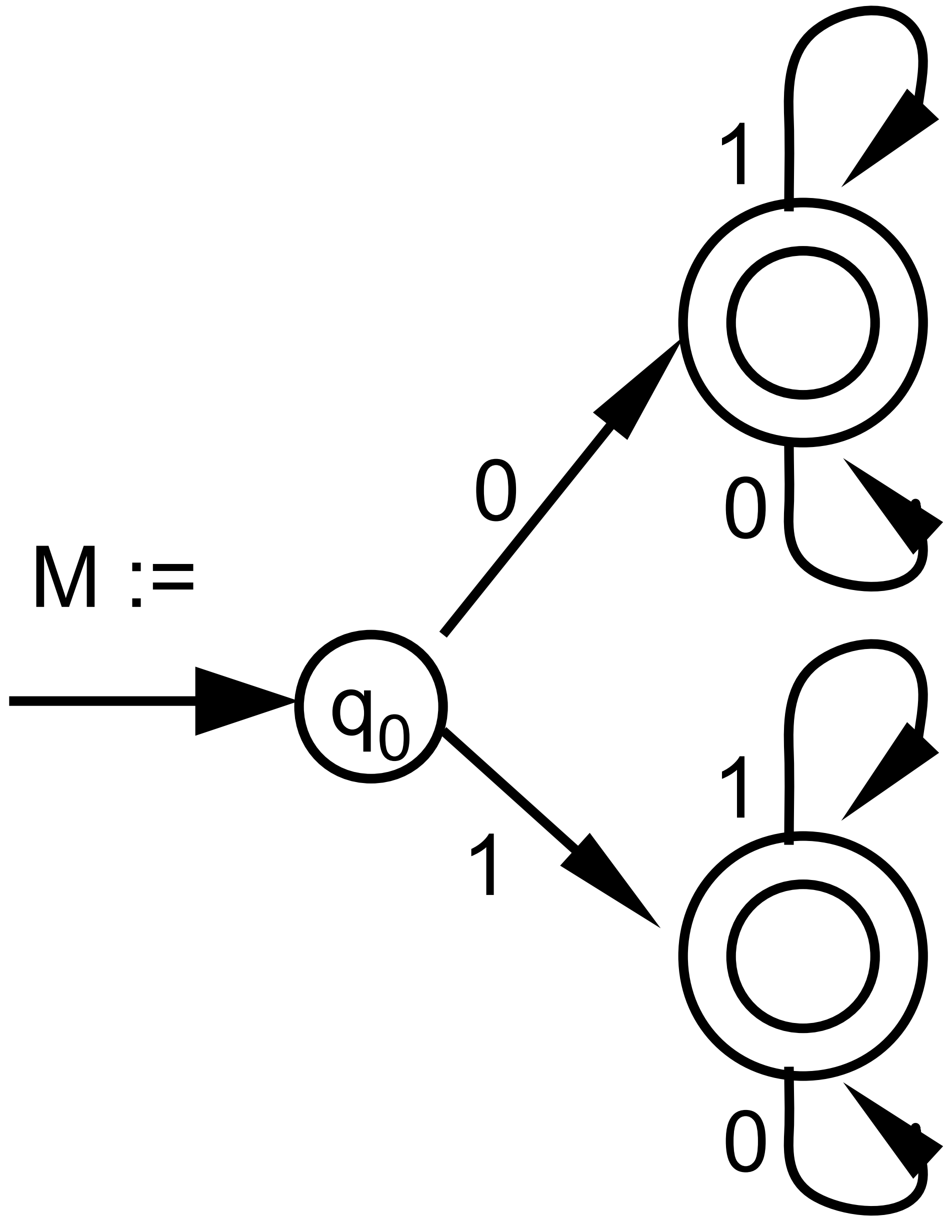


- L(M) = ?

Example

$\Sigma = \{0,1\}$

M :=

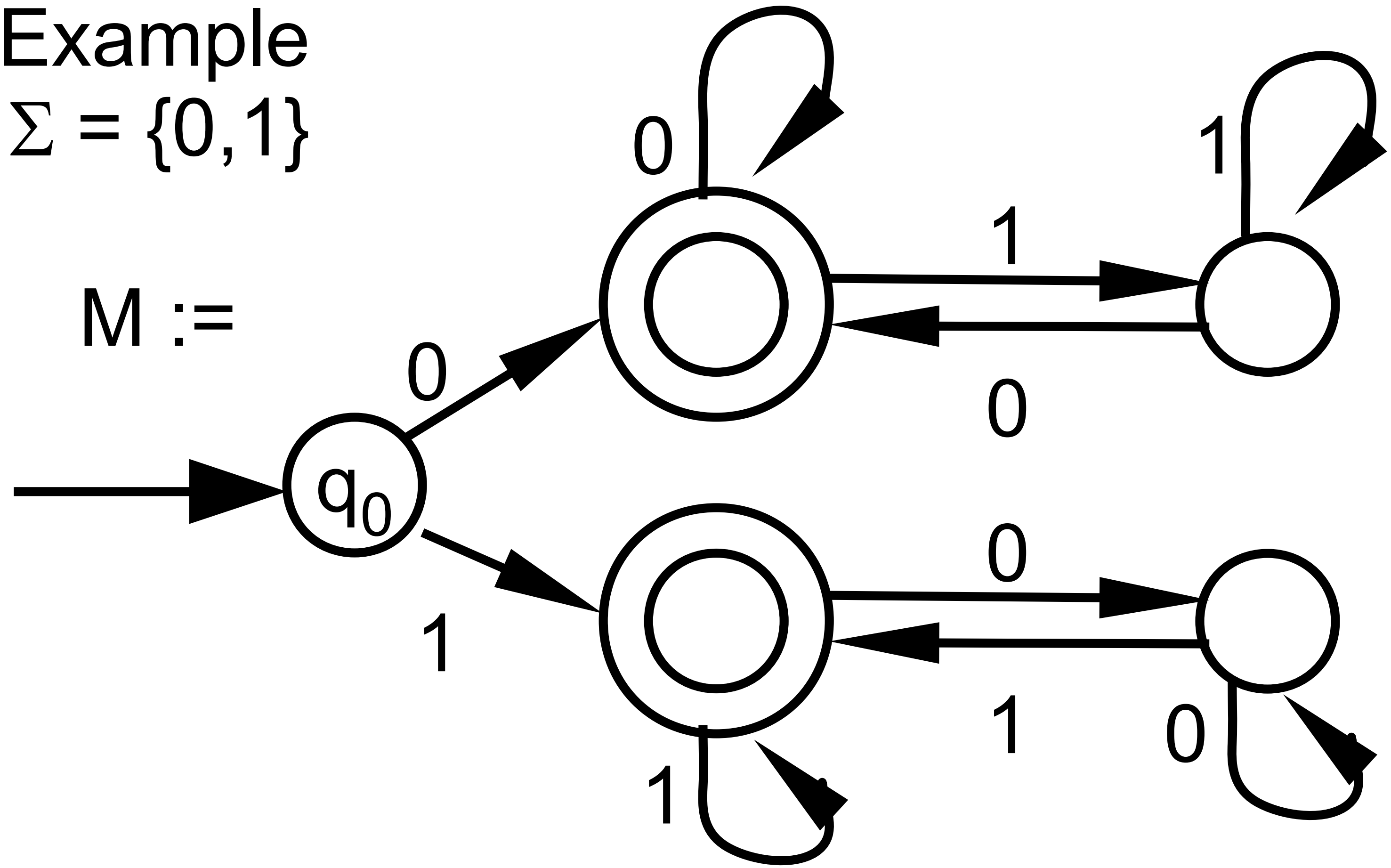


- L(M) = all strings over {0,1} except empty string ε

  $= \{0,1\}^* - \{ \varepsilon \}$

Example
$\Sigma = \{0,1\}$

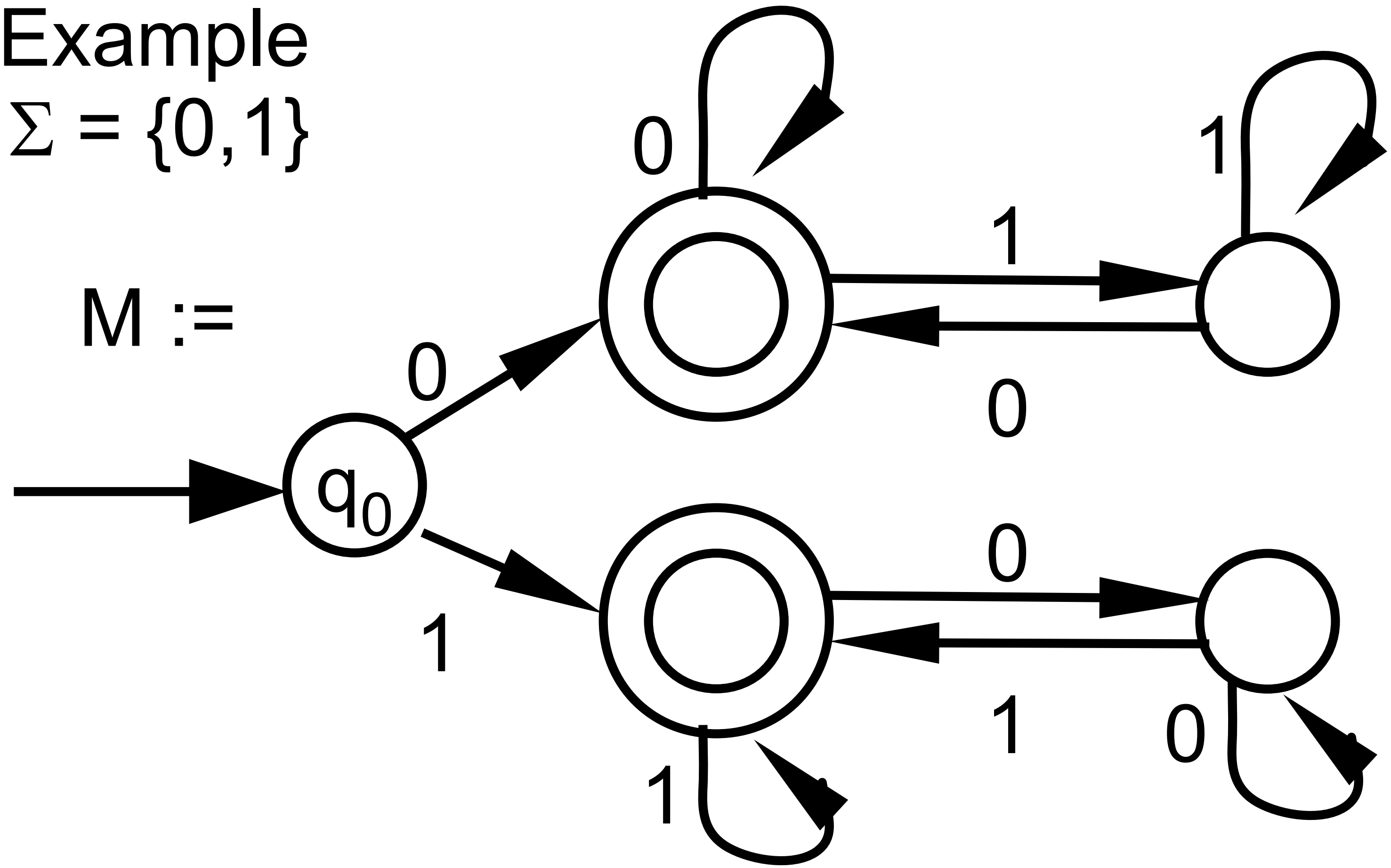
M :=



• L(M) = ?

Example
$\Sigma = \{0,1\}$

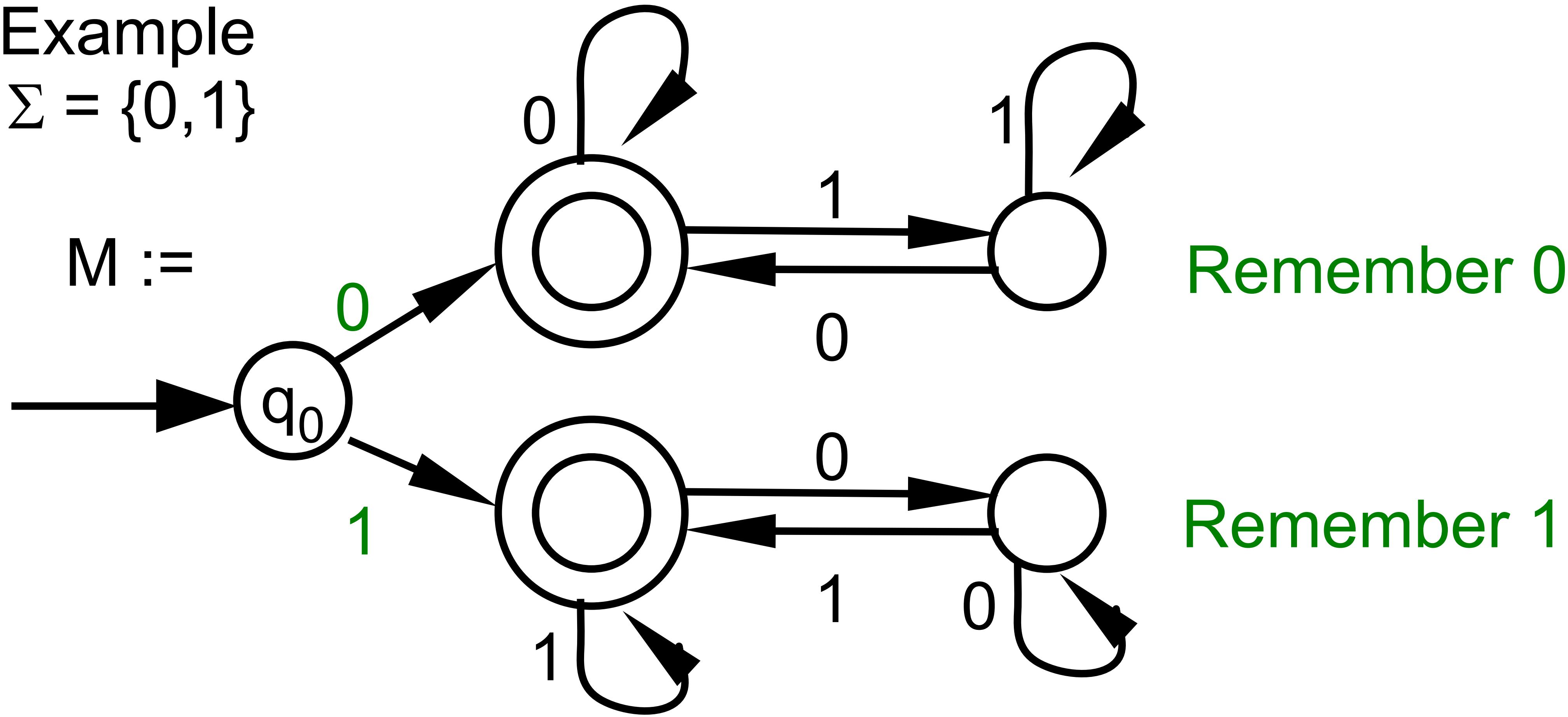
M :=



- L(M) = { w : w starts and ends with same symbol }
- Memory is encoded in … what ?

Example

$\Sigma = \{0,1\}$

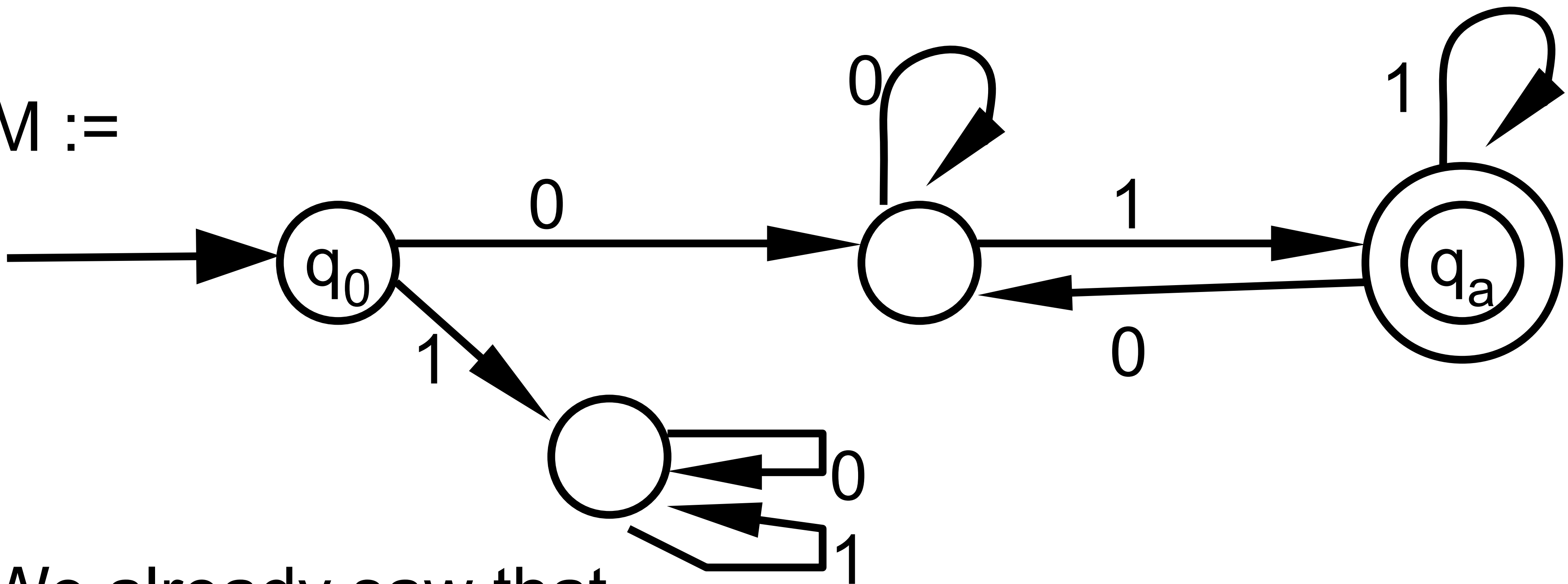


- $L(M) = \{ w : w$ starts and ends with same symbol $\}$
- Memory is encoded in states.

DFA have finite states, so finite memory

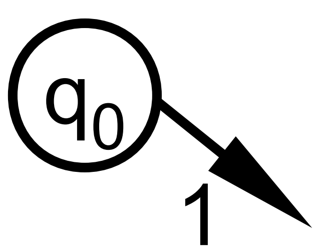
Convention:

M :=



We already saw that

L(M) = { w : w starts with 0 and ends with 1 }

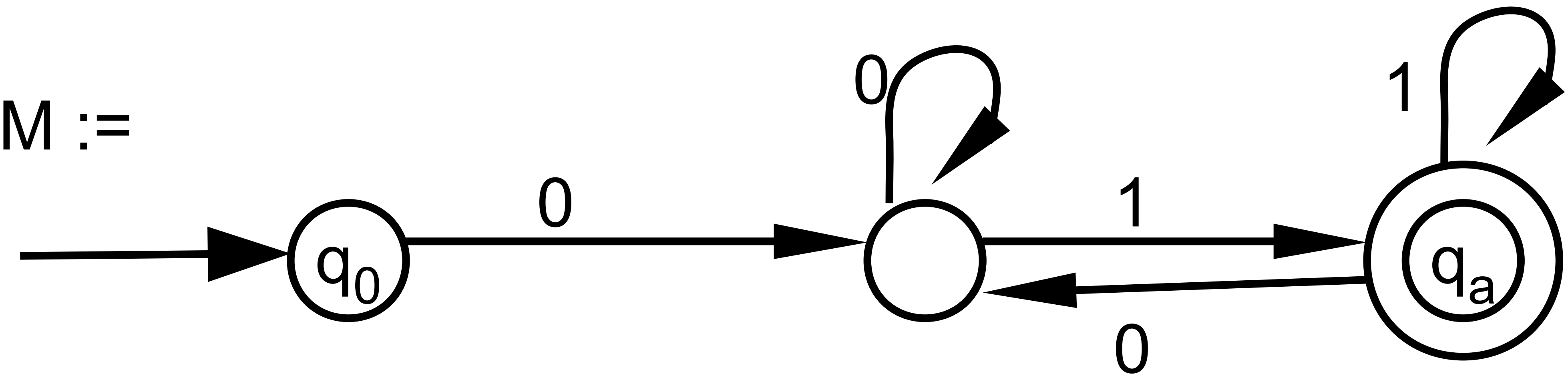
The arrow 

leads to a “sink” state.

If followed, M can never accept

Convention:

M :=



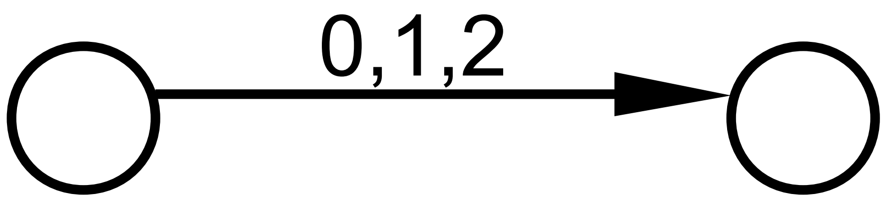
Don't need to write such arrows:

If, from some state, read symbol with no corresponding arrow, imagine M goes into “sink state” that is not shown, and REJECT.

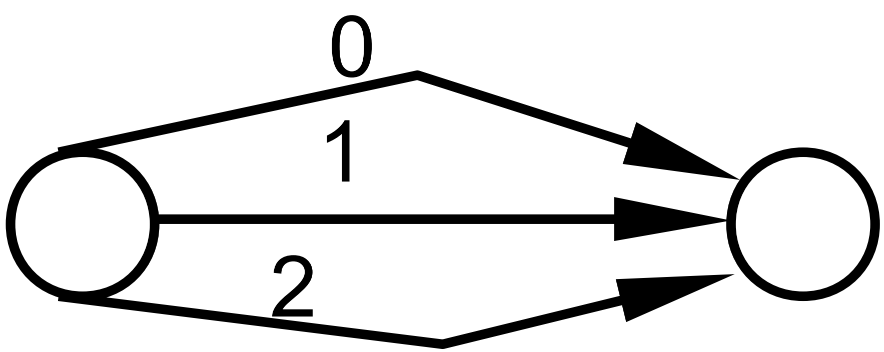
This makes pictures more compact.

Another convention:

List multiple transition on same arrow:



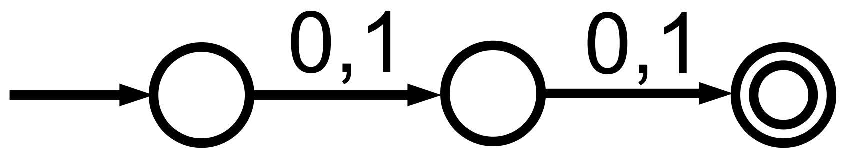
Means



This makes pictures more compact.

Example $\sum = \{0,1\}$

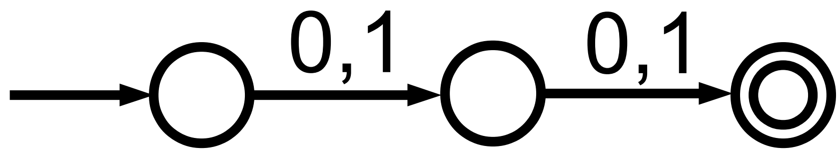
M =



L(M) = ?

Example $\sum = \{0,1\}$

M =

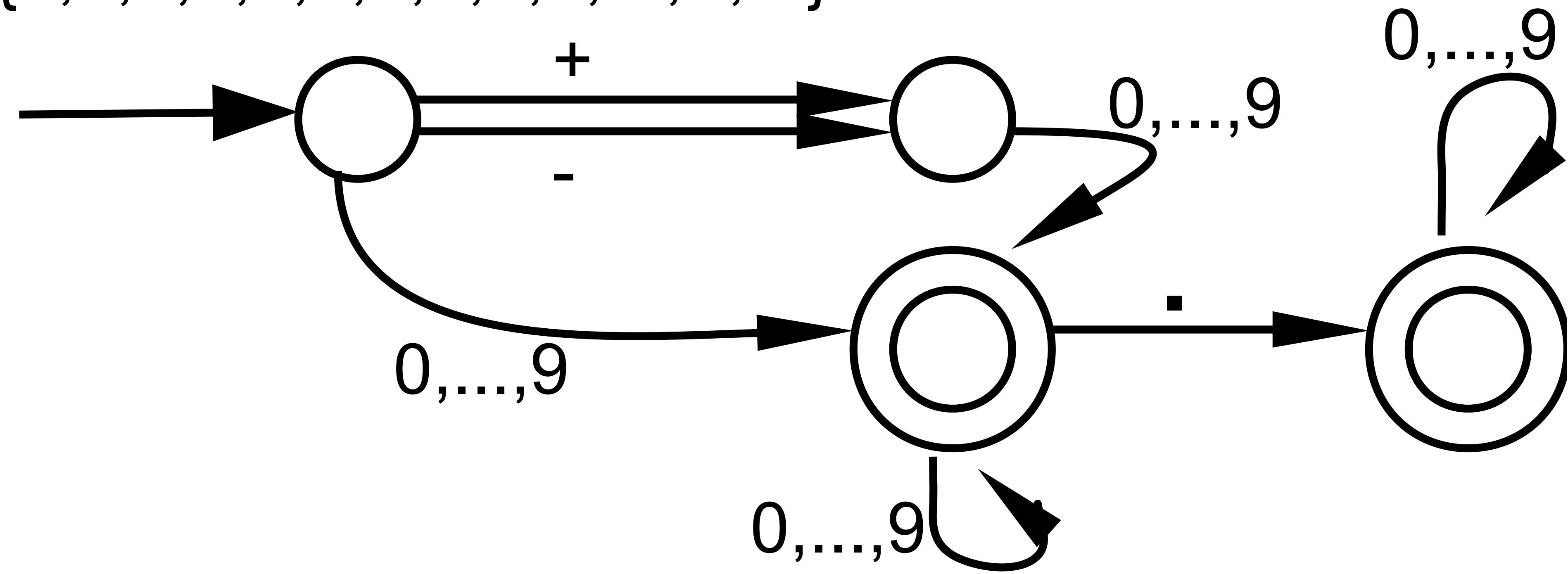


$L(M) = \sum^2 = \{00,01,10,11\}$

Example from programming languages:

Recognize strings representing numbers:

Σ = {0,1,2,3,4,5,6,7,8,9, +, -, . }

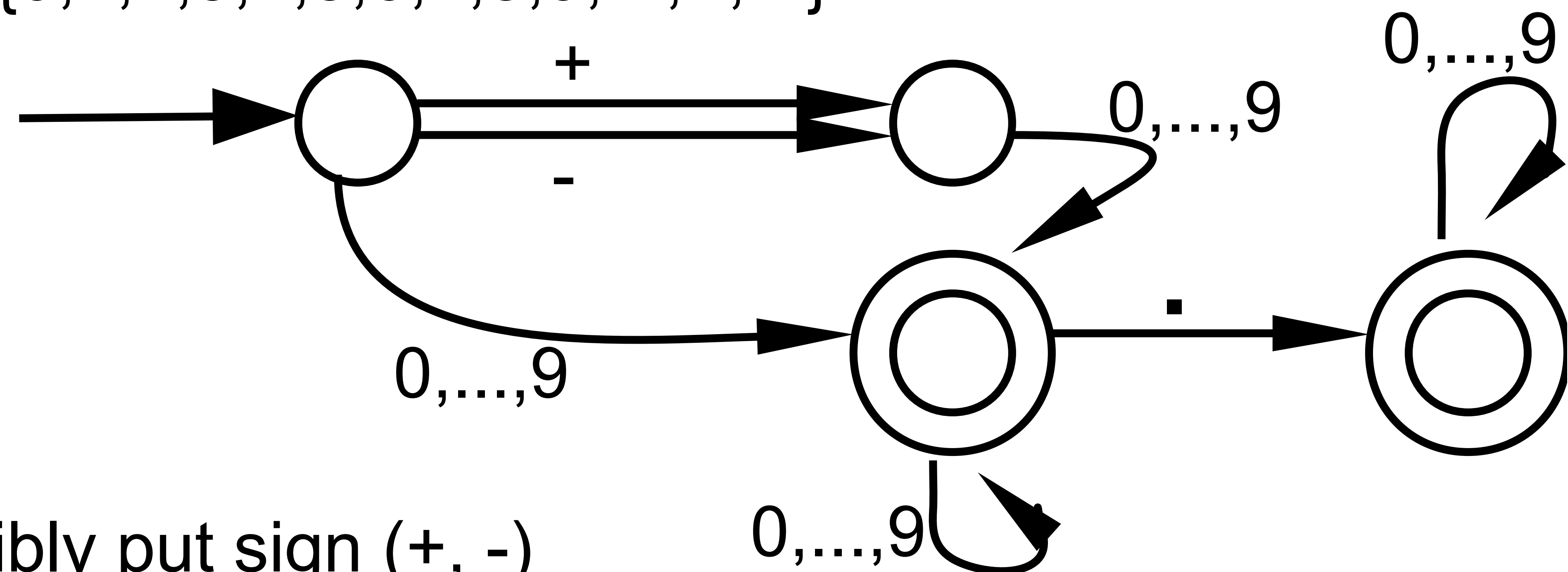


Note: 0,…,9 means 0,1,2,3,4,5,6,7,8,9: 10 transitions

Example from programming languages:

Recognize strings representing numbers:

$\Sigma = \{0,1,2,3,4,5,6,7,8,9, +, -, .\}$



Possibly put sign (+, -)

Follow with arbitrarily many digits, but at least one

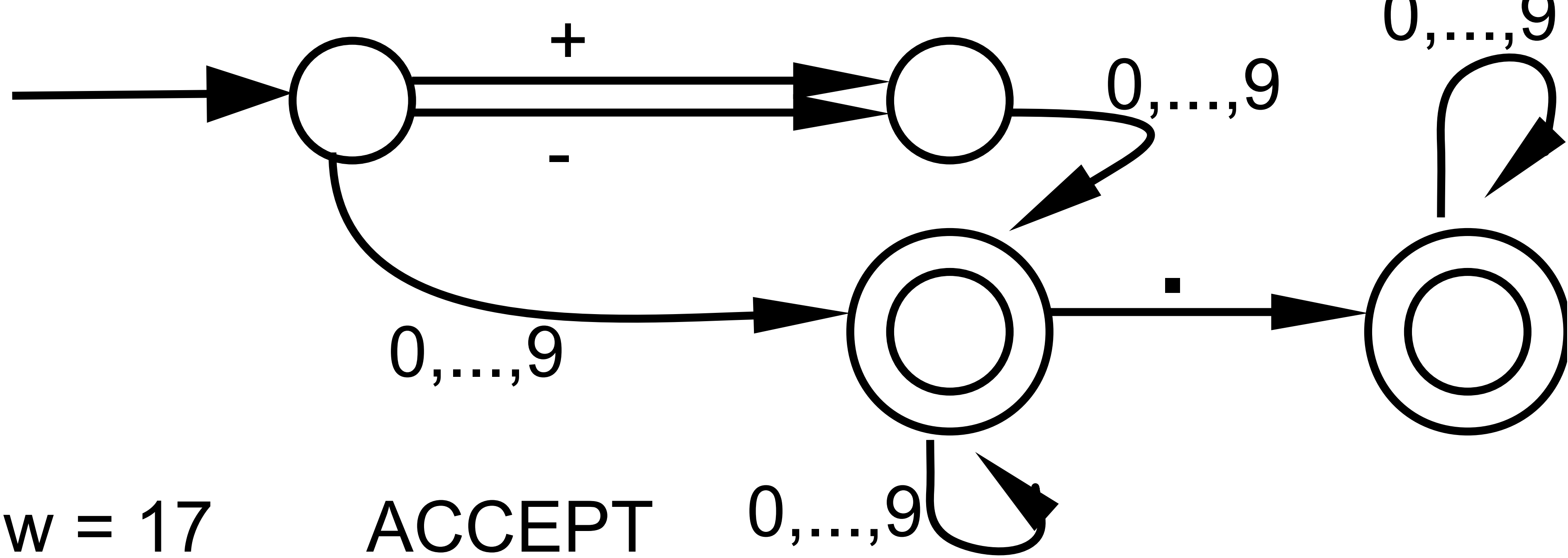
Possibly put decimal point

Follow with arbitrarily many digits, possibly none

Example from programming languages:

Recognize strings representing numbers:

$\Sigma = \{0,1,2,3,4,5,6,7,8,9, +, -, .\}$



Input w = 17 ACCEPT

Input w = + REJECT

Input w = -3.25 ACCEPT

Input w = +2.35-. REJECT

Example
$\Sigma = \{0,1\}$

- What about { w : w has same number of 0 and 1 }

- Can you design a DFA that recognizes that?

- It seems you need infinite memory

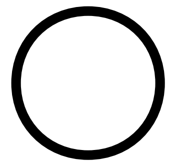
- We will prove later that

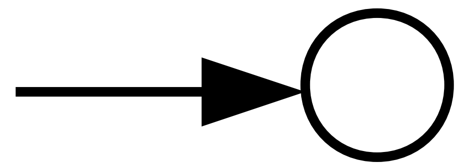
there is no DFA that recognizes that language !

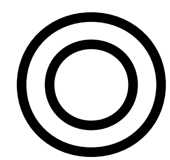
Next: formal definition of DFA

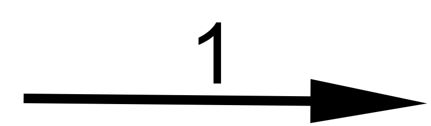
- Useful to prove various properties of DFA
- Especially important to prove that things CANNOT be recognized by DFA.
- Useful to practice mathematical notation

State diagram of a DFA:

- One or more states 

- Exactly one start state 

- Some number of accept states 

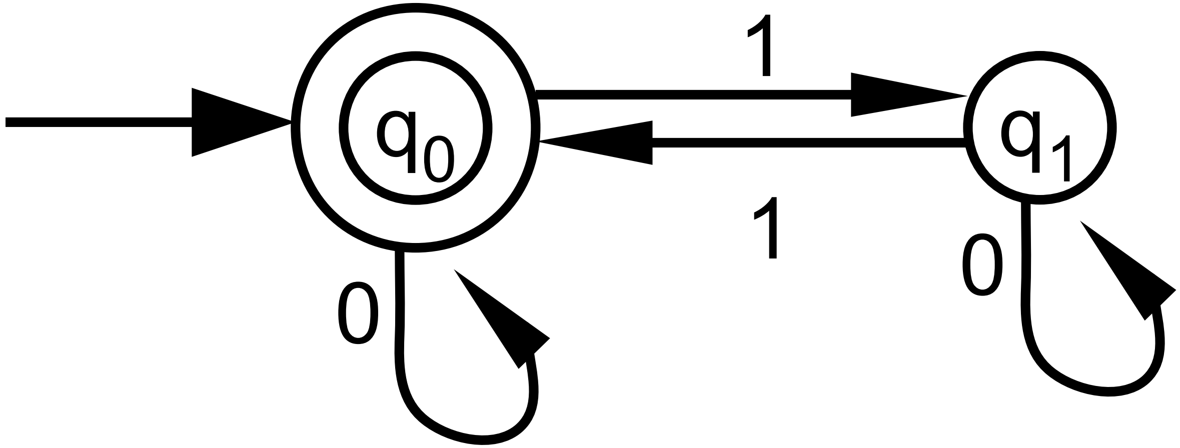
- Labelled transitions exiting each state, for every symbol in Σ 

- Definition: A finite automaton (DFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

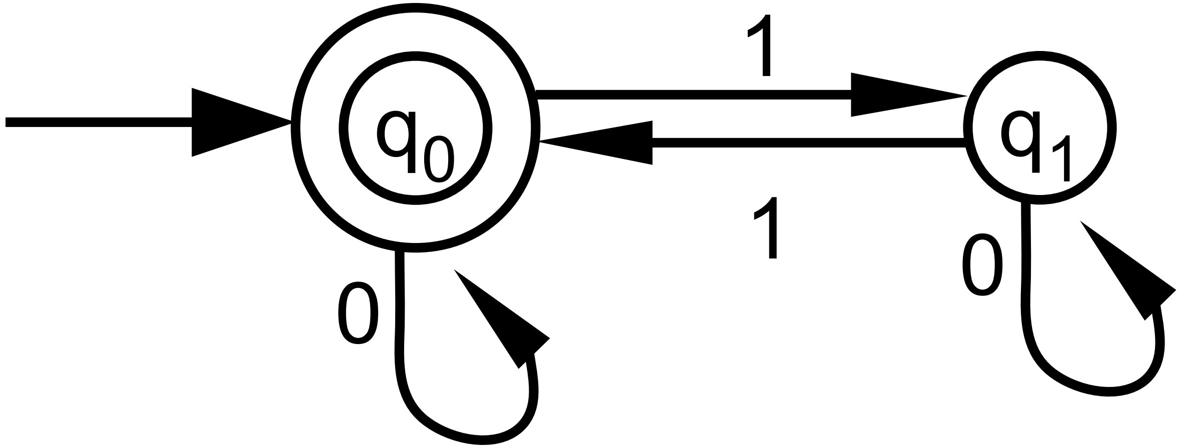
- Q is a finite set of states
- Σ is the input alphabet
- $\delta : Q \times \Sigma \rightarrow Q$ is the transition function
- q_0 in Q is the start state
- $F \subseteq Q$ is the set of accept states

$Q \times \Sigma$ is the set of ordered pairs $(a,b) : a \in Q,\ b \in \Sigma$

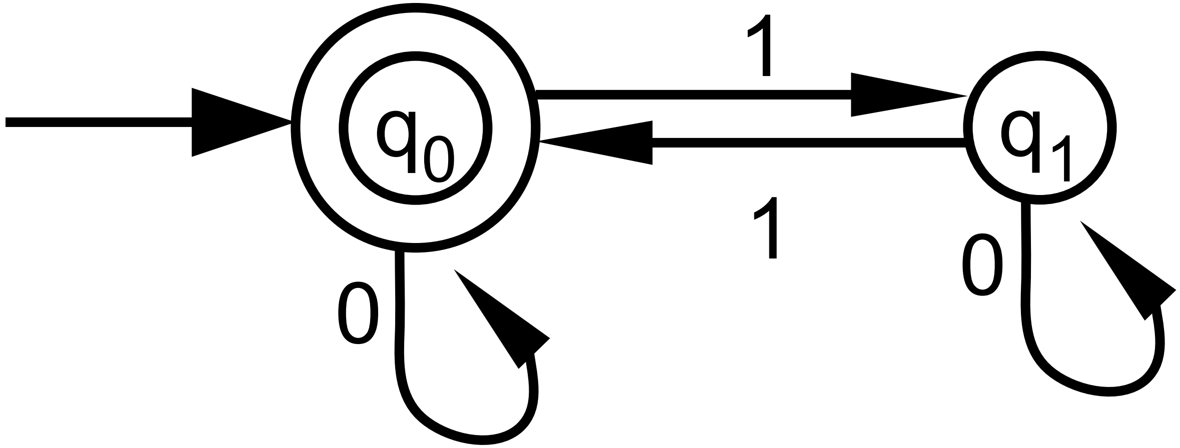
Example $\{q,r,s\} \times \{0,1\} = \{(q,0),(q,1),(r,0),(r,1),(s,0),(s,1)\}$



- Example: above DFA is 5-tuple (Q, Σ, δ, q_0, F) where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0, 0) = ?$

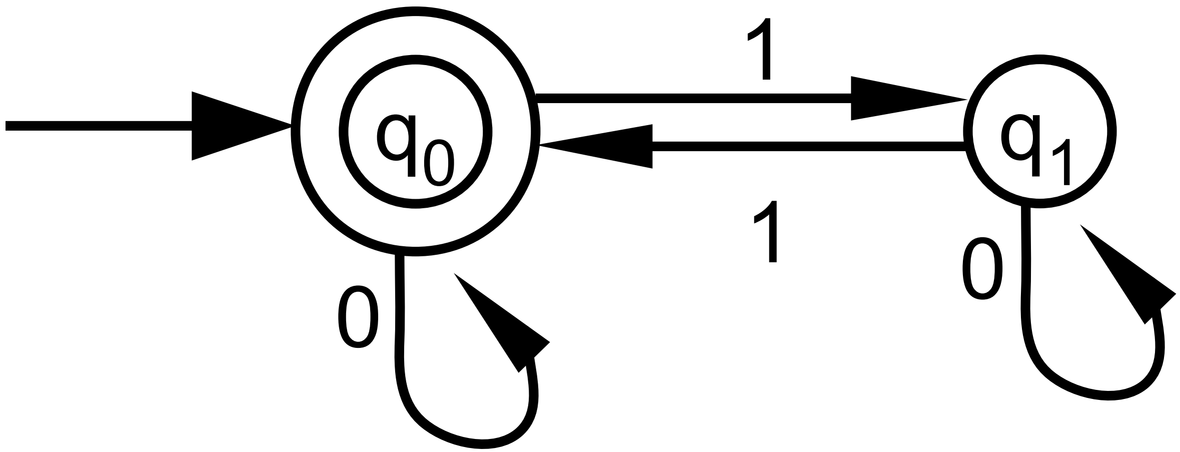


- Example: above DFA is 5-tuple (Q, Σ, δ, q_0, F) where
- $Q = \{ q_0, q_1 \}$
- $\Sigma = \{0,1\}$
- $\delta(q_0, 0) = q_0$ $\quad\delta(q_0, 1) = ?$



- Example: above DFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0 ,0) = q_0$ $\delta(q_0 ,1) = q_1$

  $\delta(q_1 ,0) = q_1$ $\delta(q_1 ,1) = q_0$
- q_0 in Q is the start state
- F = ?



- Example: above DFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0 ,0) = q_0 \quad \delta(q_0 ,1) = q_1$

  $\delta(q_1 ,0) = q_1 \quad \delta(q_1 ,1) = q_0$
- q_0 in Q is the start state
- $F = \{ q_0\} \subseteq Q$ is the set of accept states

- Definition: A DFA $(Q, \Sigma, \delta, q_0, F)$ accepts a string w if

- $w = w_1 w_2 \ldots w_k$ where, $\forall\ 1 \leq i \leq k$, w_i is in Σ (the k symbols of w)

- The sequence of k+1 states $r_0, r_1, .., r_k$ such that:

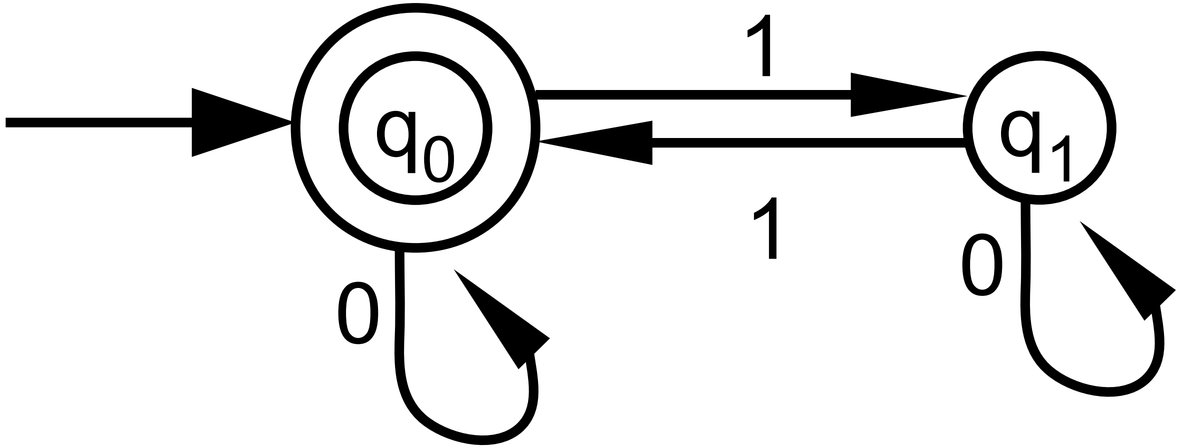
  (1) $r_0 = q_0$, and

  (2) $r_{i+1} = \delta(r_i, w_{i+1})$ $\forall\ 0 \leq i < k$

  has r_k in F

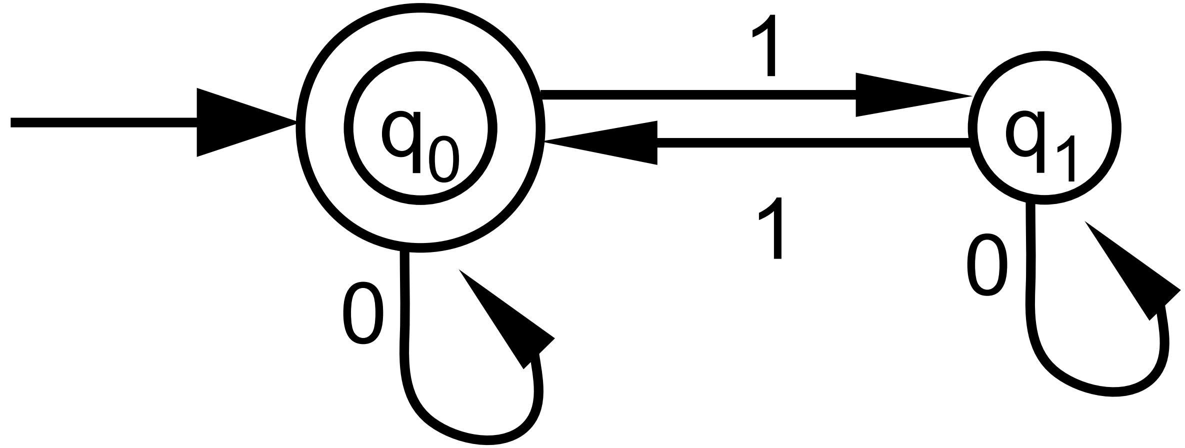
  (r_i = state DFA is in after reading i-th symbol in w)

Example



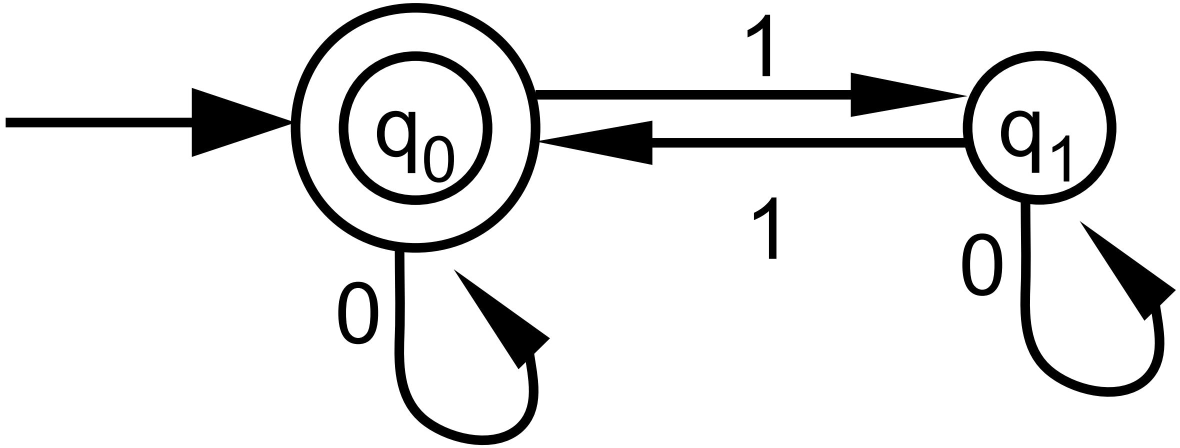
- Above DFA $(Q, \Sigma, \delta, q_0, F)$ accepts $w = 011$

Example



- Above DFA $(Q, \Sigma, \delta, q_0, F)$ accepts $w = 011$
- $w = 011 = w_1 w_2 w_3$ $w_1 = 0$ $w_2 = 1$ $w_3 = 1$

Example



- Above DFA $(Q, \Sigma, \delta, q_0, F)$ accepts $w = 011$
- $w = 011 = w_1\ w_2\ w_3$ $\quad w_1 = 0 \quad w_2 = 1 \quad w_3 = 1$

We must show that

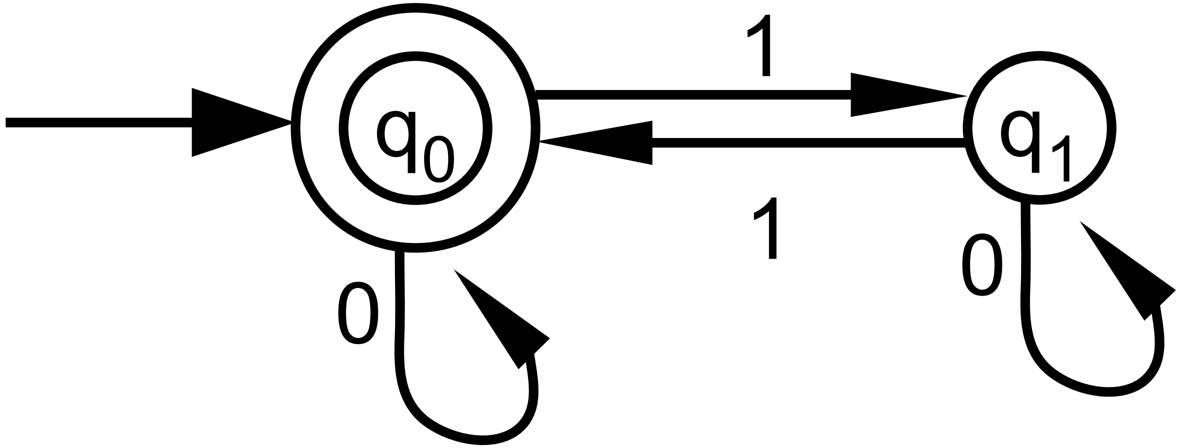
- The sequence of 3+1=4 states r_0, r_1, r_2, r_3 such that:

(1) $r_0 = q_0$

(2) $r_{i+1} = \delta(r_i, w_{i+1})\ \forall\ 0 \le i < 3$

has r_3 in F

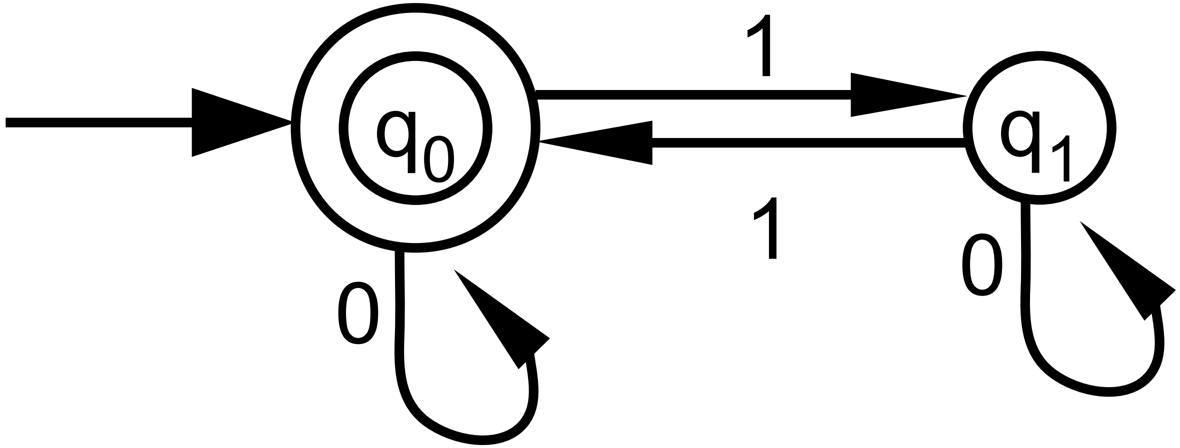
Example



- Above DFA $(Q, \Sigma, \delta, q_0, F)$ accepts $w = 011$
- $w = 011 = w_1\, w_2\, w_3$ $\qquad w_1 = 0 \quad w_2 = 1 \quad w_3 = 1$

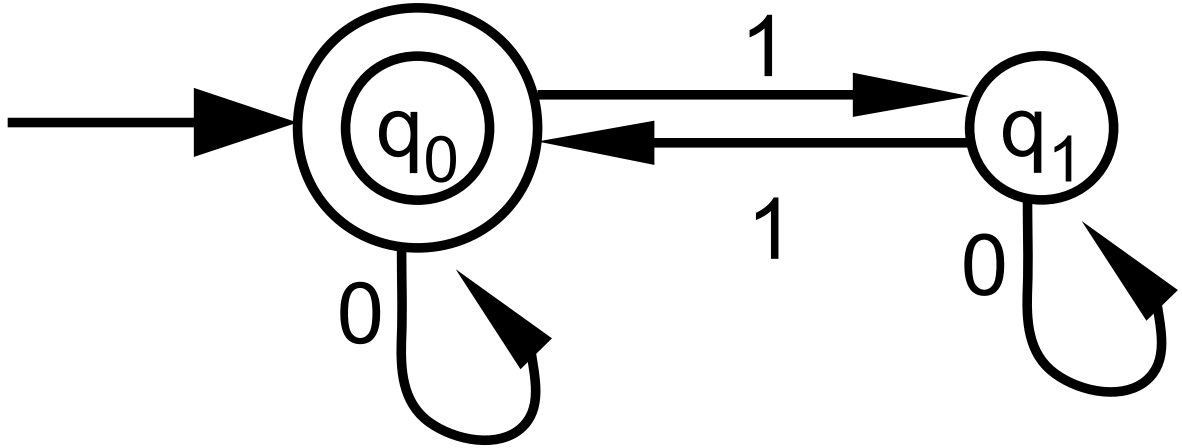
- $r_0 = q_0$
- $r_1 :=$?

Example



- Above DFA $(Q, \Sigma, \delta, q_0, F)$ accepts $w = 011$
- $w = 011 = w_1\ w_2\ w_3$ $w_1 = 0$ $w_2 = 1$ $w_3 = 1$
- $r_0 = q_0$
- $r_1 = \delta(r_0, w_1) = \delta(q_0, 0) = q_0$
- $r_2 :=$?

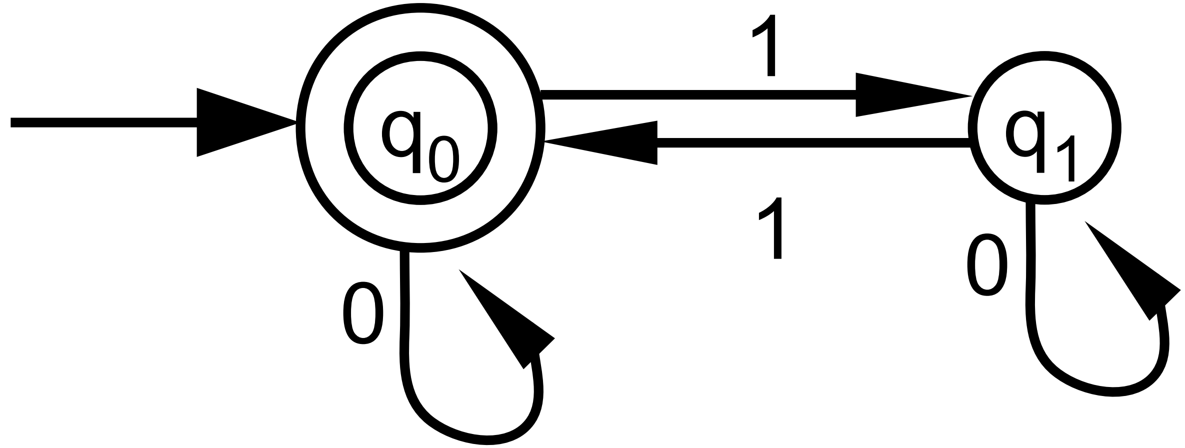
Example



- Above DFA $(Q, \Sigma, \delta, q_0, F)$ accepts w = 011
- $w = 011 = w_1 \, w_2 \, w_3$ $\qquad w_1 = 0 \quad w_2 = 1 \quad w_3 = 1$

- $r_0 = q_0$
- $r_1 = \delta(r_0, w_1) = \delta(q_0, 0) = q_0$
- $r_2 = \delta(r_1, w_2) = \delta(q_0, 1) = q_1$
- $r_3 :=$?

Example



- Above DFA $(Q, \Sigma, \delta, q_0, F)$ accepts $w = 011$
- $w = 011 = w_1 w_2 w_3$ $w_1 = 0$ $w_2 = 1$ $w_3 = 1$

- $r_0 = q_0$
- $r_1 = \delta(r_0, w_1) = \delta(q_0, 0) = q_0$
- $r_2 = \delta(r_1, w_2) = \delta(q_0, 1) = q_1$
- $r_3 = \delta(r_2, w_3) = \delta(q_1, 1) = q_0$
- $r_3 = q_0$ in F OK DONE!

- Definition: For a DFA M, we denote by L(M) the set of strings accepted by M:

$$L(M) := \{ w : M \text{ accepts } w\}$$

We say M accepts or recognizes the language L(M)

- Definition: A language L is regular

$$\text{if } \exists \text{ DFA } M : L(M) = L$$

In the next lectures we want to:

- Understand power of regular languages

- Develop alternate, compact notation to specify regular languages

Example: Unix command ***grep '\<c.*h\>'*** *file* selects all words starting with c and ending with h in *file*

- Understand power of regular languages:

- Suppose A, B are regular languages, what about
- not A := { w : w is not in A }
- A U B := { w : w in A or w in B }
- A o B := { $w_1 w_2$: w_1 in A and w_2 in B }
- A* := { $w_1 w_2 \ldots w_k$: $k \geq 0$, w_i in A for every i }

- Are these languages regular?

- Understand power of regular languages:

- Suppose A, B are regular languages, what about
- not A := { w : w is not in A }
- A U B := { w : w in A or w in B }
- A o B := { w_1 w_2 : w_1 in A and w_2 in B }
- A* := { w_1 w_2 … w_k : $k \geq 0$, w_i in A for every i }

- Terminology: Are regular languages closed under not, U, o, * ?

- Theorem:

  If A is a regular language, then so is (not A)

- Theorem:

  If A is a regular language, then so is (not A)

- Proof idea: ?????????? the set of accept states

- Theorem:

    If A is a regular language, then so is (not A)

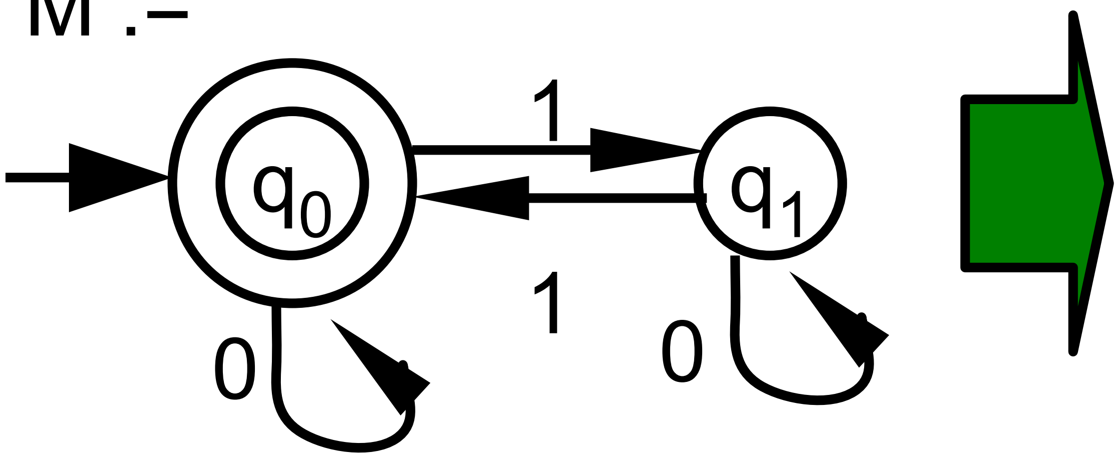
- Proof idea: Complement the set of accept states
- Example

- Theorem:

  If A is a regular language, then so is (not A)

- Proof idea: Complement the set of accept states
- Example:

M :=



L(M) =

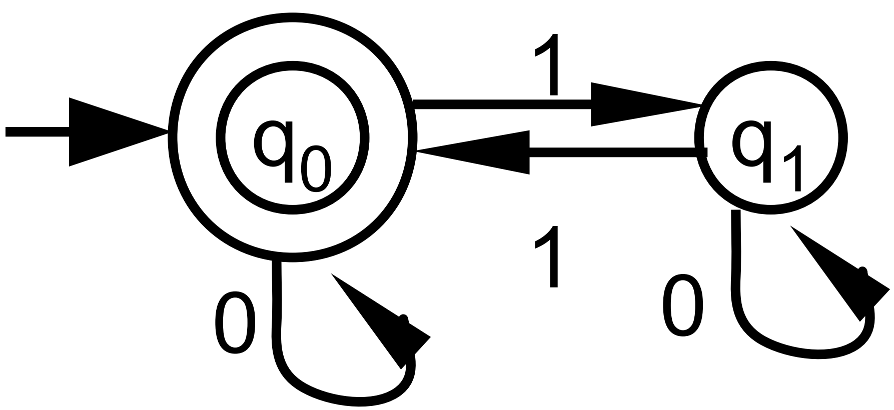
{ w : w has even number of 1}

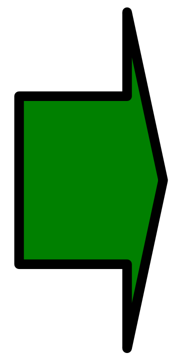
- **Theorem**:

  If A is a regular language, then so is (not A)

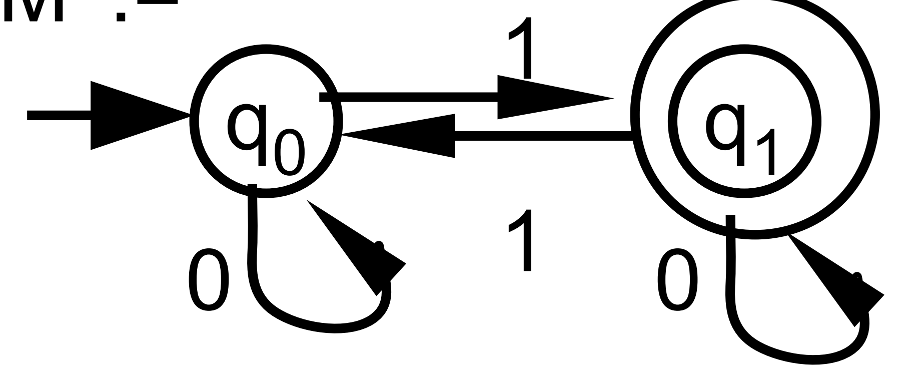
- Proof idea: Complement the set of accept states
- Example:

M :=





M' :=



L(M) =

{ w : w has even number of 1}

L(M') = not L(M) =

{ w : w has odd number of 1}

- Theorem: If A is a regular language, then so is (not A)
- Proof:

Given DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$.

Define DFA M' = ?????????????????????????????

This definition is the creative step of this proof,

the rest is (perhaps complicated but) mechanical

“unwrapping definitions”

- Theorem: If A is a regular language, then so is (not A)
- Proof:

  Given DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$.

  Define DFA $M' = (Q, \Sigma, \delta, q_0, F')$, where $F' :=$ not F.
- We need to show $L(M') =$ not $L(M)$, that is:

  for any w, ??????????????????????????

- Theorem: If A is a regular language, then so is (not A)
- Proof:

Given DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$.

Define DFA $M' = (Q, \Sigma, \delta, q_0, F')$, where $F' :=$ not F.

- We need to show L(M') = not L(M), that is:

for any w, M' accepts w ⟷ M does not accept w.

- So let w be any string of length k, and consider the k+1 states $r_0, r_1, .., r_k$ from the definition of accept:

(1) $r_0 = q_0$, and

(2) $r_{i+1} = \delta(r_i, w_{i+1}) \ \forall\ 0 \le i < k$.

How do we conclude?

- Theorem: If A is a regular language, then so is (not A)
- Proof:

Given DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$.

Define DFA $M' = (Q, \Sigma, \delta, q_0, F')$, where $F' := \text{not } F$.

- We need to show L(M') = not L(M), that is:

for any w, M' accepts w ⟺ M does not accept w

- So let w be any string of length k, and consider the k+1 states $r_0, r_1, .., r_k$ from the definition of accept:

(1) $r_0 = q_0$, and

(2) $r_{i+1} = \delta(r_i, w_{i+1}) \; \forall \; 0 \le i < k$.

Note that r_k in F' ⟺ r_k not in F, since F' = not F. ■

What is a proof?

- A proof is an explanation, written in English, of why something is true.
- Every sentence must be logically connected to the previous ones, often by “so”, “hence”, “since”, etc.
- Your audience is a human being, NOT a machine.

- Theorem: If A is a regular language, then so is (not A)
- Proof:

  DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$

  DFA $M' = (Q, \Sigma, \delta, q_0, F')$, where $F' := \text{not } F$.

  $L(M') = \text{not } L(M)$

  M' accepts w ⟺ M does not accept w

k+1 states $r_0, r_1, \ldots, r_k$

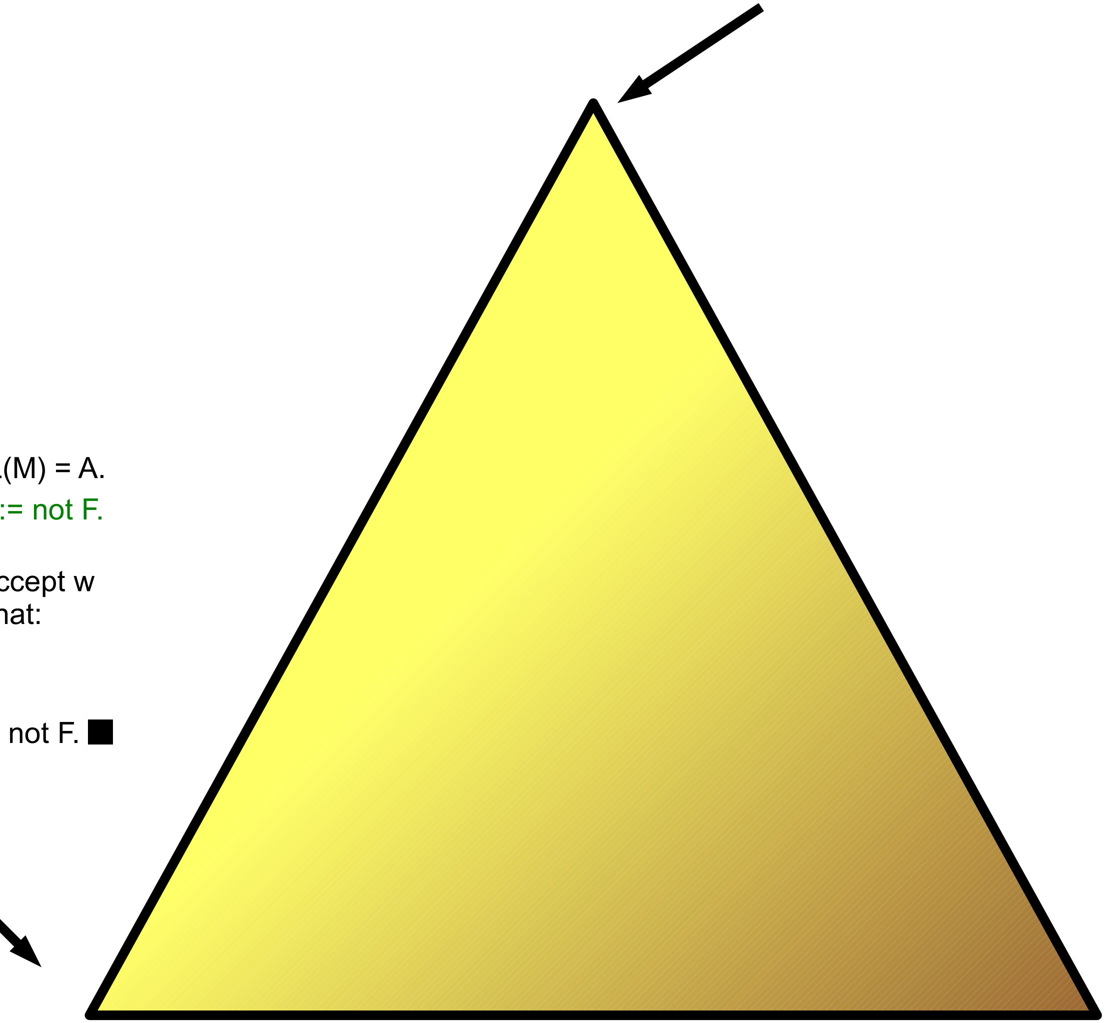
(1) $r_0 = q_0$, and

(2) $r_{i+1} = \delta(r_i, w_{i+1}) \ \forall\ 0 \le i < k$.

r_k in F' ⟺ r_k not in F, $F' = \text{not } F$. ■

What is a proof?

Complement the set of accept states



Given DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = A$.

Define DFA $M' = (Q, \Sigma, \delta, q_0, F')$, where $F' :=$ not F.

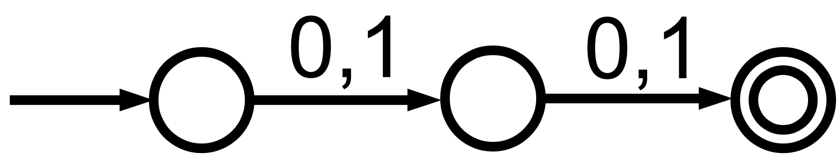
- We need to show $L(M') =$ not $L(M)$, that is:
  for any w, M' accepts w $\Leftrightarrow$ M does not accept w
- Consider the k+1 states $r_0, r_1, .., r_k$ such that:
  (1) $r_0 = q_0$, and
  (2) $r_{i+1} = \delta(r_i, w_{i+1})\ \forall\ 0 \le i < k$.

Note that r_k in F' $\Leftrightarrow$ r_k not in F, since $F' =$ not F. ■

To know a proof means to know all the pyramid

Example $\sum = \{0,1\}$

M =



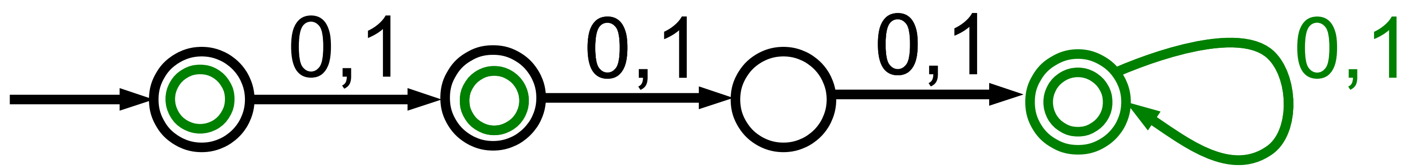
$L(M) = \sum^2 = \{00,01,10,11\}$

What is a DFA M' :

$L(M') = \text{not } \sum^2$ = all strings except those of length 2 ?

Example $\Sigma = \{0,1\}$

M' =



$L(M') = \text{not } \Sigma^2 = \{0,1\}^* - \{00,01,10,11\}$

Do not forget the convention about the sink state!

- Suppose A, B are regular languages, what about
- not A := { w : w is not in A }   REGULAR
- A U B := { w : w in A or w in B }
- A o B := { w_1 w_2 : w_1 in A and w_2 in B }
- A* := { w_1 w_2 … w_k : $k \geq 0$, w_i in A for every i }

•Theorem: If A, B are regular, then so is A U B

•Proof idea: Take Cartesian product of states

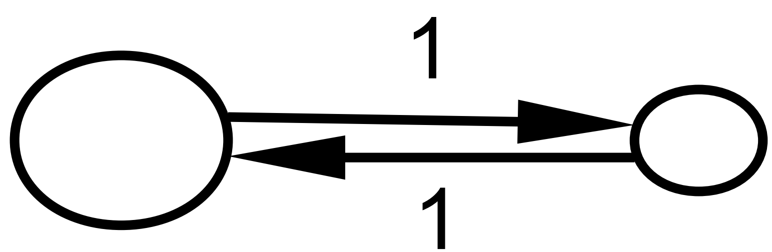
In a pair (q,q'),

q tracks DFA for A,

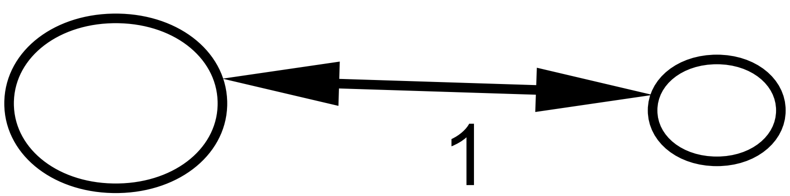
q' tracks DFA for B.

•Next we see an example.

In it we abbreviate

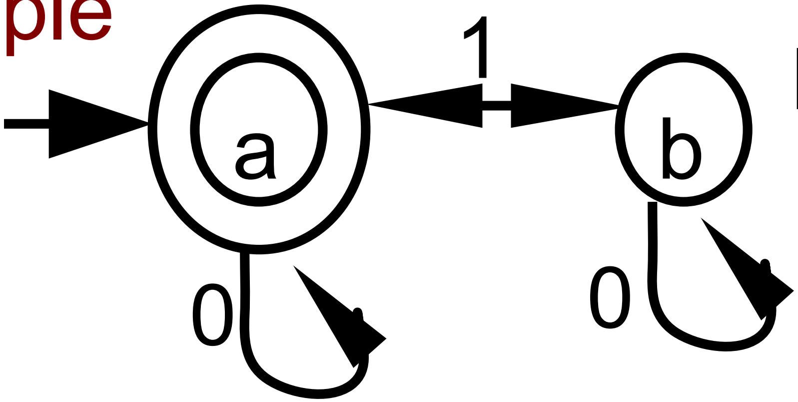


with

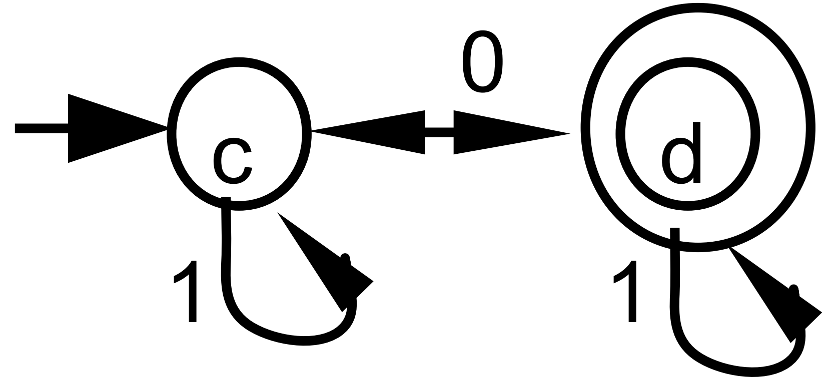


Example

$M_A :=$



$M_B :=$

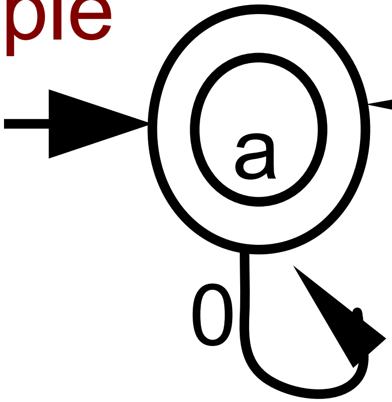


$L(M_A) = A =$?

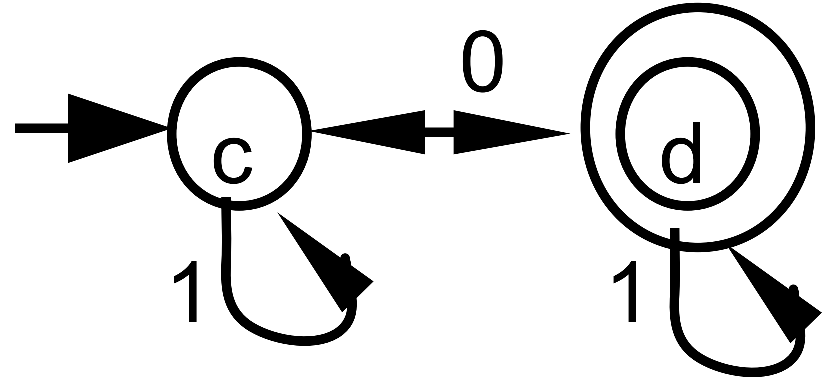
$L(M_B) = B =$?

Example

M_A :=



M_B :=



$L(M_A) = A =$

{ w : w has even number of 1}

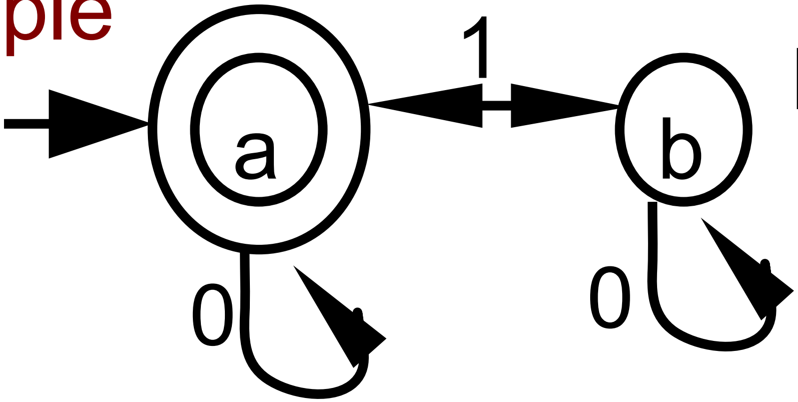
$L(M_B) = B =$

{ w : w has odd number of 0}

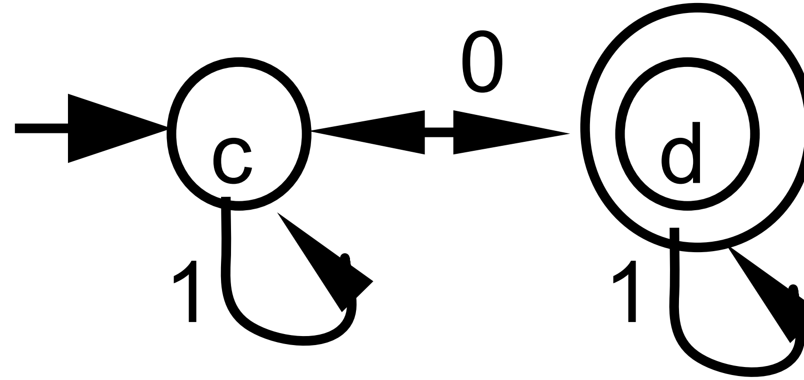
$M_{A \cup B}$:= How many states?

Example

$M_A :=$



$M_B :=$



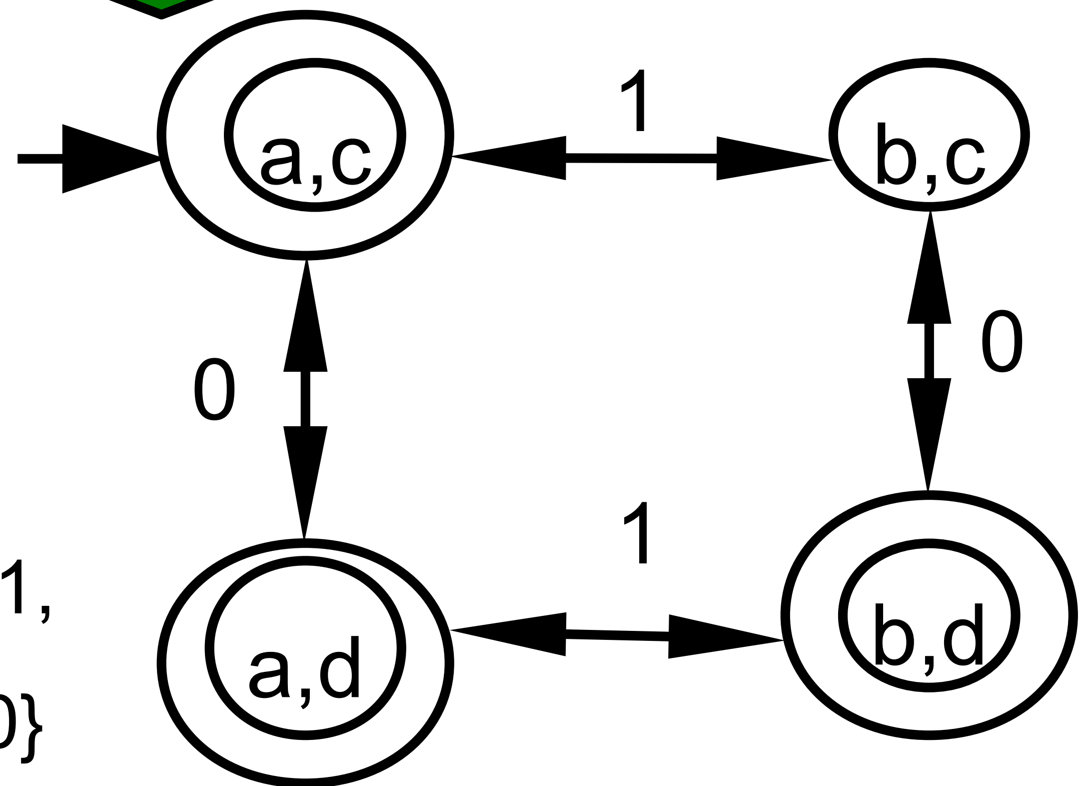
$L(M_A) = A =$

{ w : w has even number of 1}

$L(M_B) = B =$

{ w : w has odd number of 0}

$M_{AUB} :=$



$L(M_{AUB}) = AUB =$

{ w : w has even number of 1,

or odd number of 0}

- **Theorem**: If A, B are regular, then so is A U B
- **Proof:**

Given DFA $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ such that L(M) = A,

DFA $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ such that L(M) = B.

Define DFA $M = (Q, \Sigma, \delta, q_0, F)$, where

$Q :=$?

- Theorem: If A, B are regular, then so is A U B
- Proof:

Given DFA $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ such that $L(M) = A$,

DFA $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ such that $L(M) = B$.

Define DFA $M = (Q, \Sigma, \delta, q_0, F)$, where

$Q := Q_A \times Q_B$

$q_0 :=$?

- Theorem: If A, B are regular, then so is A U B
- Proof:

Given DFA $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ such that L(M) = A,

DFA $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ such that L(M) = B.

Define DFA $M = (Q, \Sigma, \delta, q_0, F)$, where

$Q := Q_A \times Q_B$

$q_0 := (q_A , q_B)$

F := ?

- **Theorem**: If A, B are regular, then so is A U B
- **Proof:**

Given DFA $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ such that L(M) = A,

DFA $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ such that L(M) = B.

Define DFA $M = (Q, \Sigma, \delta, q_0, F)$, where

$Q := Q_A \times Q_B$

$q_0 := (q_A, q_B)$

$F := \{(q,q') \in Q : q \in F_A \text{ or } q' \in F_B\}$

$\delta((q,q'), v) := (?, ?)$

- Theorem: If A, B are regular, then so is A U B
- Proof:

Given DFA $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ such that L(M) = A,

DFA $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ such that L(M) = B.

Define DFA $M = (Q, \Sigma, \delta, q_0, F)$, where

$Q := Q_A \times Q_B$

$q_0 := (q_A , q_B)$

$F := \{(q,q') \in Q : q \in F_A \text{ or } q' \in F_B \}$

$\delta((q,q'), v) := (\delta_A (q,v), \delta_B (q',v))$

- We need to show L(M) = A U B that is, for any w:
  M accepts w ⟷ M_A accepts w or M_B accepts w

- Proof of M accepts w $\Rightarrow M_A$ accepts w or M_B accepts w
- Suppose that M accepts w of length k.
- From the definitions of accept and M, the sequence

$(s_0, t_0) = q_0 = (q_A, q_B)$,

$(s_{i+1}, t_{i+1}) = \delta((s_i, t_i), w_{i+1}) = (\delta_A(s_i, w_{i+1}), \delta_B(t_i, w_{i+1}) \; \forall 0 \leq i < k$

has $(s_k, t_k) \in$?

- Proof of M accepts w $\rightarrow M_A$ accepts w or M_B accepts w
- Suppose that M accepts w of length k.
- From the definitions of accept and M, the sequence $(s_0, t_0) = q_0 = (q_A, q_B)$, $(s_{i+1}, t_{i+1}) = \delta((s_i, t_i), w_{i+1}) = (\delta_A(s_i, w_{i+1}), \delta_B(t_i, w_{i+1})\ \forall 0 \leq i < k$ has $(s_k, t_k) \in F$.
- By our definition of F, what can we say about (s_k, t_k) ?

- Proof of M accepts w $\rightarrow M_A$ accepts w or M_B accepts w
- Suppose that M accepts w of length k.
- From the definitions of accept and M, the sequence $(s_0, t_0) = q_0 = (q_A, q_B)$, $(s_{i+1}, t_{i+1}) = \delta((s_i, t_i), w_{i+1}) = (\delta_A(s_i, w_{i+1}), \delta_B(t_i, w_{i+1})\ \forall 0 \le i < k$ has $(s_k, t_k) \in F$.
- By our definition of F, $s_k \in F_A$ or $t_k \in F_B$.
- Without loss of generality, assume $s_k \in F_A$.
- Then M_A accepts w because the sequence $s_0 = q_A$, $s_{i+1} = \delta_A(s_i, w_{i+1})\ \forall 0 \le i < k$, has $s_k \in F_A$.

- Proof of M accepts w $\Leftarrow$ M_A accepts w or M_B accepts w
- W/out loss of generality, assume M_A accepts w, |w|=k.
- From the definition of M_A accepts w, the sequence

$r_0 := q_A$, $r_{i+1} := \delta_A(r_i, w_{i+1})$ $\forall\, 0 \le i < k$, has r_k in ?

- Proof of M accepts w $\Leftarrow$ M_A accepts w or M_B accepts w
- W/out loss of generality, assume M_A accepts w, $|w|=k$.
- From the definition of M_A accepts w, the sequence

  $r_0 := q_A$, $r_{i+1} := \delta_A(r_i, w_{i+1})$ $\forall\, 0 \leq i < k$, has r_k in F_A.

- Define the sequence of k+1 states

  $t_0 := q_B$, $t_{i+1} := \delta_B(t_i, w_{i+1})$ $\forall\, 0 \leq i < k$.

- M accepts w because the sequence

  ?????????? (recall states in M are pairs)

- Proof of M accepts w $\Leftarrow$ M_A accepts w or M_B accepts w
- W/out loss of generality, assume M_A accepts w, |w|=k.
- From the definition of M_A accepts w, the sequence

$r_0 := q_A$, $r_{i+1} := \delta_A(r_i, w_{i+1})$ $\forall\ 0 \le i < k$, has r_k in F_A .

- Define the sequence of k+1 states

$$t_0 := q_B,\ t_{i+1} := \delta_B(t_i, w_{i+1})\ \forall\ 0 \le i < k.$$

- M accepts w because the sequence

$(r_0, t_0) = q = (q_A, q_B)$,

$(r_{i+1}, t_{i+1}) = \delta((r_i, t_i), w_{i+1}) = (\delta_A(r_i, w_{i+1}), \delta_B(t_i, w_{i+1}))\ \forall 0 \le i < k$

has (r_k, t_k) in F, by our definition of F. ■

- Suppose A, B are regular languages, what about
- not A := { w : w is not in A }   REGULAR
- A U B := { w : w in A or w in B }   REGULAR
- A o B := { w_1 w_2 : w_1 in A and w_2 in B }
- A* := { w_1 w_2 … w_k : $k \geq 0$, w_i in A for every i }

- Other two are more complicated!

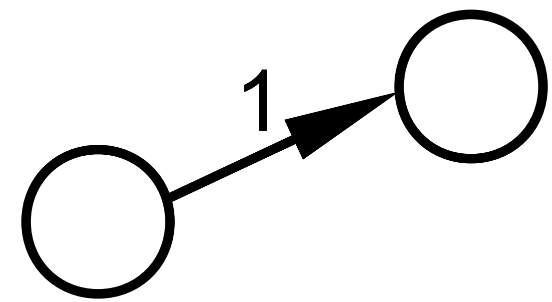
- Plan: we introduce NFA

  prove that NFA are equivalent to DFA

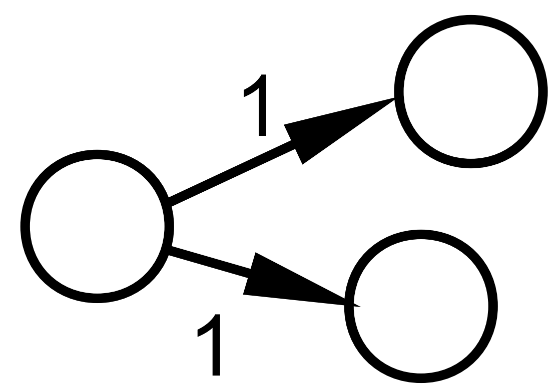
  reprove A U B, prove A o B, A* regular, using NFA

Non deterministic finite automata (NFA)

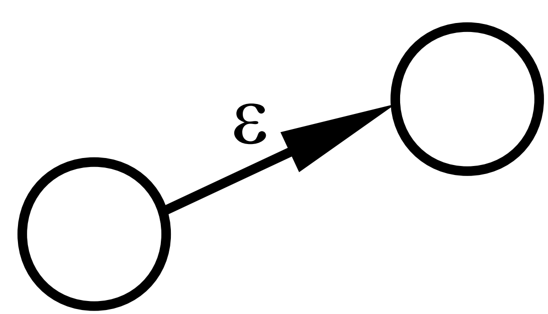
- DFA: given state and input symbol, unique choice for next state, deterministic:



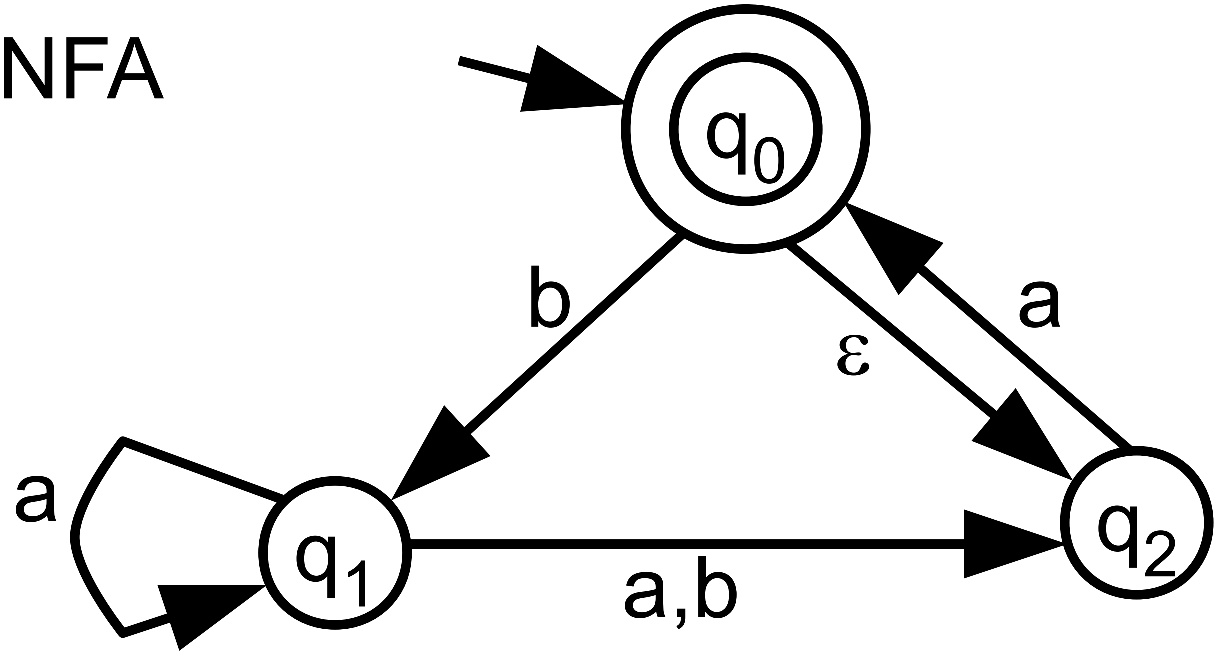
- Next we allow multiple choices, non-deterministic



- We also allow ε-transitions: can follow without reading anything



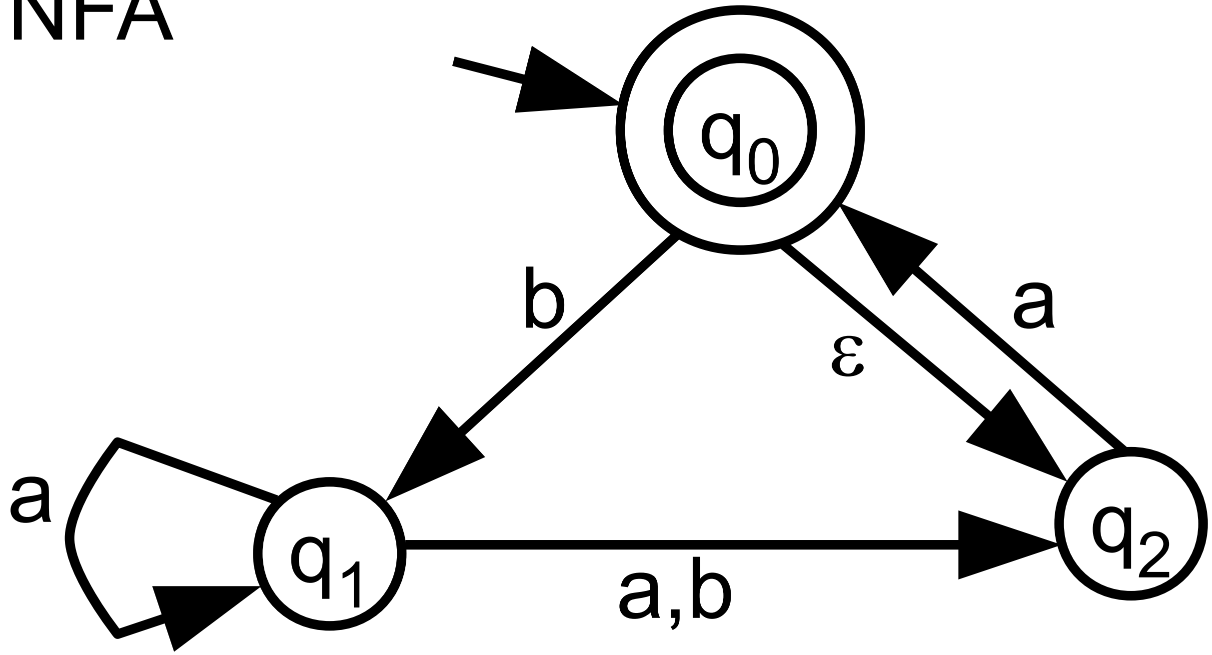
Example of NFA



Intuition of how it computes:

- Accept string w if there is a way to follow transitions that ends in accept state
- Transitions labelled with symbol in $\Sigma = \{a,b\}$ must be matched with input
- ε transitions can be followed without matching

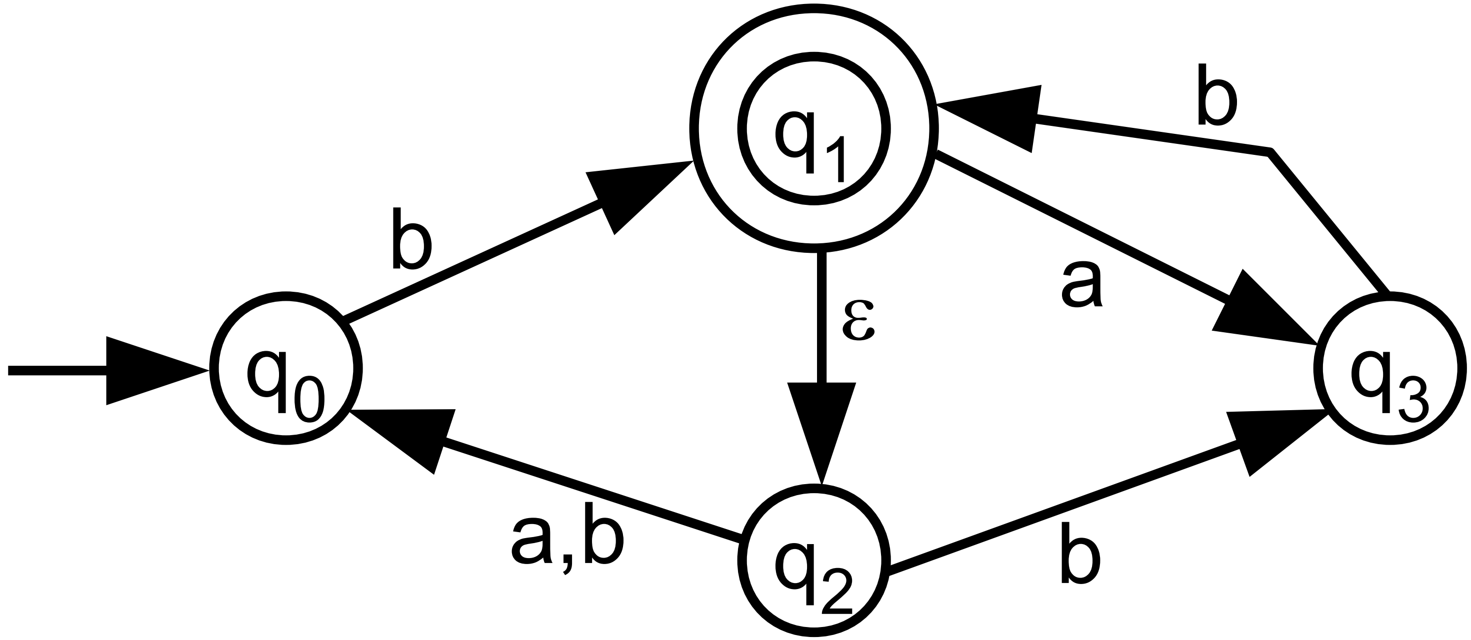
Example of NFA



Example:

- Accept a (first follow ε-transition)
- Accept baaa

ANOTHER Example of NFA



Example:

- Accept bab (two accepting paths, one uses the ε-transition)
- Reject ba (two possible paths, but neither has final state = q_1)

- **Definition:** A non-deterministic finite automaton (NFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

- Q is a finite set of states
- Σ is the input alphabet
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \text{Powerset}(Q)$
- q_0 in Q is the start state
- $F \subseteq Q$ is the set of accept states

- Recall: Powerset(Q) = set of all subsets of Q

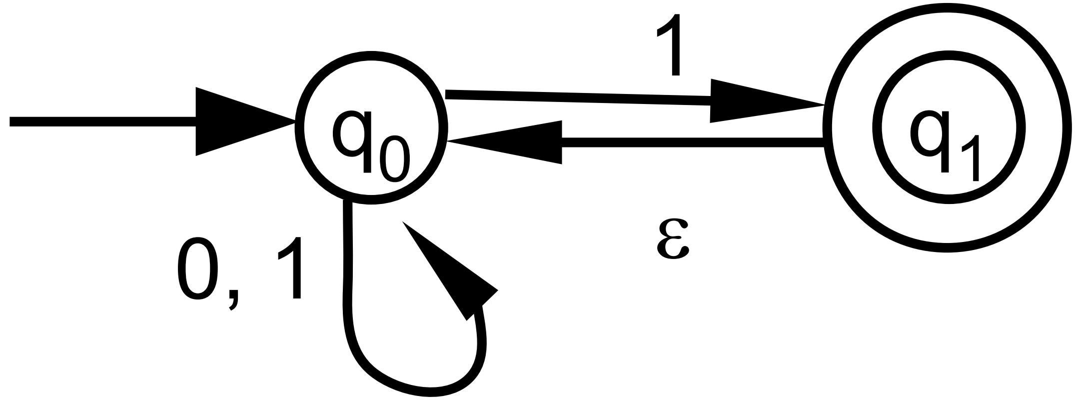
  Example: Powerset({1,2}) = ?

- Definition: A non-deterministic finite automaton (NFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

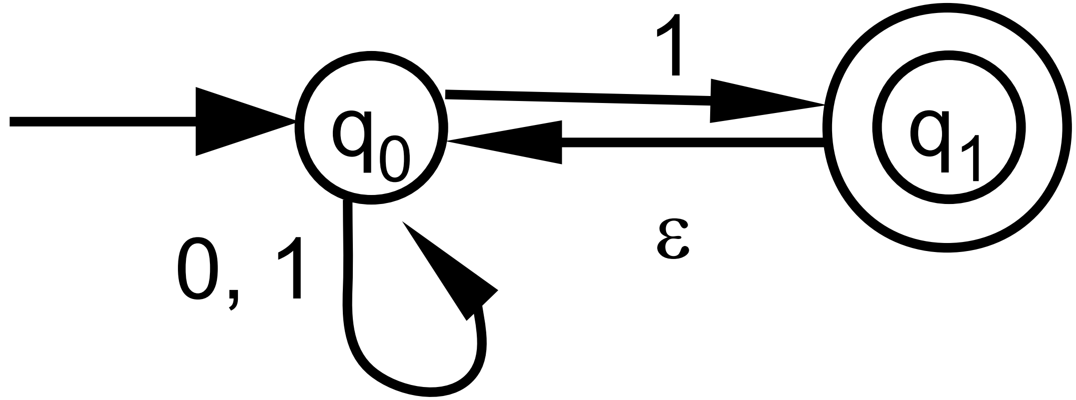
- Q is a finite set of states
- Σ is the input alphabet
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \text{Powerset}(Q)$
- q_0 in Q is the start state
- $F \subseteq Q$ is the set of accept states

- Recall: Powerset(Q) = set of all subsets of Q

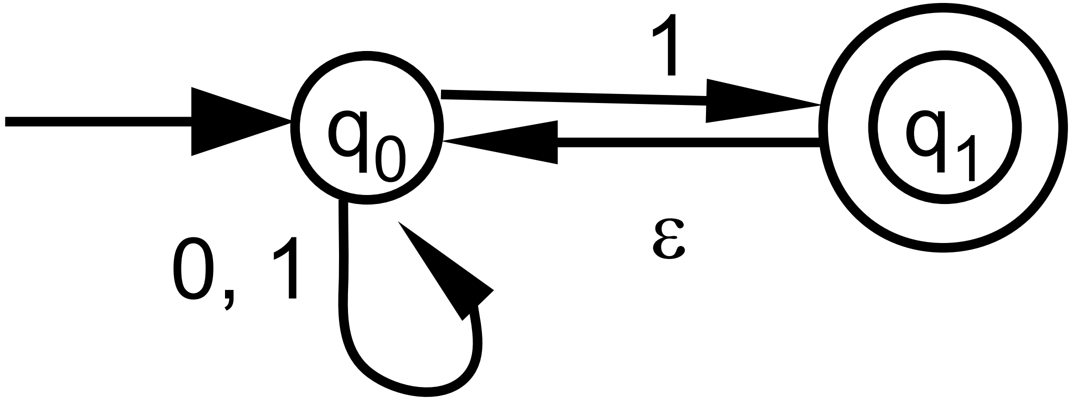
  Example: Powerset({1,2}) = $\{\emptyset, \{1\}, \{2\}, \{1,2\}\}$



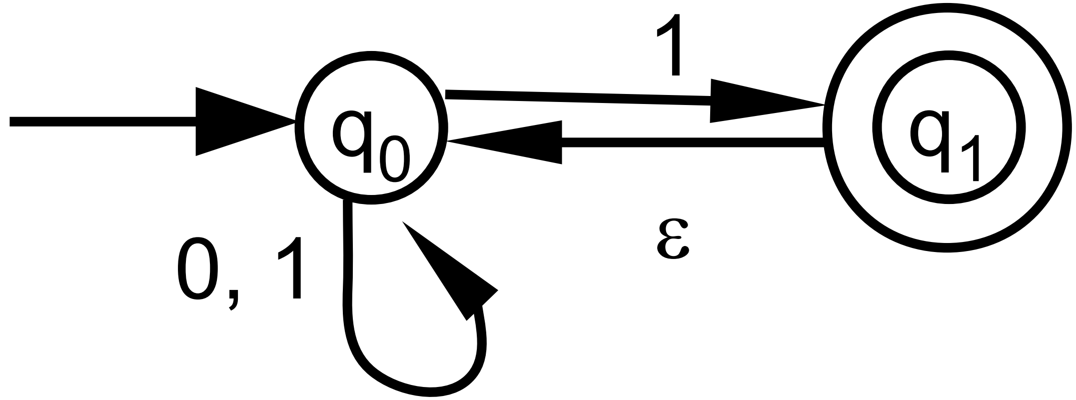
- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1 \}$
- $\Sigma = \{0,1\}$
- $\delta(q_0, 0) =$?



- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0 ,0) = \{q_0\}$ $\delta(q_0 ,1) =$?

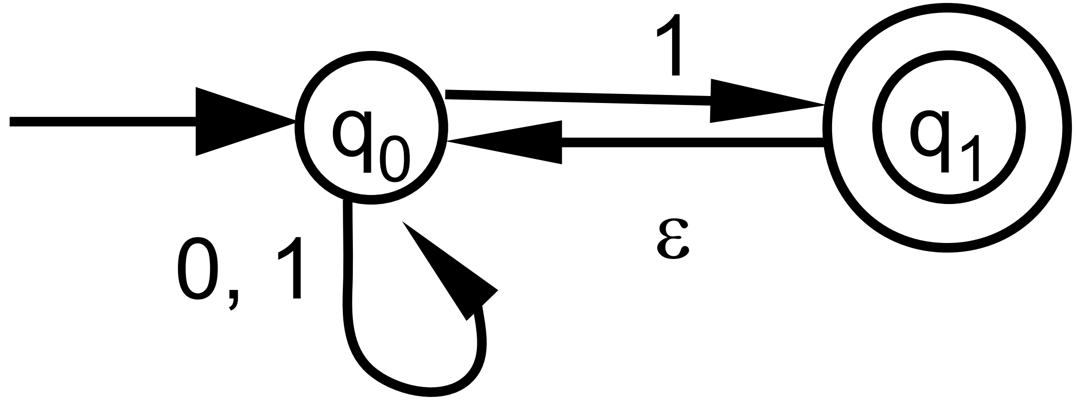


- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0, 0) = \{q_0\}$ $\delta(q_0, 1) = \{q_0, q_1\}$ $\delta(q_0, \varepsilon) =$?



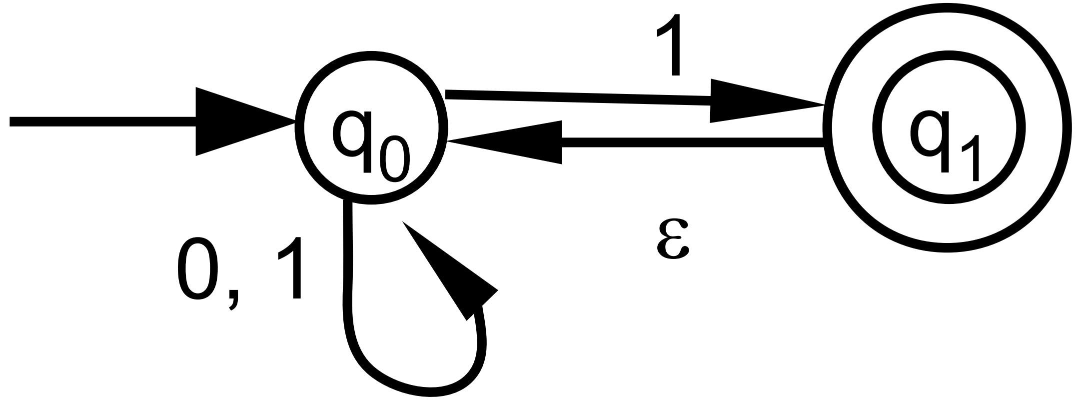
- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0 ,0) = \{q_0\}$ $\delta(q_0 ,1) = \{q_0, q_1\}$ $\delta(q_0 ,\varepsilon) = \emptyset$

  $\delta(q_1 ,0) =$?



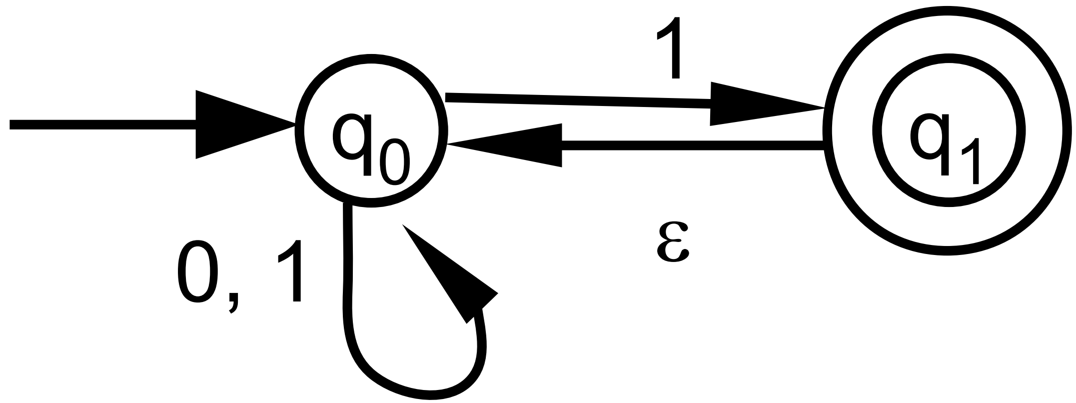
- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0, 0) = \{q_0\}$ $\delta(q_0, 1) = \{q_0, q_1\}$ $\delta(q_0, \varepsilon) = \varnothing$

  $\delta(q_1, 0) = \varnothing$ $\delta(q_1, 1) = ?$



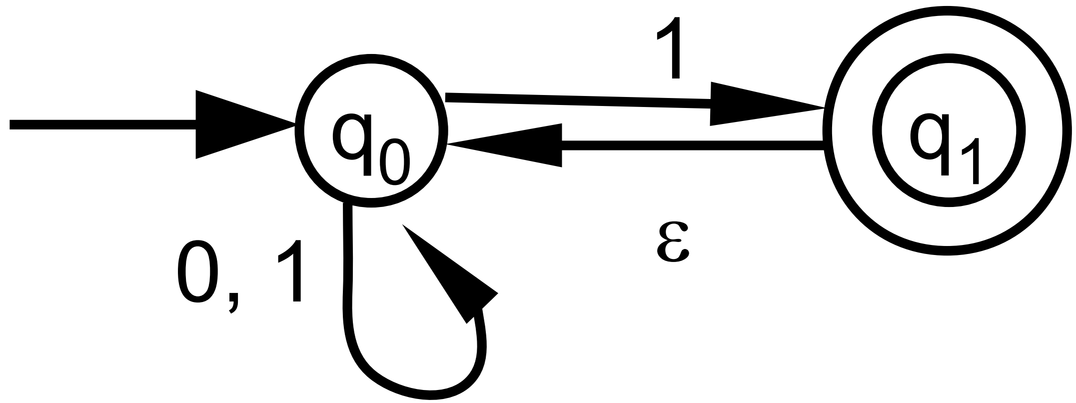
- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0 ,0) = \{q_0\}$ $\delta(q_0 ,1) = \{q_0, q_1\}$ $\delta(q_0 ,\varepsilon) = \varnothing$

  $\delta(q_1 ,0) = \varnothing$ $\delta(q_1 ,1) = \varnothing$ $\delta(q_1 ,\varepsilon) =$?



- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1 \}$
- $\Sigma = \{0,1\}$
- $\delta(q_0, 0) = \{q_0\}$ $\delta(q_0, 1) = \{q_0, q_1\}$ $\delta(q_0, \varepsilon) = \varnothing$

  $\delta(q_1, 0) = \varnothing$ $\delta(q_1, 1) = \varnothing$ $\delta(q_1, \varepsilon) = \{q_0\}$
- q_0 in Q is the start state
- F = ?



- Example: above NFA is 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
- $Q = \{ q_0, q_1\}$
- $\Sigma = \{0,1\}$
- $\delta(q_0 ,0) = \{q_0\}$ $\delta(q_0 ,1) = \{q_0, q_1\}$ $\delta(q_0 ,\varepsilon) = \emptyset$

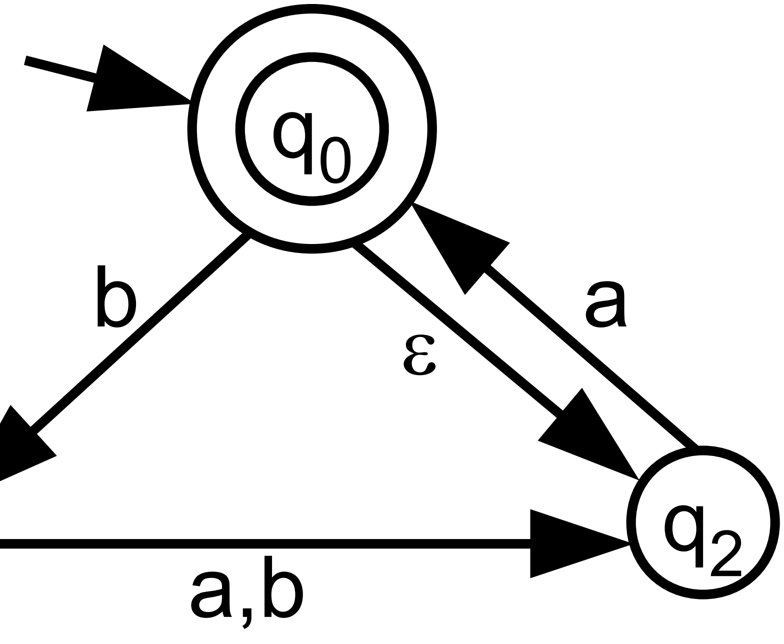
  $\delta(q_1 ,0) = \emptyset$ $\delta(q_1 ,1) = \emptyset$ $\delta(q_1 ,\varepsilon) = \{q_0\}$
- q_0 in Q is the start state
- $F = \{ q_1\} \subseteq Q$ is the set of accept states

- Definition: A NFA $(Q, \Sigma, \delta, q_0, F)$ accepts a string w if
  $\exists$ integer k, $\exists$ k strings $w_1, w_2, \ldots, w_k$ such that
- $w = w_1 w_2 \ldots w_k$ where $\forall\ 1 \le i \le k$, $w_i \in \Sigma \cup \{\varepsilon\}$
  (the symbols of w, or ε)

- $\exists$ sequence of k+1 states $r_0, r_1, .., r_k$ in Q such that:
- $r_0 = q_0$
- $r_{i+1} \in \delta(r_i, w_{i+1})\ \forall\ 0 \le i < k$
- r_k is in F

- Differences with DFA are in green

Back to first example NFA:



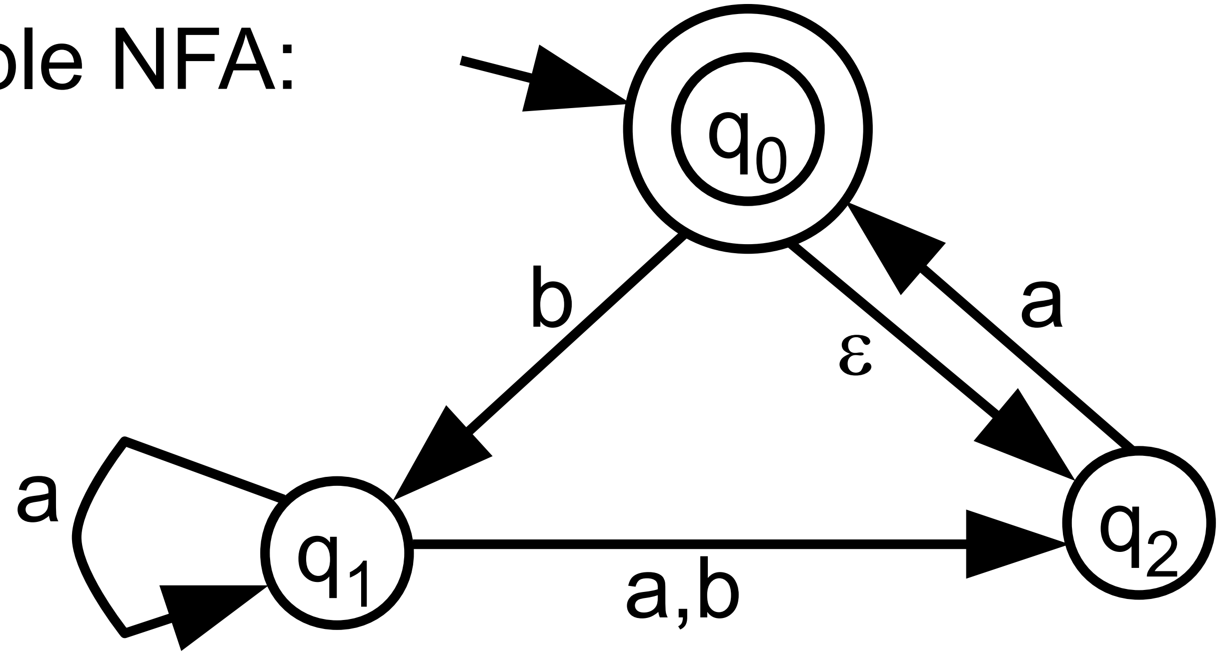
Accepts w = baaa

w_1 = b, w_2 = a, w_3 = a, w_4= ε, w_5 = a

Accepting sequence of 5+1 = 6 states:

r_0 = ?

Back to first example NFA:



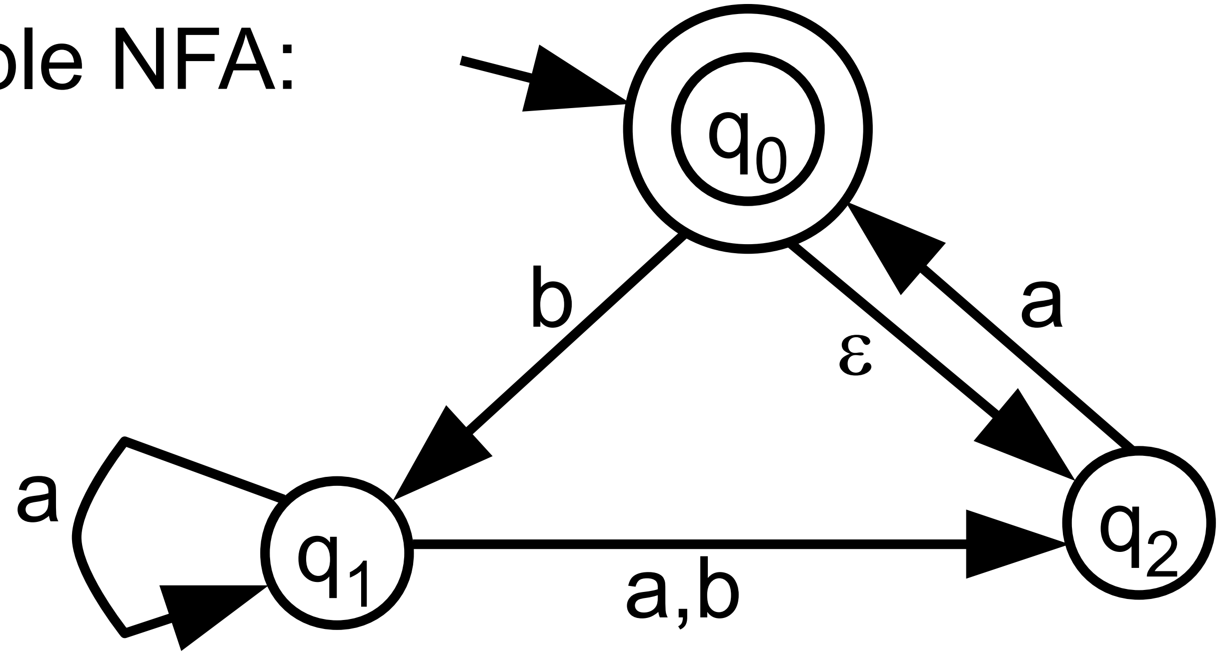
Accepts w = baaa

$w_1 = b$, $w_2 = a$, $w_3 = a$, $w_4 = \varepsilon$, $w_5 = a$

Accepting sequence of 5+1 = 6 states:

$r_0 = q_0$, $r_1 =$?

Back to first example NFA:



Accepts w = baaa

$w_1 = b$, $w_2 = a$, $w_3 = a$, $w_4 = \varepsilon$, $w_5 = a$

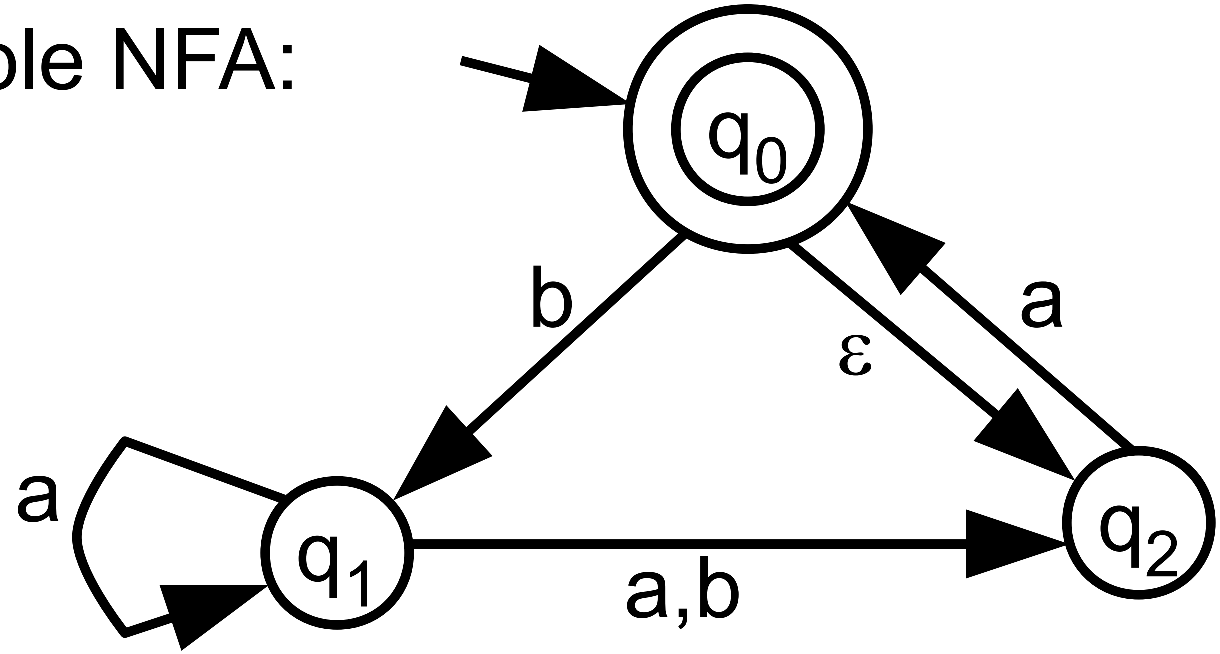
Accepting sequence of 5+1 = 6 states:

$r_0 = q_0$, $r_1 = q_1$, $r_2 =$?

Transitions:

$r_1 \in \delta(r_0, b) = \{q_1\}$

Back to first example NFA:



Accepts w = baaa

$w_1 = b$, $w_2 = a$, $w_3 = a$, $w_4 = \varepsilon$, $w_5 = a$

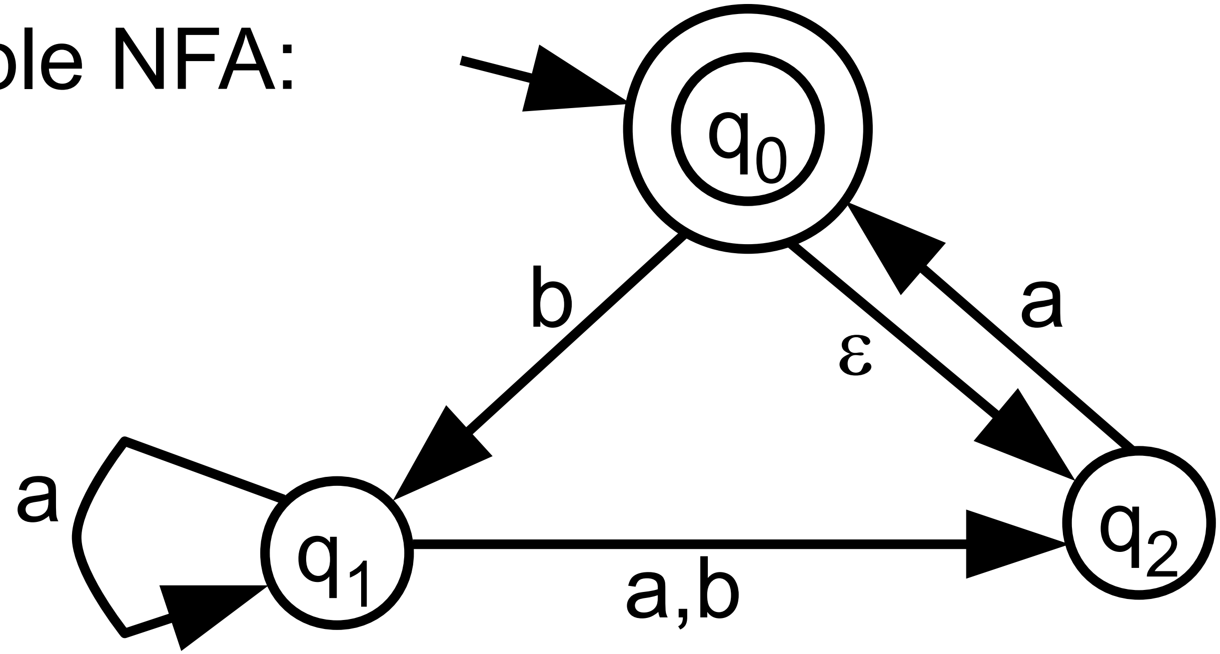
Accepting sequence of 5+1 = 6 states:

$r_0 = q_0$, $r_1 = q_1$, $r_2 = q_2$, $r_3 =$?

Transitions:

$r_1 \in \delta(r_0,b) = \{q_1\}$ $r_2 \in \delta(r_1,a) = \{q_1,q_2\}$

Back to first example NFA:



Accepts w = baaa

$w_1 = b$, $w_2 = a$, $w_3 = a$, $w_4 = \varepsilon$, $w_5 = a$

Accepting sequence of 5+1 = 6 states:

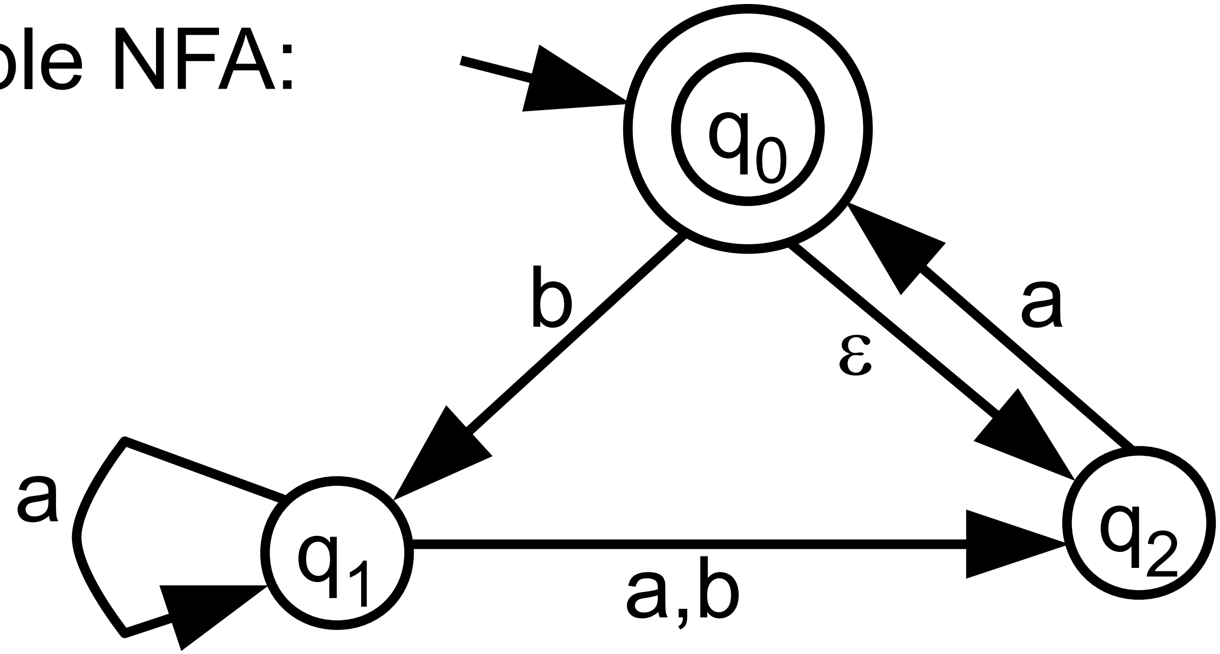
$r_0 = q_0$, $r_1 = q_1$, $r_2 = q_2$, $r_3 = q_0$, $r_4 =$?

Transitions:

$r_1 \in \delta(r_0,b) = \{q_1\}$ $r_2 \in \delta(r_1,a) = \{q_1,q_2\}$

$r_3 \in \delta(r_2,a) = \{q_0\}$

Back to first example NFA:



Accepts w = baaa

$w_1 = b, \quad w_2 = a, \quad w_3 = a, \quad w_4 = \varepsilon, \quad w_5 = a$

Accepting sequence of 5+1 = 6 states:

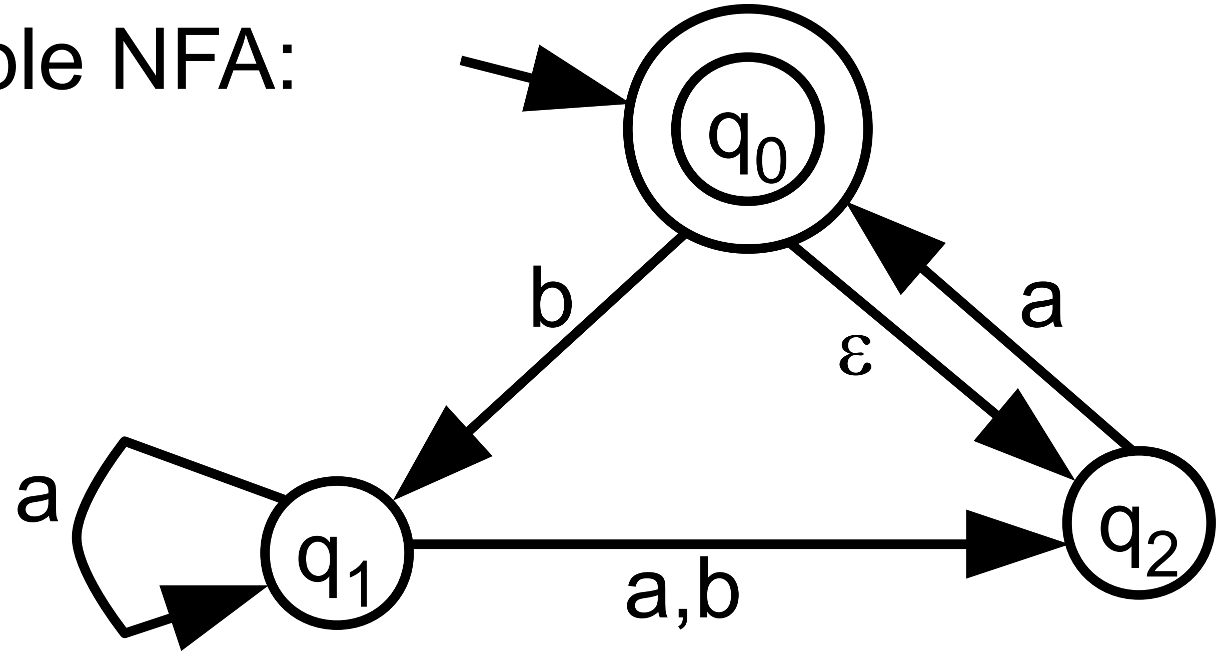
$r_0 = q_0, \quad r_1 = q_1, \quad r_2 = q_2, \quad r_3 = q_0, \quad r_4 = q_2, \quad r_5 =$?

Transitions:

$r_1 \in \delta(r_0,b) = \{q_1\} \quad r_2 \in \delta(r_1,a) = \{q_1,q_2\}$

$r_3 \in \delta(r_2,a) = \{q_0\} \quad r_4 \in \delta(r_3,\varepsilon) = \{q_2\}$

Back to first example NFA:



Accepts w = baaa

$w_1 = b, \quad w_2 = a, \quad w_3 = a, \quad w_4 = \varepsilon, \quad w_5 = a$

Accepting sequence of 5+1 = 6 states:

$r_0 = q_0, \quad r_1 = q_1, \quad r_2 = q_2, \quad r_3 = q_0, \quad r_4 = q_2, \quad r_5 = q_0$

Transitions:

$r_1 \in \delta(r_0, b) = \{q_1\} \quad r_2 \in \delta(r_1, a) = \{q_1, q_2\}$

$r_3 \in \delta(r_2, a) = \{q_0\} \quad r_4 \in \delta(r_3, \varepsilon) = \{q_2\} \quad r_5 \in \delta(r_4, a) = \{q_0\}$

- NFA are at least as powerful as DFA,
  because DFA are a special case of NFA

- Are NFA more powerful than DFA?

- Surprisingly, they are not:

- Theorem:

For every NFA N there is DFA M : L(M) = L(N)

- Theorem:

For every NFA N there is DFA M : L(M) = L(N)

- Construction without ε transitions
- Given NFA N $(Q, \Sigma, \delta, q, F)$
- Construct DFA M $(Q', \Sigma, \delta', q', F')$ where:
- $Q' :=$ Powerset(Q)
- $q' = \{q\}$
- $F' = \{ S : S \in Q'$ and S contains an element of F$\}$
- $\delta'(S, a) := \cup_{s \in S} \delta(s,a)$

$$= \{ t : t \in \delta(s,a) \text{ for some } s \in S \}$$

- It remains to deal with ε transitions

- Definition: Let S be a set of states.

E(S) := { q : q can be reached from some state
    s in S traveling along 0 or more ε transitions }

- We think of following ε transitions at beginning, or right after reading an input symbol in Σ

- Theorem:

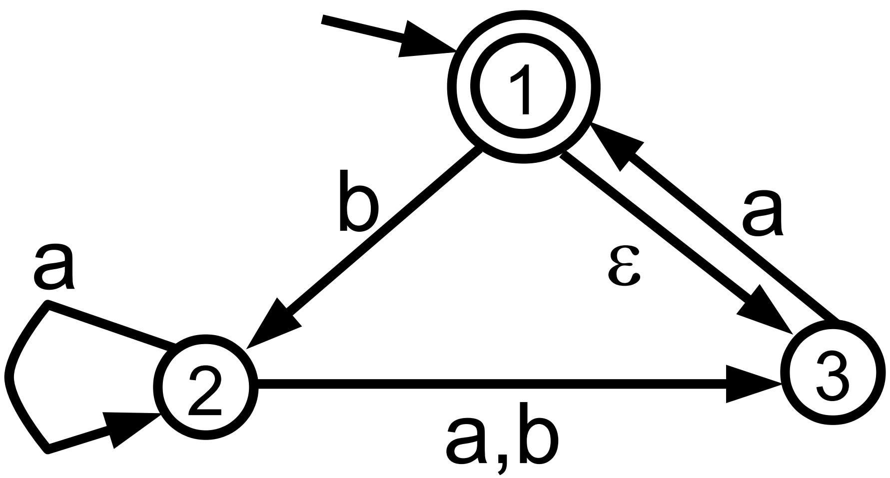
For every NFA N there is DFA M : L(M) = L(N)

- Construction including ε transitions
- Given NFA N $(Q, \Sigma, \delta, q, F)$
- Construct DFA M $(Q', \Sigma, \delta', q', F')$ where:
- $Q' :=$ Powerset(Q)
- $q' = E(\{q\})$
- $F' = \{ S : S \in Q'$ and S contains an element of $F\}$
- $\delta'(S, a) := E(\cup_{s \in S} \delta(s,a))$

$$= \{ t : t \in E(\delta (s,a)) \text{ for some } s \in S \}$$

Example: NFA → DFA conversion

NFA

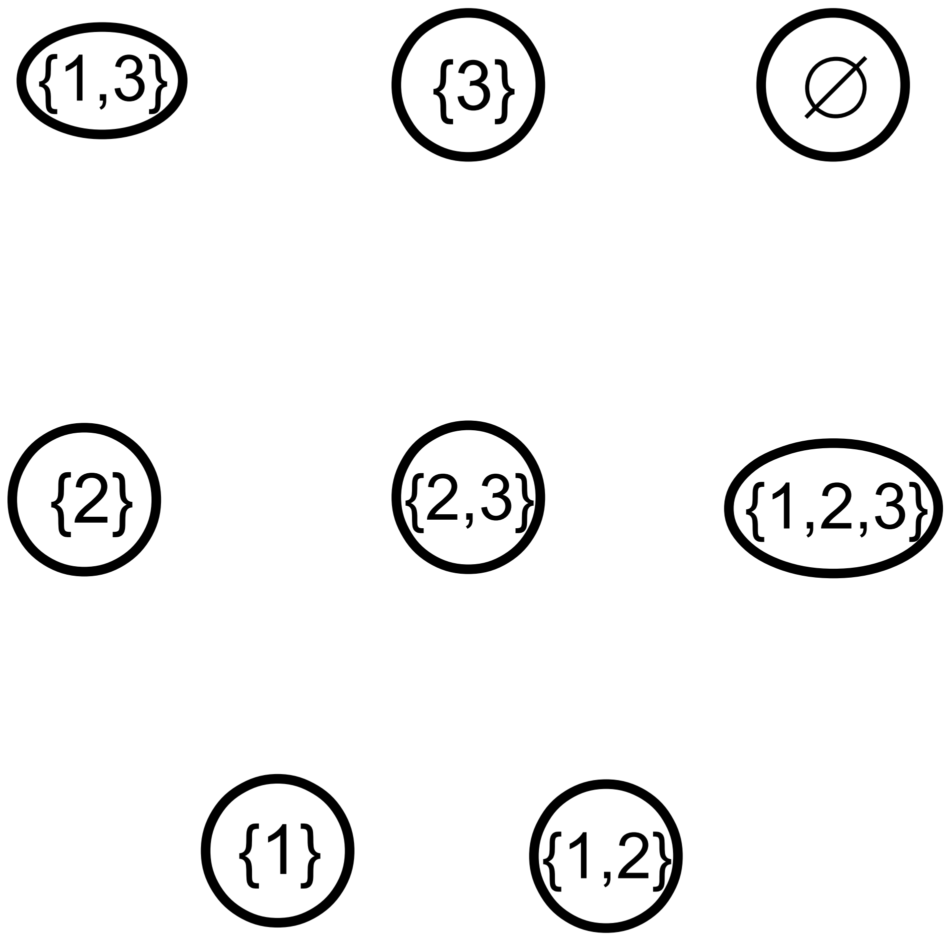


Q_{DFA} = Powerset(Q_{NFA})

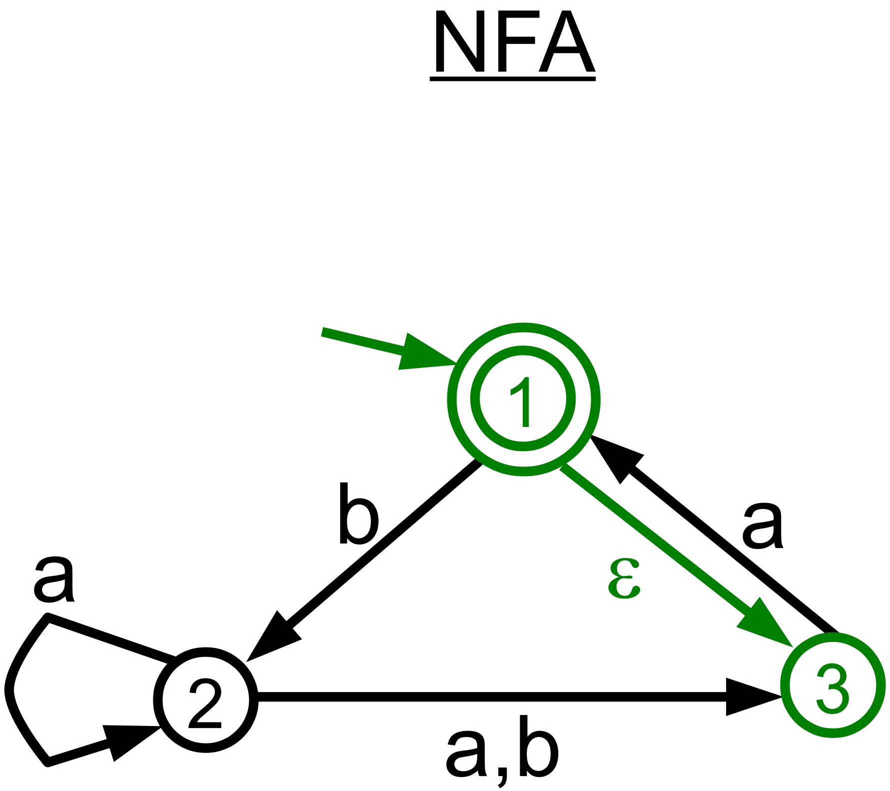
= Powerset({1,2,3})

= {∅,{1},{2},{3},{1,2}...}

DFA



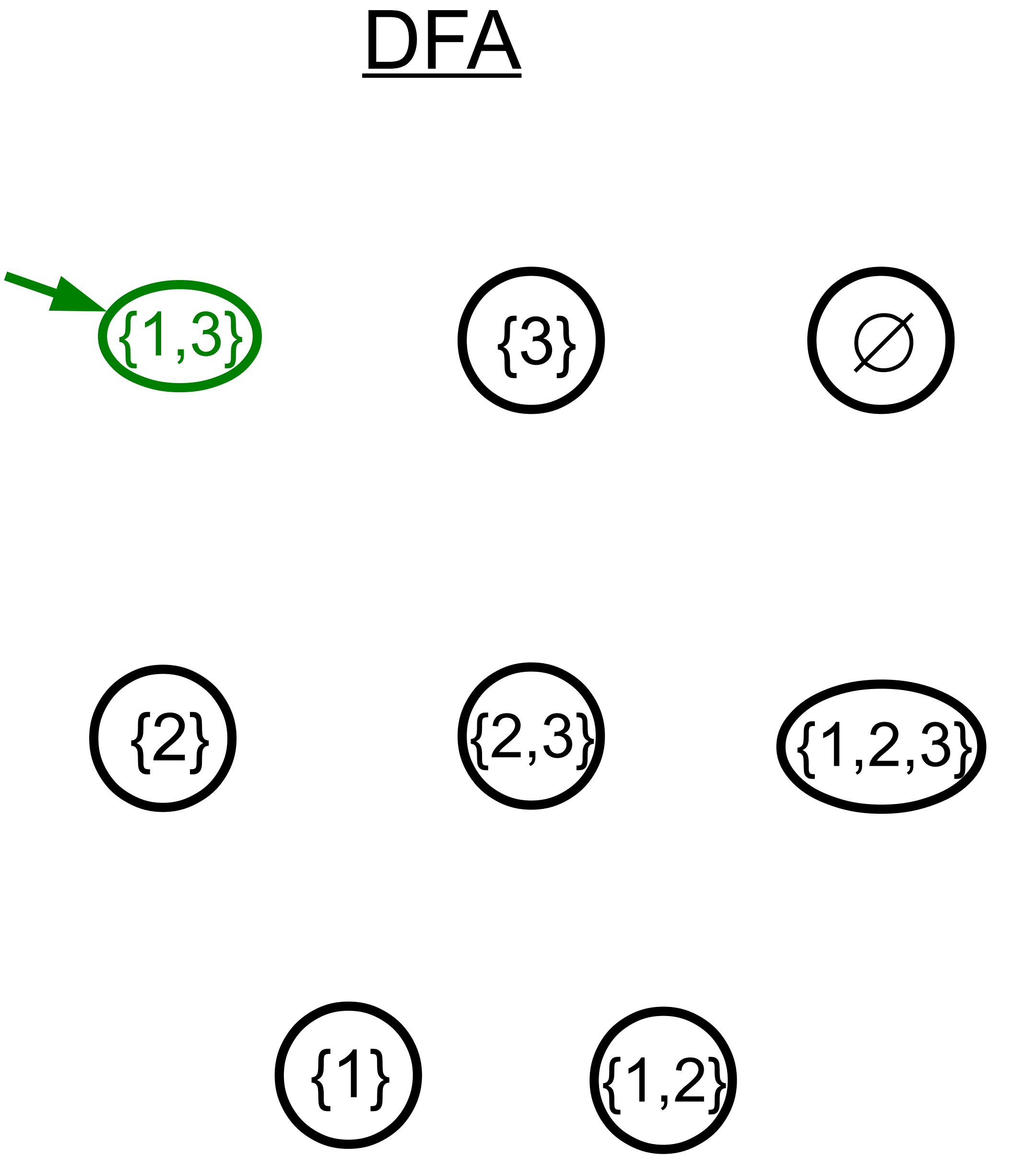
Example: NFA → DFA conversion



$q_{DFA} = E(\{q_{NFA}\})$

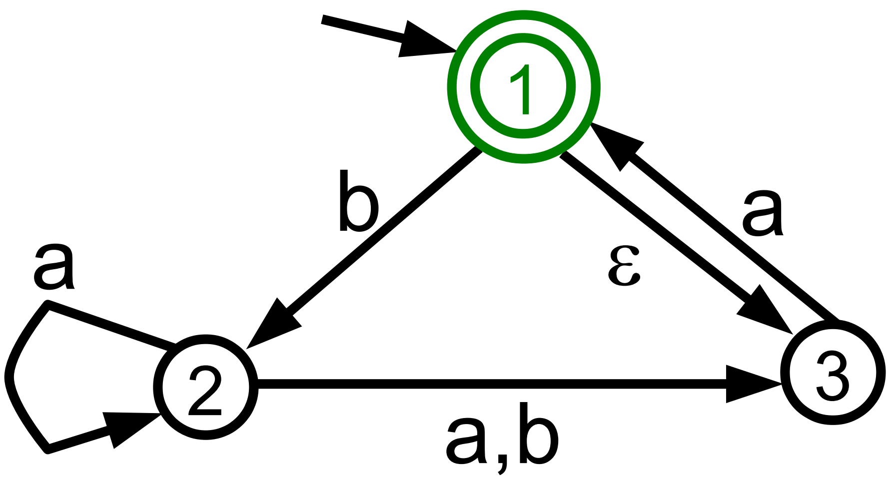
$= E(\{1\})$

$= \{1,3\}$

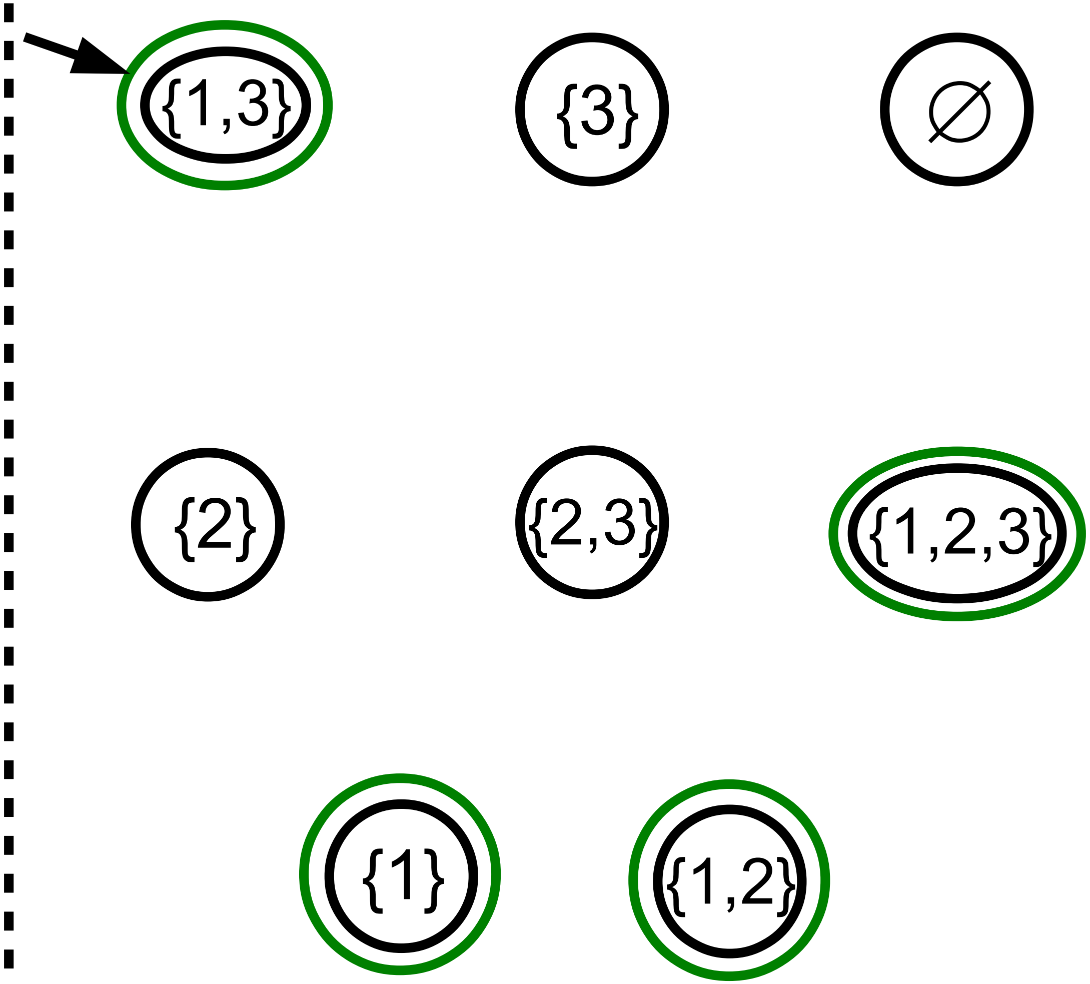


Example: NFA → DFA conversion

NFA



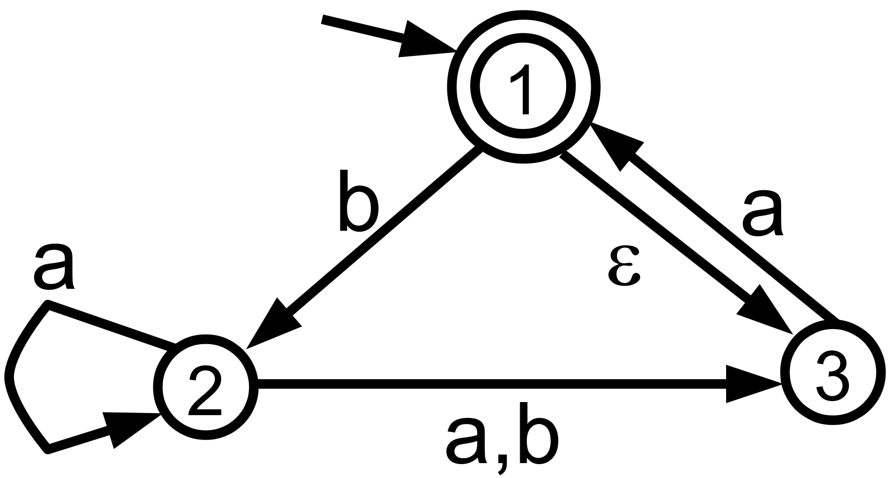
DFA



F_{DFA} = {S : S contains an element of F_{NFA}}

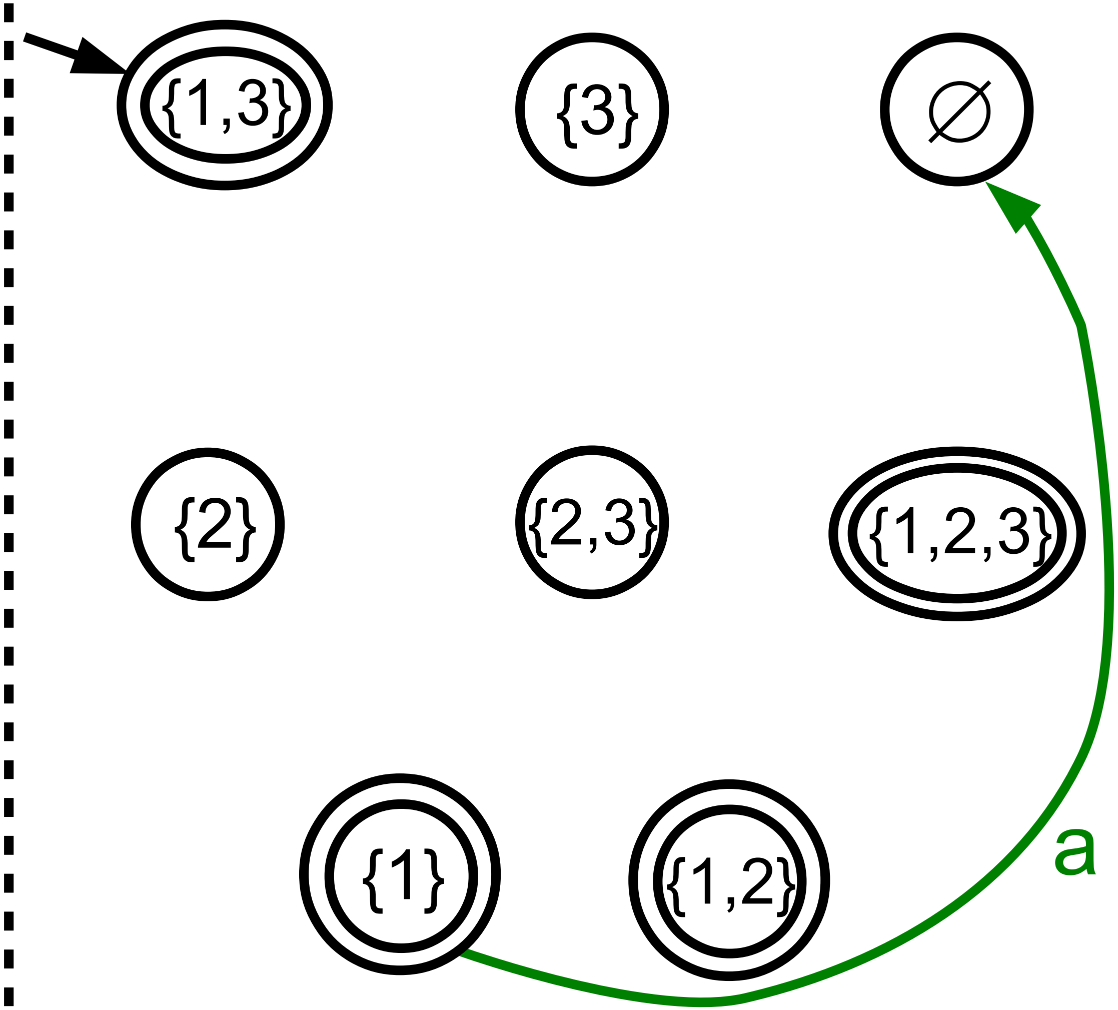
Example: NFA → DFA conversion

NFA

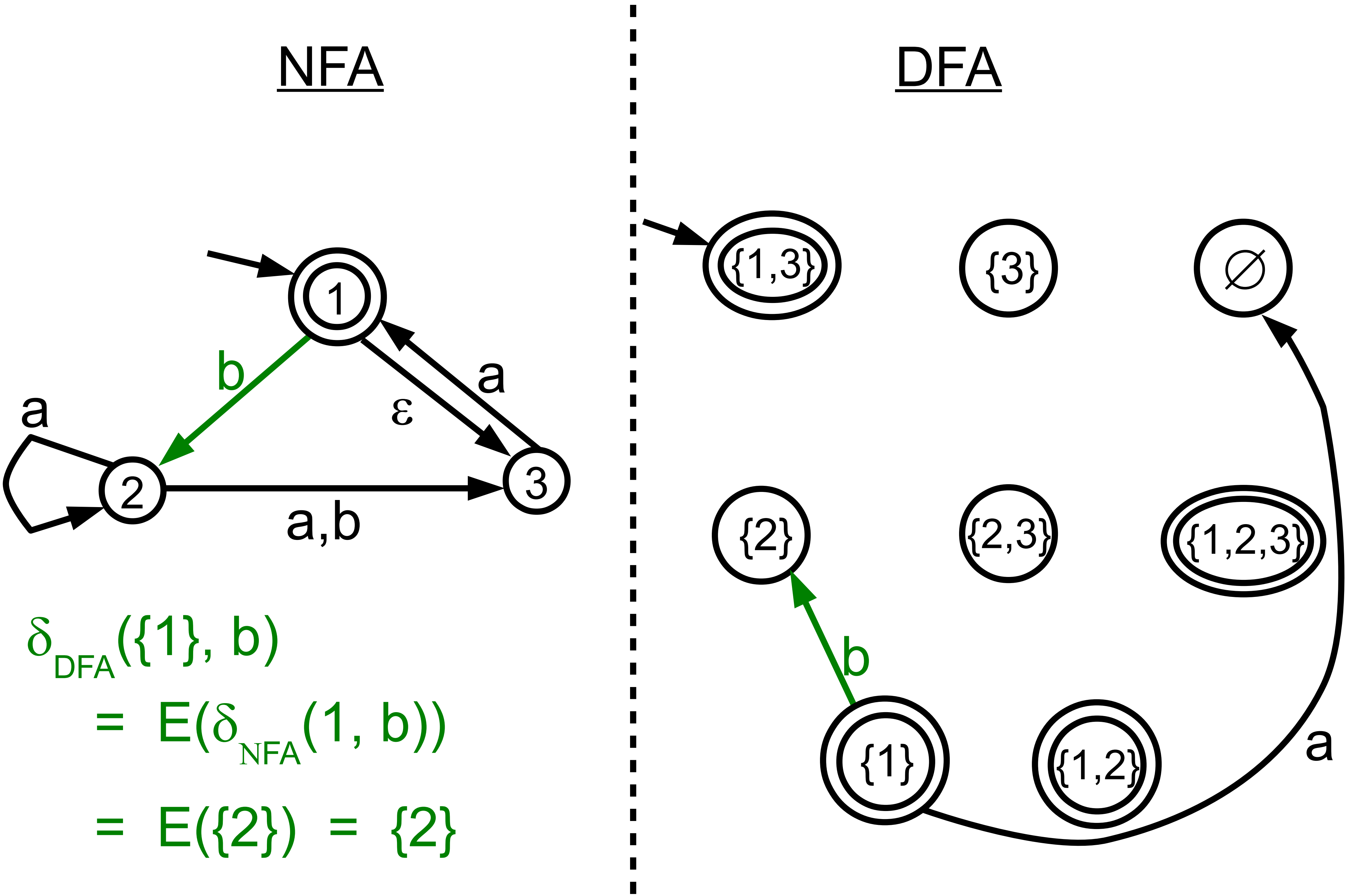


$$\delta_{DFA}(\{1\}, a)$$
$$= E(\delta_{NFA}(1, a))$$
$$= E(\varnothing) = \varnothing$$

DFA



Example: NFA → DFA conversion



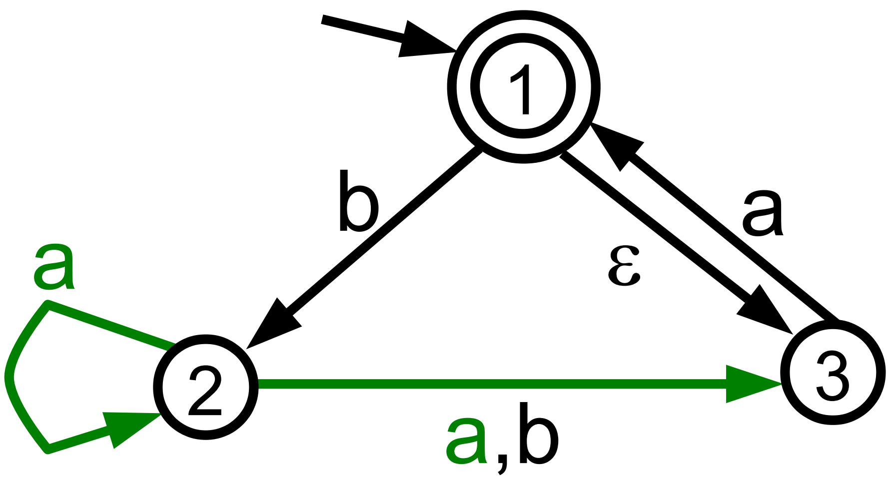
$\delta_{DFA}(\{1\}, b)$

$= E(\delta_{NFA}(1, b))$

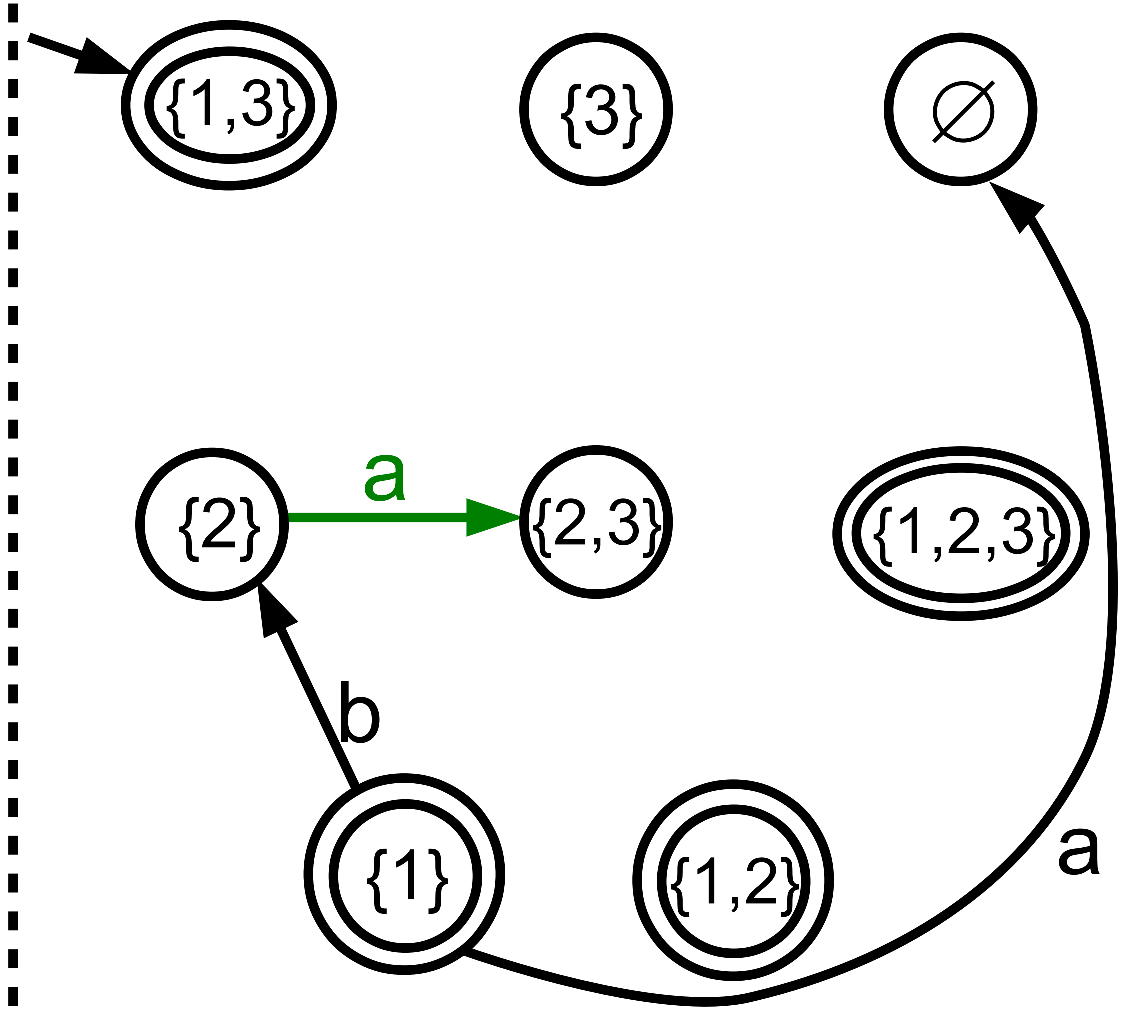
$= E(\{2\}) = \{2\}$

Example: NFA → DFA conversion

NFA



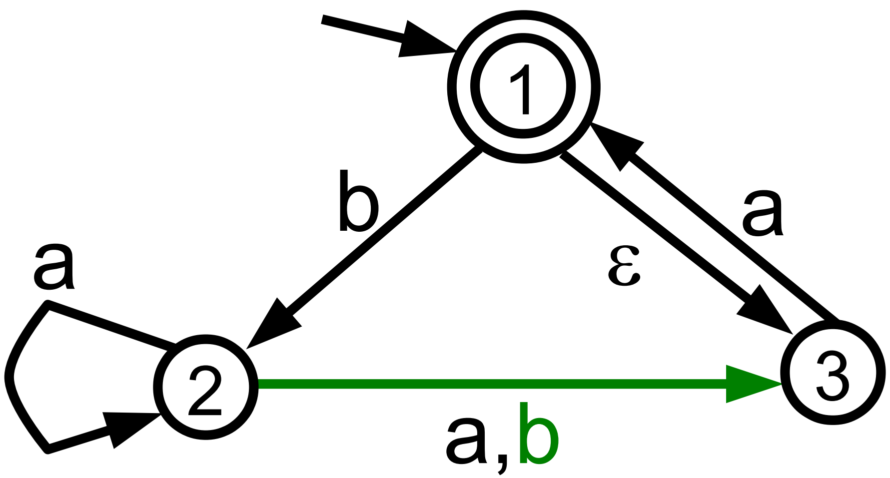
DFA



$$\delta_{DFA}(\{2\}, a)$$
$$= E(\delta_{NFA}(2, a))$$
$$= E(\{2,3\}) = \{2,3\}$$

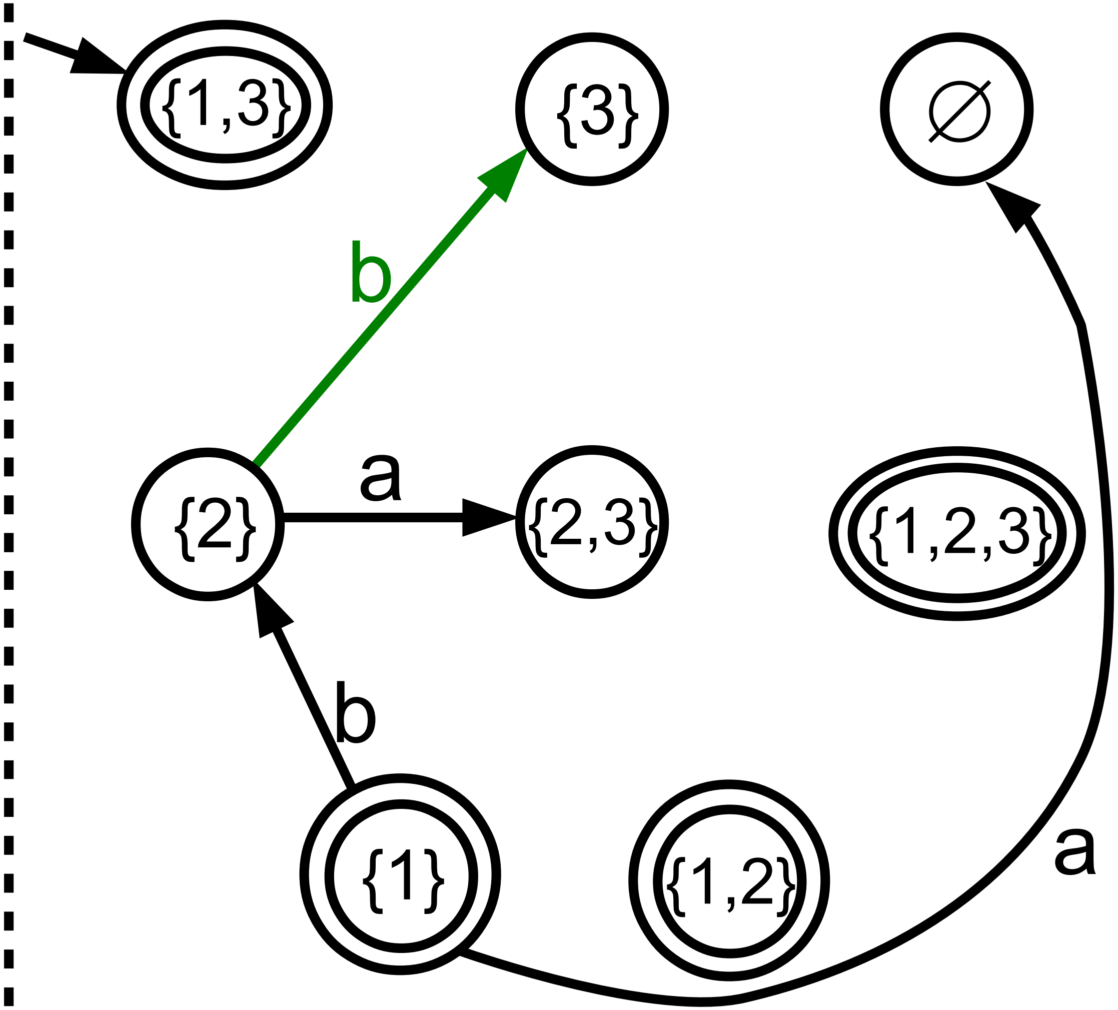
Example: NFA → DFA conversion

NFA



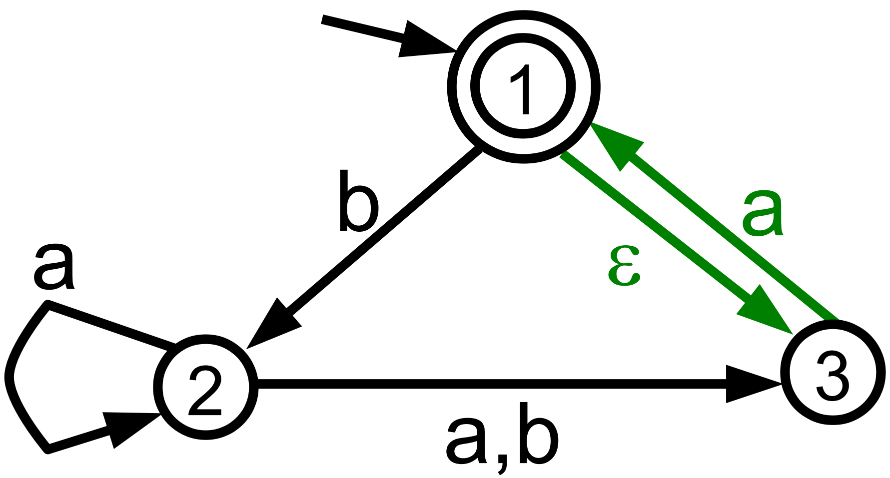
$$\delta_{DFA}(\{2\}, b)$$
$$= E(\delta_{NFA}(2, b))$$
$$= E(\{3\}) = \{3\}$$

DFA



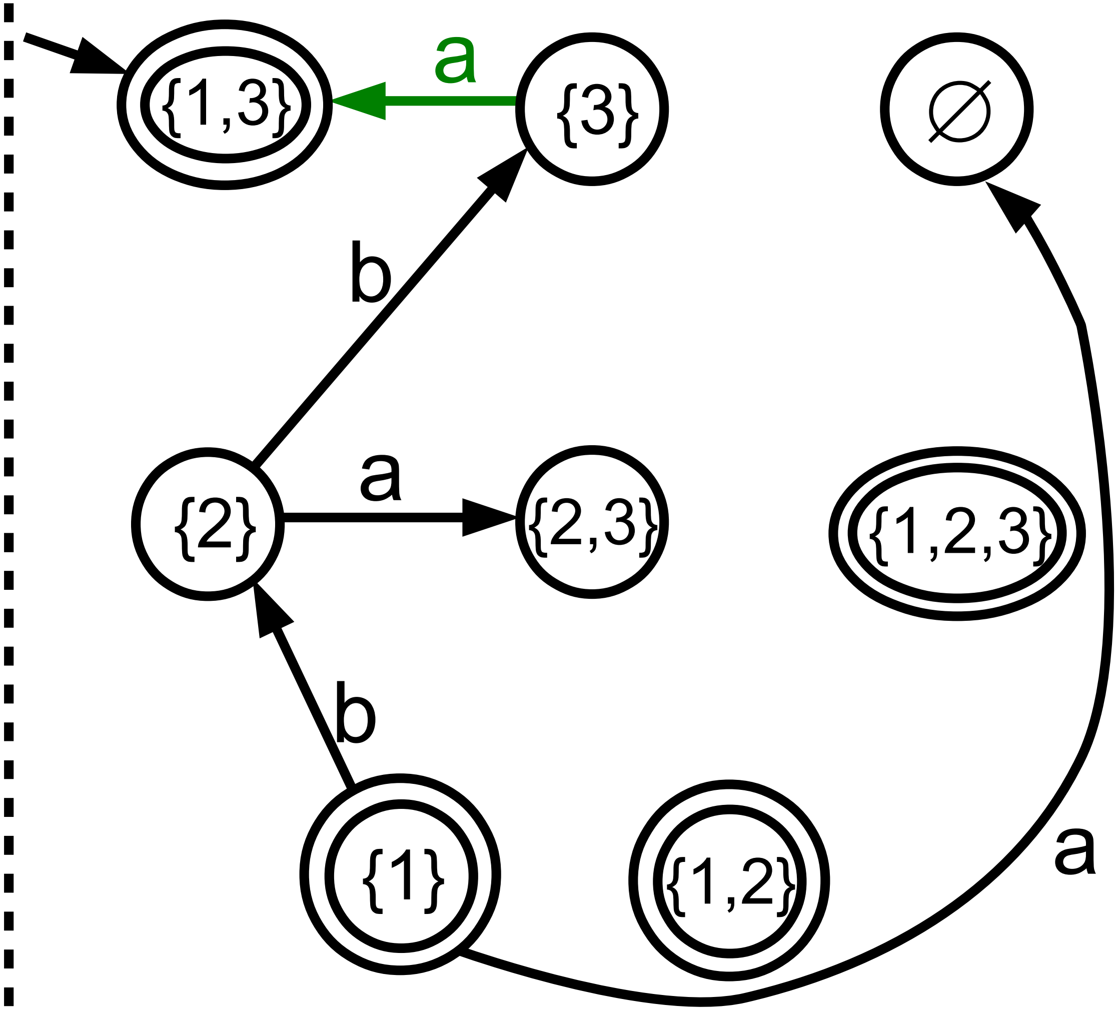
Example: NFA → DFA conversion

NFA



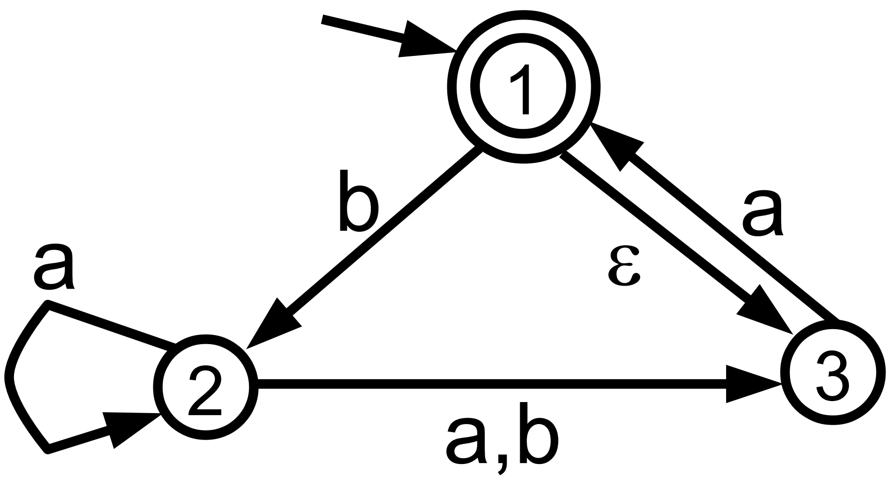
$$\delta_{DFA}(\{3\}, a)$$
$$= E(\delta_{NFA}(3, a))$$
$$= E(\{1\}) = \{1,3\}$$

DFA



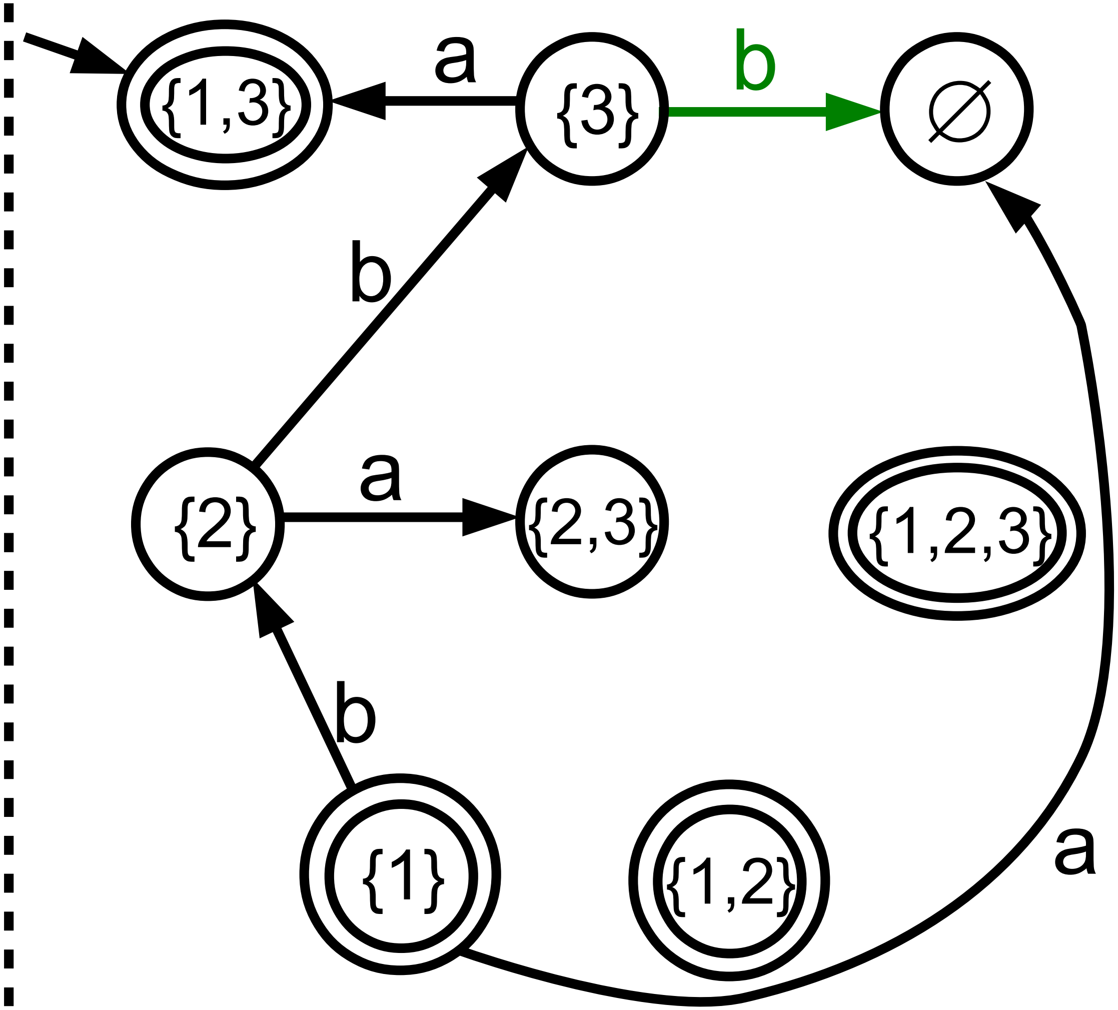
Example: NFA → DFA conversion

NFA



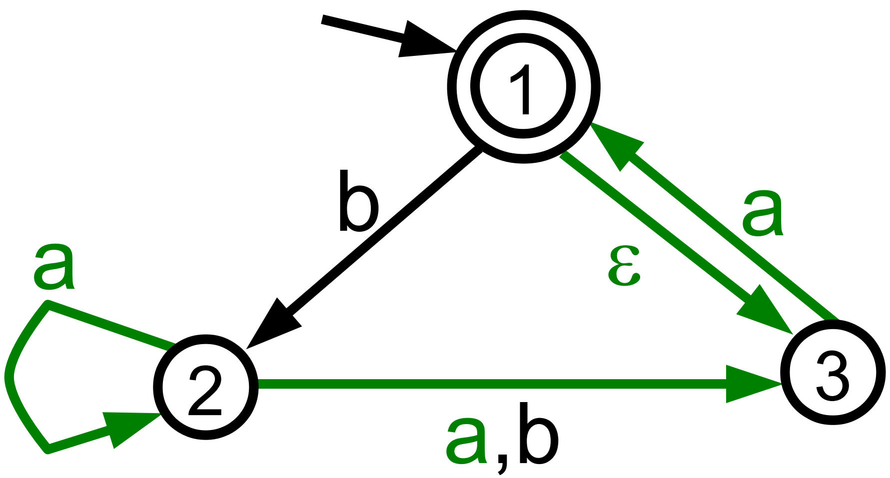
$$\delta_{DFA}(\{3\}, b)$$
$$= E(\delta_{NFA}(3, b))$$
$$= E(\varnothing) = \varnothing$$

DFA



Example: NFA → DFA conversion

NFA

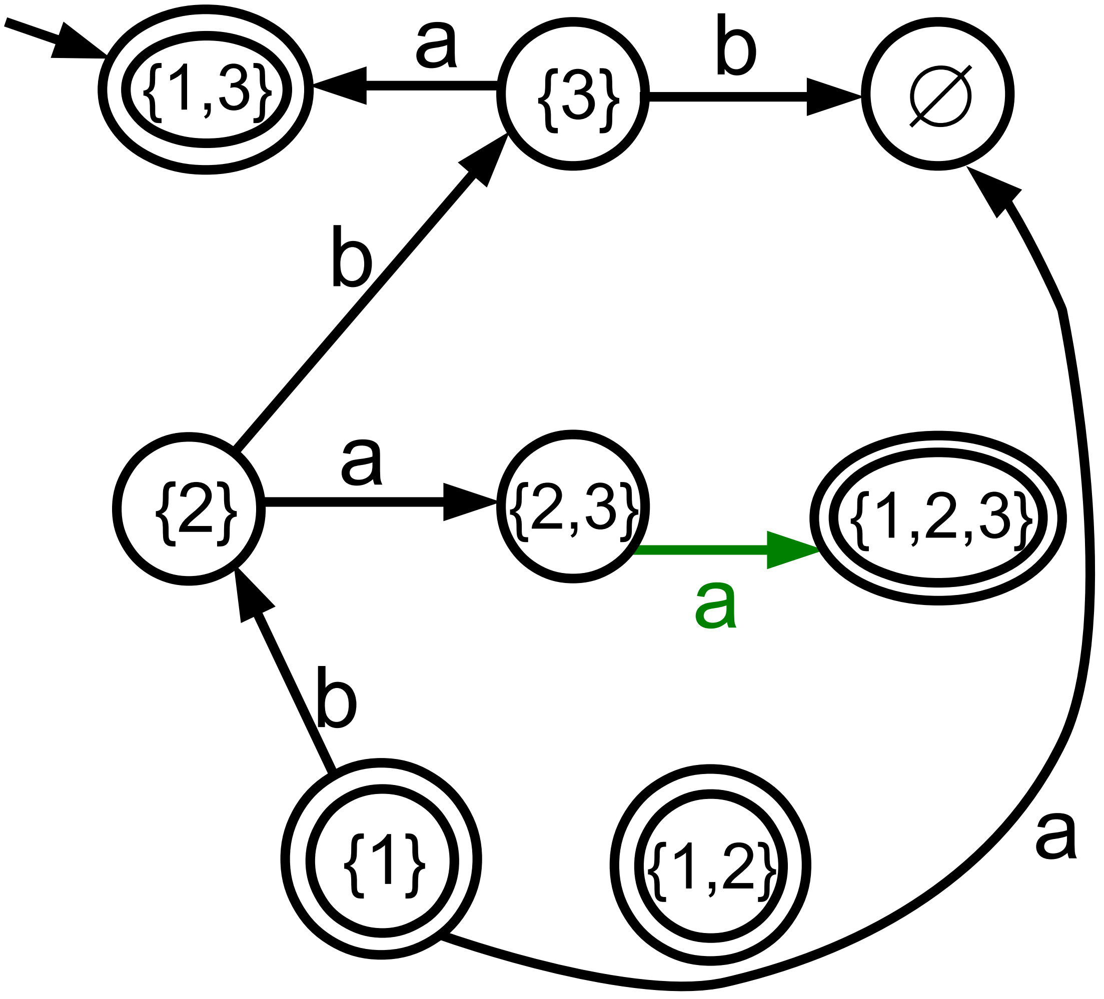


$$\delta_{DFA}(\{2,3\}, a)$$

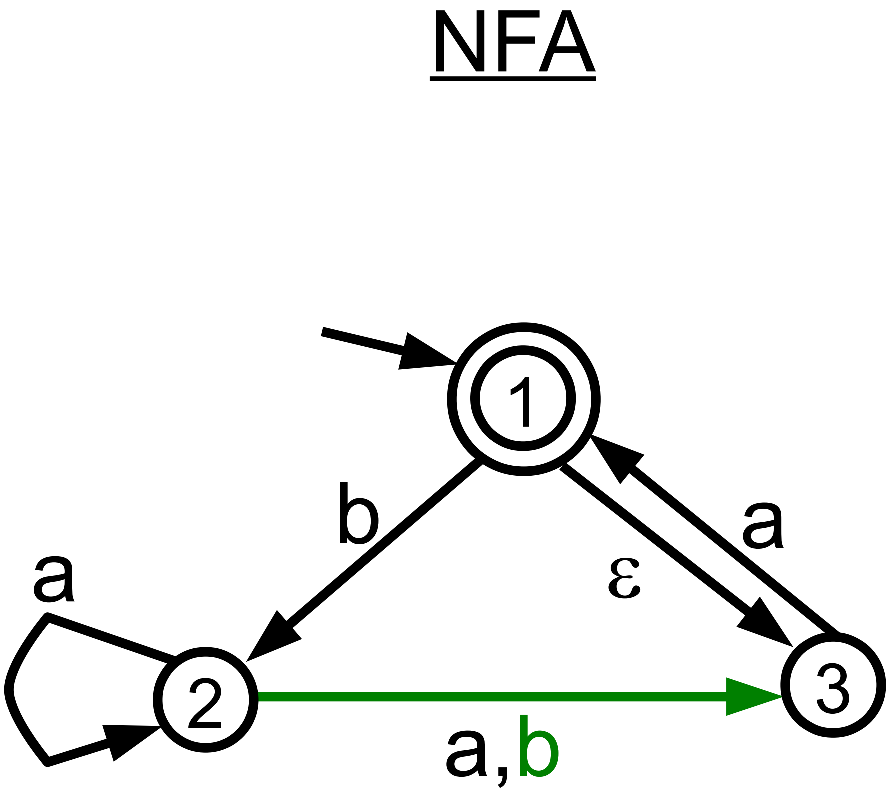
$$= E(\delta_{NFA}(2,a) \cup \delta_{NFA}(3,a))$$

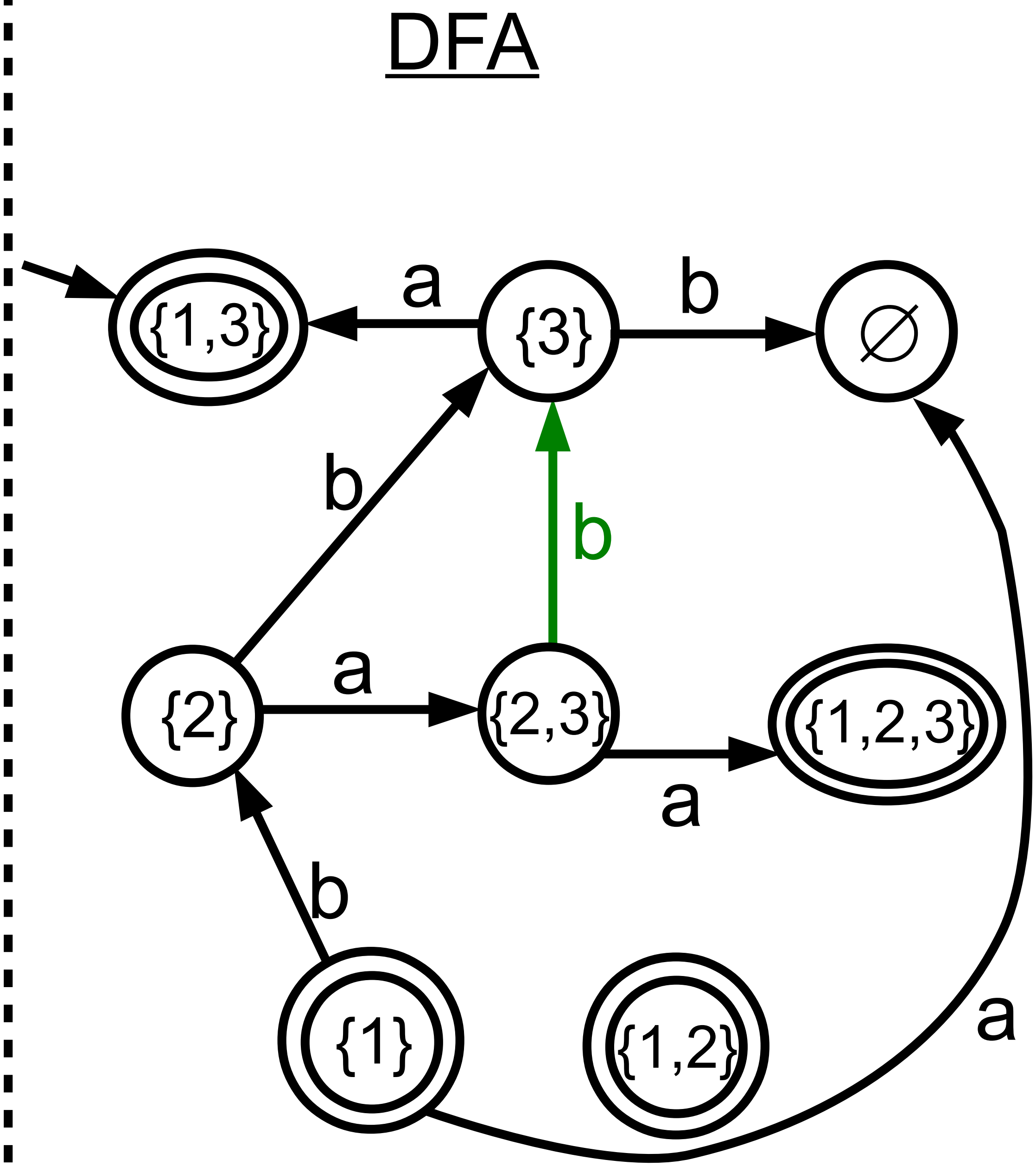
$$= E(\{2,3\} \cup \{1\}) = \{1,2,3\}$$

DFA



Example: NFA → DFA conversion



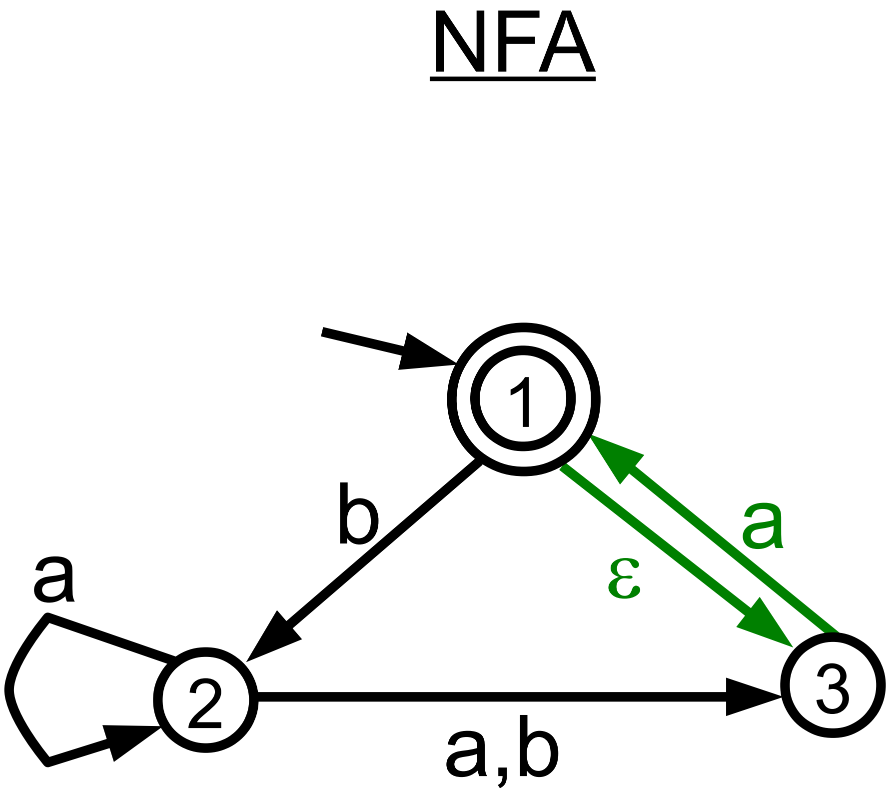


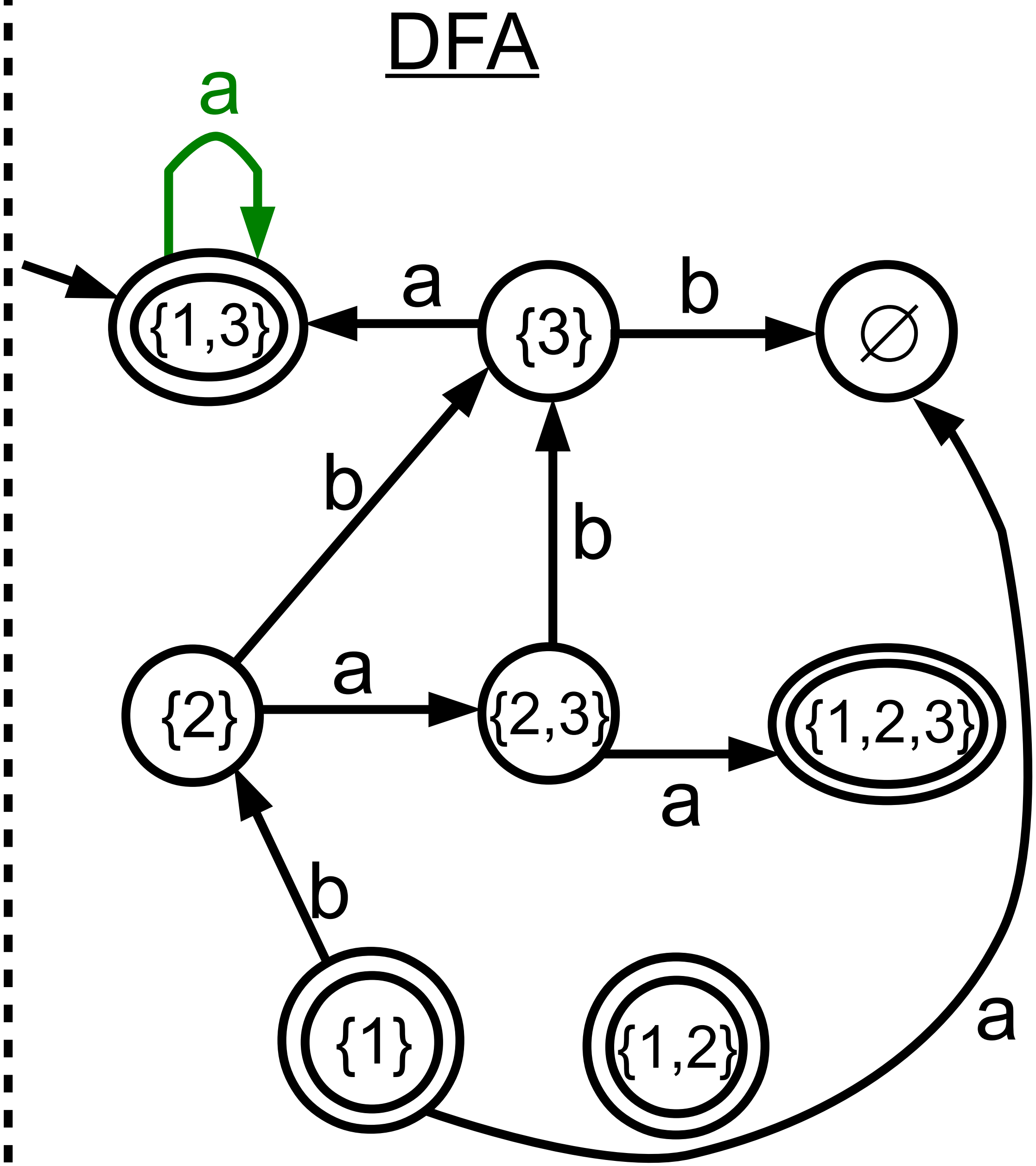
$\delta_{DFA}(\{2,3\}, b)$

$= E(\delta_{NFA}(2,b) \cup \delta_{NFA}(3,b))$

$= E(\{3\} \cup \varnothing) = \{3\}$

Example: NFA → DFA conversion



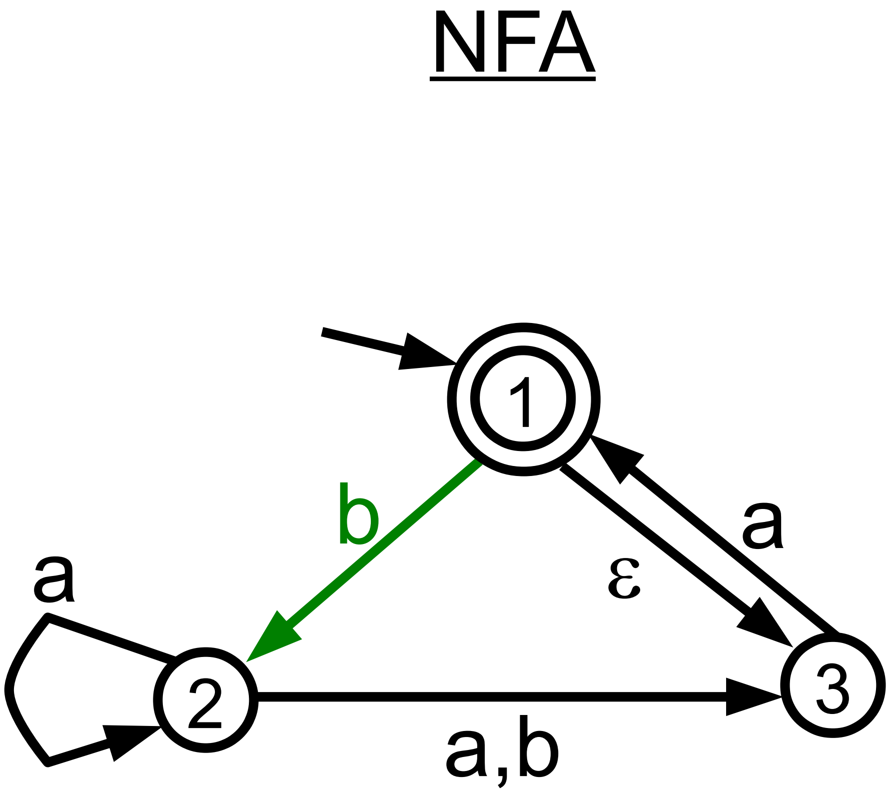


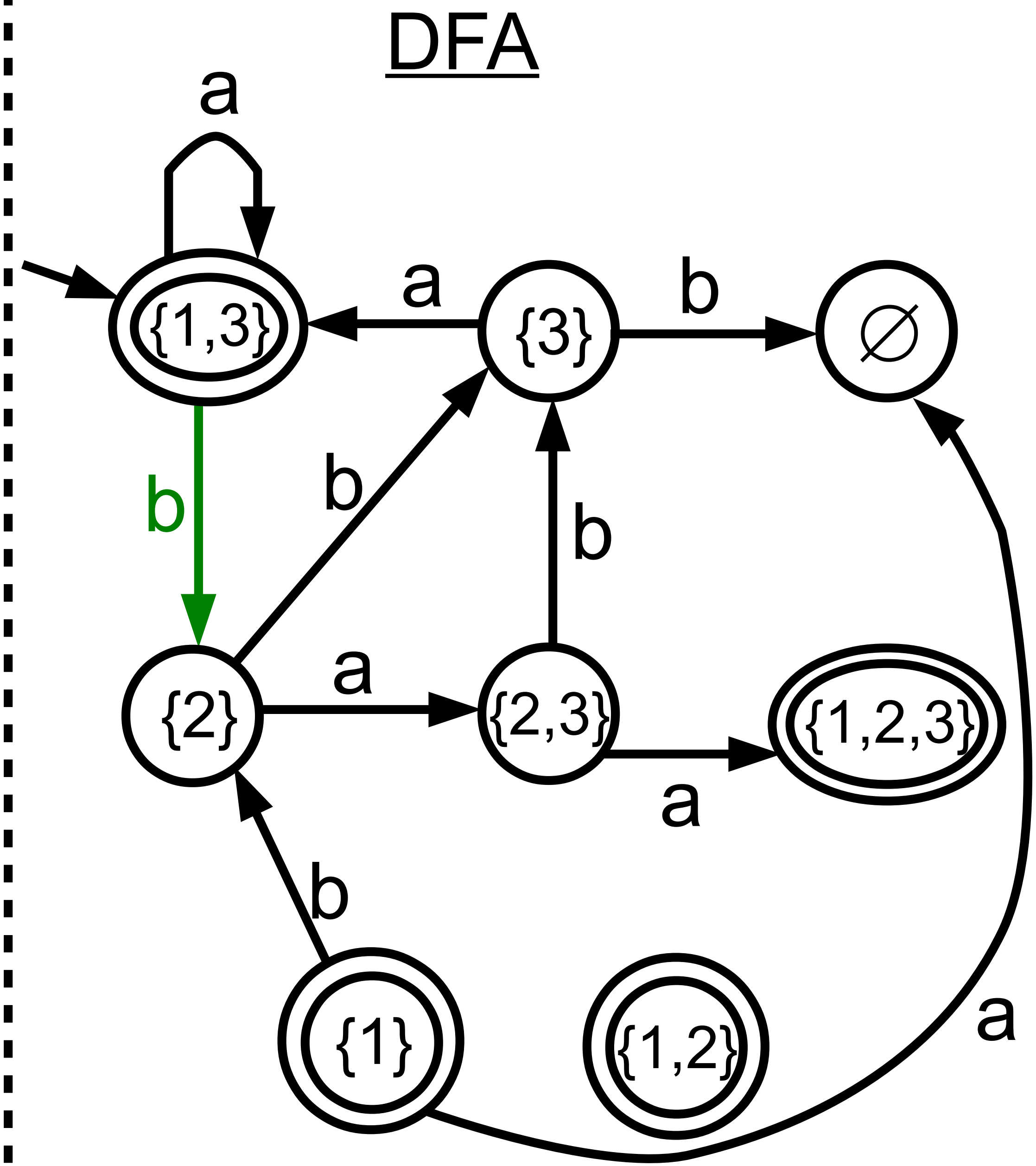
$\delta_{DFA}(\{1,3\}, a)$

$= E(\delta_{NFA}(1,a) \cup \delta_{NFA}(3,a))$

$= E(\varnothing \cup \{1\}) = \{1,3\}$

Example: NFA → DFA conversion



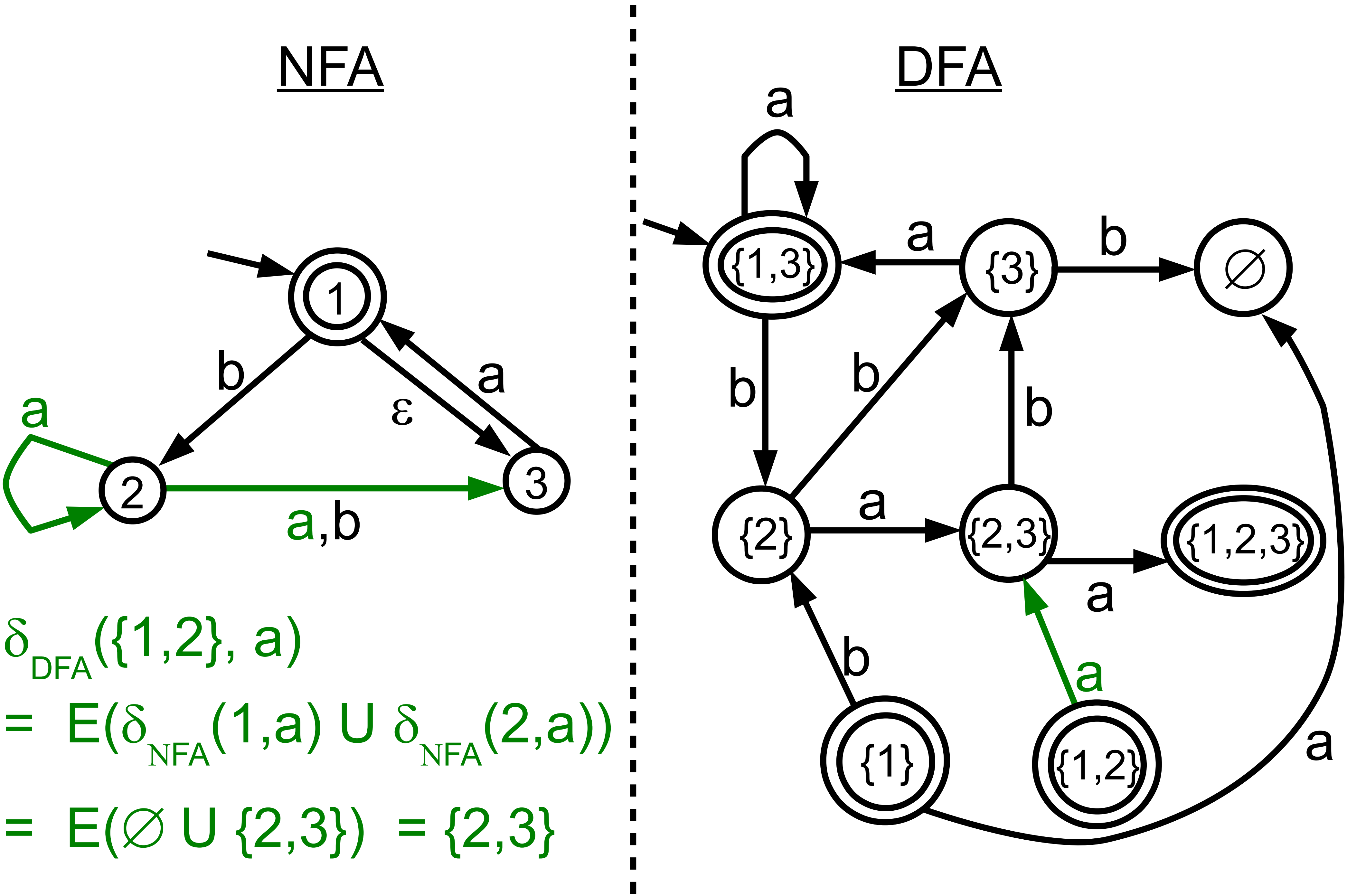


$\delta_{DFA}(\{1,3\}, b)$

$= E(\delta_{NFA}(1,b) \cup \delta_{NFA}(3,b))$

$= E(\{2\} \cup \varnothing) = \{2\}$

Example: NFA → DFA conversion



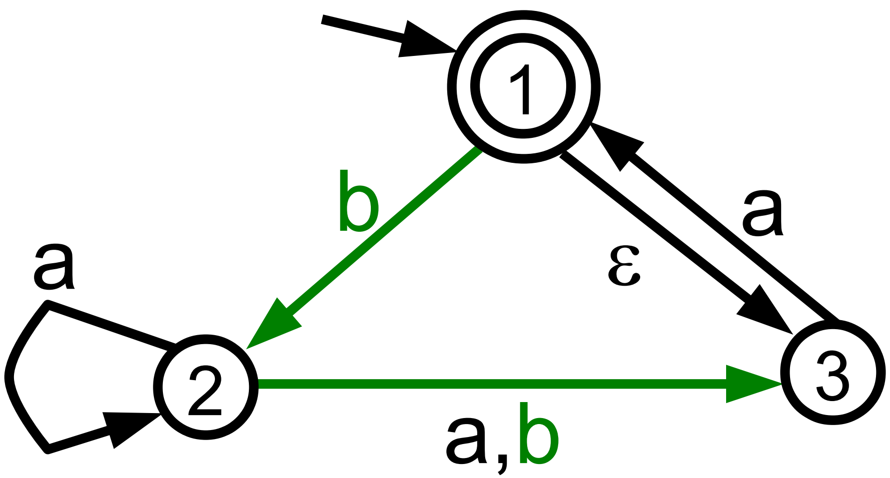
$\delta_{DFA}(\{1,2\}, a)$

$= E(\delta_{NFA}(1,a) \cup \delta_{NFA}(2,a))$

$= E(\emptyset \cup \{2,3\}) = \{2,3\}$

Example: NFA → DFA conversion

NFA

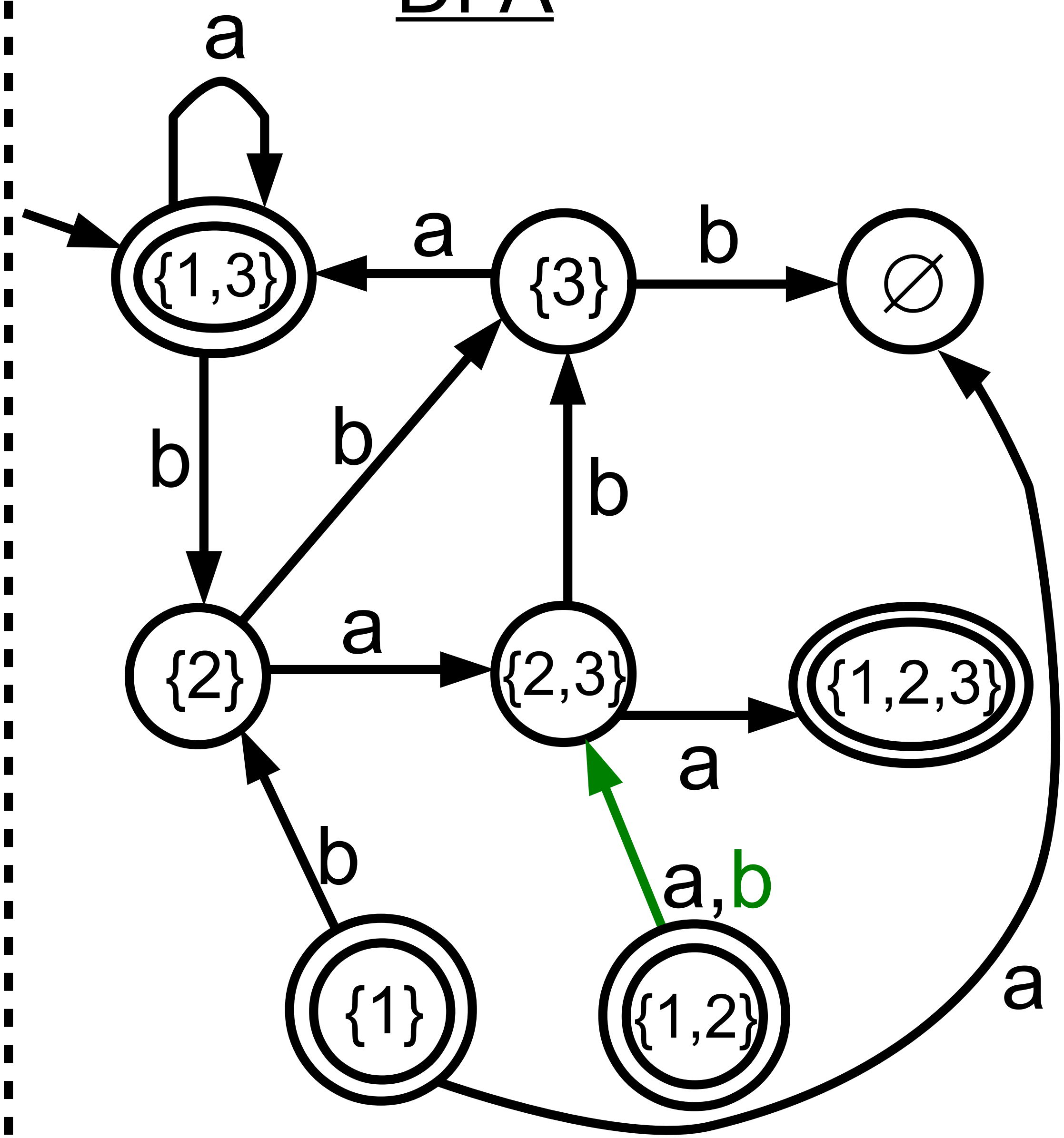


$$\delta_{DFA}(\{1,2\}, b)$$

$$= E(\delta_{NFA}(1,b) \cup \delta_{NFA}(2,b))$$

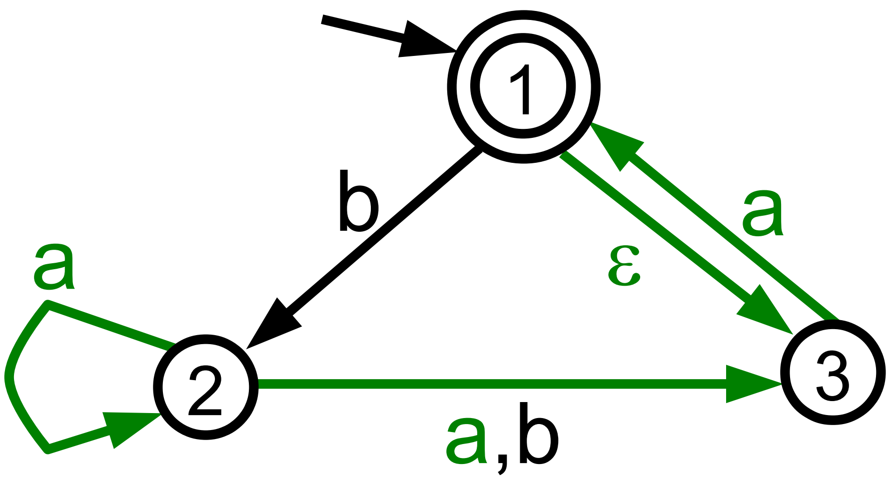
$$= E(\{2\} \cup \{3\}) = \{2,3\}$$

DFA



Example: NFA → DFA conversion

NFA

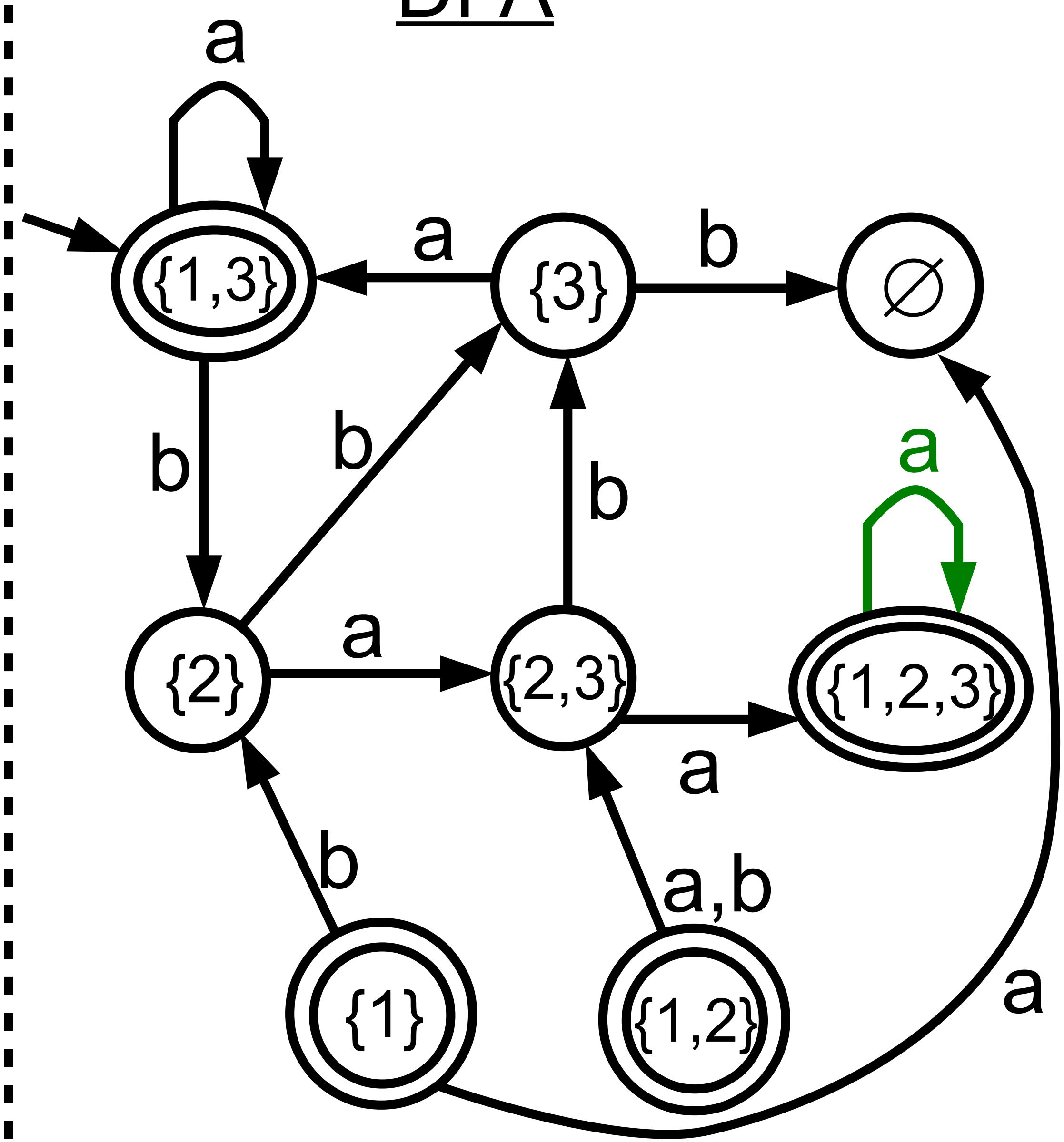


$$\delta_{DFA}(\{1,2,3\}, a)$$

$$= E(\delta_{NFA}(1,a) \cup \delta_{NFA}(2,a) \cup \delta_{NFA}(3,a))$$

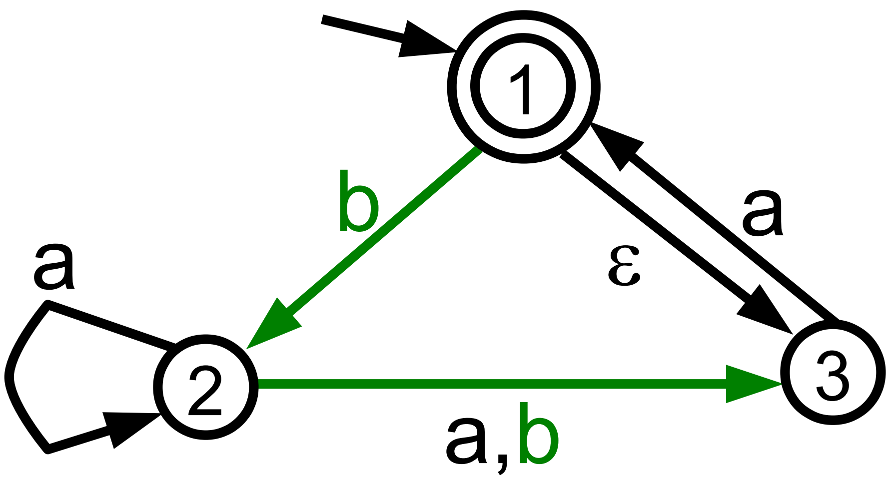
$$= E(\varnothing \cup \{2,3\} \cup \{1\}) = \{1,2,3\}$$

DFA

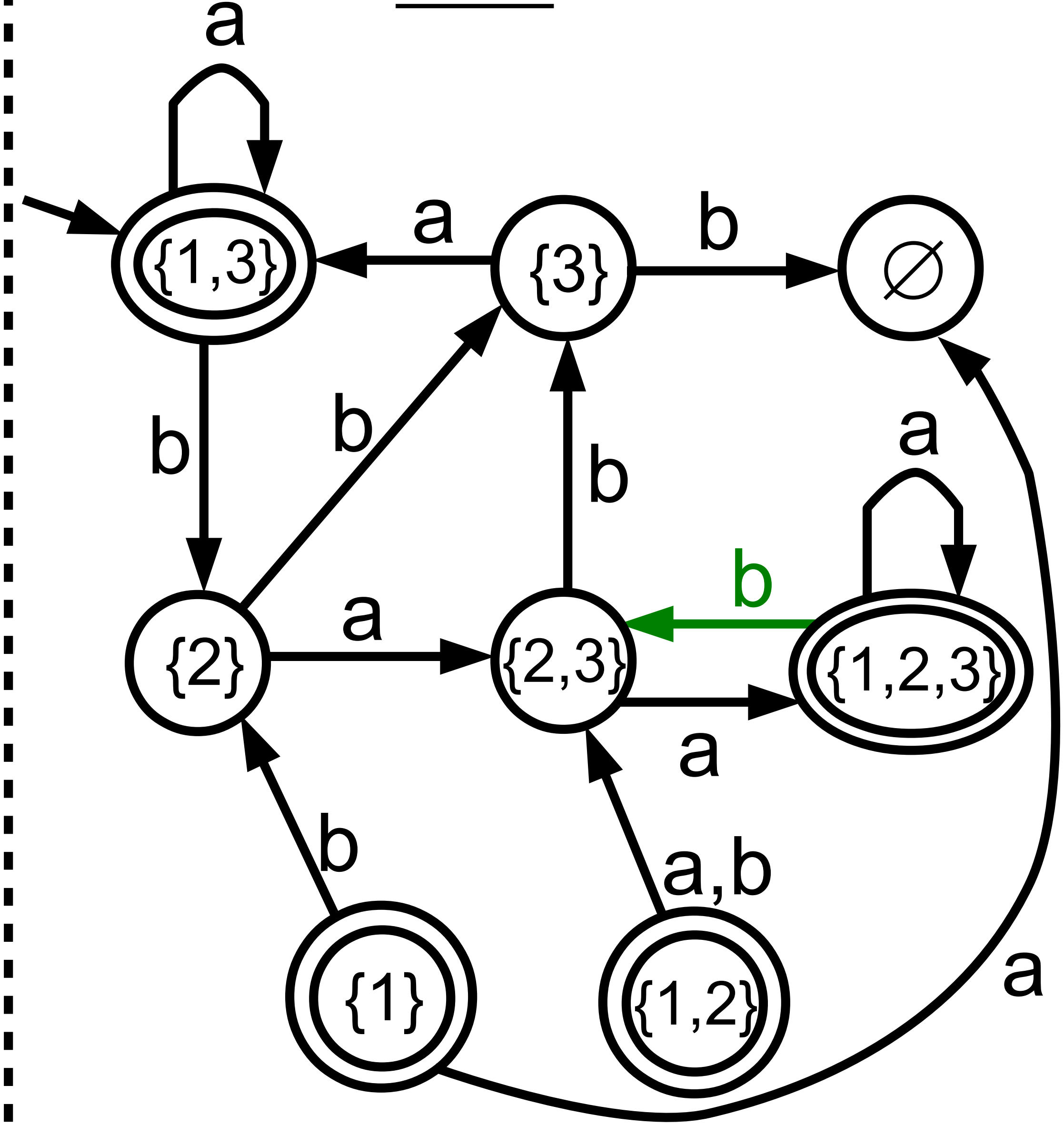


Example: NFA → DFA conversion

NFA



DFA

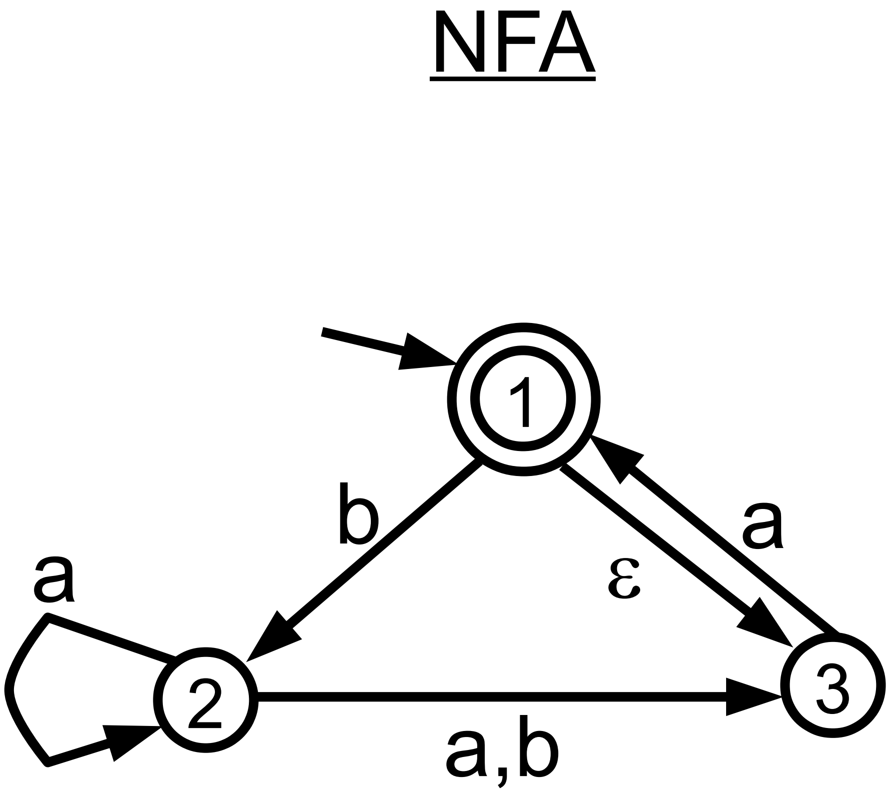


$\delta_{DFA}(\{1,2,3\}, b)$

$= E(\delta_{NFA}(1,b) \cup \delta_{NFA}(2,b) \cup \delta_{NFA}(3,b))$

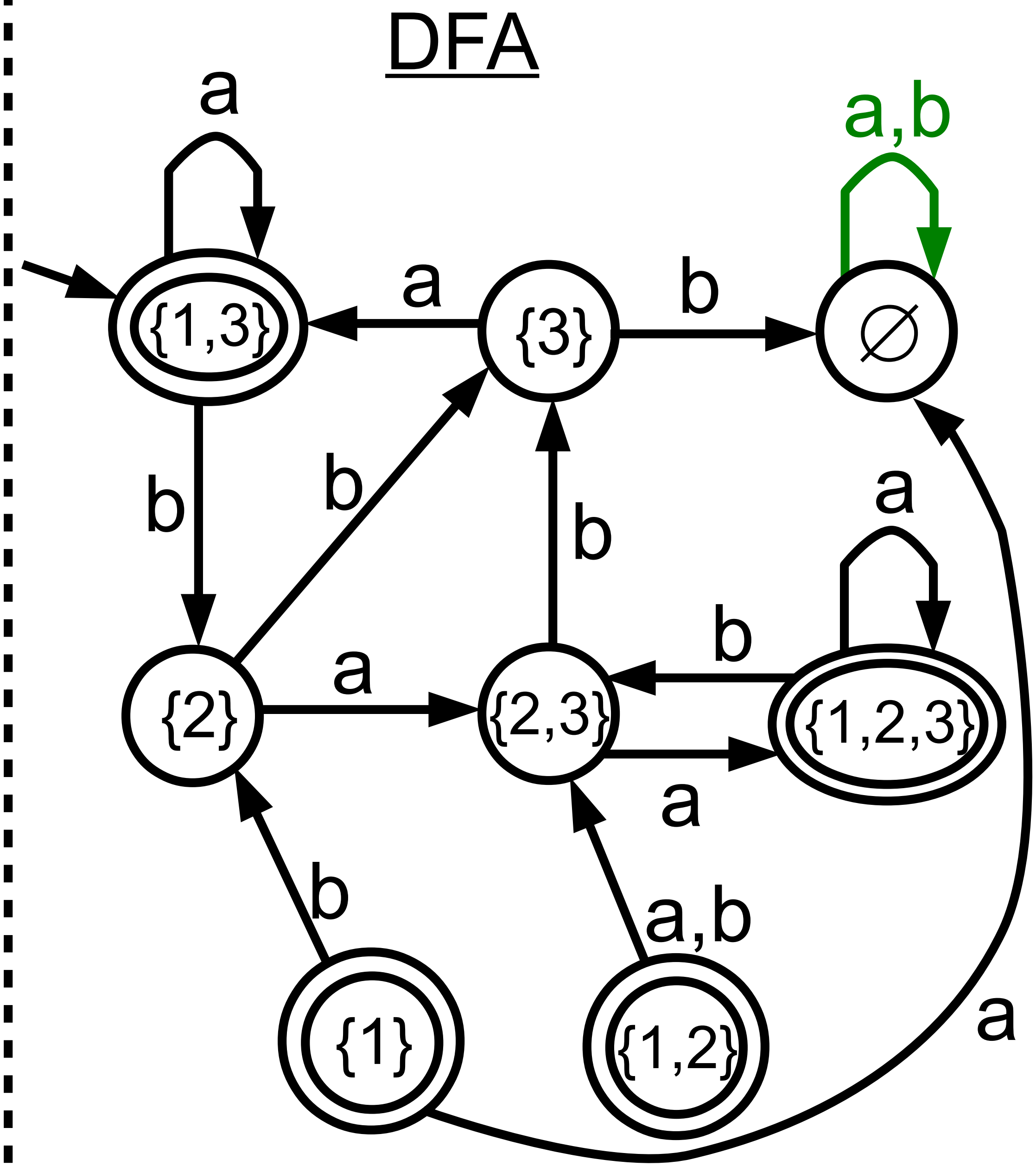
$= E(\{2\} \cup \{3\} \cup \varnothing) = \{2,3\}$

Example: NFA → DFA conversion



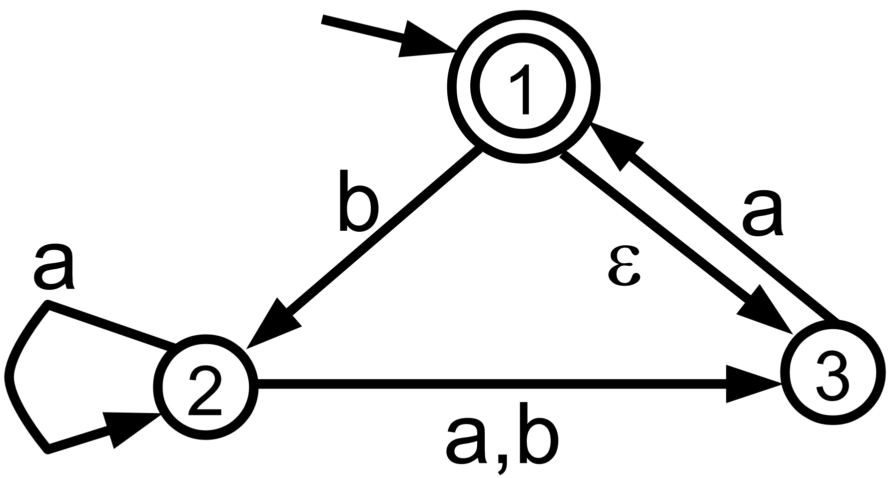
$\delta_{DFA}(\varnothing, a) = \varnothing$

$\delta_{DFA}(\varnothing, b) = \varnothing$

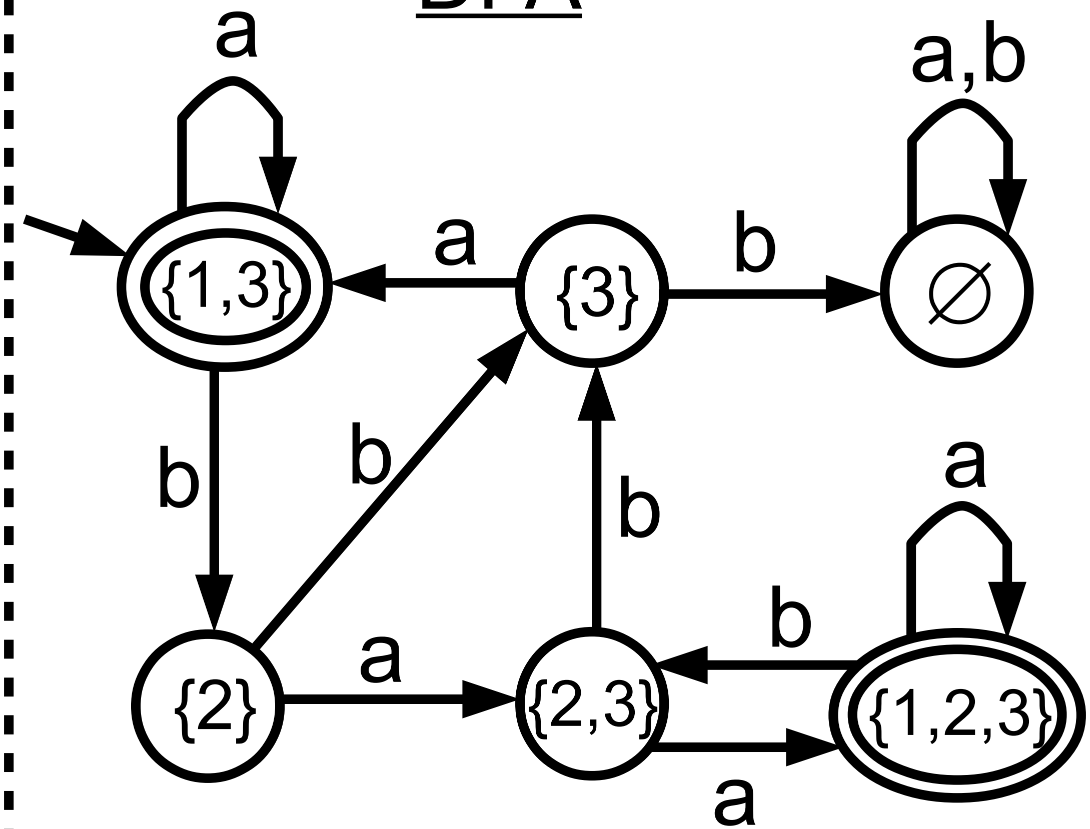


Example: NFA → DFA conversion

NFA



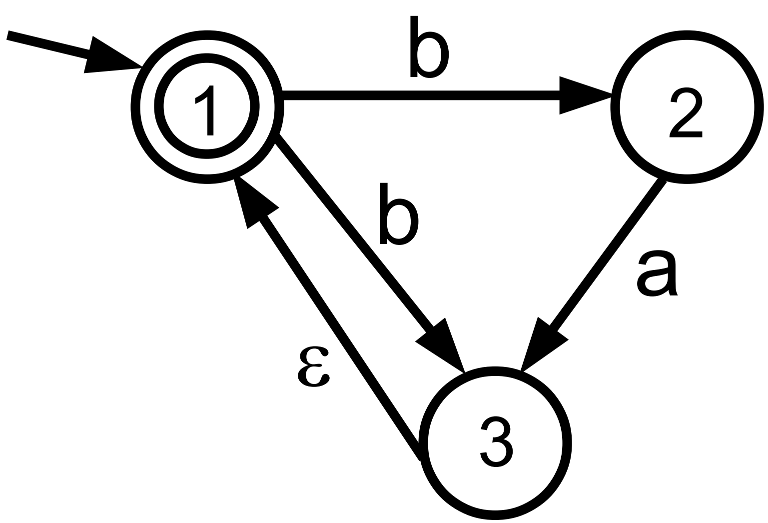
DFA



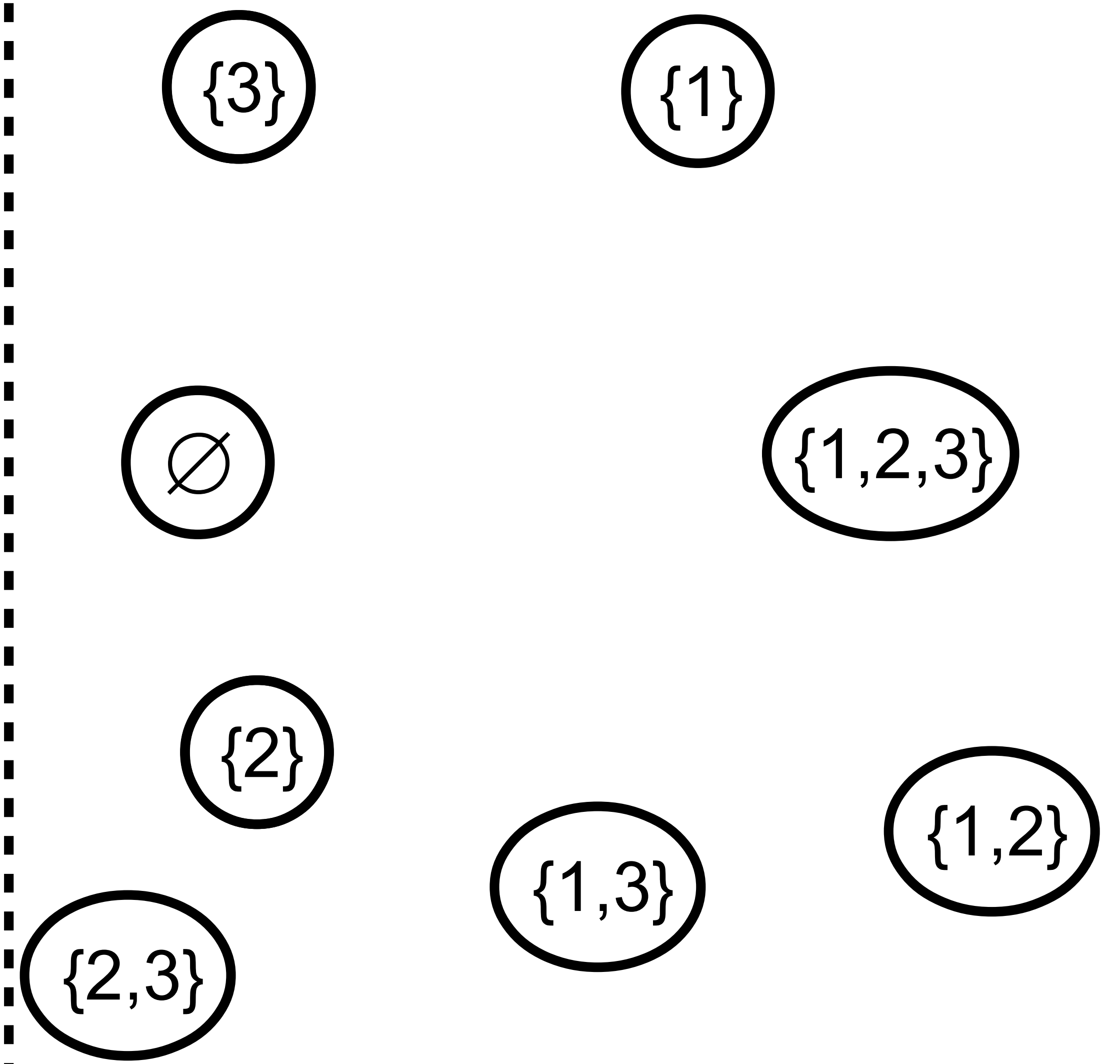
We can delete the unreachable states.

ANOTHER Example: NFA → DFA conversion

NFA



DFA

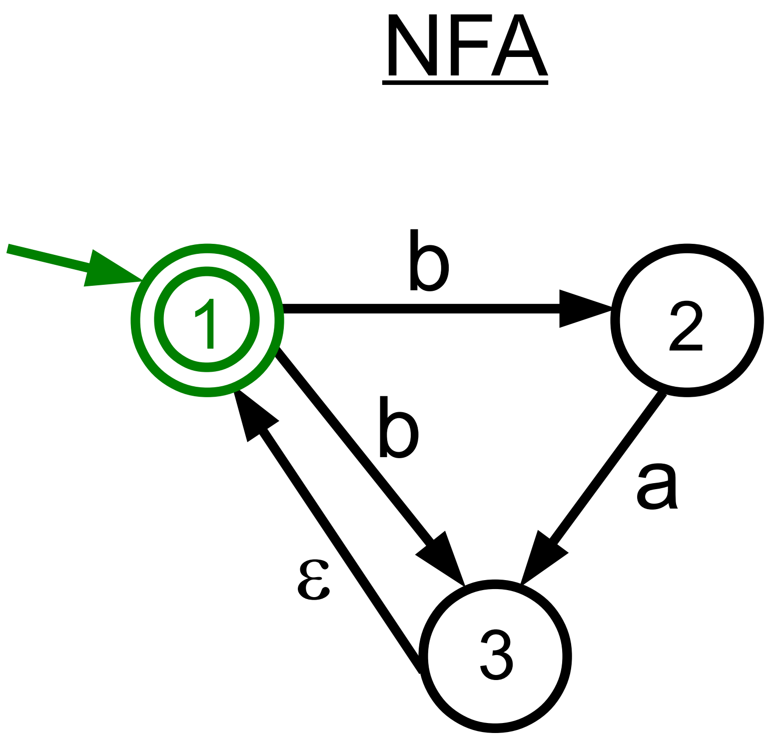


Q_{DFA} = Powerset(Q_{NFA})

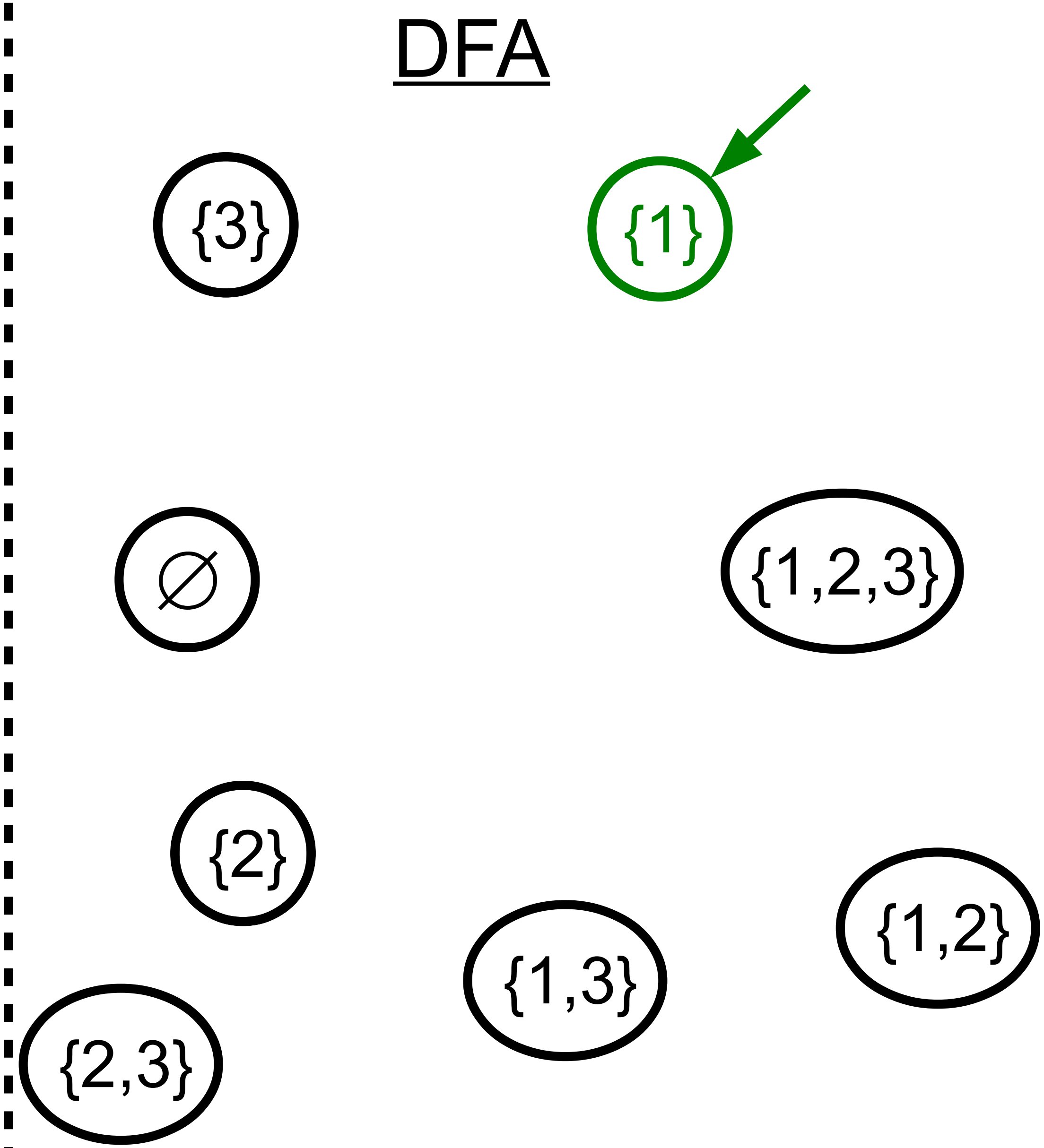
= Powerset({1,2,3})

= {∅,{1},{2},{3},{1,2}...}

ANOTHER Example: NFA → DFA conversion

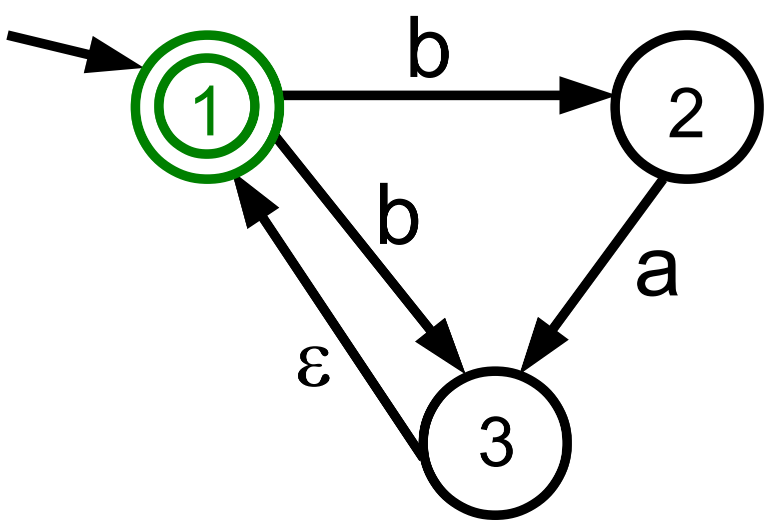


$$q_{DFA} = E(\{q_{NFA}\})$$
$$= E(\{1\})$$
$$= \{1\}$$

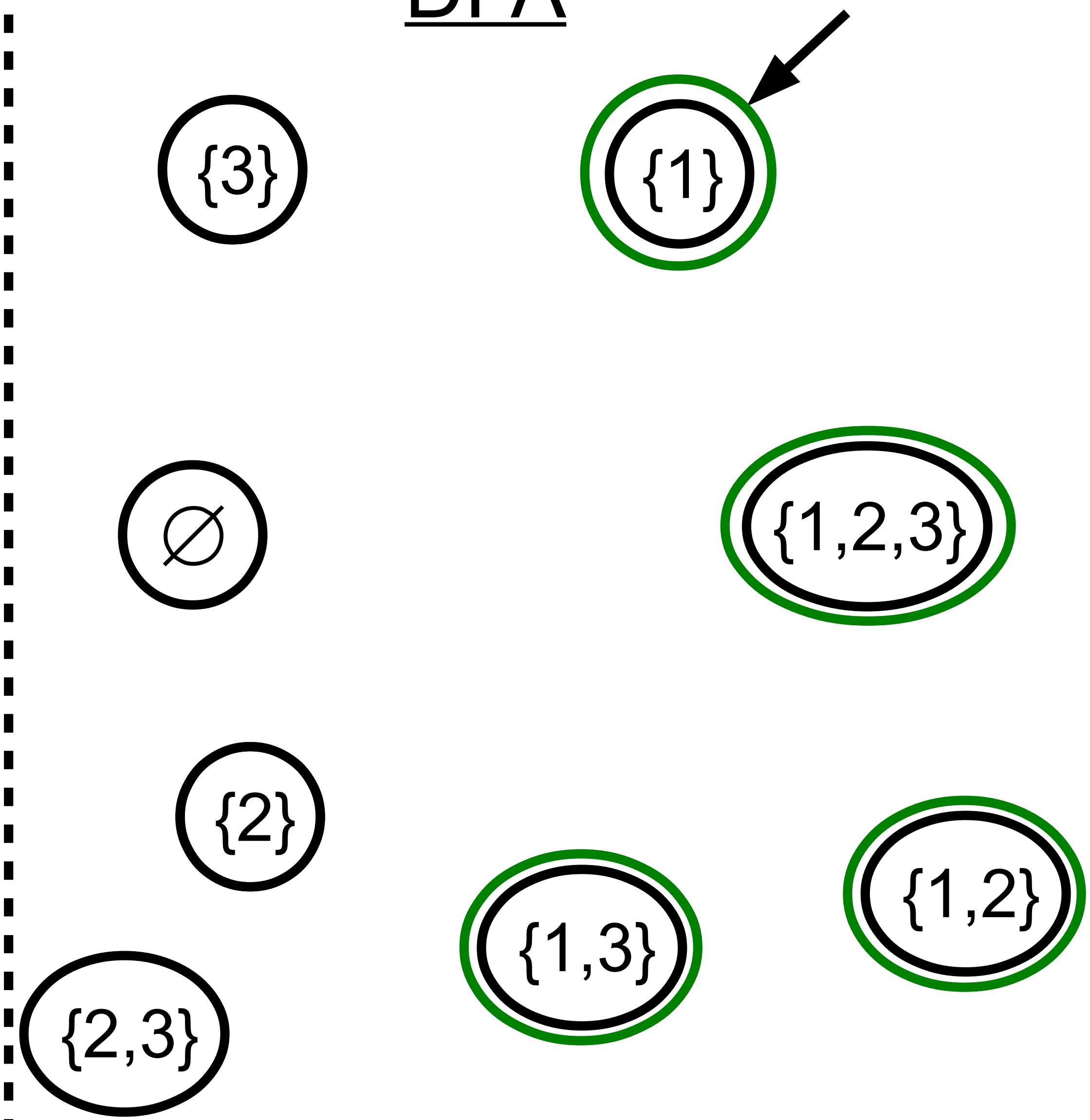


ANOTHER Example: NFA → DFA conversion

NFA



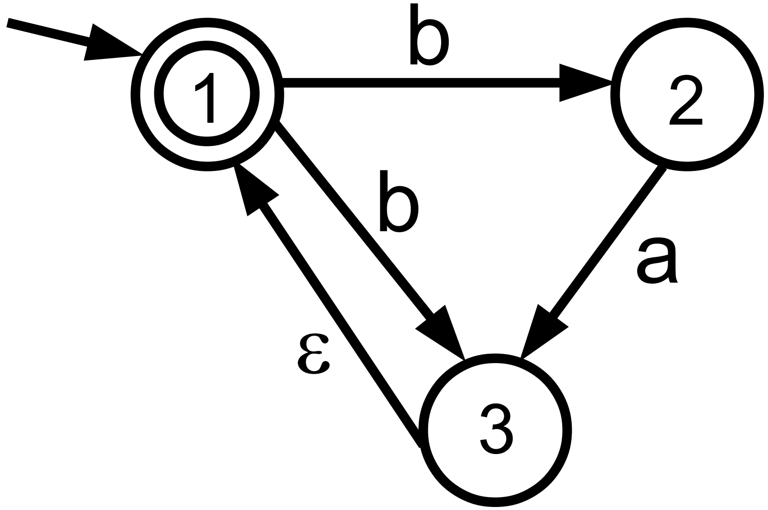
DFA



F_{DFA} = {S : S contains an element of F_{NFA}}

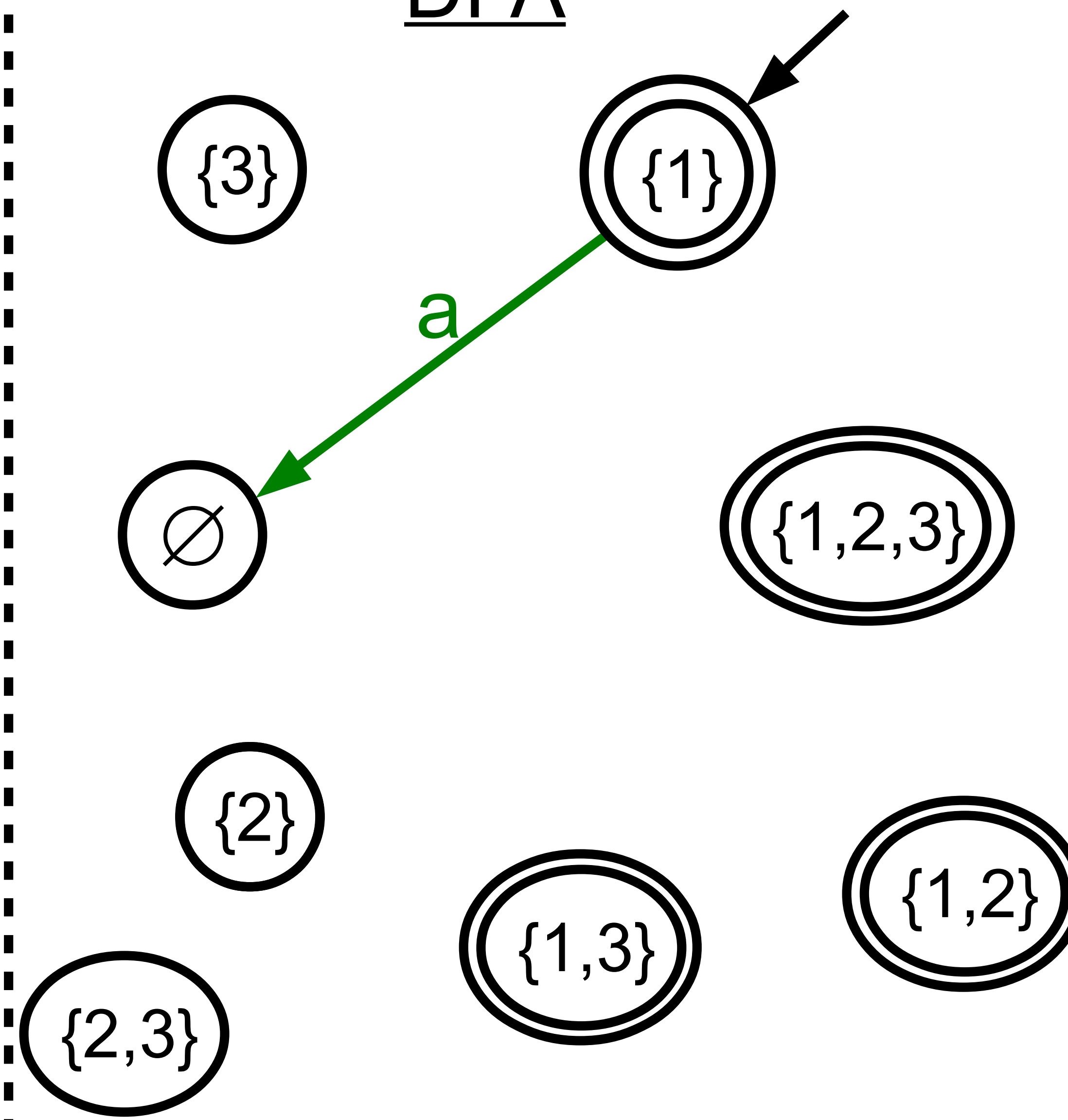
ANOTHER Example: NFA → DFA conversion

NFA



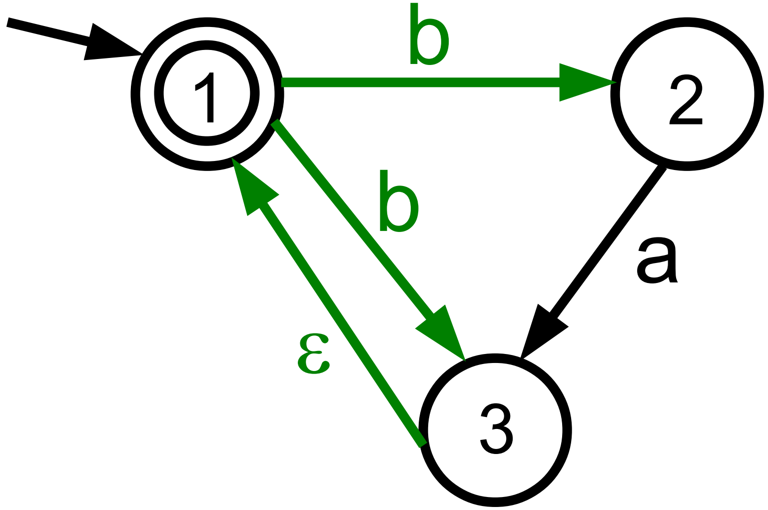
$$\delta_{DFA}(\{1\}, a)$$
$$= E(\delta_{NFA}(1, a))$$
$$= E(\varnothing) = \varnothing$$

DFA

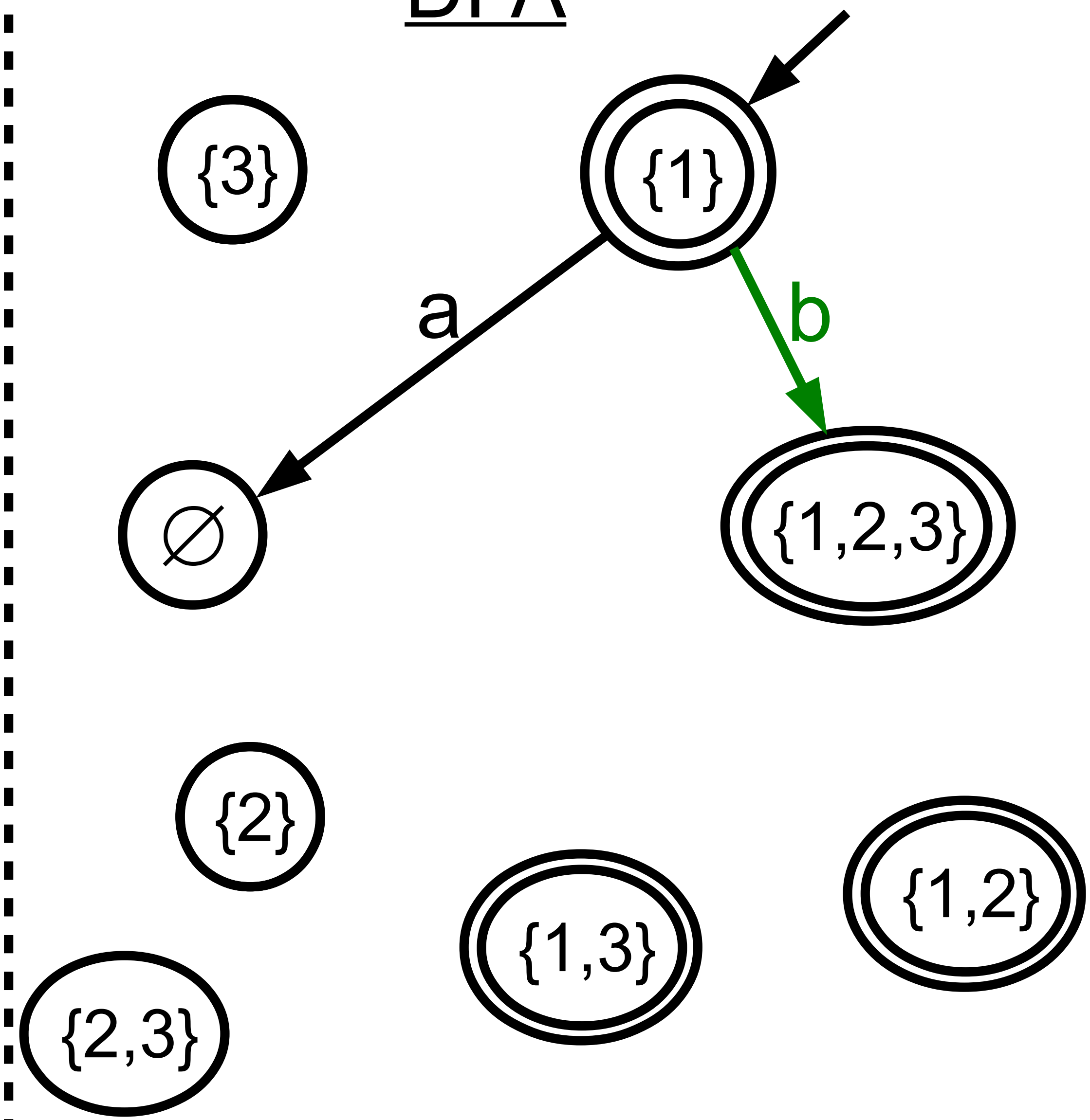


ANOTHER Example: NFA → DFA conversion

NFA



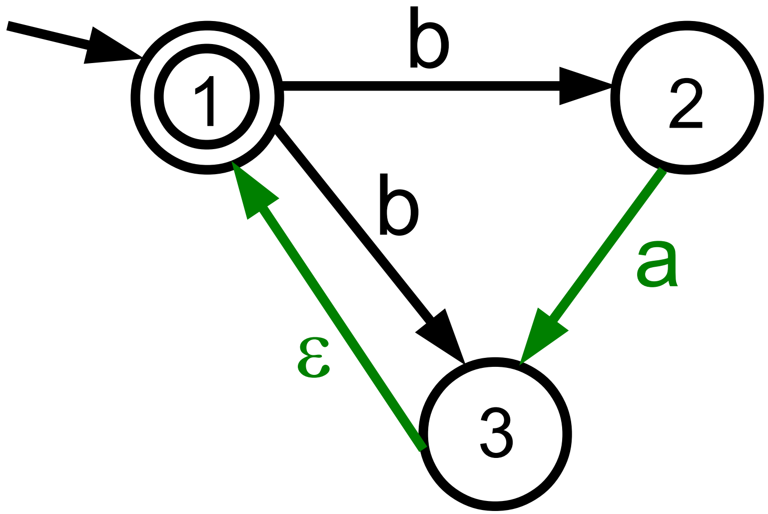
DFA



$$\delta_{DFA}(\{1\}, b)$$
$$= E(\delta_{NFA}(1, b))$$
$$= E(\{2,3\}) = \{1,2,3\}$$

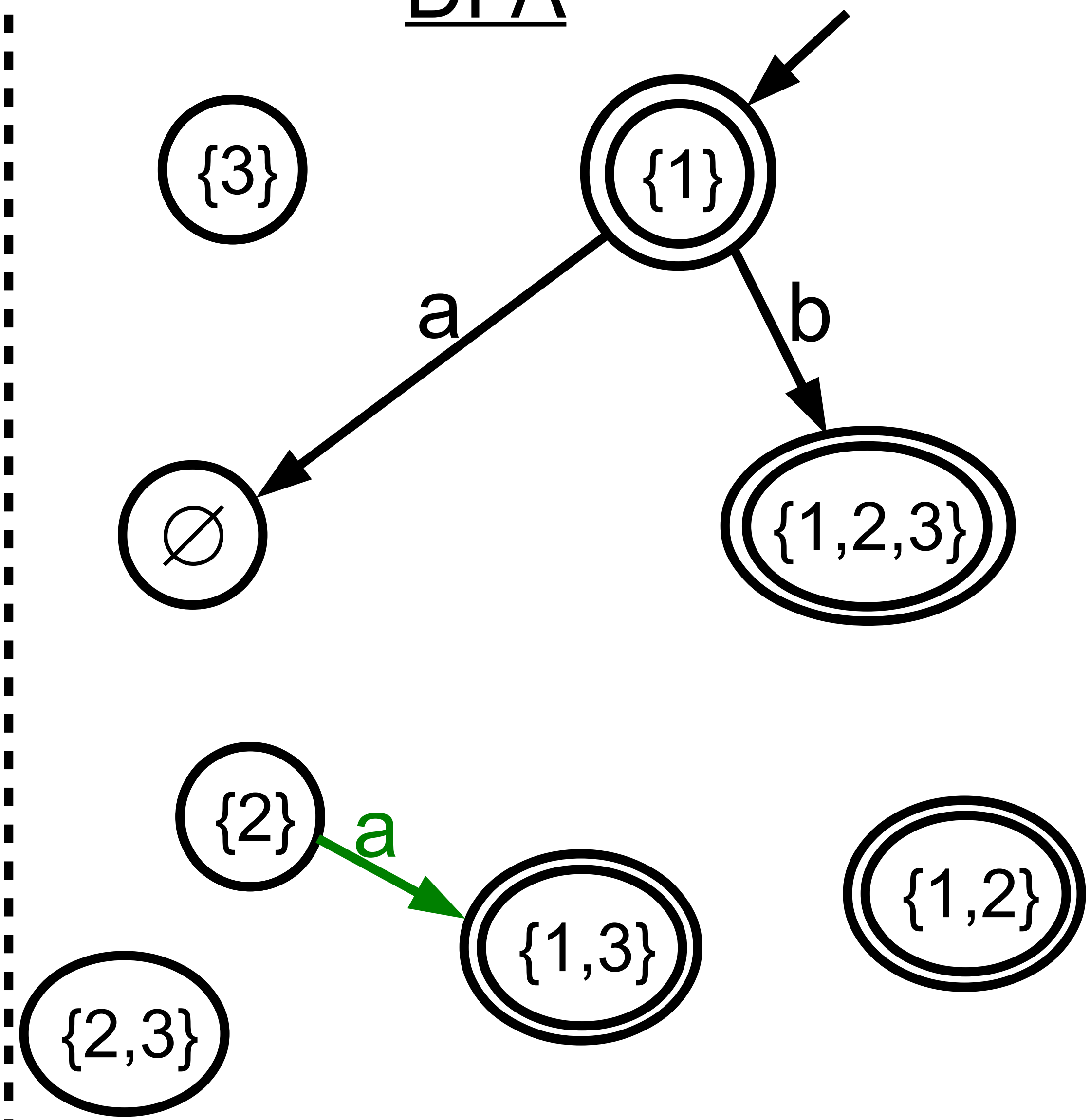
ANOTHER Example: NFA → DFA conversion

NFA



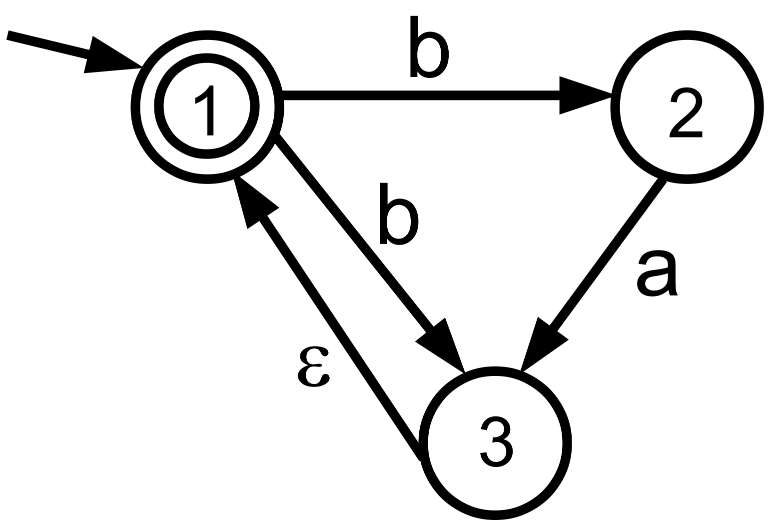
$$\delta_{DFA}(\{2\}, a) = E(\delta_{NFA}(2, a)) = E(\{3\}) = \{1,3\}$$

DFA

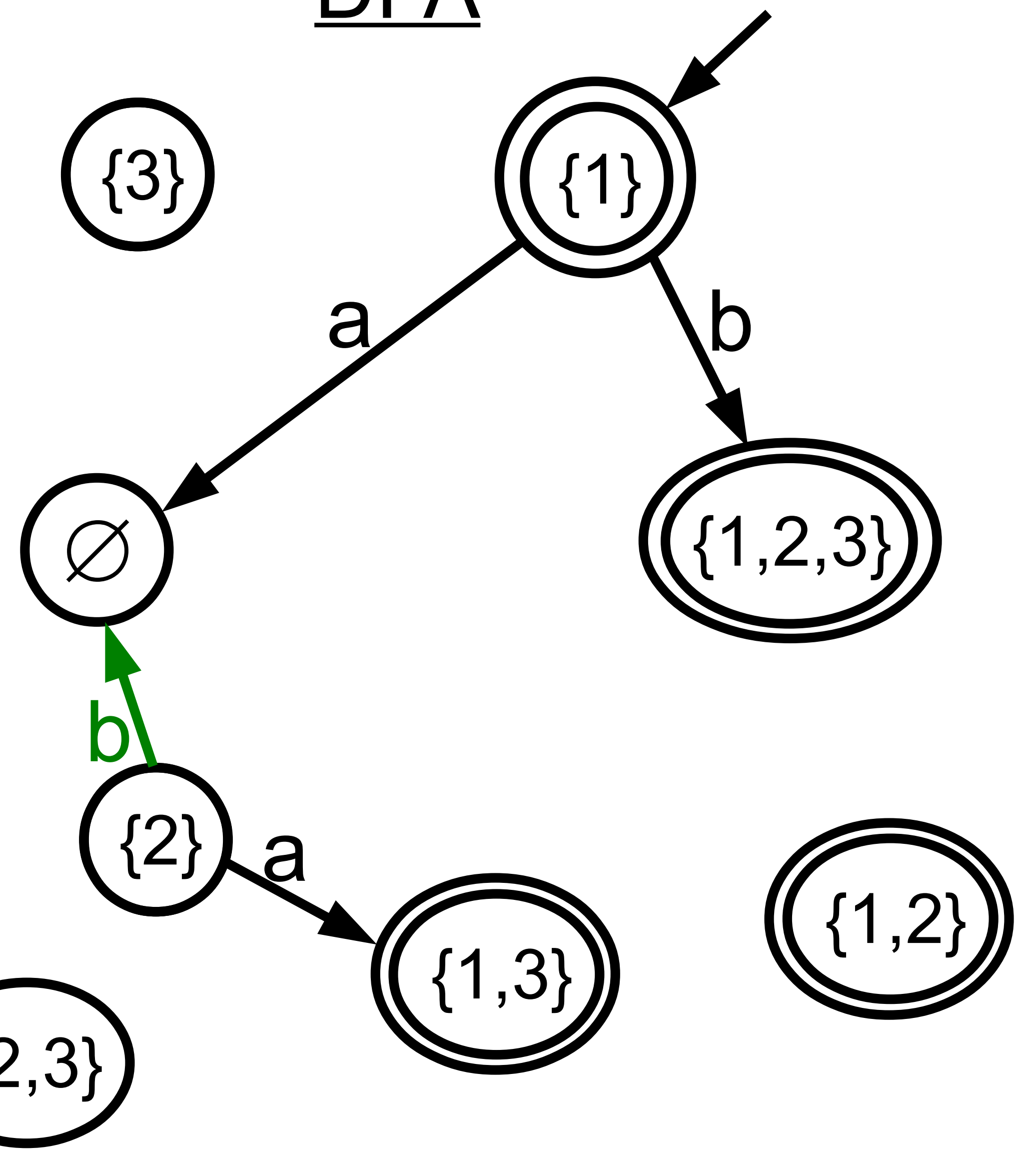


ANOTHER Example: NFA → DFA conversion

NFA



DFA

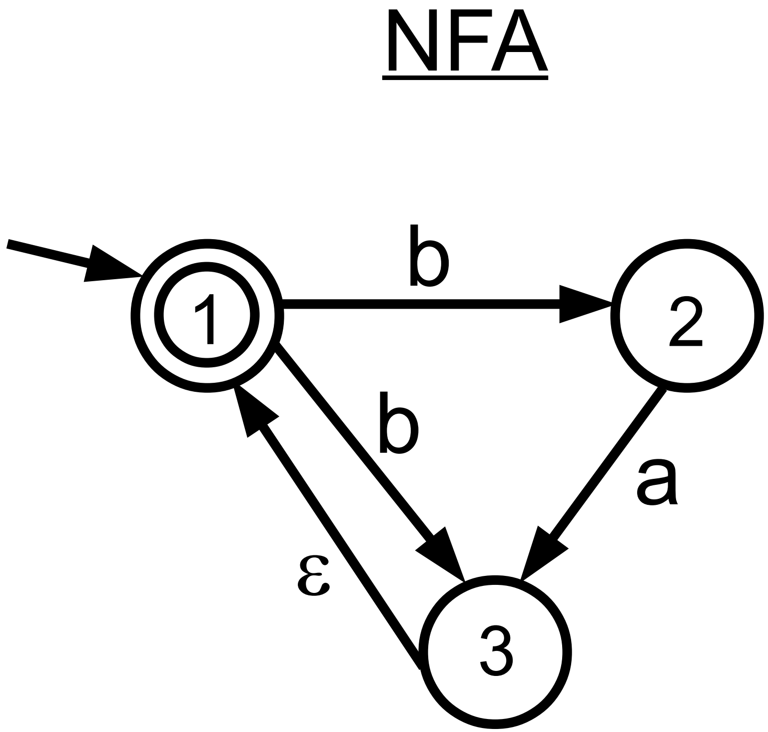


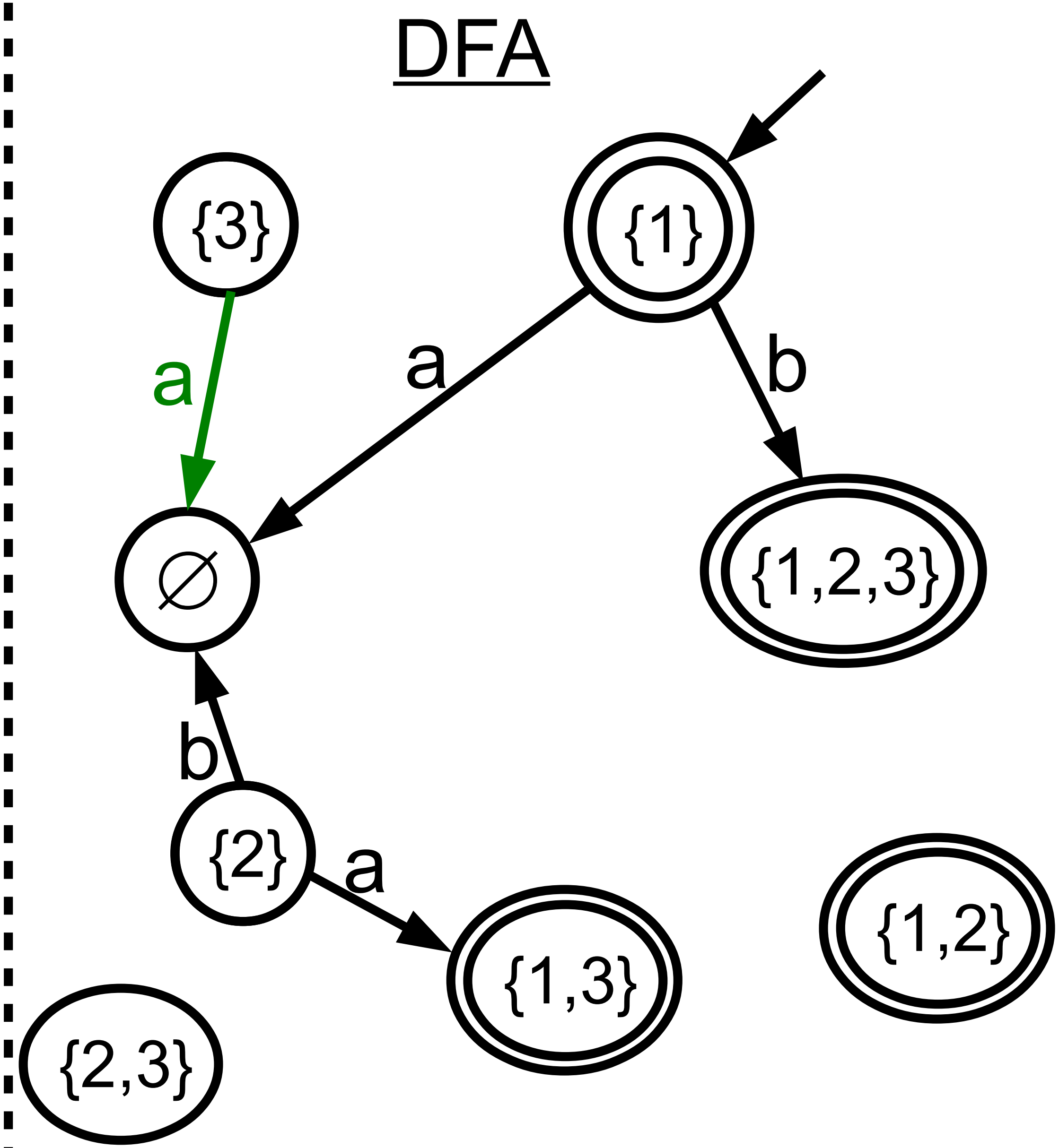
$\delta_{DFA}(\{2\}, b)$

$= E(\delta_{NFA}(2, b))$

$= E(\emptyset) = \emptyset$

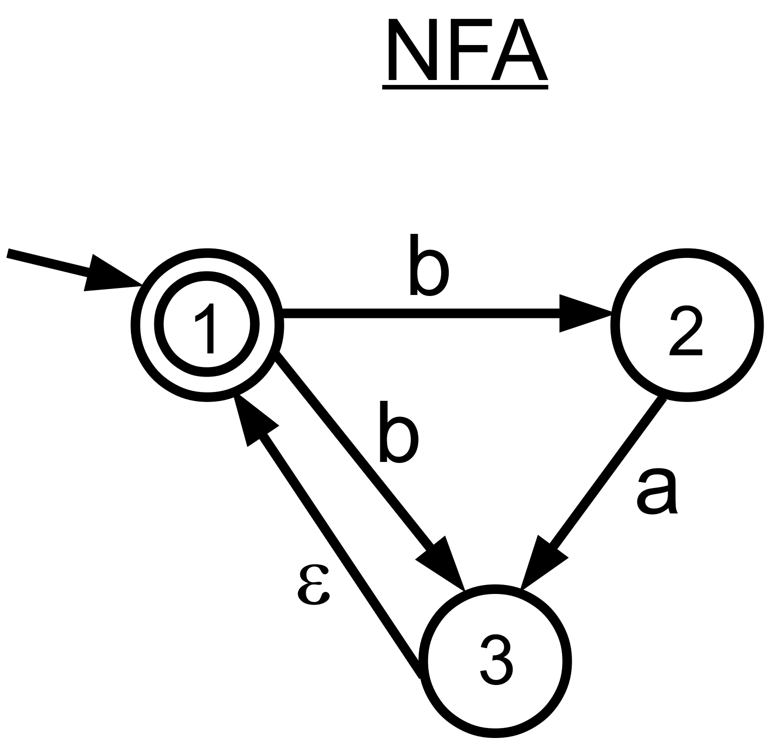
ANOTHER Example: NFA → DFA conversion

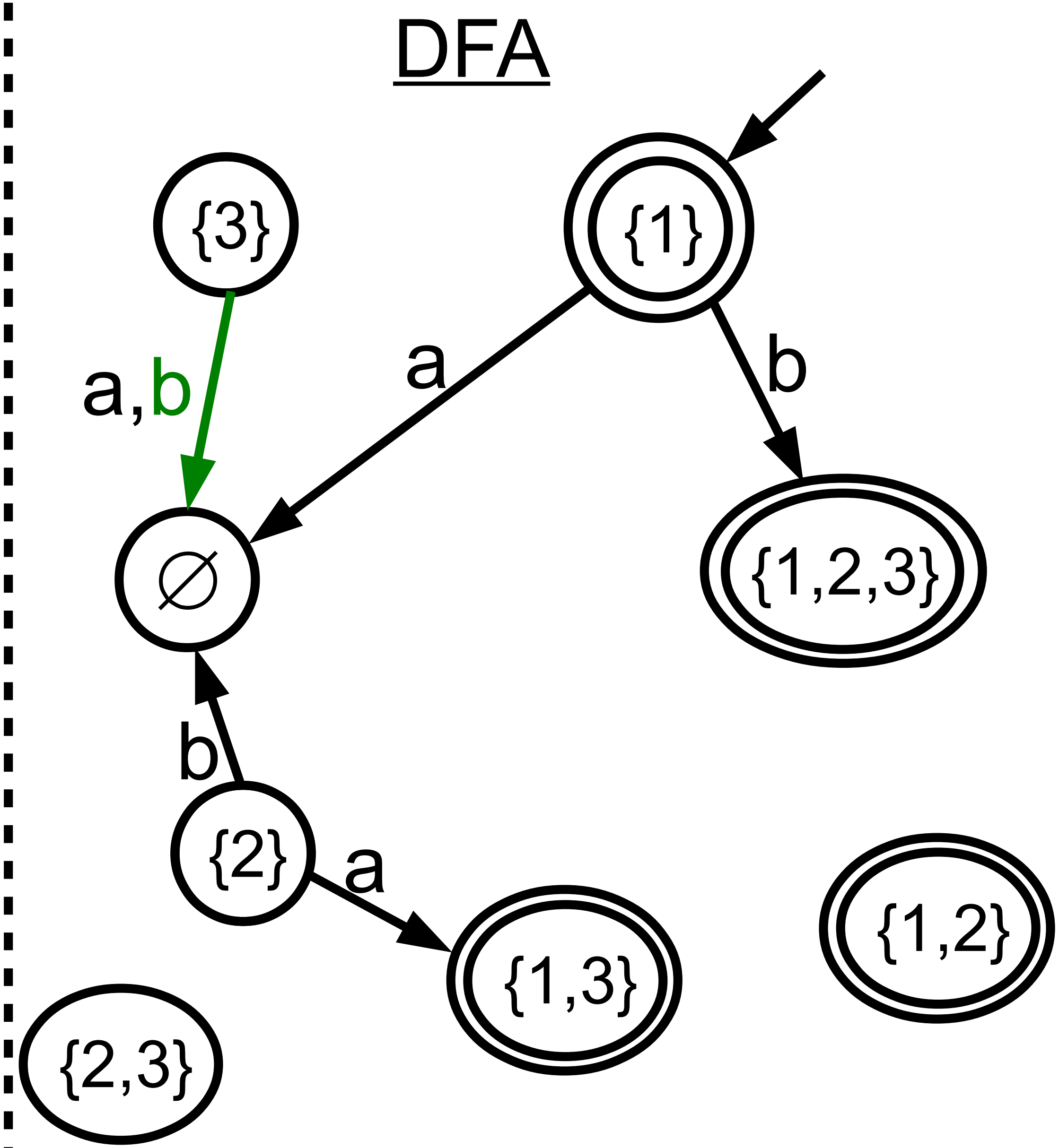




$$\delta_{DFA}(\{3\}, a)$$
$$= E(\delta_{NFA}(3, a))$$
$$= E(\emptyset) = \emptyset$$

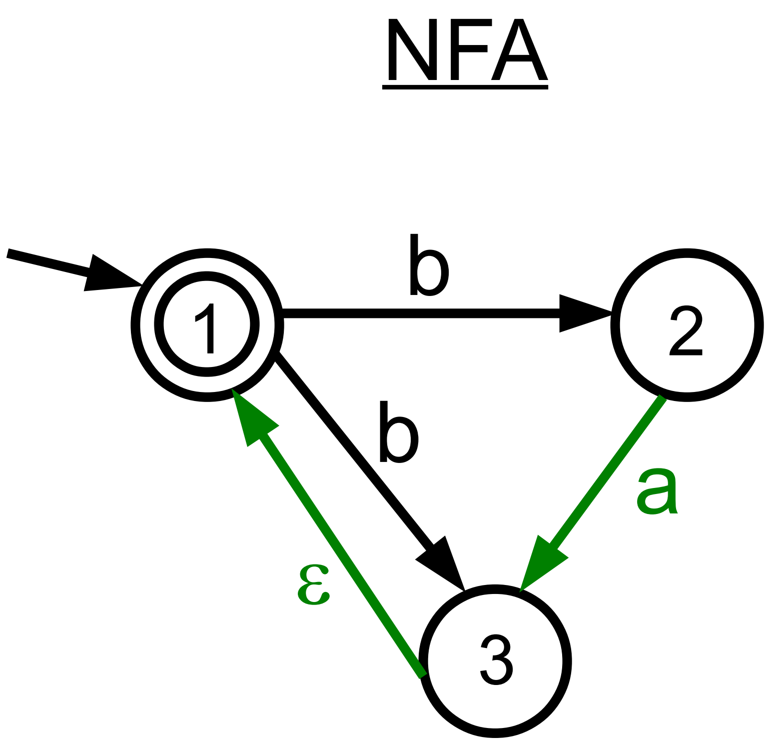
ANOTHER Example: NFA → DFA conversion

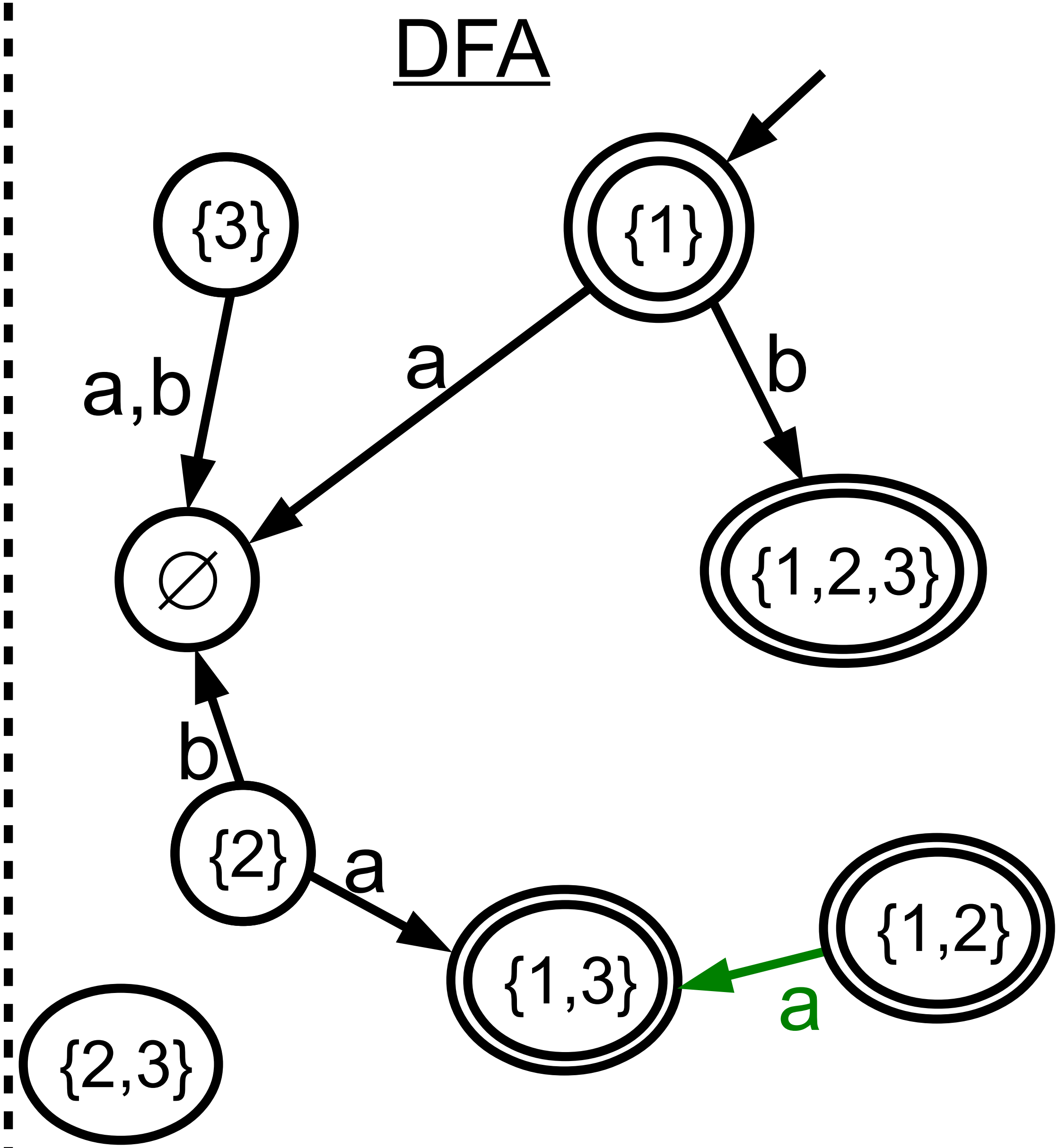




$$\delta_{DFA}(\{3\}, b)$$
$$= E(\delta_{NFA}(3, b))$$
$$= E(\emptyset) = \emptyset$$

ANOTHER Example: NFA → DFA conversion





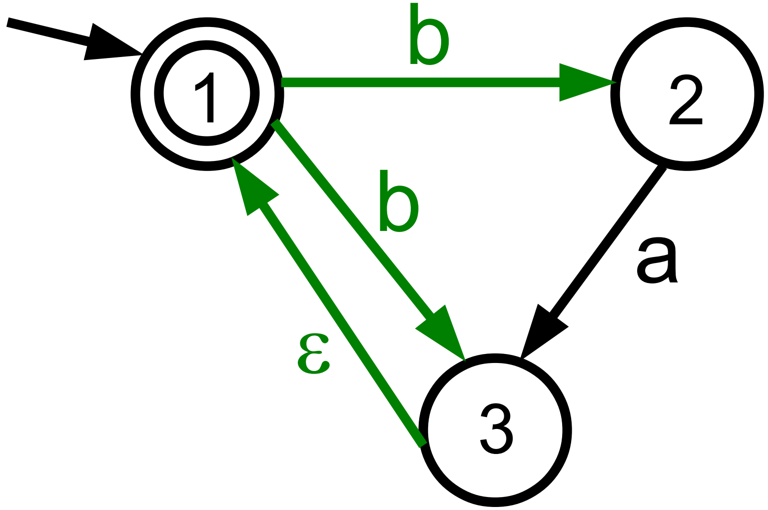
$\delta_{DFA}(\{1,2\}, a)$

$= E(\delta_{NFA}(1,a) \cup \delta_{NFA}(2,a))$

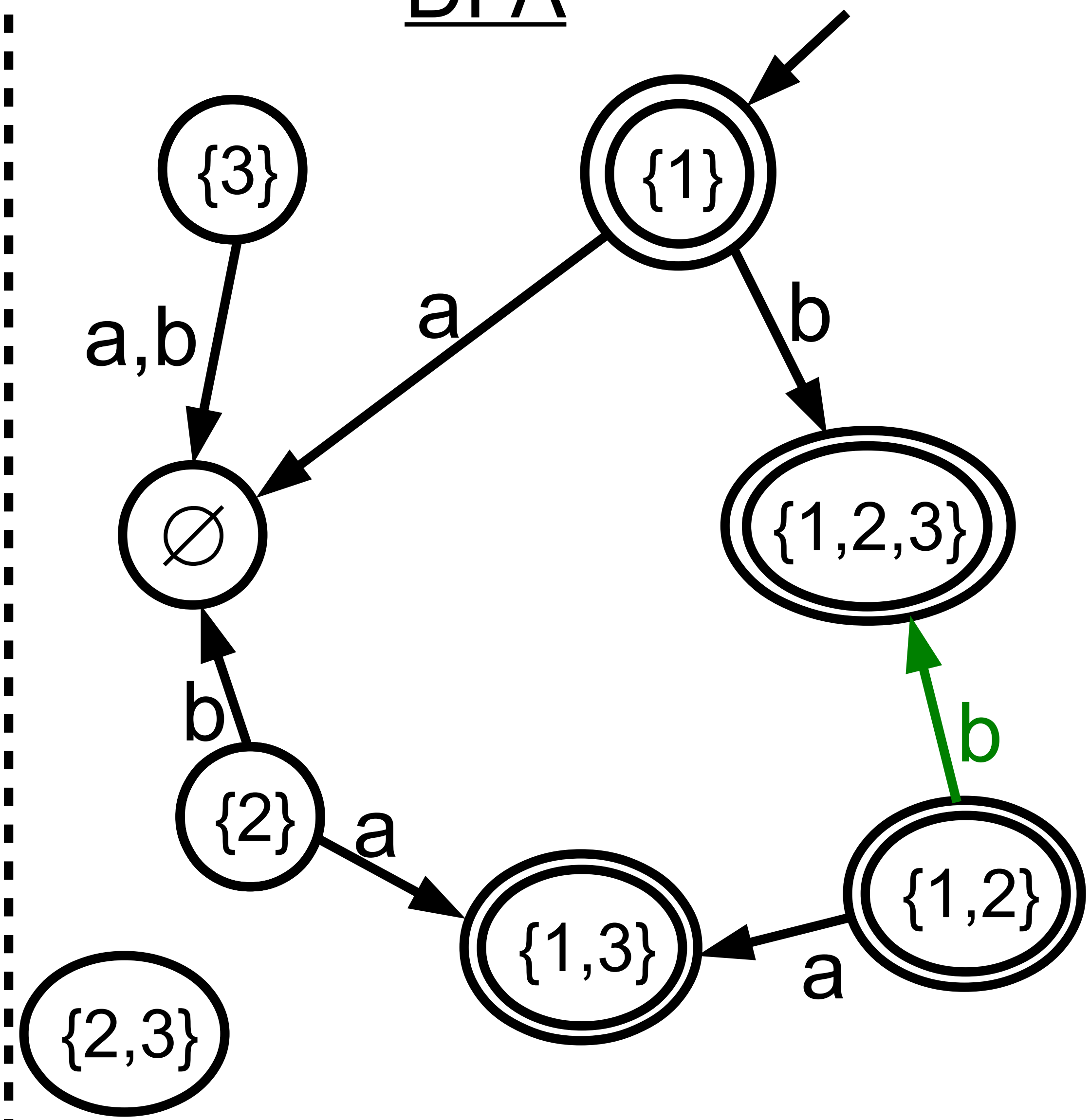
$= E(\varnothing \cup \{3\}) = \{1,3\}$

ANOTHER Example: NFA → DFA conversion

NFA



DFA



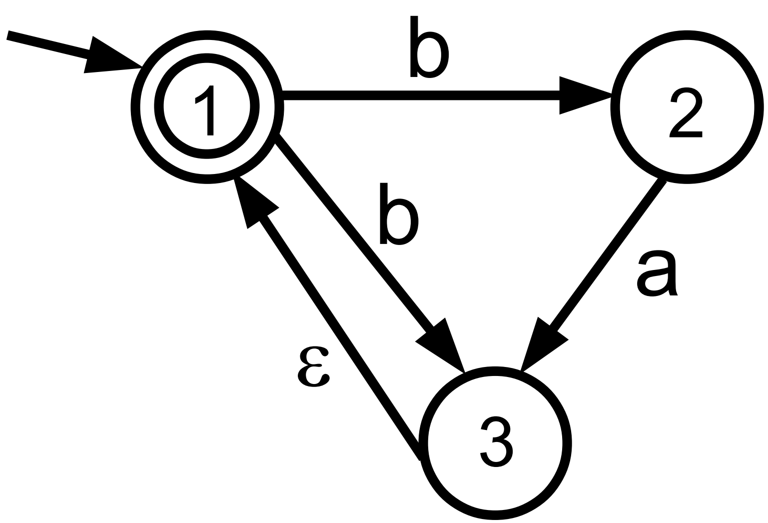
$\delta_{DFA}(\{1,2\}, b)$

$= E(\delta_{NFA}(1,b) \cup \delta_{NFA}(2,b))$

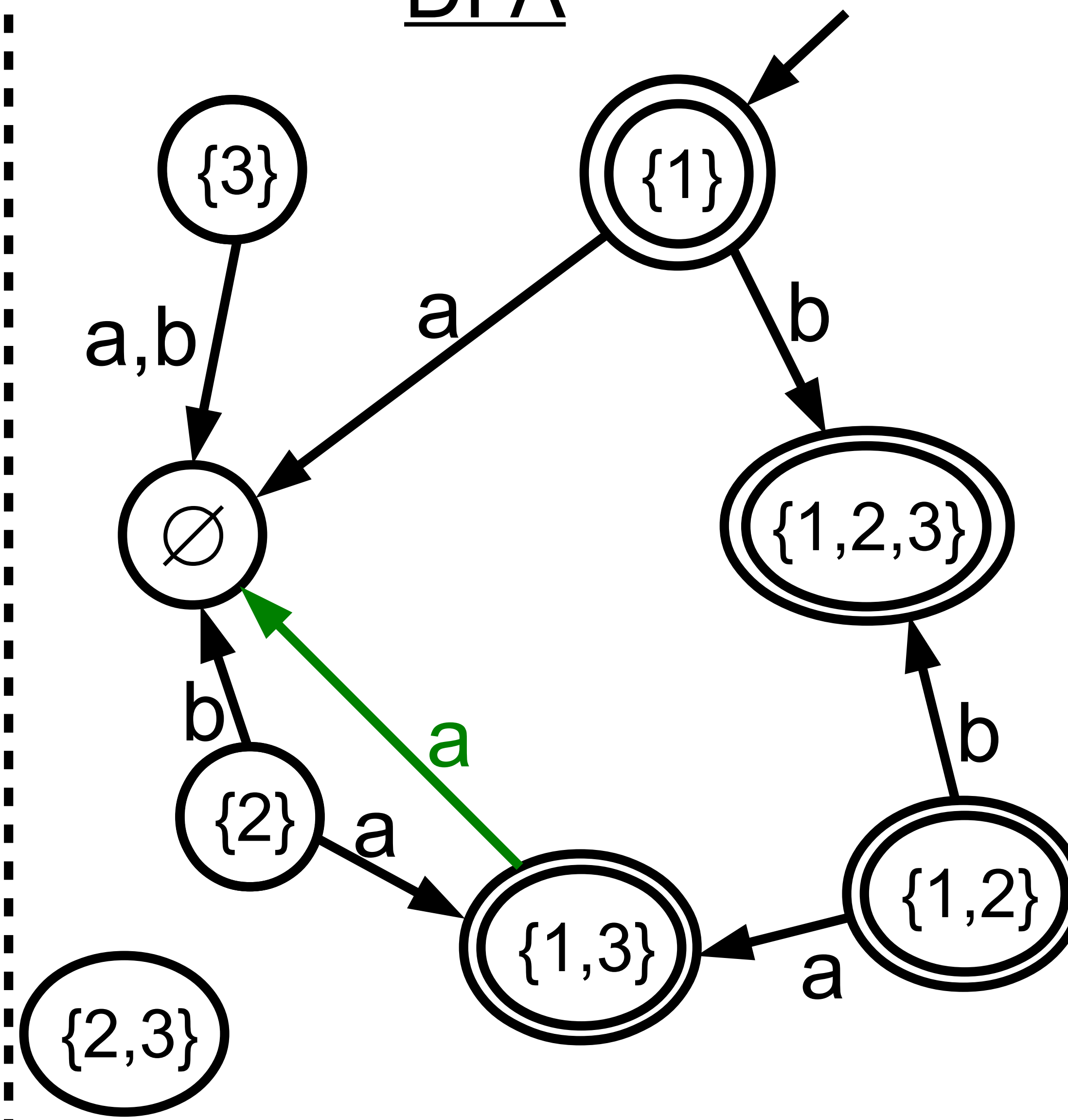
$= E(\{2,3\} \cup \varnothing) = \{1,2,3\}$

ANOTHER Example: NFA → DFA conversion

NFA



DFA



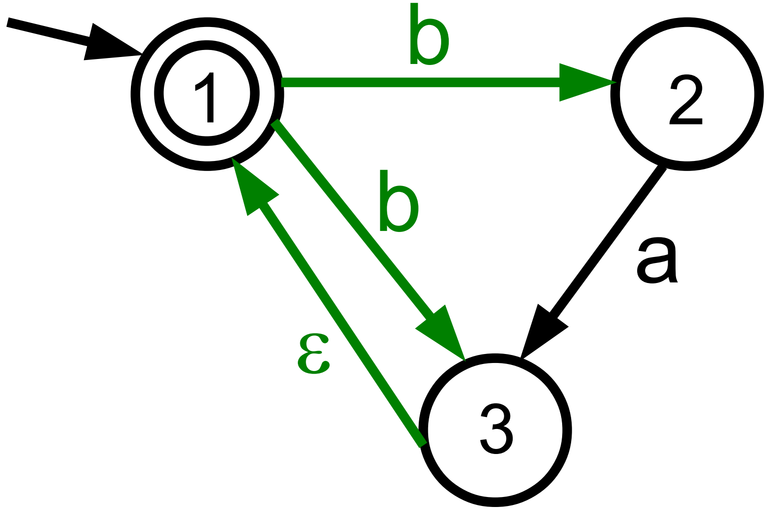
$\delta_{DFA}(\{1,3\}, a)$

$= E(\delta_{NFA}(1,a) \cup \delta_{NFA}(3,a))$

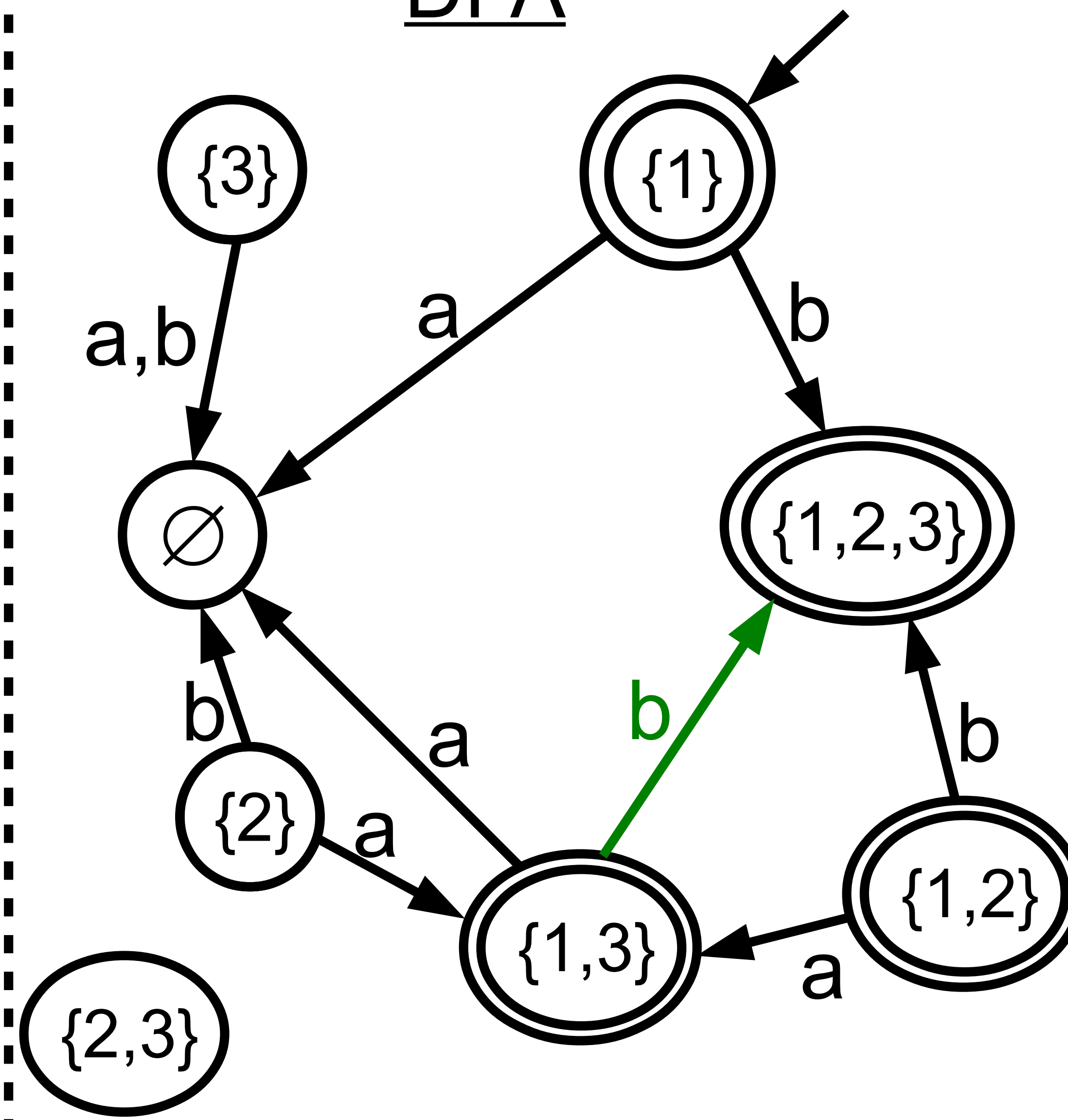
$= E(\varnothing \cup \varnothing) = \varnothing$

ANOTHER Example: NFA → DFA conversion

NFA



DFA



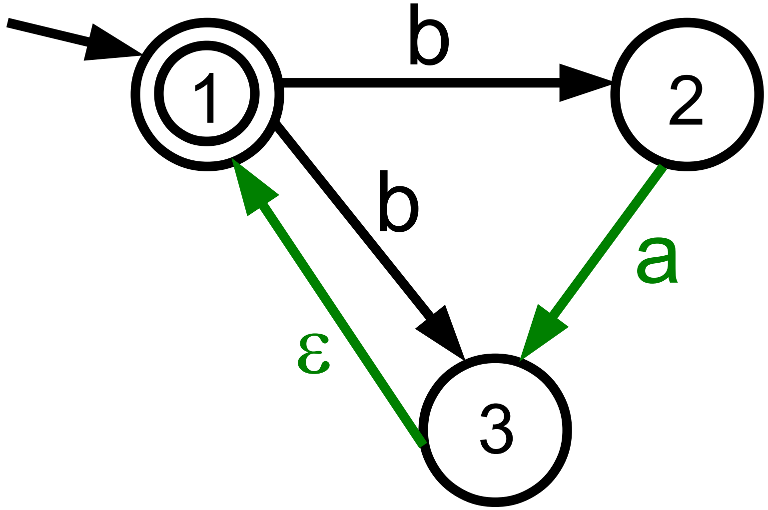
$\delta_{DFA}(\{1,3\}, b)$

$= E(\delta_{NFA}(1,b) \cup \delta_{NFA}(3,b))$

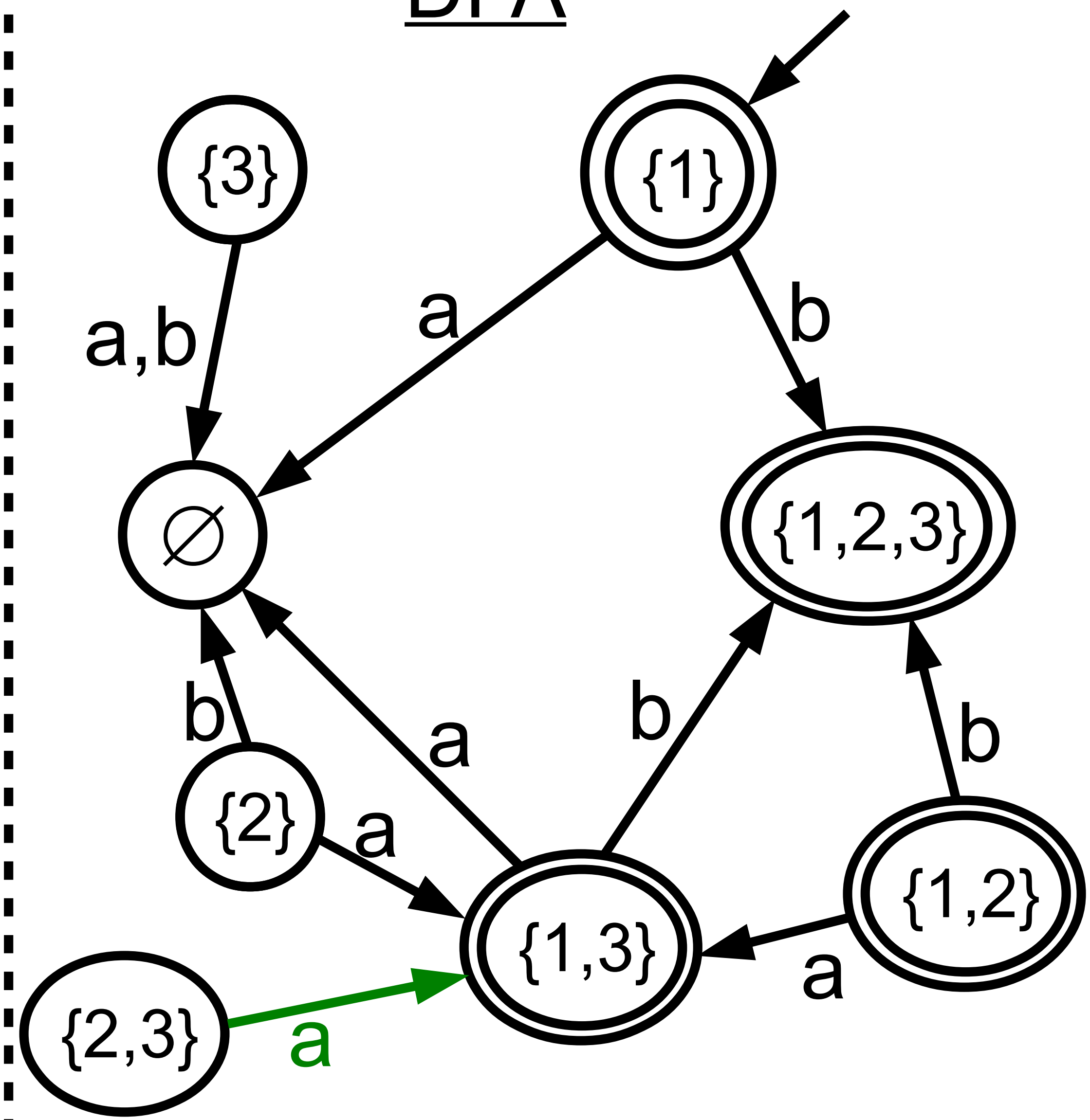
$= E(\{2,3\} \cup \varnothing) = \{1,2,3\}$

ANOTHER Example: NFA → DFA conversion

NFA



DFA



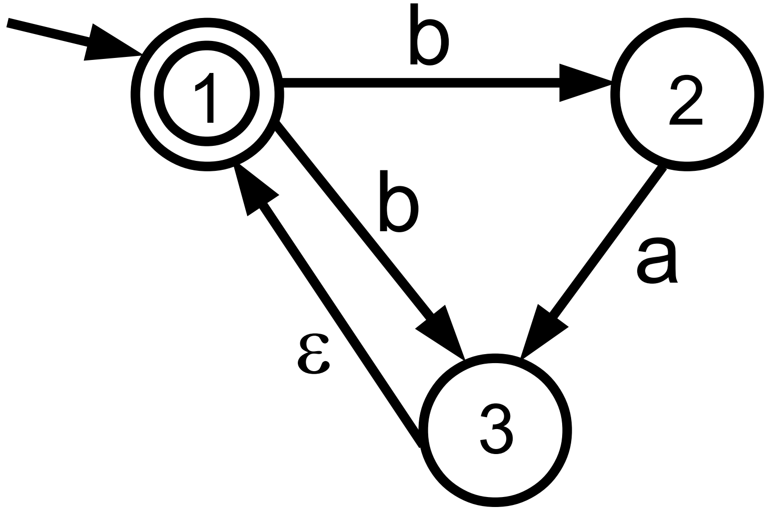
$\delta_{DFA}(\{2,3\}, a)$

$= E(\delta_{NFA}(2,a) \cup \delta_{NFA}(3,a))$

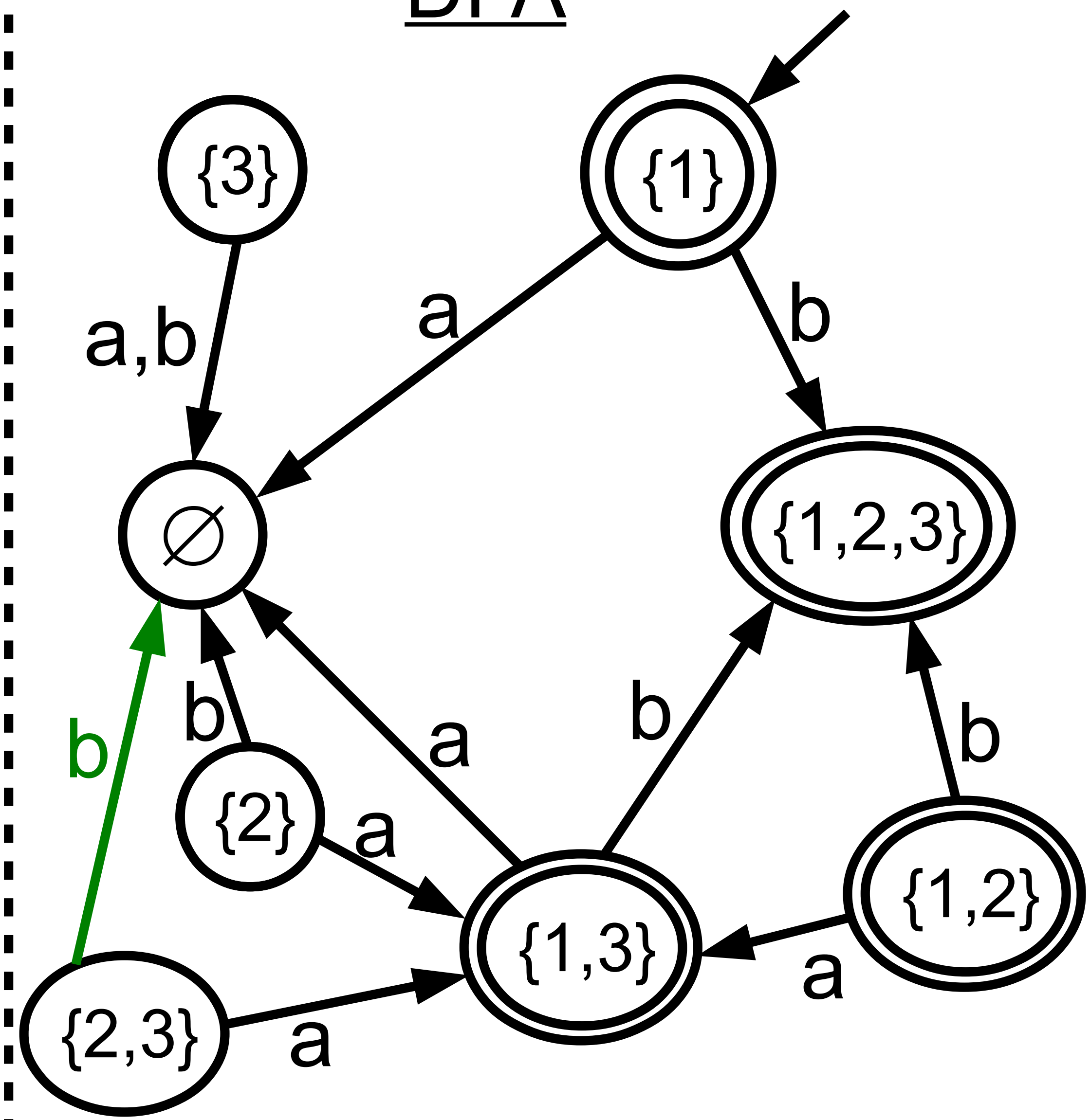
$= E(\{3\} \cup \emptyset) = \{1,3\}$

ANOTHER Example: NFA → DFA conversion

NFA



DFA



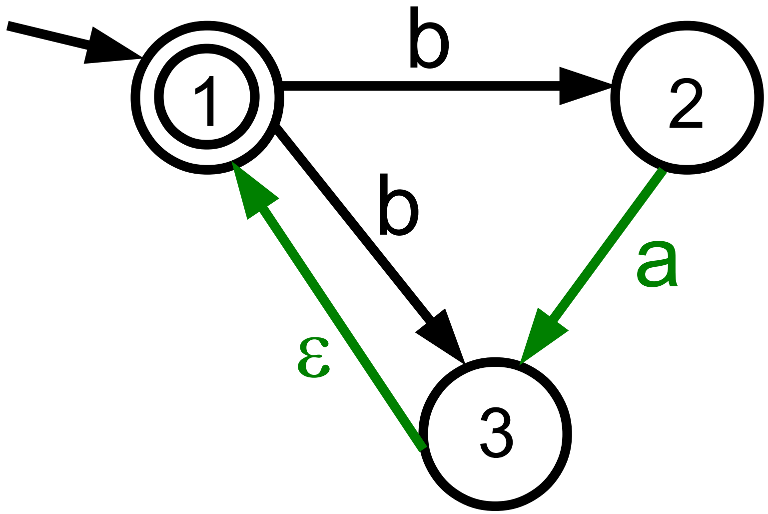
$\delta_{DFA}(\{2,3\}, b)$

$= E(\delta_{NFA}(2,b) \cup \delta_{NFA}(3,b))$

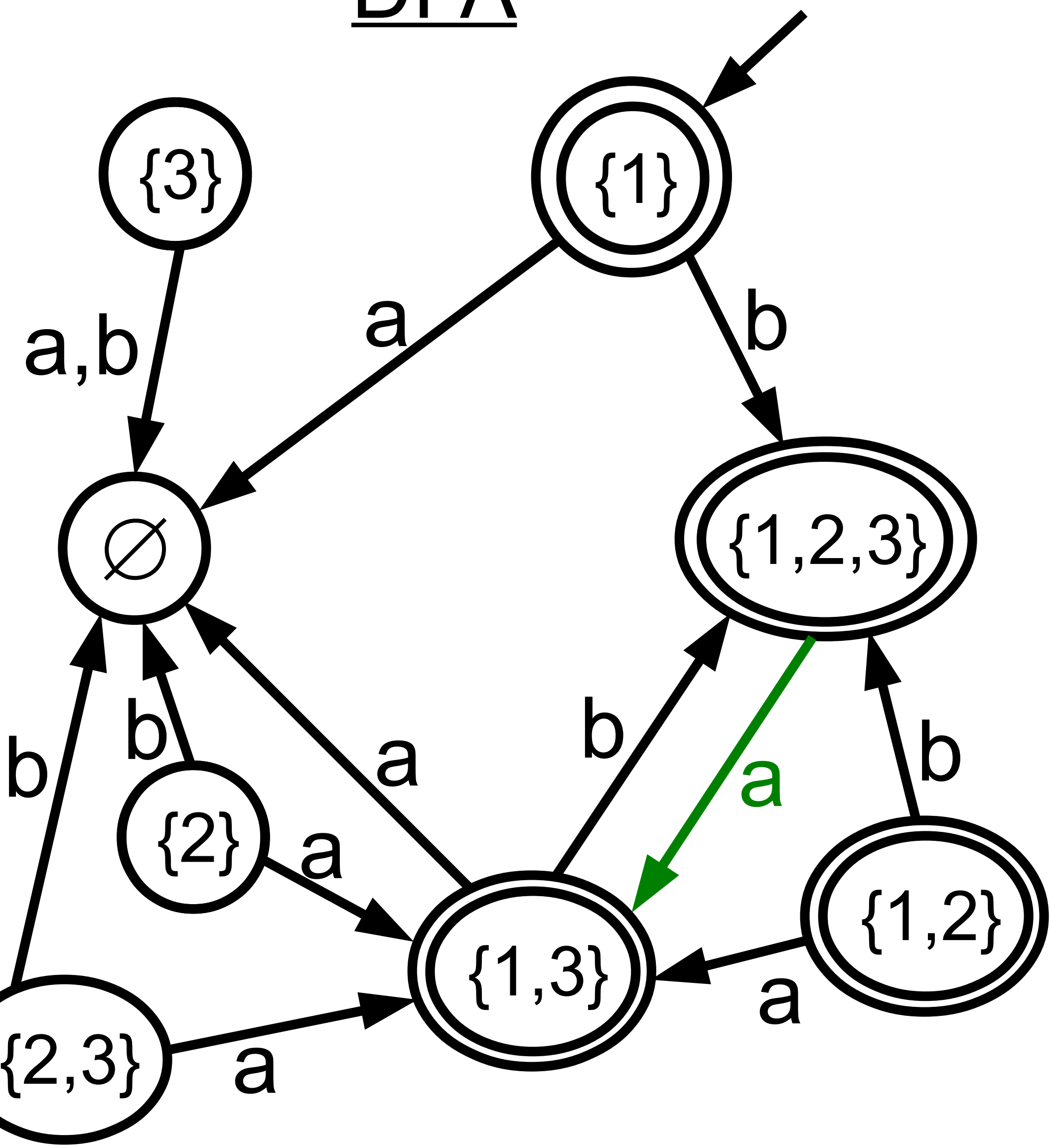
$= E(\varnothing \cup \varnothing) = \varnothing$

ANOTHER Example: NFA → DFA conversion

NFA



DFA



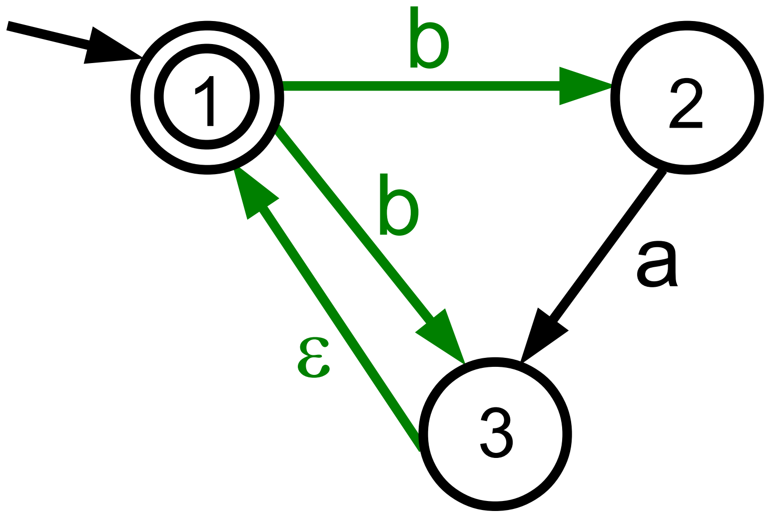
$\delta_{DFA}(\{1,2,3\}, a)$

$=E(\delta_{NFA}(1,a) \cup \delta_{NFA}(2,a) \cup \delta_{NFA}(3,a))$

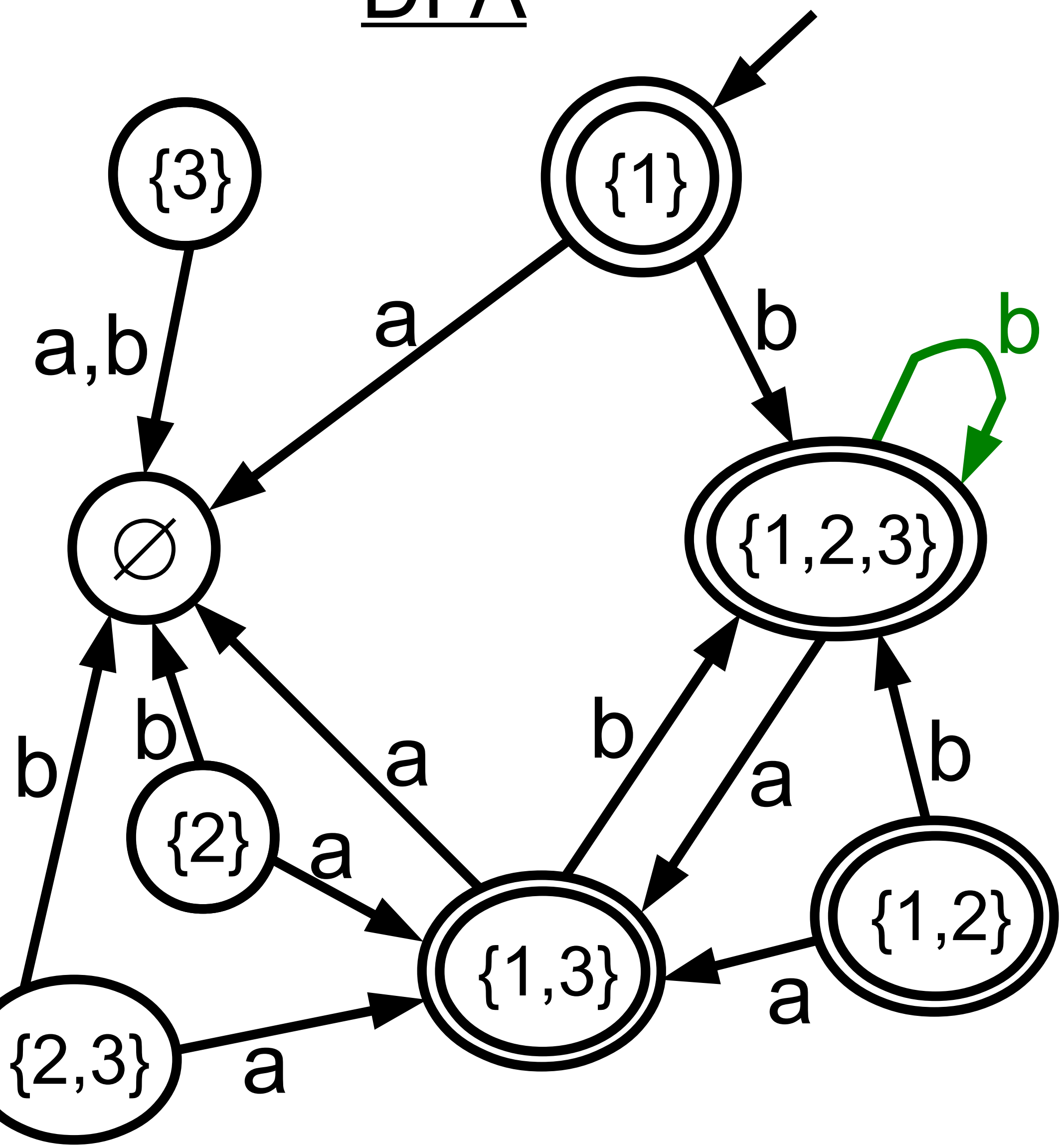
$=E(\varnothing \cup \{3\} \cup \varnothing) = \{1,3\}$

ANOTHER Example: NFA → DFA conversion

NFA



DFA

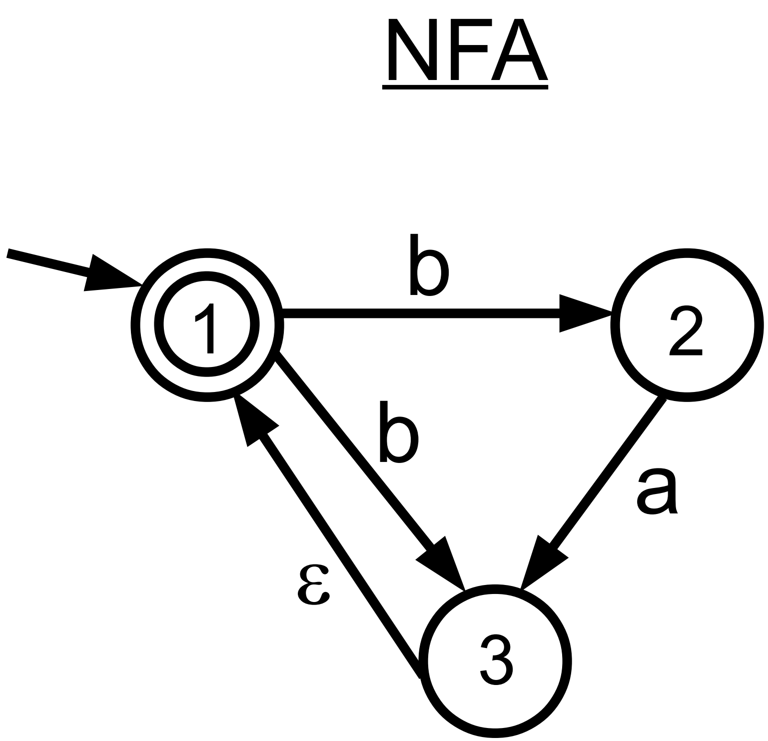


$$\delta_{DFA}(\{1,2,3\}, b)$$

$$= E\left(\delta_{NFA}(1,b) \cup \delta_{NFA}(2,b) \cup \delta_{NFA}(3,b)\right)$$

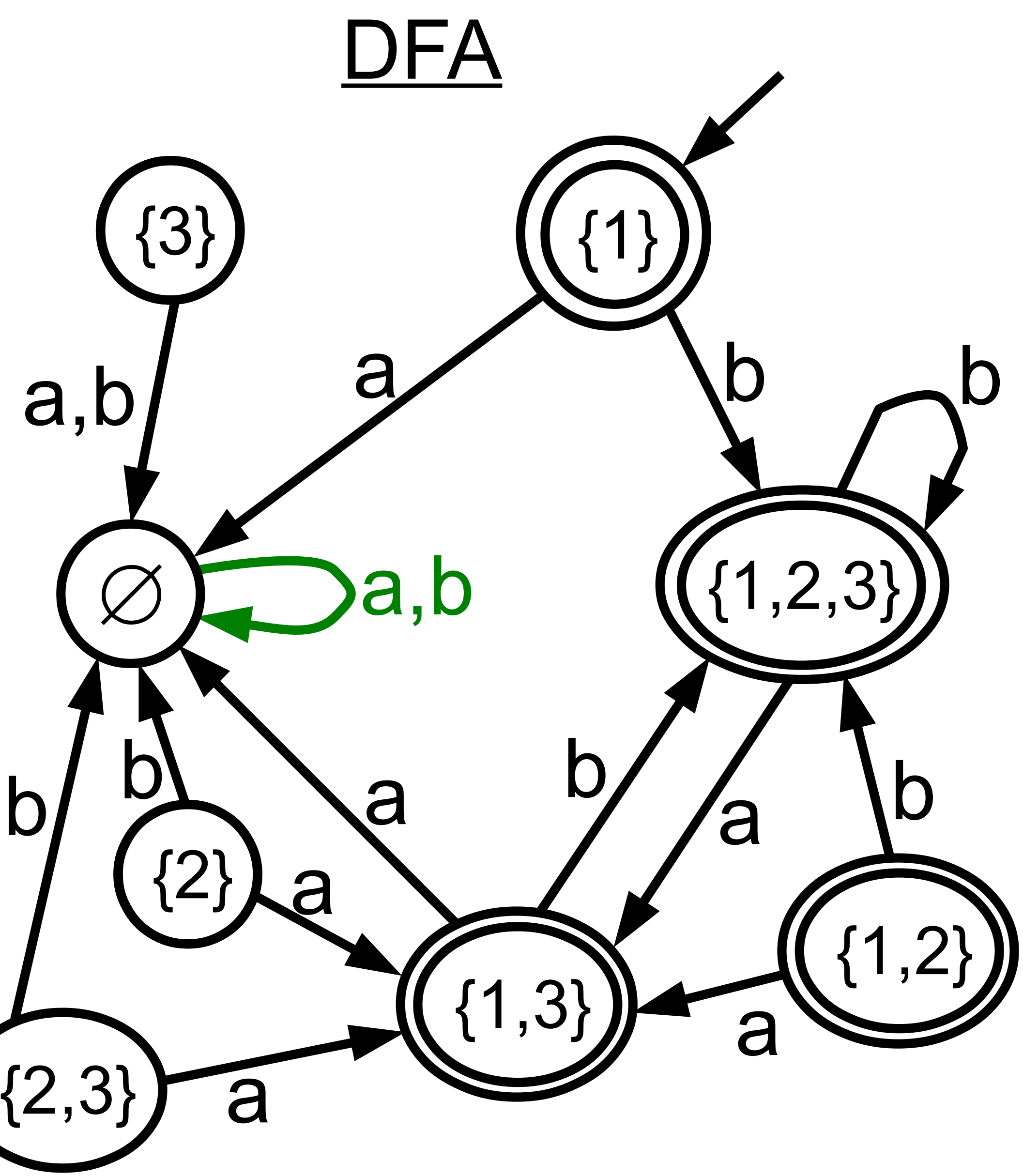
$$= E\left(\{2,3\} \cup \varnothing \cup \varnothing\right) = \{1,2,3\}$$

ANOTHER Example: NFA → DFA conversion



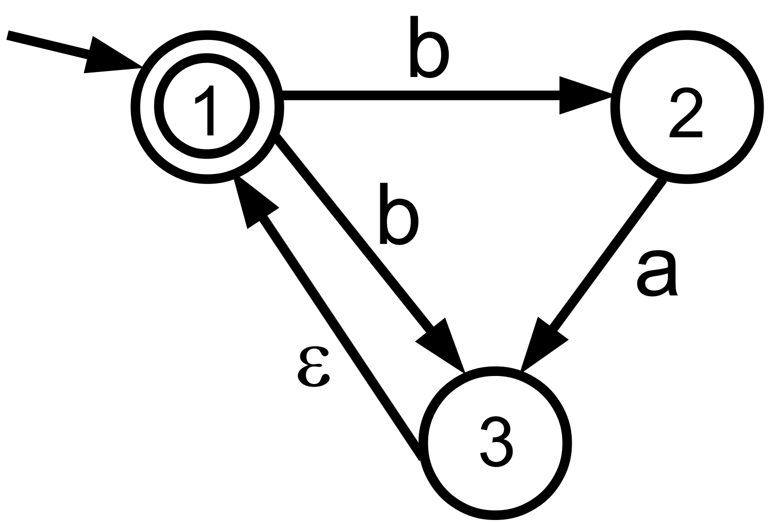
$\delta_{DFA}(\varnothing, a) = \varnothing$

$\delta_{DFA}(\varnothing, b) = \varnothing$



ANOTHER Example: NFA → DFA conversion

NFA



We can delete the unreachable states.

DFA