

Lecture 4 – Nondeterministic Finite Automata (NFA)

COSE215: Theory of Computation

Jihyeok Park



2025 Spring

① Deterministic Finite Automata (DFA)

- Definition
- Transition Diagram and Transition Table
- Extended Transition Function
- Acceptance of a Word
- Language of DFA (Regular Language)
- Examples

1. Nondeterministic Finite Automata (NFA)

Definition

Transition Diagram and Transition Table

Extended Transition Function

Language of NFA

Examples

2. Equivalence of DFA and NFA

DFA \rightarrow NFA

DFA \leftarrow NFA (Subset Construction)

1. Nondeterministic Finite Automata (NFA)

Definition

Transition Diagram and Transition Table

Extended Transition Function

Language of NFA

Examples

2. Equivalence of DFA and NFA

DFA \rightarrow NFA

DFA \leftarrow NFA (Subset Construction)

Definition (Nondeterministic Finite Automaton (NFA))

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

$$N = (Q, \Sigma, \delta, q_0, F)$$

- Q is a finite set of **states**
- Σ is a finite set of **symbols**
- $\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)$ is the **transition function**
- $q_0 \in Q$ is the **initial state**
- $F \subseteq Q$ is the set of **final states**

$$N = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$$

$$\delta(q_0, a) = \{q_0, q_1\}$$

$$\delta(q_1, a) = \{q_2\}$$

$$\delta(q_2, a) = \emptyset$$

$$\delta(q_0, b) = \{q_0\}$$

$$\delta(q_1, b) = \emptyset$$

$$\delta(q_2, b) = \emptyset$$

```
// The definition of NFA
case class NFA(
  states: Set[State],
  symbols: Set[Symbol],
  trans: Map[(State, Symbol), Set[State]],
  initState: State,
  finalStates: Set[State],
)
```

```
// The definition of NFA
case class NFA(
  states: Set[State],
  symbols: Set[Symbol],
  trans: Map[(State, Symbol), Set[State]],
  initState: State,
  finalStates: Set[State],
)
```

```
// An example of NFA
val nfa1: NFA = NFA(
  states      = Set(0, 1, 2),
  symbols     = Set('a', 'b'),
  trans       = Map(
    (0, 'a') -> Set(0, 1), (1, 'a') -> Set(2), // (2, 'a') -> Set(),
    (0, 'b') -> Set(0), // (1, 'b') -> Set(), (2, 'b') -> Set(),
  ).withDefaultValue(Set()),
  initState   = 0,
  finalStates = Set(2),
)
```

You can **skip empty transitions** using `withDefaultValue` method.

$$N_1 = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$$

$$\delta(q_0, a) = \{q_0, q_1\}$$

$$\delta(q_1, a) = \{q_2\}$$

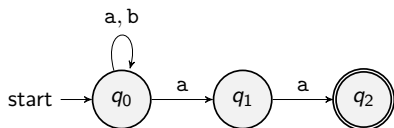
$$\delta(q_2, a) = \emptyset$$

$$\delta(q_0, b) = \{q_0\}$$

$$\delta(q_1, b) = \emptyset$$

$$\delta(q_2, b) = \emptyset$$

Transition Diagram



Transition Table

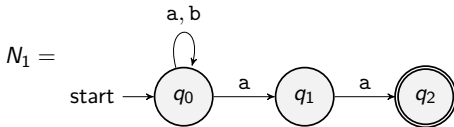
q	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	\emptyset
$*q_2$	\emptyset	\emptyset

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



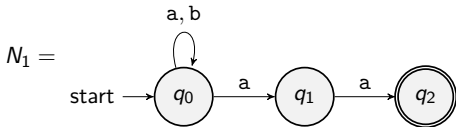
$$\delta^*({q_0}, \text{baa})$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



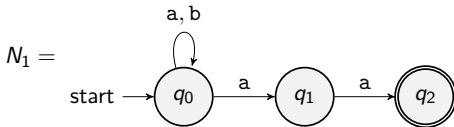
$$\delta^*({q_0}, baa) = \delta^*(\delta(q_0, b), aa) = \delta^*({q_0}, aa)$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



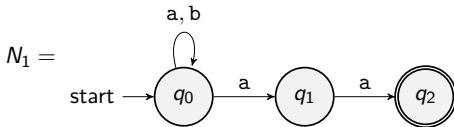
$$\begin{aligned}
 \delta^*({q_0}, baa) &= \delta^*(\delta(q_0, b), aa) &&= \delta^*({q_0}, aa) \\
 &= \delta^*(\delta(q_0, a), a) &&= \delta^*({q_0, q_1}, a)
 \end{aligned}$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



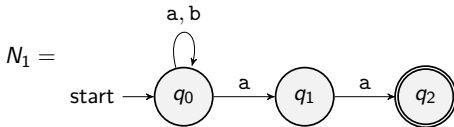
$$\begin{aligned}
 \delta^*({q_0}, baa) &= \delta^*(\delta(q_0, b), aa) &&= \delta^*({q_0}, aa) \\
 &= \delta^*(\delta(q_0, a), a) &&= \delta^*({q_0}, q_1), a) \\
 &= \delta^*(\delta(q_0, a) \cup \delta(q_1, a), \epsilon) &&= \delta^*({q_0}, q_1, q_2}, \epsilon)
 \end{aligned}$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



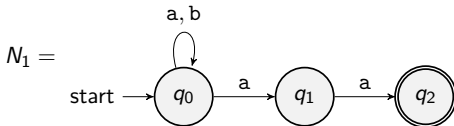
$$\begin{aligned}
 \delta^*({q_0}, baa) &= \delta^*(\delta(q_0, b), aa) &&= \delta^*({q_0}, aa) \\
 &= \delta^*(\delta(q_0, a), a) &&= \delta^*({q_0}, q_1, a) \\
 &= \delta^*(\delta(q_0, a) \cup \delta(q_1, a), \epsilon) &&= \delta^*({q_0}, q_1, q_2}, \epsilon) \\
 &= \{q_0, q_1, q_2\}
 \end{aligned}$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



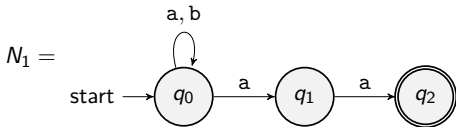
$$\delta^*({q_0}, aba)$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



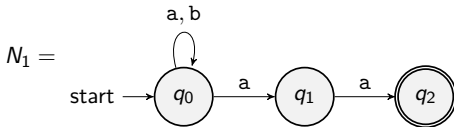
$$\delta^*({q_0}, aba) = \delta^*(\delta(q_0, a), ba) = \delta^*({q_0, q_1}, ba)$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



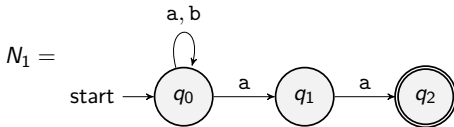
$$\begin{aligned}
 \delta^*({q_0}, aba) &= \delta^*(\delta(q_0, a), ba) &&= \delta^*({q_0, q_1}, ba) \\
 &= \delta^*(\delta(q_0, b) \cup \delta(q_1, b), a) &&= \delta^*({q_0}, a)
 \end{aligned}$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$



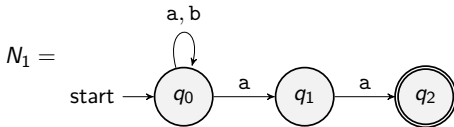
$$\begin{aligned}
 \delta^*({q_0}, aba) &= \delta^*(\delta(q_0, a), ba) &&= \delta^*({q_0, q_1}, ba) \\
 &= \delta^*(\delta(q_0, b) \cup \delta(q_1, b), a) &&= \delta^*({q_0}, a) \\
 &= \delta^*(\delta(q_0, a), \epsilon) &&= \delta^*({q_0, q_1}, \epsilon)
 \end{aligned}$$

Definition (Extended Transition Function)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **extended transition function** $\delta^* : \mathcal{P}(Q) \times \Sigma^* \rightarrow \mathcal{P}(Q)$ is defined as follows:

- **(Basis Case)** $\delta^*(S, \epsilon) = S$
- **(Induction Case)** $\delta^*(S, aw) = \delta^*(\bigcup_{q \in S} \delta(q, a), w)$

where $S \subseteq Q$, $a \in \Sigma$, and $w \in \Sigma^*$

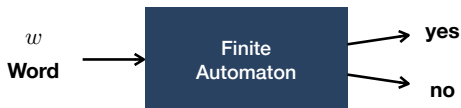


$$\begin{aligned}
 \delta^*({q_0}, aba) &= \delta^*(\delta(q_0, a), ba) &&= \delta^*({q_0, q_1}, ba) \\
 &= \delta^*(\delta(q_0, b) \cup \delta(q_1, b), a) &&= \delta^*({q_0}, a) \\
 &= \delta^*(\delta(q_0, a), \epsilon) &&= \delta^*({q_0, q_1}, \epsilon) \\
 &= {q_0, q_1}
 \end{aligned}$$

```
// The type definition of words
type Word = String
case class NFA(...):
  ...
  // The extended transition function of NFA
  def extTrans(qs: Set[State], w: Word): Set[State] = w match
    case ""      => qs
    case x <| w => extTrans(qs.flatMap(q => trans(q, x)), w)

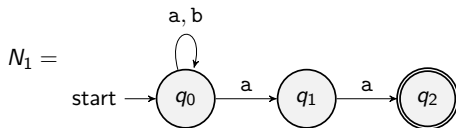
// An example transition for a word "baa"
nfa1.extTrans(Set(0), "baa") // Set(0, 1, 2)

// An example transition for a word "aba"
nfa1.extTrans(Set(0), "aba") // Set(0, 1)
```



Definition (Acceptance of a Word)

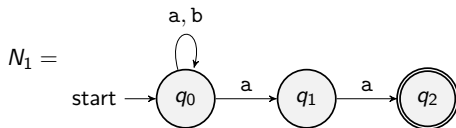
For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, we say that N **accepts** a word $w \in \Sigma^*$ if and only if $\delta^*({q_0}, w) \cap F \neq \emptyset$





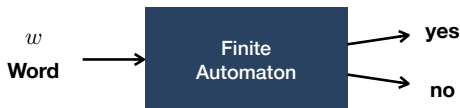
Definition (Acceptance of a Word)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, we say that N **accepts** a word $w \in \Sigma^*$ if and only if $\delta^*({q_0}, w) \cap F \neq \emptyset$



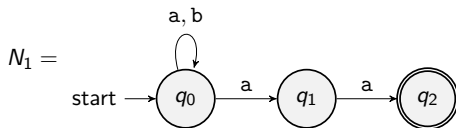
$$\delta^*({q_0}, baa) \cap F = \{q_0, q_1, q_2\} \cap \{q_2\} = \{q_2\} \neq \emptyset$$

It means that N_1 **accepts** baa.



Definition (Acceptance of a Word)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, we say that N **accepts** a word $w \in \Sigma^*$ if and only if $\delta^*({q_0}, w) \cap F \neq \emptyset$

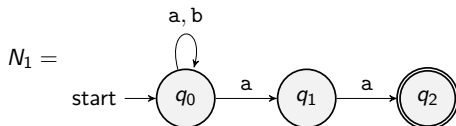


$$\delta^*({q_0}, \text{baa}) \cap F = \{q_0, q_1, q_2\} \cap \{q_2\} = \{q_2\} \neq \emptyset$$

It means that N_1 **accepts** baa.

$$\delta^*({q_0}, \text{aba}) \cap F = \{q_0, q_1\} \cap \{q_2\} = \emptyset$$

It means that N_1 does **not accept** aba.



```
case class NFA(...):
  ...
  // The acceptance of a word by NFA
  def accept(w: Word): Boolean =
    extTrans(Set(initState), w).intersect(finalStates).nonEmpty

// An example acceptance of a word "baa"
nfa1.accept("baa") // true

// An example non-acceptance of a word "aba"
nfa1.accept("aba") // false
```

Definition (Language of NFA)

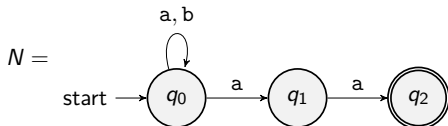
For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **language** of N is defined as:

$$L(N) = \{w \in \Sigma^* \mid N \text{ accepts } w\}$$

Definition (Language of NFA)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **language** of N is defined as:

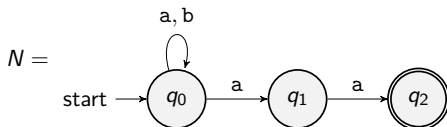
$$L(N) = \{w \in \Sigma^* \mid N \text{ accepts } w\}$$



Definition (Language of NFA)

For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, the **language** of N is defined as:

$$L(N) = \{w \in \Sigma^* \mid N \text{ accepts } w\}$$



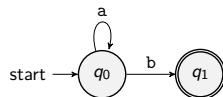
$$L(N) = \{waa \mid w \in \{a, b\}^*\}$$

Examples

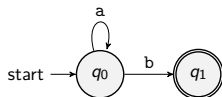
$$L = \{a^n b \mid n \geq 0\}$$

Examples

$$L = \{a^n b \mid n \geq 0\}$$

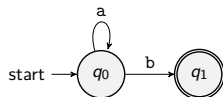


$$L = \{a^n b \mid n \geq 0\}$$

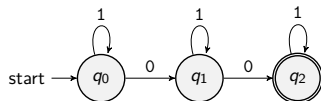


$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two } 0\text{'s}\}$$

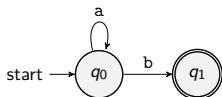
$$L = \{a^n b \mid n \geq 0\}$$



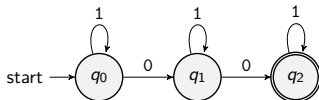
$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two } 0\text{'s}\}$$



$$L = \{a^n b \mid n \geq 0\}$$



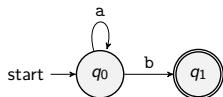
$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two 0's}\}$$



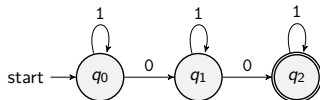
$$L = \{w \in \{a, b\}^* \mid w \text{ contains three consecutive a's}\}$$

Examples

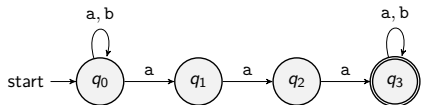
$$L = \{a^n b \mid n \geq 0\}$$



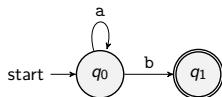
$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two } 0\text{'s}\}$$



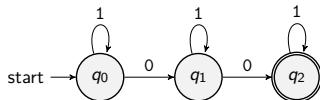
$$L = \{w \in \{a, b\}^* \mid w \text{ contains three consecutive } a\text{'s}\}$$



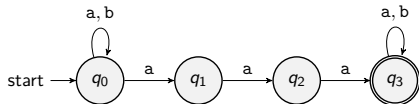
$$L = \{a^n b \mid n \geq 0\}$$



$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two 0's}\}$$



$$L = \{w \in \{a, b\}^* \mid w \text{ contains three consecutive a's}\}$$

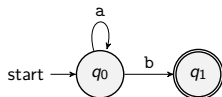


$$L = \{w \in \{0, 1\}^* \mid \mathbb{N}(w) \equiv 0 \pmod{3}\}$$

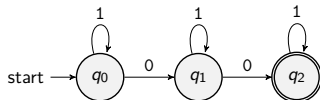
where $\mathbb{N}(w)$ is the natural number represented by w in binary

Examples

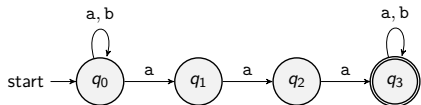
$$L = \{a^n b \mid n \geq 0\}$$



$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two 0's}\}$$

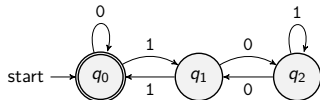


$$L = \{w \in \{a, b\}^* \mid w \text{ contains three consecutive a's}\}$$

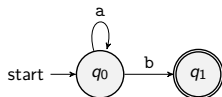


$$L = \{w \in \{0, 1\}^* \mid \mathbb{N}(w) \equiv 0 \pmod{3}\}$$

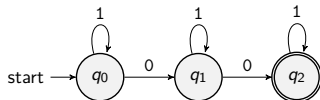
where $\mathbb{N}(w)$ is the natural number represented by w in binary



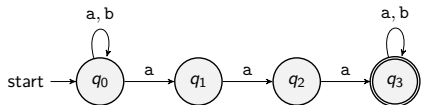
$$L = \{a^n b \mid n \geq 0\}$$



$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two 0's}\}$$

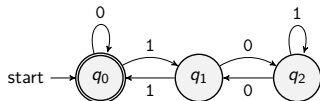


$$L = \{w \in \{a, b\}^* \mid w \text{ contains three consecutive a's}\}$$



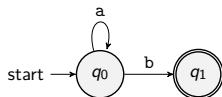
$$L = \{w \in \{0, 1\}^* \mid \mathbb{N}(w) \equiv 0 \pmod{3}\}$$

where $\mathbb{N}(w)$ is the natural number represented by w in binary

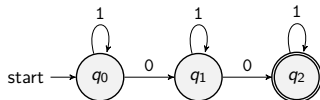


$$L = \{a^n b^n \mid n \geq 0\}$$

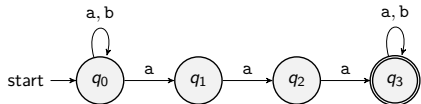
$$L = \{a^n b \mid n \geq 0\}$$



$$L = \{w \in \{0, 1\}^* \mid w \text{ contains exactly two 0's}\}$$

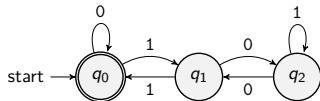


$$L = \{w \in \{a, b\}^* \mid w \text{ contains three consecutive a's}\}$$



$$L = \{w \in \{0, 1\}^* \mid \mathbb{N}(w) \equiv 0 \pmod{3}\}$$

where $\mathbb{N}(w)$ is the natural number represented by w in binary



$$L = \{a^n b^n \mid n \geq 0\}$$

IMPOSSIBLE (\nexists NFA N . $L(N) = L$)

1. Nondeterministic Finite Automata (NFA)

Definition

Transition Diagram and Transition Table

Extended Transition Function

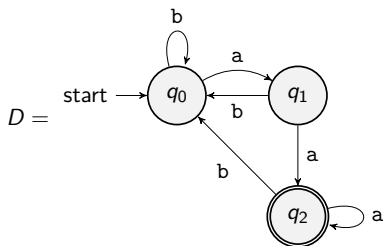
Language of NFA

Examples

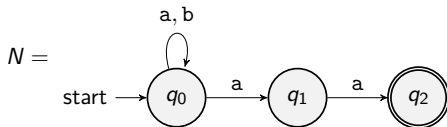
2. Equivalence of DFA and NFA

DFA \rightarrow NFA

DFA \leftarrow NFA (Subset Construction)

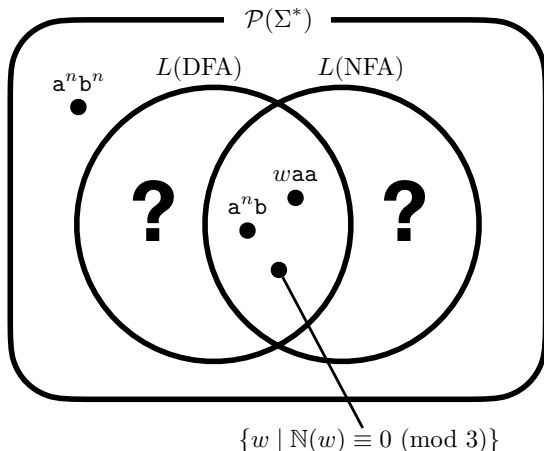


$$L(D) = \{waa \mid w \in \{a, b\}^*\} = L(N)$$

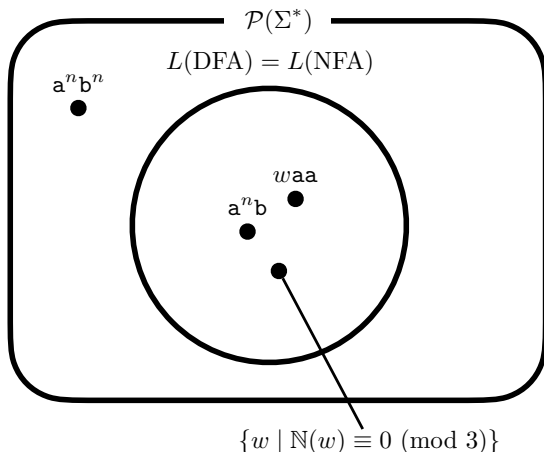


Equivalence of DFA and NFA

Is there any language that is the language of a DFA but not the language of an NFA, or vice versa?



Is there any language that is the language of a DFA but not the language of an NFA, or vice versa? **No! DFA and NFA are equivalent.**



Theorem (Equivalence of DFA and NFA)

A language L is the language $L(D)$ of a DFA D if and only if L is the language $L(N)$ of an NFA N .

Theorem (Equivalence of DFA and NFA)

A language L is the language $L(D)$ of a DFA D if and only if L is the language $L(N)$ of an NFA N .

Proof) By the following two theorems.

Theorem (DFA to NFA)

For a given DFA $D = (Q, \Sigma, \delta, q, F)$, \exists NFA N . $L(D) = L(N)$.

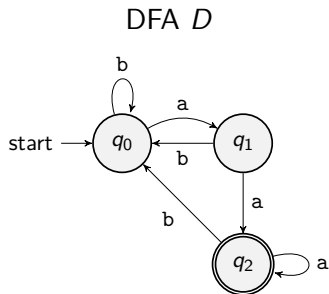
It means ① we can always construct an NFA equivalent to a given DFA.

Theorem (NFA to DFA – Subset Construction)

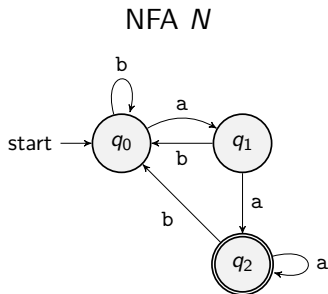
For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, \exists DFA D . $L(D) = L(N)$.

It means ② we can always construct a DFA equivalent to a given NFA.

① Let's learn how to construct an NFA equivalent to a given DFA.



q	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_2	q_0
$*q_2$	q_2	q_0



q	a	b
$\rightarrow q_0$	$\{q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_0\}$
$*q_2$	$\{q_2\}$	$\{q_0\}$

Theorem (DFA to NFA)

For a given DFA $D = (Q, \Sigma, \delta_D, q_0, F)$, \exists NFA N . $L(D) = L(N)$.

Proof) Consider the following NFA:

$$N = (Q, \Sigma, \delta_N, q_0, F)$$

where $\forall q \in Q. \forall a \in \Sigma$.

$$\delta_N(q, a) = \{\delta_D(q, a)\}$$

Then,

$$\begin{aligned}
 w \in L(D) &\iff \delta_D^*(q_0, w) \in F && (\because \text{definition of } L(D)) \\
 &\iff \{\delta_D^*(q_0, w)\} \cap F \neq \emptyset && (\because \text{set theory}) \\
 &\iff \delta_N^*(\{q_0\}, w) \cap F \neq \emptyset && (\because \text{lemma in the next slide}) \\
 &\iff w \in L(N) && (\because \text{definition of } L(N)) \quad \square
 \end{aligned}$$

Lemma

$$\forall q \in Q. \forall w \in \Sigma^*. \delta_N^*({q}, w) = \{\delta_D^*(q, w)\}.$$

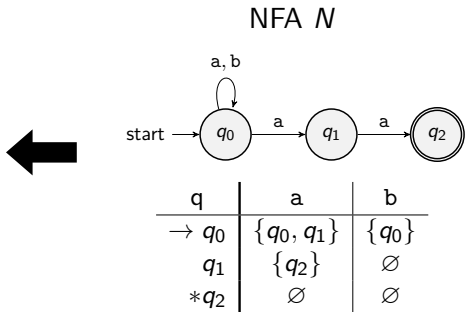
Proof) By induction on the **length of word**.

- **(Base Case)** $\delta_N^*({q}, \epsilon) = \{q\} = \{\delta_D^*(q, \epsilon)\}$.
- **(Inductive Case)** Assume it holds for w (I.H.).

$$\begin{aligned} \delta_N^*({q}, aw) &= \delta_N^*(\delta_N(q, a), w) && (\because \text{definition of } \delta_N^*) \\ &= \delta_N^*({\delta_D(q, a)}, w) && (\because \text{definition of } \delta_N) \\ &= \{\delta_D^*(\delta_D(q, a), w)\} && (\because \text{I.H.}) \\ &= \{\delta_D^*(q, aw)\} && (\because \text{definition of } \delta_D^*) \quad \square \end{aligned}$$

② Let's learn how to construct a DFA equivalent to a given NFA.

We will use **subsets of states** in the NFA as **states** in the DFA.
(This is called the **subset construction** approach.)

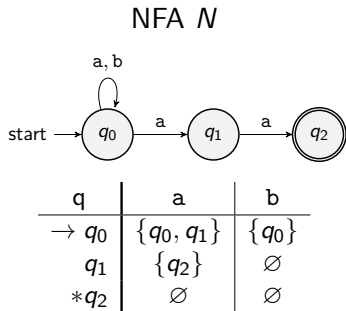


② Let's learn how to construct a DFA equivalent to a given NFA.

We will use **subsets of states** in the NFA as **states** in the DFA.
 (This is called the **subset construction** approach.)

DFA D

q	a	b
\emptyset		
$\{q_0\}$		
$\{q_1\}$		
$\{q_2\}$		
$\{q_0, q_1\}$		
$\{q_0, q_2\}$		
$\{q_1, q_2\}$		
$\{q_0, q_1, q_2\}$		



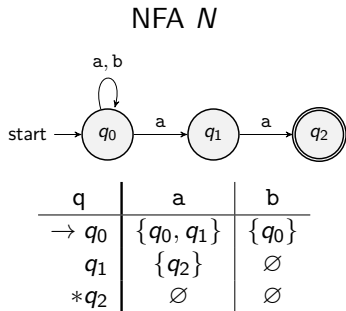
Let's enumerate all possible **set of NFA states** as **DFA states**.

② Let's learn how to construct a DFA equivalent to a given NFA.

We will use **subsets of states** in the NFA as **states** in the DFA.
(This is called the **subset construction** approach.)

DFA D

q	a	b
\emptyset		
$\rightarrow \{q_0\}$		
$\{q_1\}$		
$*\{q_2\}$		
$\{q_0, q_1\}$		
$*\{q_0, q_2\}$		
$*\{q_1, q_2\}$		
$*\{q_0, q_1, q_2\}$		



The **initial** DFA state is $\{q_0\}$.

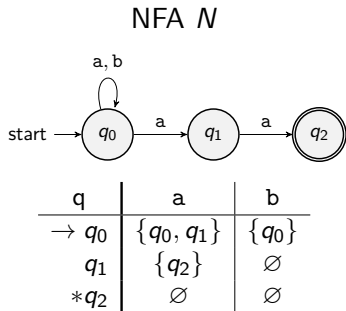
Any DFA state containing NFA final states (i.e., q_2) is a **final** DFA state.

② Let's learn how to construct a DFA equivalent to a given NFA.

We will use **subsets of states** in the NFA as **states** in the DFA.
(This is called the **subset construction** approach.)

DFA D

q	a	b
\emptyset	\emptyset	\emptyset
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\{q_2\}$	\emptyset
$*\{q_2\}$	\emptyset	\emptyset
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	$\{q_2\}$	\emptyset
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$



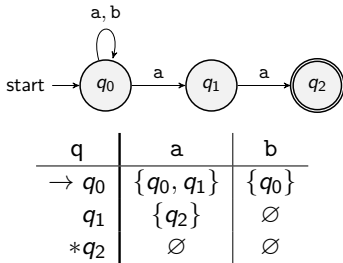
The **transition** in the DFA is defined by point-wise transition in the NFA.

② Let's learn how to construct a DFA equivalent to a given NFA.

We will use **subsets of states** in the NFA as **states** in the DFA.
(This is called the **subset construction** approach.)

DFA D

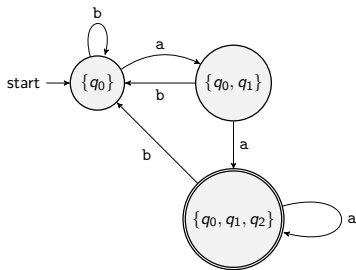
q	a	b
\emptyset	\emptyset	\emptyset
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\{q_2\}$	\emptyset
$*\{q_2\}$	\emptyset	\emptyset
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	$\{q_2\}$	\emptyset
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$

NFA N 

Let's remove **unreachable** DFA states from the initial DFA state.

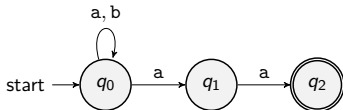
② Let's learn how to construct a DFA equivalent to a given NFA.

DFA D



q	a	b
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$

NFA N



q	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	\emptyset
$*q_2$	\emptyset	\emptyset



Theorem (NFA to DFA – Subset Construction)

For a given NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$, \exists DFA D . $L(D) = L(N)$.

Proof) Define a DFA

$$D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$$

where

- $Q_D = \mathcal{P}(Q_N)$
- $\forall S \in Q_D. \forall a \in \Sigma.$

$$\delta_D(S, a) = \bigcup_{q \in S} \delta_N(q, a)$$

- $F_D = \{S \in Q_D \mid S \cap F_N \neq \emptyset\}$

Then,

$$\begin{aligned}
 w \in L(D) &\iff \delta_D^*(\{q_0\}, w) \in F_D && (\because \text{definition of } L(D)) \\
 &\iff \delta_D^*(\{q_0\}, w) \cap F_N \neq \emptyset && (\because \text{definition of } F_D) \\
 &\iff \delta_N^*(\{q_0\}, w) \cap F_N \neq \emptyset && (\because \text{lemma in the next slide}) \\
 &\iff w \in L(N) && (\because \text{definition of } L(N)) \quad \square
 \end{aligned}$$

Lemma

$$\forall S \in Q_D. \forall w \in \Sigma^*. \delta_D^*(S, w) = \delta_N^*(S, w)$$

Proof) By induction on the **length of word**.

- **(Base Case)** $\delta_D^*(S, \epsilon) = S = \delta_N^*(S, \epsilon)$.
- **(Inductive Case)** Assume it holds for w (I.H.).

$$\delta_D^*(S, aw) = \delta_D^*(\delta_D(S, a), w) \quad (\because \text{definition of } \delta_D^*)$$

$$= \delta_D^*(\bigcup_{q \in S} \delta_N(q, a), w) \quad (\because \text{definition of } \delta_D)$$

$$= \delta_N^*(\bigcup_{q \in S} \delta_N(q, a), w) \quad (\because \text{I.H.})$$

$$= \delta_N^*(S, aw) \quad (\because \text{definition of } \delta_D^*) \quad \square$$

1. Nondeterministic Finite Automata (NFA)

Definition

Transition Diagram and Transition Table

Extended Transition Function

Language of NFA

Examples

2. Equivalence of DFA and NFA

DFA \rightarrow NFA

DFA \leftarrow NFA (Subset Construction)

- ϵ -Nondeterministic Finite Automata (ϵ -NFA)

Jihyeok Park

jihyeok_park@korea.ac.kr

<https://plrg.korea.ac.kr>