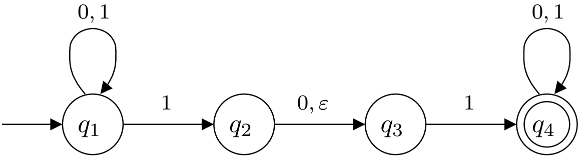
Finite Automata 03

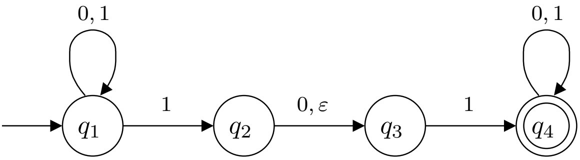
Thumrongsak Kosiyatrakul
tkosiyat@cs.pitt.edu

Nondeterministic Finite Automaton



- Let $\Sigma = \{0, 1\}$
- Different between Deterministic Finite Automaton (DFA) and Nondeterministic Finite Automaton (NFA):
  - DFA always has exactly one exiting transition arrow for each symbol in the alphabet
    - NFA may have none, one, or many exiting arrows for each symbol
  - DFA have no arrow with the label ε
    - NFA may have Zero, one, or many arrows exiting from each state with the label ε.

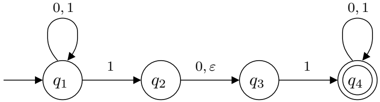
Compute an NFA



- If you encounter a state with multiple way to proceed for a regular input:
    - The machine splits into multiple copies of itself
    - The machines follow all the possibilities in parallel.
    - Each copy of the machine takes one of the possible ways.
- If you encounter a state with an ε symbol as an exiting arrow:
    - Without reading any input, the machine splits into multiple copies.
    - Each follows each of the exiting ε-labeled arrows, and
    - One stays at the current state.

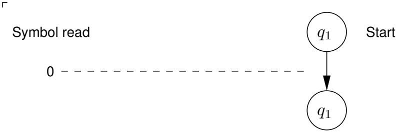
Compute an NFA (Computational Tree)

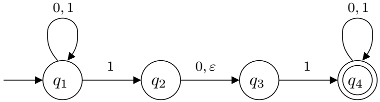
- Computation of the machine on input 010110



Compute an NFA (Computational Tree)

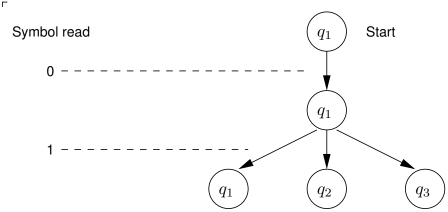
- Computation of the machine on input $\underline{0}10110$

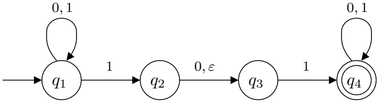




Compute an NFA (Computational Tree)

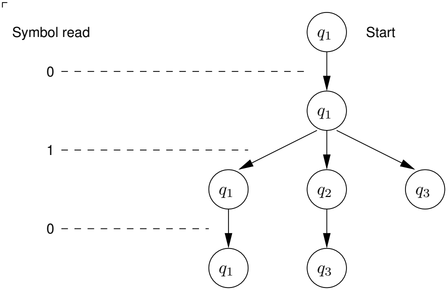
- Computation of the machine on input $0\underline{1}0110$

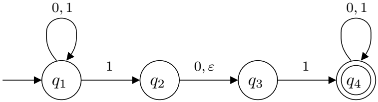




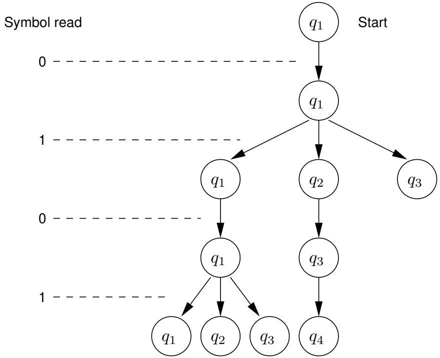
Compute an NFA (Computational Tree)

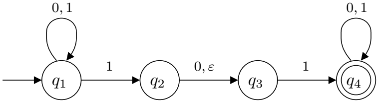
- Computation of the machine on input $01\underline{0}110$



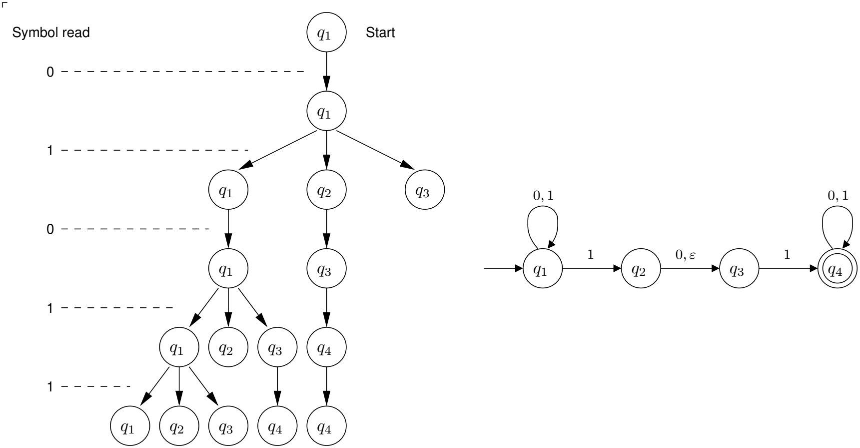


- Computation of the machine on input $010\underline{1}10$



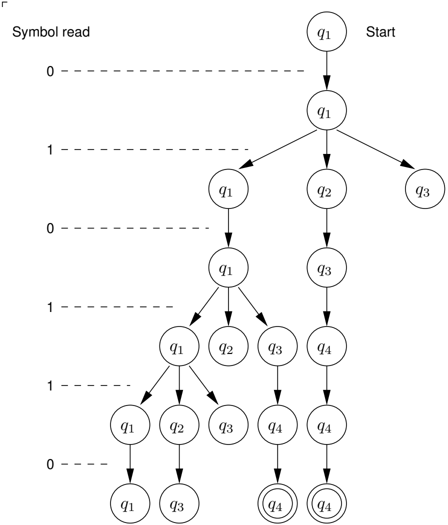


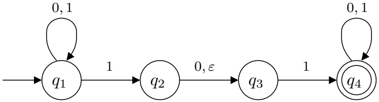
- Computation of the machine on input $0101\underline{1}0$



Compute an NFA (Computational Tree)

- Computation of the machine on input $01011\underline{0}$



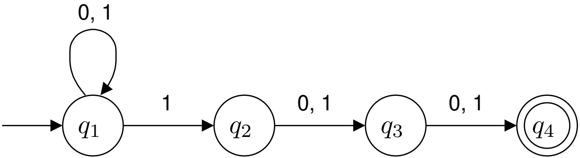


If there is at least one copy in an accept state, the machine accept the input string.

Example

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 in A).

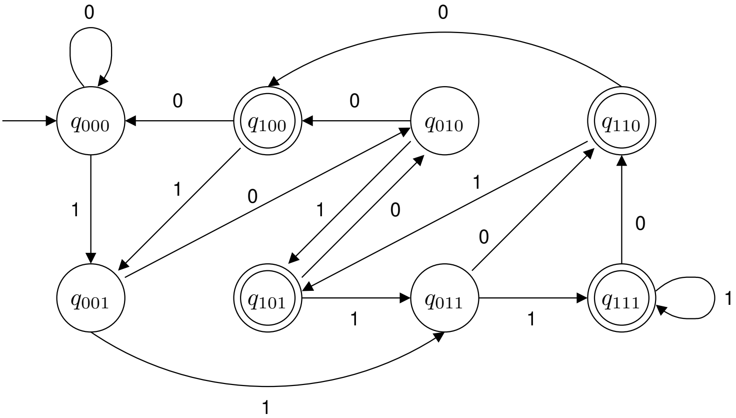
- Design a DFA for this problem is quite complicate
- Design an NFA is easier



- The transition from q_1 to q_2 is our guess that this is the 1 in the third position from the end.
- If our guess is wrong:
  - The input string is shorter, it will end at reject state.
  - The input string is longer, the machine will die but other one remains alive.

Example

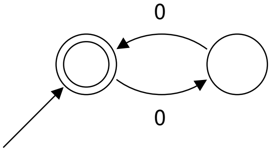
- An equivalent DFA machine



- Name states according to the last three symbols

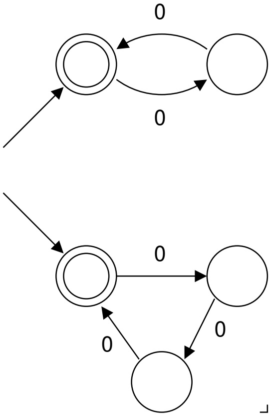
Example

Suppose the alphabet Σ is $\{0\}$. Design a machine that recognizes the language A where A is an empty string or all strings over Σ that their length is a multiple of 2 or 3.



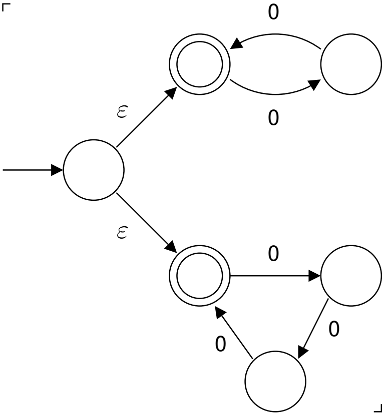
Example

Suppose the alphabet Σ is $\{0\}$. Design a machine that recognizes the language A where A is an empty string or all strings over Σ that their length is a multiple of 2 or 3.



Example

Suppose the alphabet Σ is $\{0\}$. Design a machine that recognizes the language A where A is an empty string or all strings over Σ that their length is a multiple of 2 or 3.



Formal Definition of A Nondeterministic Finite Automaton

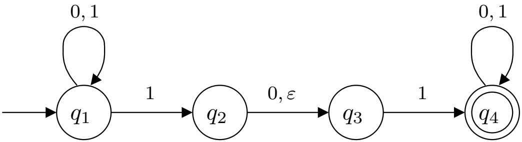
- A **nondeterministic finite automaton** is a 5-tuple

$$(Q, \Sigma, \delta, q0, F)$$

  1. Q is a finite set of states
  2. Σ is a finite alphabet
  3. $\delta : Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$ is the transition function,
     - $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$ and
     - $\mathcal{P}(Q)$ is the **powerset** of Q (set of set of states).
  4. $q_0 \in Q$ is the start state
  5. $F \subseteq Q$ is the set of accept states.

- **Notes**
  - In an NFA, one input symbol can change the state of the machine to multiple states.
    - Split to multiple copies with different current states
    - Example: $\delta(q_0, 1) = \{q_0, q_1\}$

Example



- $Q = \{q_1, q_2, q_3, q_4\}$
- $\Sigma = \{0, 1\}$ and $\Sigma_\varepsilon = \{0, 1, \varepsilon\}$
- δ is given as

δ	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	$\{q_4\}$	$\{q_4\}$	$\emptyset$

  - We treat ε as a regular input symbol
  - If there is no ε transitions, we can ignore the ε column
- q_1 is the start state
- $F = \{q_4\}$

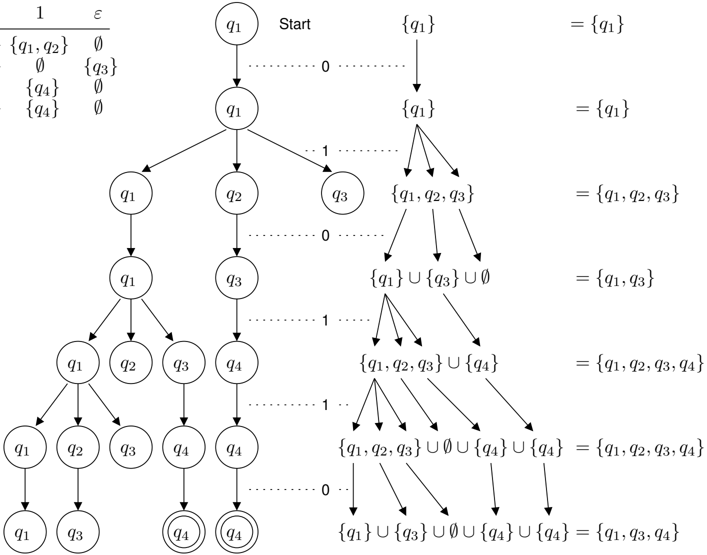
NFA and DFA

- NFA is a slightly different computation model compared to DFA
  - NFA can split into multiple copies
  - NFA may have ε transitions
- Is there a language that can be recognized by an NFA but cannot be recognized by any DFAs?
- In theory of computation, we try to see whether we can capture the behavior of an NFA using a DFA

Simulating and NFA with a DFA

- Simulate 010110

δ	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	$\{q_4\}$	$\{q_4\}$	$\emptyset$



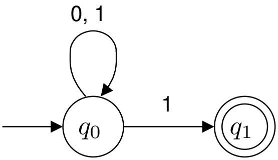
Equivalence of NFAs and DFAs

Every nondeterministic finite automaton has an equivalent deterministic finite automaton.

- Let $N = (Q, \Sigma, \delta, q_0, F)$ be the NFA recognizing some language A
- We are going to construct a DFA $M = (Q', \Sigma, \delta', q_0', F')$ recognizing A
- Let's consider the case where N has no ε transitions.
  1. $Q' = \mathcal{P}(Q)$
  2. $\delta'(R, a) = \bigcup_{r \in R} \delta(r, a)$
  3. $q_0' = \{q_0\}$
  4. $F' = \{R \in Q' \mid R \text{ contains an accept state of } N\}$

Example

- Let Σ be $\{0, 1\}$. The following NFA N recognizes the language A where A is a set of strings that end with a 1.



- $N = (Q, \Sigma, \delta, q_0, F)$
  1. $Q = \{q_0, q_1\}$,
  2. $\Sigma = \{0, 1\}$
  3. δ is given as

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\emptyset$	$\emptyset$

  4. q_0 is the start state
  5. $F = \{q_1\}$

Example

- Construct a DFA $M = (Q', \Sigma, \delta', q_0', F')$
  - $Q' = \mathcal{P}(Q) = \mathcal{P}(\{q_0, q_1\})$

$$Q' = \{\emptyset, \{q_0\}, \{q_1\}, \{q_0, q_1\}\}$$

  - We will construct δ' later
  - $q_0' = \{q_0\}$ where q_0 is the start state of the NFA
  - $F' = \{R \in Q' \mid R \text{ contains an accept state of } N\}$

$$F' = \{\{q_1\}, \{q_0, q_1\}\}$$

where $F = \{q_1\}$

Example

- Let's focus on transition functions
- The transition function δ of the NFA is as follows:

	0	1
q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\emptyset$	$\emptyset$

- Recall that the set of state of the equivalent DFA is the power set of set of state of the NFA

δ'	0	1
$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
$\{q_1\}$	$\emptyset$	$\emptyset$
$\{q_0, q_1\}$	$\{q_0\}$	$\{q_0, q_1\}$

Example

- Machine $M = (Q', \Sigma, \delta', q_0', F')$ equivalent to N can be defined as follows:
  - $Q' = \mathcal{P}(Q) = \{\emptyset, \{q_0\}, \{q_1\}, \{q_0, q_1\}\}$,
  - $\delta'(R, a) = \bigcup_{r \in R} \delta(r, a)$
    - $\delta'(\emptyset, 0) = \bigcup_{r \in \emptyset} \delta(r, 0) = \emptyset$
    - $\delta'(\emptyset, 1) = \bigcup_{r \in \emptyset} \delta(r, 1) = \emptyset$
    - $\delta'(\{q_0\}, 0) = \bigcup_{r \in \{q_0\}} \delta(r, 0) = \delta(q_0, 0) = \{q_0\}$
    - $\delta'(\{q_0\}, 1) = \bigcup_{r \in \{q_0\}} \delta(r, 1) = \delta(q_0, 1) = \{q_0, q_1\}$
    - $\delta'(\{q_1\}, 0) = \bigcup_{r \in \{q_1\}} \delta(r, 0) = \delta(q_1, 0) = \emptyset$
    - $\delta'(\{q_1\}, 1) = \bigcup_{r \in \{q_1\}} \delta(r, 1) = \delta(q_1, 1) = \emptyset$

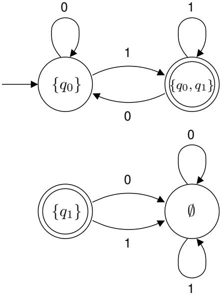
Example

- Machine M (Continue)
  - δ' (Continue)
    - $\delta'(\{q_0, q_1\}, 0) = \bigcup_{r \in \{q_0, q_1\}} \delta(r, 0) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0\} \cup \emptyset = \{q_0\}$
    - $\delta'(\{q_0, q_1\}, 1) = \bigcup_{r \in \{q_0, q_1\}} \delta(r, 0) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$
  - Thus δ' is given by

δ'	0	1
$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
$\{q_1\}$	$\emptyset$	$\emptyset$
$\{q_0, q_1\}$	$\{q_0\}$	$\{q_0, q_1\}$

Example

- The state diagram of the machine $M = (Q', \Sigma, \delta', q'_0, F')$ equivalent to N $(L(M) = L(N))$ is shown below:



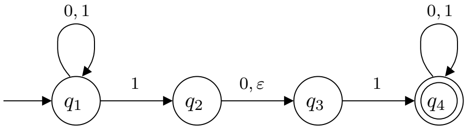
- Without bottom part, it is the same as one of our previous example
- It is okay for a DFA to have unreachable states

NFA to DFA with ε Symbol

- Let $E(R)$ be the collection of states that can be reached from members of R by going only along ε arrows, including the members of R themselves.

$$E(R) = \{q \mid q \text{ can be reached from } R \text{ by travel along } 0 \text{ or more } \varepsilon \text{ arrows}\}$$

- Note that a state q can be reached from its own state q by travel along no ε arrow ($R \subseteq E(R)$)
- Example:



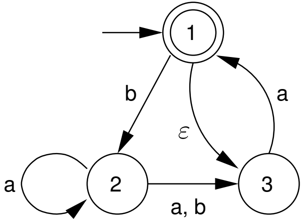
  - $E(\{q_1\}) = \{q_1\}$
  - $E(\{q_2\}) = \{q_2, q_3\}$
  - $E(\{q_1, q_2\}) = \{q_1, q_2, q_3\}$
  - $E(\{q_1, q_3\}) = \{q_1, q_3\}$

NFA to DFA with ε Symbol

- Let NFA $N = (Q, \Sigma, \delta, q_0, F)$ with ε transitions that recognizes a language A
- We can construct a DFA $M = (Q', \Sigma, \delta', q_0', F')$ as
  1. $Q' = \mathcal{P}(Q)$
  2. δ' is given by
  $$\delta'(R, a) = \bigcup_{r \in R} E(\delta(r, a))$$
  3. $q_0' = E(\{q_0\})$
     - If q_0 has no exiting arrow for ε, $q_0' = \{q_0\}$
  4. $F' = \{R \in Q' \mid R \text{ contains an accept state of } N\}$

Example

- Consider the following NFA machine:



- $N = (Q, \Sigma, \delta, q_0, F)$
  - $Q = \{1, 2, 3\}$
  - $\Sigma = \{a, b\}$
  - δ is given by

	a	b	ε
1	$\emptyset$	$\{2\}$	$\{3\}$
2	$\{2, 3\}$	$\{3\}$	$\emptyset$
3	$\{1\}$	$\emptyset$	$\emptyset$

  - $q_0 = 1$
  - $F = \{1\}$

Example

- Machine $M = (Q', \Sigma, \delta', q_0', F')$ equivalent to N:
  - $Q' = \mathcal{P}(Q) = \mathcal{P}(\{1,2,3\})$

$$Q' = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$$

  - Start state is $E(\{1\}) = \{1,3\}$
  - F' is a set of set of states that contain accept states of N ($F = \{1\}$).

$$F' = \{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}\}$$

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

Example

- Let's focus on transition functions
- The transition function δ of the NFA is as follows:

δ	a	b	ε
1	$\emptyset$	$\{2\}$	$\{3\}$
2	$\{2,3\}$	$\{3\}$	$\emptyset$
3	$\{1\}$	$\emptyset$	$\emptyset$

- Recall that the set of state of the equivalent DFA is the power set of set of state of the NFA

δ'	a	b
$\emptyset$	$\emptyset$	$\emptyset$
$\{1\}$	$\emptyset$	$\{2\}$
$\{2\}$	$\{2,3\}$	$\{3\}$
$\{3\}$	$\{1,3\}$	$\emptyset$
$\{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{1,3\}$	$\{1,3\}$	$\{2\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$

Machine M's δ'

$$
\begin{aligned}
\delta'(\emptyset, a) &= \bigcup_{r \in \emptyset} E(\delta(r, a)) \\
&= \emptyset \\
\delta'(\emptyset, b) &= \bigcup_{r \in \emptyset} E(\delta(r, b)) \\
&= \emptyset \\
\delta'(\{1\}, a) &= \bigcup_{r \in \{1\}} E(\delta(r, a)) \\
&= E(\delta(1, a)) \\
&= E(\emptyset) \\
&= \emptyset \\
\delta'(\{1\}, b) &= \bigcup_{r \in \{1\}} E(\delta(r, b)) \\
&= E(\delta(1, b)) \\
&= E(\{2\}) \\
&= \{2\} \\
\delta'(\{2\}, a) &= \bigcup_{r \in \{2\}} E(\delta(r, a)) \\
&= E(\delta(2, a)) \\
&= E(\{2, 3\}) \\
&= \{2, 3\}
\end{aligned}
$$

$$
\begin{aligned}
\delta'(\{2\}, b) &= \bigcup_{r \in \{2\}} E(\delta(r, b)) \\
&= E(\delta(2, b)) \\
&= E(\{3\}) \\
&= \{3\} \\
\delta'(\{3\}, a) &= \bigcup_{r \in \{3\}} E(\delta(r, a)) \\
&= E(\delta(3, a)) \\
&= E(\{1\}) \\
&= \{1, 3\} \\
\delta'(\{3\}, b) &= \bigcup_{r \in \{3\}} E(\delta(r, b)) \\
&= E(\delta(3, b)) \\
&= E(\emptyset) \\
&= \emptyset \\
\delta'(\{1, 2\}, a) &= \bigcup_{r \in \{1,2\}} E(\delta(r, a)) \\
&= E(\delta(1, a)) \cup E(\delta(2, a)) \\
&= \emptyset \cup \{2, 3\} \\
&= \{2, 3\}
\end{aligned}
$$

Machine M's δ'

$$
\begin{aligned}
\delta'(\{1,2\},b) &= \bigcup_{r\in\{1,2\}} E(\delta(r,b)) \\
&= E(\delta(1,b)) \cup E(\delta(2,b)) \\
&= \{2\} \cup \{3\} \\
&= \{2,3\} \\
\delta'(\{1,3\},a) &= \bigcup_{r\in\{1,3\}} E(\delta(r,a)) \\
&= E(\delta(1,a)) \cup E(\delta(3,a)) \\
&= \emptyset \cup \{1,3\} \\
&= \{1,3\} \\
\delta'(\{1,3\},b) &= \bigcup_{r\in\{1,3\}} E(\delta(r,b)) \\
&= E(\delta(1,b)) \cup E(\delta(3,b)) \\
&= \{2\} \cup \emptyset \\
&= \{2\} \\
\delta'(\{2,3\},a) &= \bigcup_{r\in\{2,3\}} E(\delta(r,a)) \\
&= E(\delta(2,a)) \cup E(\delta(3,a)) \\
&= \{2,3\} \cup \{1,3\} \\
&= \{1,2,3\}
\end{aligned}
$$

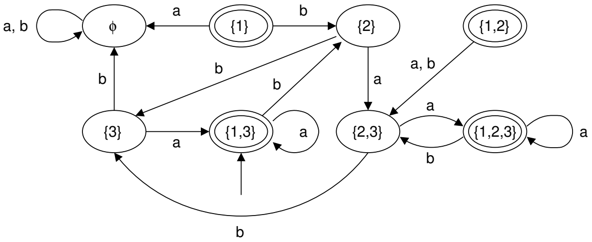
$$
\begin{aligned}
\delta'(\{2,3\},b) &= \bigcup_{r\in\{2,3\}} E(\delta(r,b)) \\
&= E(\delta(2,b)) \cup E(\delta(3,b)) \\
&= \{3\} \cup \emptyset \\
&= \{3\} \\
\delta'(\{1,2,3\},a) &= \bigcup_{r\in\{1,2,3\}} E(\delta(r,a)) \\
&= E(\delta(1,a) \cup E(\delta(2,a)) \cup \\
&\quad E(\delta(3,a)) \\
&= \emptyset \cup \{2,3\} \cup \{1,3\} \\
&= \{1,2,3\} \\
\delta'(\{1,2,3\},b) &= \bigcup_{r\in\{1,2,3\}} E(\delta(r,b)) \\
&= E(\delta(1,b) \cup E(\delta(2,b)) \cup \\
&\quad E(\delta(3,b)) \\
&= \{2\} \cup \{3\} \cup \emptyset \\
&= \{2,3\}
\end{aligned}
$$

Example

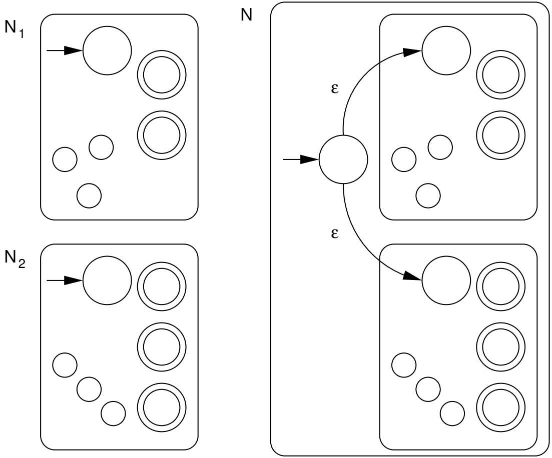
- Recall the transition function of M

	a	b
$\emptyset$	$\emptyset$	$\emptyset$
$\{1\}$	$\emptyset$	$\{2\}$
$\{2\}$	$\{2,3\}$	$\{3\}$
$\{3\}$	$\{1,3\}$	$\emptyset$
$\{1,2\}$	$\{2,3\}$	$\{2,3\}$
$\{1,3\}$	$\{1,3\}$	$\{2\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$

- The state diagram of M



Closure Under Union Operation



- N_1 recognizes a regular language A
- N_2 recognizes a regular language B
- N recognizes $A \cup B$ ($A \cup B$ is regular)

Closure Under Union Operation

- Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognizes A_1.
- Let $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognizes A_2.
- To construct $N = (Q, \Sigma, \delta, q_0, F)$ to recognizes $A_1 \cup A_2$:
  1. $Q = \{q_0\} \cup Q_1 \cup Q_2$
  2. The state q_0 is the start state of N
  3. The set of accept state $F = F_1 \cup F_2$
  4. δ is given by

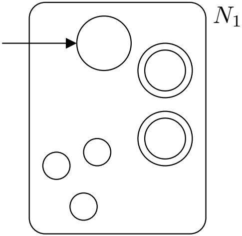
$$\delta(q, a) = \begin{cases} \delta_1(q, a) & q \in Q_1 \\ \delta_2(q, a) & q \in Q_2 \\ \{q_1, q_2\} & q = q_0 \text{ and } a = \varepsilon \\ \emptyset & q = q_0 \text{ and } a \neq \varepsilon \end{cases}$$

Closure Under Union Operation

- Given state diagrams of finite-state machines N_1 and N_2
- To draw a state diagram of a new machine N where $L(N) = L(N_1) \cup L(N_2)$:
  1. Draw the state diagram of N_1 on the top half
  2. Draw the state diagram of N_2 on the bottom half
  3. Add a new start state
  4. Add ε transitions from the new start state to the start states of N_1 and N_2, respectively

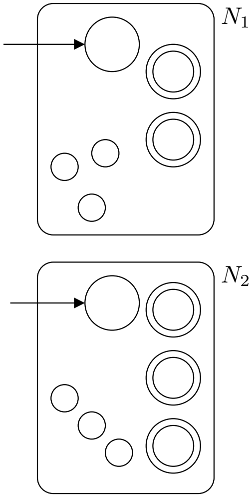
Closure Under Union Operation

- Draw N_1 on the top half



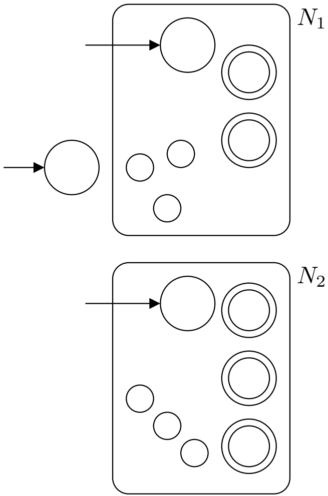
Closure Under Union Operation

- Draw N_2 on the bottom half

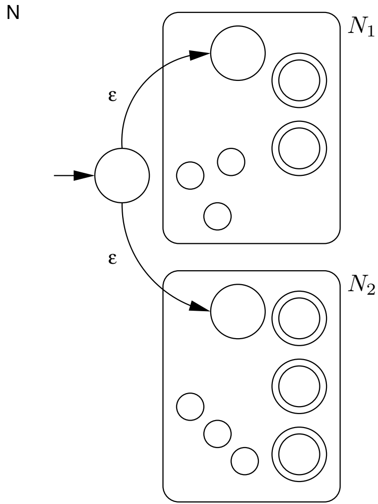


Closure Under Union Operation

- Add new start state



- Add ε transitions



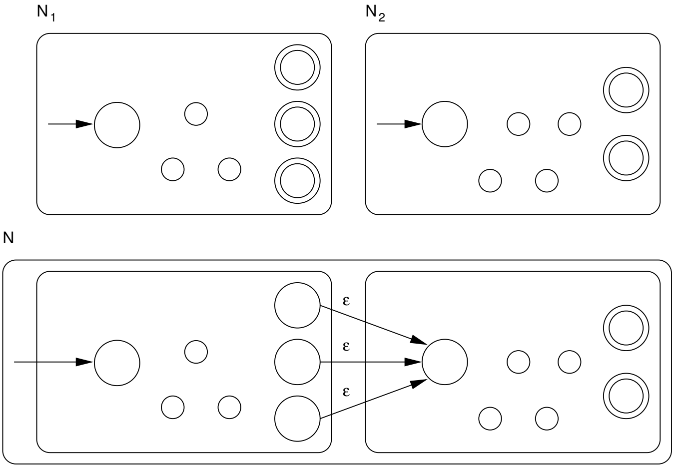
Closure Under Concatenation Operation

- Let N_1 recognizes A and N_2 recognizes B
- Given a string w, how do we know whether $w \in AB$?
- Recall the definition of AB (A concatenated by B)

$$AB = \{xy \mid x \in A \text{ and } y \in B\}$$

- For w to be in AB, w must be divided into two strings x and y where $w = xy$ such that $x \in A$ and $y \in B$
  - If $x \in A$, $x \in L(N_1)$
    - Since $x \in L(N_1)$, by simulating N_1 on input x, the simulation will end in an accept state of N_1 (N_1 accepts x)
  - But if $x \notin A$, simulation will end in a non-accept state of N_1
  - If $y \in B$, $y \in L(N_2)$
    - Since $y \in L(N_2)$, by simulating N_2 on input y, the simulation will end in an accept state of N_2 (N_2 accepts y)
  - But if $y \notin B$, simulation will end in a non-accept state of N_2

Closure Under Concatenation Operation



- N_1 recognizes a regular language A
- N_2 recognizes a regular language B
- N recognizes AB (AB is regular)

Closure Under Concatenation Operation

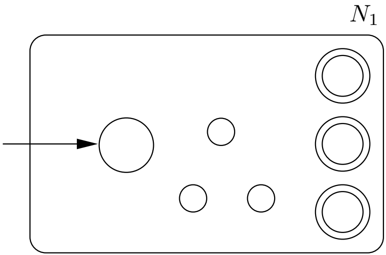
- Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognizes A_1.
- Let $N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognizes A_2.
- To construct $N = (Q, \Sigma, \delta, q_0, F)$ to recognizes $A_1 \circ A_2$:
  1. $Q = Q_1 \cup Q_2$
  2. The state q_1 is the start state of N
  3. The set of accept state $F = F_2$
  4. δ is given by

$$\delta(q, a) = \begin{cases} \delta_1(q, a) & q \in Q_1 \text{ and } q \notin F_1 \\ \delta_2(q, a) & q \in Q_2 \text{ and } a \neq \varepsilon \\ \delta_1(q, a) \cup \{q_2\} & q \in F_1 \text{ and } a = \varepsilon \\ \delta_2(q, a) & q \in Q_2 \end{cases}$$

Closure Under Concatenation Operation

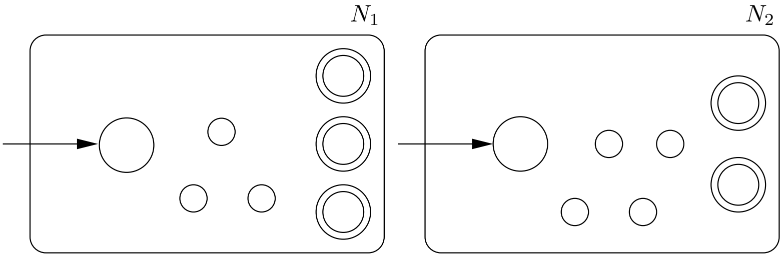
- Given state diagrams of finite-state machines N_1 and N_2
- To draw a state diagram of a new machine N where $L(N) = L(N_1) \circ L(N_2)$:
  1. Draw the state diagram of N_1 on the left side
  2. Draw the state diagram of N_2 on the right side
  3. For every accept state of N_1, add the ε transition to the start state of N_2
  4. Change all accept states of N_1 to non-accept states

- Draw the state diagram of N_1 on the left side



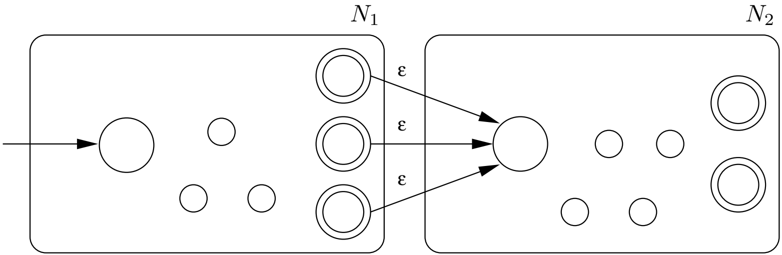
Closure Under Concatenation Operation

- Draw the state diagram of N_2 on the right side



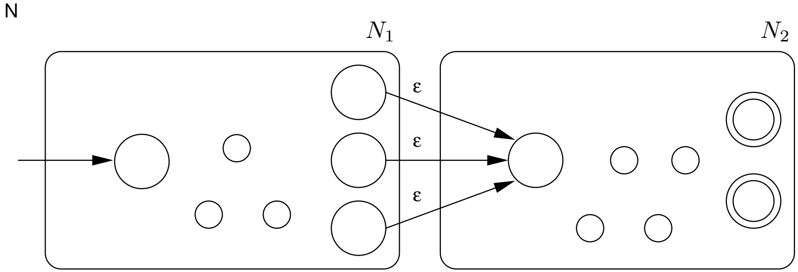
Closure Under Concatenation Operation

- ε from accept states of N_1 to start state of N_2



Closure Under Concatenation Operation

- Accept states of N_1 to non-accept states



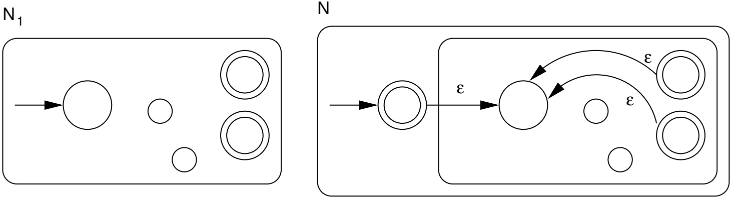
Closure Under Star Operation

- Star operator is similar to concatenation except that it can be concatenated any number of times:

$$A^* = \{x_1x_2x_3 \ldots x_k \mid k \geq 0 \text{ and } x_i \in A\}$$

and

$$\varepsilon \in A^* \qquad \text{for any language } A$$



- N_1 recognizes a regular language A
- N recognizes A^* (A^* is regular)

Closure Under Star Operation

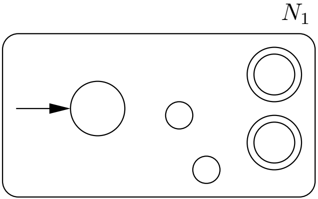
- Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognizes A_1.
- To construct $N = (Q, \Sigma, \delta, q_0, F)$ to recognizes A_1^*:
  1. $Q = \{q_0\} \cup Q_1$
  2. The state q_0 is the start state of N
  3. The set of accept state $F = \{q_0\} \cup F_1$
  4. δ is given by

$$\delta(q,a) = \begin{cases} \delta_1(q,a) & q \in Q_1 \text{ and } q \notin F_1 \\ \delta_1(q,a) & q \in F_1 \text{ and } a \neq \varepsilon \\ \delta_1(q,a) \cup \{q_1\} & q \in F_1 \text{ and } a = \varepsilon \\ \{q_1\} & q = q_0 \text{ and } a = \varepsilon \\ \emptyset & q = q_0 \text{ and } a \notin \varepsilon \end{cases}$$

Closure Under Star Operation

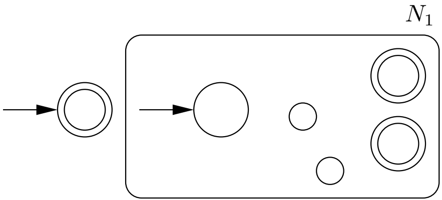
- Given state diagrams of finite-state machines N_1
- To draw a state diagram of a new machine N where $L(N) = L(N_1)^*$:
  1. Draw the state diagram of N_1
  2. Add a new start state and make it an accept state
  3. Add ε transition from the new start state to the start state of N_1
  4. For each **original** accept state of N_1, add ε transition to the **original** start state of N_1

- Draw the state diagram of N_1



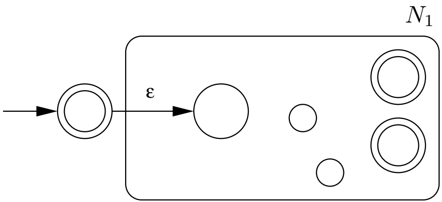
Closure Under Star Operation

- Add a new start state and make it an accept state



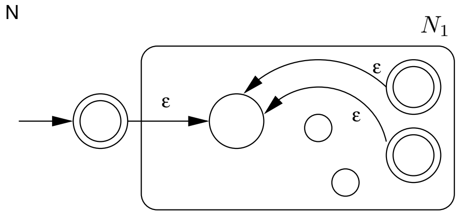
Closure Under Star Operation

- Add ε from thew new start state to the original start state of N_1



Closure Under Star Operation

- For each **original** accept state of N_1, add ε transition to the **original** start state of N_1



Conclusions

- A Nondeterministic Finite Automata (NFA) has an equivalent Deterministic Finite Automata (DFA)
  - The algorithm how to convert from an NFA N to an equivalent DFA D where $L(N) = L(D)$ has been discussed
- The set of all regular languages is closed under union, concatenation, and star operations:
  - If A and B are regular languages, $A \cup B$ is a regular language
  - If A and B are regular languages, AB ($A \circ B$) is a regular language
  - If A is a regular language, A^* is a regular language
- The proof process also gives us an algorithm how to construct DFAs