Lecture 4 – Nondeterministic Finite Automata (NFA)

COSE215: Theory of Computation

Jihyeok Park

PLRG

2024 Spring

Recall

1. Deterministic Finite Automata (DFA)
   - Definition
   - Transition Diagram and Transition Table
   - Extended Transition Function
   - Acceptance of a Word
   - Language of DFA (Regular Language)
   - Examples

Contents

Contents

PLRG

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

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

$$\begin{array}{lll} \delta(q_0, \mathtt{a}) = \{q_0, q_1\} & \delta(q_1, \mathtt{a}) = \{q_2\} & \delta(q_2, \mathtt{a}) = \varnothing \\ \delta(q_0, \mathtt{b}) = \{q_0\} & \delta(q_1, \mathtt{b}) = \varnothing & \delta(q_2, \mathtt{b}) = \varnothing \end{array}$$

Definition of NFA

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

```scala
// 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),
)
```

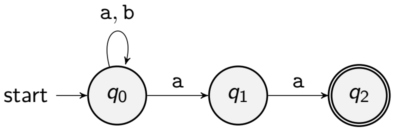
You can **skip empty transitions** using `withDefaultValue` method.

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

$$\delta(q_0, \mathtt{a}) = \{q_0, q_1\} \qquad \delta(q_1, \mathtt{a}) = \{q_2\} \qquad \delta(q_2, \mathtt{a}) = \varnothing$$

$$\delta(q_0, \mathtt{b}) = \{q_0\} \qquad \delta(q_1, \mathtt{b}) = \varnothing \qquad \delta(q_2, \mathtt{b}) = \varnothing$$

Transition Diagram



Transition Table

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

Extended Transition Function

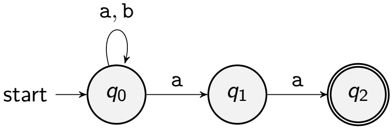
Definition (Extended Transition Function)

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

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

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

$N_1 =$



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

Extended Transition Function

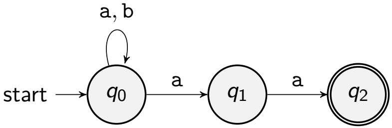
Definition (Extended Transition Function)

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

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

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

$N_1 =$



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

Extended Transition Function

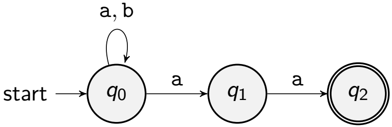
```scala
// The type definition of words
type Word = String
case class NFA(...):
  ...
  // The extended transition function of NFA
  def extTrans(qs: Set[State], w: Word): Set[State] = w match
    case ""     => qs
    case x <| w => extTrans(qs.flatMap(q => trans(q, x)), w)

// 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)
```

Acceptance of a Word

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 $\delta^*(\{q_0\}, w) \cap F \neq \varnothing$

$N_1 =$



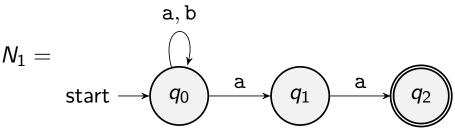
$$\delta^*(\{q_0\}, \mathtt{baa}) \cap F = \{q_0, q_1, q_2\} \cap \{q_2\} = \{q_2\} \neq \varnothing$$

It means that N_1 **accepts** `baa`.

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

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

Acceptance of a Word




```
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
```

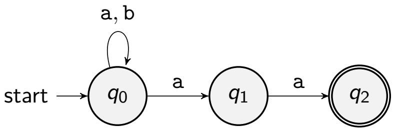
Language of NFA

Definition (Language of NFA)

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

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

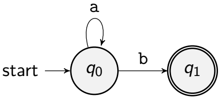
$N =$



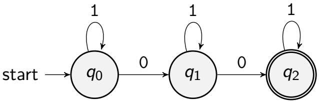
$$L(N) = \{w\mathtt{aa} \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$$

Examples

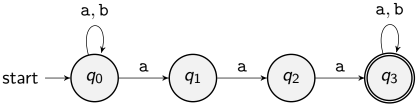
$L = \{\mathtt{a}^n\mathtt{b} \mid n \geq 0\}$



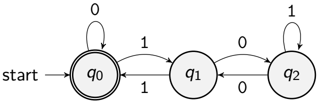
$L = \{w \in \{0,1\}^* \mid w$ contains exactly two 0's $\}$



$L = \{w \in \{\mathtt{a},\mathtt{b}\}^* \mid w$ 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



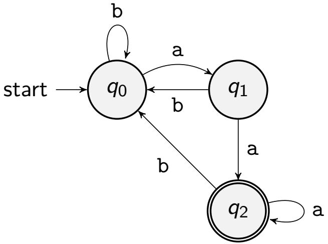
$L = \{\mathtt{a}^n\mathtt{b}^n \mid n \geq 0\}$

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

Contents

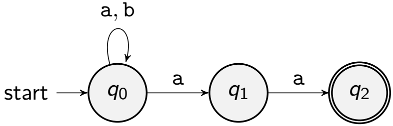
Equivalence of DFA and NFA

$D =$



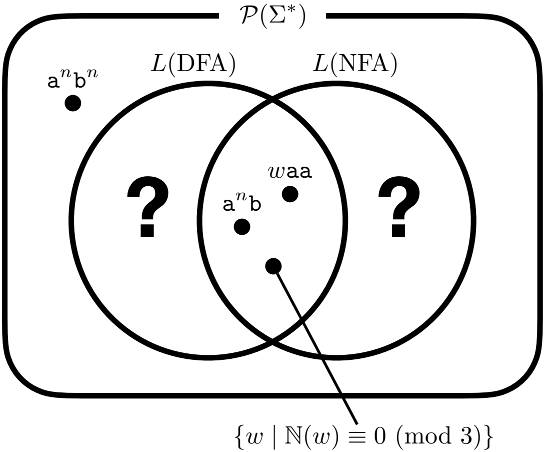
$$L(D) = \{w\mathtt{aa} \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\} = L(N)$$

$N =$



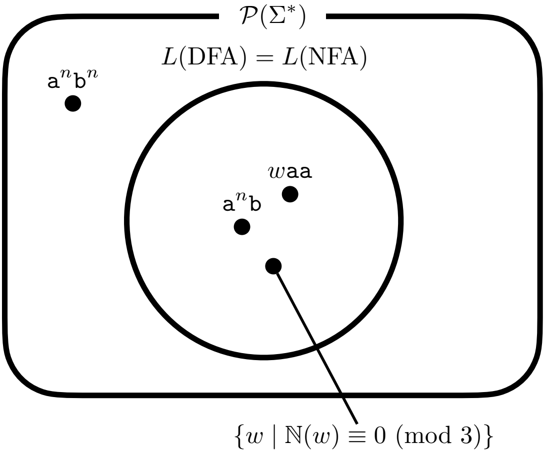
Equivalence of DFA and NFA

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



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? No! DFA and NFA are **equivalent**.



Equivalence of DFA and NFA

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)

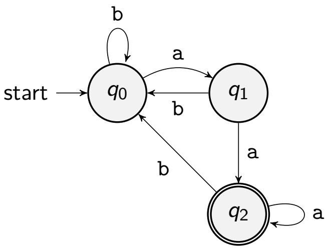
For a given NFA $N = (Q, \Sigma, \delta, q_0, F)$, $\exists$ DFA D. $L(D) = L(N)$.

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

DFA $\rightarrow$ NFA – Example

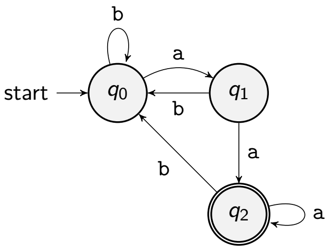
① **Let's learn how to construct an NFA equivalent to a given DFA.**

DFA D



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

NFA N



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 $\forall q \in Q.\ \forall x \in \Sigma.$

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

Then,

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

PLRG

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.).

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

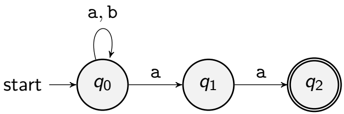
DFA $\leftarrow$ NFA (Subset Construction) – Example

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
$\varnothing$	$\varnothing$	$\varnothing$
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\{q_2\}$	$\varnothing$
$*\{q_2\}$	$\varnothing$	$\varnothing$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	$\{q_2\}$	$\varnothing$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$

NFA N

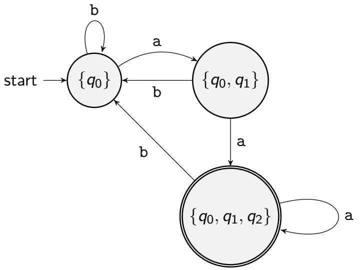


q	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\varnothing$
$*q_2$	$\varnothing$	$\varnothing$

DFA $\leftarrow$ NFA (Subset Construction) – Example

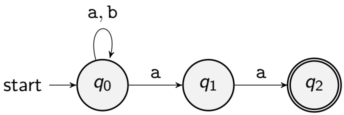
② **Let's learn how to construct a DFA equivalent to a given NFA.**
We will use **subsets of states** in the NFA as **states** in the DFA.
(This is called the **subset construction** approach.)

DFA D



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

NFA N



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

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

- $Q_D = \mathcal{P}(Q_N)$
- $\forall S \in Q_D.\ \forall x \in \Sigma.$
$$\delta_D(S, x) = \bigcup_{q \in S} \delta_N(q, x)$$
- $F_D = \{S \in Q_D \mid S \cap F_N \neq \varnothing\}$

Then,

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

DFA $\leftarrow$ NFA (Subset Construction) – Lemma

PLRG

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.).

$$\begin{aligned}
\delta_D^*(S, xw) &= \delta_D^*(\delta_D(S, x), w) && (\because \text{definition of } \delta_D^*) \\
&= \delta_D^*(\textstyle\bigcup_{q\in S} \delta_N(q, x), w) && (\because \text{definition of } \delta_D) \\
&= \delta_N^*(\textstyle\bigcup_{q\in S} \delta_N(q, x), w) && (\because \text{I.H.}) \\
&= \delta_N^*(S, xw) && (\because \text{definition of } \delta_N^*) \quad \square
\end{aligned}$$

Summary

PLRG

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)

Next Lecture

- ϵ-Nondeterministic Finite Automata (ϵ-NFA)

Jihyeok Park
jihyeok_park@korea.ac.kr
https://plrg.korea.ac.kr