Lecture 4 – Nondeterministic Finite Automata (NFA)

COSE215: Theory of Computation

Jihyeok Park

PLRG

2023 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

Definition of NFA

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**

$$N = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$

$$\begin{array}{lll} \delta(q_0, 0) = \{q_0, q_1\} & \delta(q_1, 0) = \{q_2\} & \delta(q_2, 0) = \varnothing \\ \delta(q_0, 1) = \{q_0\} & \delta(q_1, 1) = \varnothing & \delta(q_2, 1) = \varnothing \end{array}$$

Definition of NFA

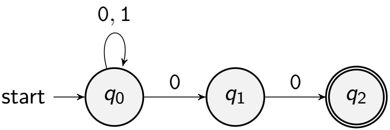
```scala
// The type definitions of states and symbols
type State = Int
type Symbol = Char
// The definition of NFA
case class NFA(
  states: Set[State],
  symbols: Set[Symbol],
  trans: Map[(State, Symbol), Set[State]],
  initState: State,
  finalStates: Set[State],
)
// An example of NFA
val nfa: NFA = NFA(
  states  = Set(0, 1, 2),
  symbols = Set('0', '1'),
  trans = Map(
    (0, '0') -> Set(0, 1), (1, '0') -> Set(2), (2, '0') -> Set(),
    (0, '1') -> Set(0),    (1, '1') -> Set(),  (2, '1') -> Set(),
  ),
  initState  = 0,
  finalStates = Set(2),
)
```

Transition Diagram and Transition Table

$$N = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$

$$\delta(q_0, 0) = \{q_0, q_1\} \qquad \delta(q_1, 0) = \{q_2\} \qquad \delta(q_2, 0) = \varnothing$$
$$\delta(q_0, 1) = \{q_0\} \qquad \delta(q_1, 1) = \varnothing \qquad \delta(q_2, 1) = \varnothing$$

Transition Diagram



Transition Table

q	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\varnothing$
$*q_2$	$\varnothing$	$\varnothing$

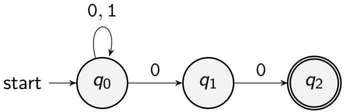
Extended Transition Function

Definition (Extended Transition Function)

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

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

$N =$



$$
\begin{aligned}
\delta^*(q_0, 100) &= \textstyle\bigcup_{q' \in \delta(q_0,1)} \delta^*(q', 00) && = \delta^*(q_0, 00) \\
&= \textstyle\bigcup_{q' \in \delta(q_0,0)} \delta^*(q', 0) && = \delta^*(q_0, 0) \cup \delta^*(q_1, 0) \\
&= \textstyle\bigcup_{q' \in \delta(q_0,0)} \delta^*(q', \epsilon) \cup \bigcup_{q' \in \delta(q_1,0)} \delta^*(q', \epsilon) && = \delta^*(q_0, \epsilon) \cup \delta^*(q_1, \epsilon) \cup \delta^*(q_2, \epsilon) \\
&= \{q_0, q_1, q_2\}
\end{aligned}
$$

Extended Transition Function

```scala
// The type definition of words
type Word = String

// A helper function to extract first symbol and rest of word
object `<|` { def unapply(w: Word) = w.headOption.map((_, w.drop(1))) }

// The extended transition function of NFA
def extTrans(nfa: NFA)(q: State, w: Word): Set[State] = w match
  case "" => Set(q)
  case a <| x => nfa.trans(q, a).flatMap(p => extTrans(nfa)(p, x))

// An example transition for a word "100"
extTrans(nfa)(0, "100") // Set(0, 1, 2)
```

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

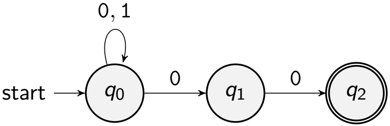
```scala
// The acceptance of a word by NFA
def accept(nfa: NFA)(w: Word): Boolean =
  val curStates: Set[State] = extTrans(nfa)(nfa.initState, w)
  curStates.intersect(nfa.finalStates).nonEmpty

// An example acceptance of a word "100"
accept(nfa)("100") // true
```

Definition (Language of NFA)

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

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

$N =$



$$L(N) = \{w00 \mid w \in \{0, 1\}^*\}$$

Examples

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

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

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

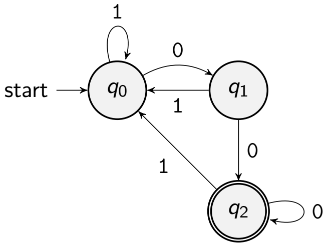
- $L = \{w \in \{0, 1^*\} \mid w \text{ has three consecutive } 0's\}$

- $L = \{w \in \{0, 1^*\} \mid \mathbb{N}(w) \equiv 0 \pmod 3\}$
  where $\mathbb{N}(w)$ is a natural number represented by w.

- $L = \{\mathtt{a}^n\mathtt{b}^n \mid n \geq 0\}$ – IMPOSSIBLE ($\nexists$ NFA N. $L(N) = L$)

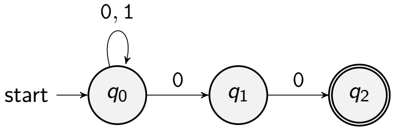
Equivalence of DFA and NFA

$D =$



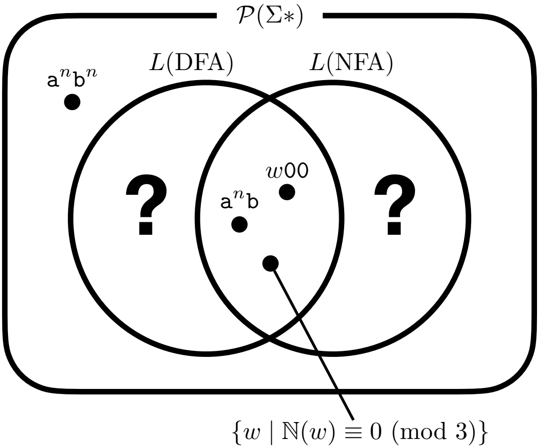
$$L(D) = \{w00 \mid w \in \{0, 1\}^*\} = L(N)$$

$N =$



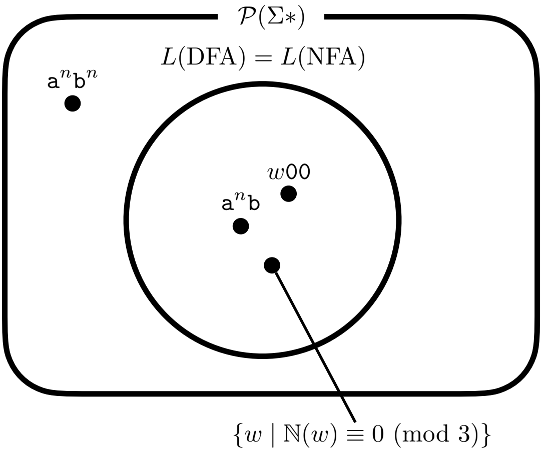
Equivalence of DFA and NFA

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. $L(D) = L(N)$.

Theorem (NFA to DFA – Subset Construction)

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

Theorem (DFA to NFA)

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

Proof) Define an NFA

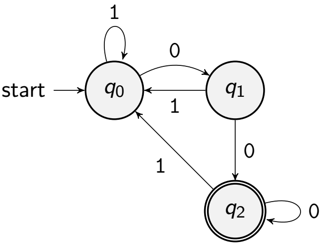
$$N = (Q, \Sigma, \delta_N, q_0, F)$$

where

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

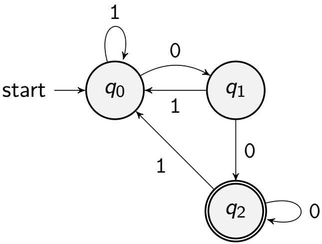
$$\delta_N(q, a) = \{\delta_D(q, a)\}$$

DFA D



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

NFA N



q	0	1
$\rightarrow q_0$	$\{q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_0\}$
$*q_2$	$\{q_2\}$	$\{q_0\}$

DFA $\to$ NFA – Proof

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.).

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

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

Theorem (NFA to DFA – Subset Construction)

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

Proof) Define a DFA

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

where

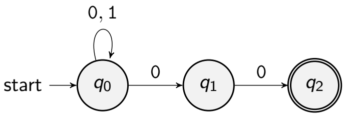
- $Q_D = \mathcal{P}(Q)$
- $\forall S \in Q_D. \ \forall a \in \Sigma.$
$$\delta_D(S, a) = \bigcup_{q \in S} \delta_N(q, a)$$
- $F_D = \{S \in Q_D \mid S \cap F \neq \varnothing\}$

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

DFA D

q	0	1
$\varnothing$	$\varnothing$	$\varnothing$
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\{q_2\}$	$\varnothing$
$*\{q_2\}$	$\varnothing$	$\varnothing$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	$\{q_2\}$	$\varnothing$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$

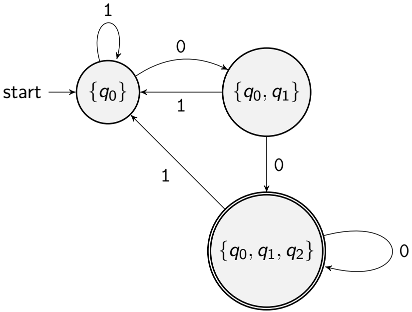
NFA N



q	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\varnothing$
$*q_2$	$\varnothing$	$\varnothing$

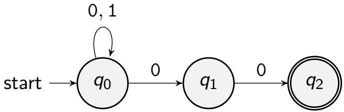
DFA $\leftarrow$ NFA (Subset Construction) – Example

DFA D



q	0	1
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$

NFA N



q	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\varnothing$
$*q_2$	$\varnothing$	$\varnothing$

DFA ← NFA (Subset Construction) – Proof

Lemma

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

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

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

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

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

Summary

1. Nondeterministic Finite Automata (NFA)
   - Definition
   - Transition Diagram and Transition Table
   - Extended Transition Function
   - Language of NFA
   - Examples
   - 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