Lecture 10 – Equivalence and Minimization of Finite Automata
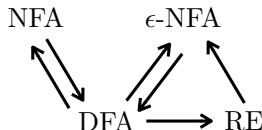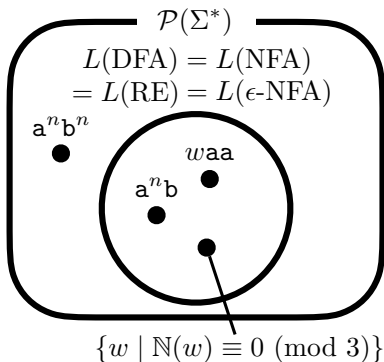
COSE215: Theory of Computation

Jihyeok Park

PLRG

2024 Spring

- Closure Properties of Regular Languages
- Pumping Lemma for Regular Languages

- How to test whether two finite automata are **equivalent**?
- How to **minimize** a finite automaton?

Contents

◆PLRG

Contents

- Are the following two DFA **equivalent** (i.e., $L(D_0) = L(D_1)$)?

- Yes, because $L(D_0) = L(D_1) = \{w\mathtt{b} \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$.
- We first define the **equivalence of states** and utilize it to test the **equivalence of DFA**.

Equivalence of States ($\equiv$)

> **Definition (Equivalence of States ($\equiv$))**
>
> For a given DFA D, q_i is **equivalent** to q_j (i.e., $q_i \equiv q_j$) if and only if
>
> $$\forall w \in \Sigma^*. \, \delta^*(q_i, w) \in F \iff \delta^*(q_j, w) \in F$$
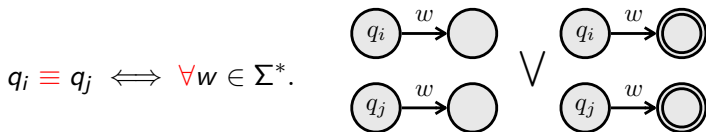
$$q_i \equiv q_j \iff \forall w \in \Sigma^*.$$

However, it is difficult to make it as an algorithm. Let's consider $q_i \not\equiv q_j$:

$$q_i \not\equiv q_j \iff \exists w \in \Sigma^*. \, (\delta^*(q_i, w) \in F \iff \delta^*(q_j, w) \notin F)$$

$$q_i \not\equiv q_j \iff \exists w \in \Sigma^*.$$

Distinguishable States ($\not\equiv$)

PLRG
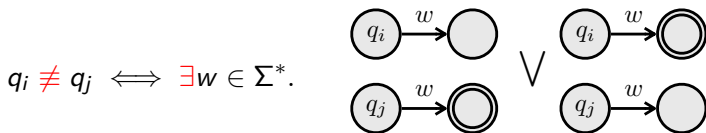
We can *inductively* test q_i is **distinguishable** with q_j (i.e., $q_i \not\equiv q_j$):

- **(Basis Case)** $w = \epsilon$

$$\boxed{q_i} \wedge \boxed{\!\boxed{q_j}\!} \quad \bigvee \quad \boxed{\!\boxed{q_i}\!} \wedge \boxed{q_j}$$

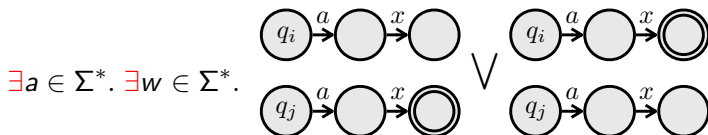$$(\; \delta^*(q_i, \epsilon) \in F \iff \delta^*(q_j, \epsilon) \notin F \;)$$

$$\iff \quad (\; q_i \in F \qquad \iff q_j \notin F \qquad)$$

- **(Induction Case)** $w = ax$

$$\exists a \in \Sigma^*. \; \exists w \in \Sigma^*. \quad \underset{q_j}{\overset{q_i}{\vphantom{x}}} \quad \bigvee \quad \underset{q_j}{\overset{q_i}{\vphantom{x}}}$$
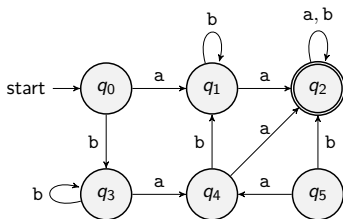
$$\exists a \in \Sigma. \; \exists x \in \Sigma^*. \; (\; \delta^*(q_i, ax) \in F \iff \delta^*(q_j, ax) \notin F \;)$$

$$\iff \quad \exists a \in \Sigma. \; \exists x \in \Sigma^*. \; (\; \delta^*(\delta(q_i, a), x) \in F \iff \delta^*(\delta(q_j, a), x) \notin F \;)$$

$$\iff \quad \exists a \in \Sigma. \; \; \delta(q_i, a) \not\equiv \delta(q_j, a)$$

Distinguishable States ($\not\equiv$)

Definition (Distinguishable States ($\not\equiv$))

For a given DFA D, q_i is **distinguishable** with q_j (i.e., $q_i \not\equiv q_j$) iff

- **(Basis Case)** $q_i \in F \iff q_j \notin F$.
- **(Induction Case)** $\exists \mathtt{a} \in \Sigma.\ \delta(q_i, \mathtt{a}) \not\equiv \delta(q_j, \mathtt{a})$.

$q_2 \not\equiv q_4$
$(\because q_2 \in F \land q_4 \notin F)$

$q_1 \not\equiv q_3$
$(\because \delta(q_1, \mathtt{a}) = q_2 \not\equiv q_4 = \delta(q_3, \mathtt{a}))$

$q_0 \not\equiv q_4$
$(\because \delta(q_0, \mathtt{b}) = q_3 \not\equiv q_1 = \delta(q_4, \mathtt{b}))$

q	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_2	q_1
$*q_2$	q_2	q_2
q_3	q_4	q_3
q_4	q_2	q_1
q_5	q_4	q_2

(Basis case) $w = \epsilon$.
$$q_i \in F \iff q_j \notin F$$

(Induction case) $w = ax$.
$$\exists a \in \Sigma.\ \delta(q_i, a) \not\equiv \delta(q_j, a)$$

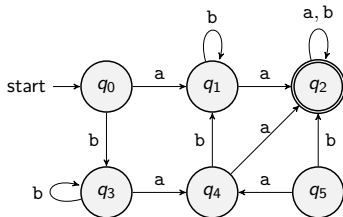| q	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_2	q_1
$*q_2$	q_2	q_2
q_3	q_4	q_3
q_4	q_2	q_1
q_5	q_4	q_2

(Basis case) $w = \epsilon$.
$$q_i \in F \iff q_j \notin F$$

(Induction case) $w = ax$.
$$\exists a \in \Sigma. \; \delta(q_i, a) \not\equiv \delta(q_j, a)$$

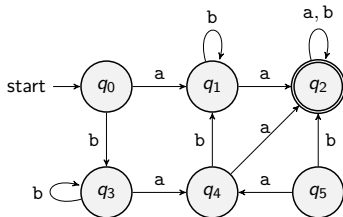| q	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_2	q_1
$*q_2$	q_2	q_2
q_3	q_4	q_3
q_4	q_2	q_1
q_5	q_4	q_2

(Basis case) $w = \epsilon$.
$$q_i \in F \iff q_j \notin F$$

(Induction case) $w = ax$.
$$\exists a \in \Sigma.\ \delta(q_i, a) \not\equiv \delta(q_j, a)$$

$$q_0 \equiv q_3 \wedge q_1 \equiv q_4$$

Equivalence of Finite Automata

Theorem (Equivalence of Finite Automata)

Consider two DFA $D = (Q, \Sigma, \delta, q_0, F)$ and $D' = (Q', \Sigma, \delta', q_0', F')$. Then,
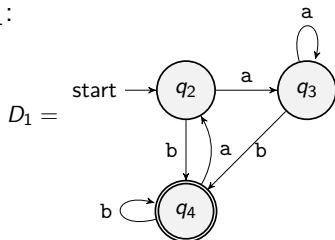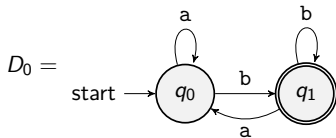
$$L(D) = L(D') \iff q_0 \equiv q_0'$$

in a DFA $D'' = (Q \uplus Q', \Sigma, \delta'', q_0, F \uplus F')$ where

$$\forall q'' \in Q \uplus Q'. \ \delta''(q, a) = \begin{cases} \delta(q'', a) & q'' \in Q \\ \delta'(q'', a) & q'' \in Q' \end{cases}$$

Proof) By the definition of equivalence of states, we have

$$
\begin{aligned}
& L(D) = L(D') \\
\iff & \forall w \in \Sigma^*. \ (D \text{ accepts } w \iff D' \text{ accepts } w) \\
\iff & \forall w \in \Sigma^*. \ (\delta^*(q_0, w) \in F \iff \delta'^*(q_0', w) \in F') \\
\iff & \forall w \in \Sigma^*. \ (\delta''^*(q_0, w) \in F \cup F' \iff \delta''^*(q_0', w) \in F \cup F') \\
\iff & q_0 \equiv q_0' \text{ in } D''
\end{aligned}
$$

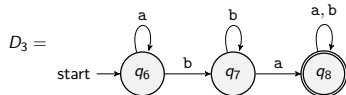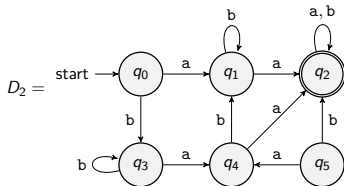Let's test the equivalence of D_0 and D_1:

Let's perform the **table-filling algorithm**:

- $q_0 \equiv q_2 \equiv q_3$
- $q_1 \equiv q_4$

$$q_0 \equiv q_2 \implies L(D_0) = L(D_1) = \{w\mathtt{b} \mid w \in \{\mathtt{a}, \mathtt{b}\}^*\}$$

Let's test the equivalence of D_2 and D_3:

Let's perform the **table-filling algorithm**:

	q_0	q_1	q_2	q_3	q_4	q_5	q_6	q_7
q_1	x							
q_2	x	x						
q_3		x	x					
q_4	x		x	x				
q_5	x	x	x	x	x			
q_6	x	x	x	x	x	x		
q_7	x		x	x		x	x	
q_8	x	x		x	x	x	x	x

- $q_0 \equiv q_3$
- $q_1 \equiv q_4 \equiv q_7$
- $q_2 \equiv q_8$
- q_5
- q_6

$q_0 \not\equiv q_6 \implies L(D_2) \neq L(D_3)$ ($\because$ ba $\notin L(D_2)$ but ba $\in L(D_3)$)

Contents

- Is it possible to **minimize** a DFA?

- Yes, let's utilize **equivalence classes** $Q/_\equiv$ of states defined with $\equiv$.
- Note that $\equiv$ is an **equivalence relation**:
  - reflexive: $\forall q \in Q.\ q \equiv q$
  - symmetric: $\forall q, q' \in Q.\ q \equiv q' \Leftrightarrow q' \equiv q$
  - transitive: $\forall q, q', q'' \in Q.\ q \equiv q' \wedge q' \equiv q'' \Leftrightarrow q \equiv q''$

Minimization Algorithm

For a given DFA $D = (Q, \sigma, \delta, q_0, F)$, the **minimization** algorithm is:

1. Remove all **unreachable states** from the initial state q_0.
2. Partition the remaining states into **equivalence classes**:

$$Q/_\equiv = \{[q]_\equiv \mid q \in Q\}$$

where the **equivalence class** of a state q is defined as:

$$[q]_\equiv = \{q' \in Q \mid q \equiv q'\}$$

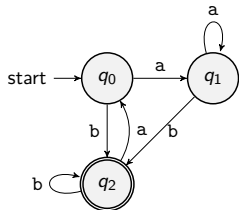3. Construct a new DFA $D/_\equiv = (Q/_\equiv, \Sigma, \delta/_\equiv, [q_0]_\equiv, F/_\equiv)$ where
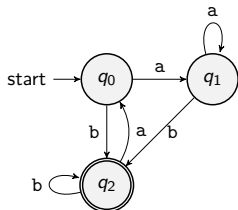
   - $\delta/_\equiv : Q/_\equiv \times \Sigma \to Q/_\equiv$ is defined by:

     $$\forall q \in Q. \ \forall a \in \Sigma. \ \delta/_\equiv([q]_\equiv, a) = [\delta(q, a)]_\equiv$$

     (We can prove $\forall q', q'' \in [q]_\equiv. \ \forall a \in \Sigma. \ [\delta_\equiv(q', a)]_\equiv = [\delta_\equiv(q'', a)]_\equiv$.)

   - $F/_\equiv = \{[q]_\equiv \mid q \in F\}$

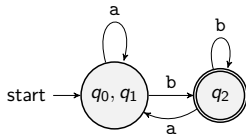# Minimization Algorithm - Example 1

① Remove unreachable states

② Partition the states into $Q/\!\!\equiv$

$$Q/\!\!\equiv = \{ \\
\{q_0, q_1\}, \quad (\because q_0 \equiv q_1) \\
\{q_2\}, \\
\}$$

③ Construct a new DFA $D/\!\!\equiv$
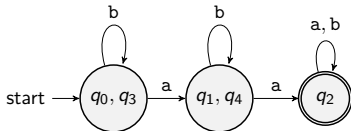
Minimization Algorithm - Example 2

① Remove unreachable states
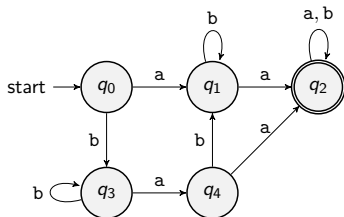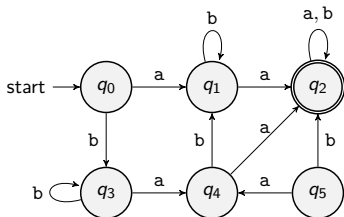
② Partition the states into $Q/_\equiv$

$$Q/_\equiv = \{ \\
\quad \{q_0, q_3\}, \quad (\because q_0 \equiv q_3) \\
\quad \{q_1, q_4\}, \quad (\because q_1 \equiv q_4) \\
\quad \{q_2\}, \\
\}$$

③ Construct a new DFA $D/_\equiv$

Theorem (Minimum-State DFA)

For a given DFA $D = (Q, \Sigma, \delta, q_0, F)$, its minimized DFA
$D/_\equiv = (Q/_\equiv, \Sigma, \delta/_\equiv, [q_0]_\equiv, F/_\equiv)$ *is a* **minimum-state DFA** *of D.*

(i.e., $\nexists$ DFA $D' = (Q', \Sigma, \delta', q_0', F')$. s.t. $L(D') = L(D) \land |Q'| < |Q/_\equiv|$).

- Assume that $\exists$ DFA D'. Then, $m < n$ when $m = |Q'|$ and $n = |Q/_\equiv|$.

- For any state $q \in Q/_\equiv$, we can find a state $q' \in Q'$ such that $q \equiv q'$.

  (We will prove it as a lemma in the next slide.)

- By Pigeonhole Principle, $\exists q_i \neq q_j \in Q/_\equiv$. $\exists q' \in Q'$. $q_i \equiv q' \land q_j \equiv q'$.

- It means that $q_i \equiv q_j$. However, it contradicts that $Q/_\equiv$ is partitioned
  into equivalence classes of states. $\qquad\square$

Proof of Minimum-State DFA – Lemma

Lemma

Consider a given DFA $D = (Q, \Sigma, \delta, q_0, F)$. Then, let

- *$D/_{\equiv} = (Q/_{\equiv}, \Sigma, \delta/_{\equiv}, [q_0]_{\equiv}, F/_{\equiv})$ be its minimized DFA*
- *$D' = (Q', \Sigma, \delta', q'_0, F')$ be another DFA such that $L(D) = L(D')$*

Then, for any state $q \in Q/_{\equiv}$, we can find a state $q' \in Q'$ such that $q \equiv q'$.

For all $q \in Q/_{\equiv}$. $\exists w = a_1 \cdots a_k$. s.t. $\delta/_{\equiv}(q_0, w) = q$. ($\because q$ is reachable.)

Let $q' = \delta'(q'_0, w)$.

Then, $\delta'^*(q'_0, a_1 \cdots a_i) \equiv \delta/_{\equiv}^*(q_0, a_1 \cdots a_i)$ for all $0 \le i \le k$.

- **(Basis Case)** $\delta'^*(q'_0, \epsilon) = q'_0 \equiv q_0 = \delta/_{\equiv}^*(q_0, \epsilon)$ ($\because L(D') = L(D/_{\equiv})$)

- **(Induction Case)** Assume $\delta'^*(q'_0, a_1 \cdots a_i) \not\equiv \delta/_{\equiv}^*(q_0, a_1 \cdots a_i)$.

  Then, by the definition of distinguishable states,
  $\delta'^*(q'_0, a_1 \cdots a_{i-1}) \not\equiv \delta/_{\equiv}^*(q_0, a_1 \cdots a_{i-1})$.

  But, it contradicts the induction hypothesis. $\square$

1. Equivalence of Finite Automata

Equivalence of States ($\equiv$)
Distinguishable States ($\not\equiv$)
Table-Filling Algorithm
Equivalence of Finite Automata
Examples

2. Minimization of Finite Automata

Minimization Algorithm
Examples
Proof of Minimum-State DFA

- Please see this document for the exercise.

https://github.com/ku-plrg-classroom/docs/tree/main/cose215/dfa-eq-min

- Please implement the following functions in Implementation.scala.

    - nonEqPairs for the **table-filling algorithm**.

    - isEqual for the **equivalence** of DFAs.

    - minimize for the **minimization** of DFAs.

- It is just an exercise, and you **don't need to submit** anything.

PLRG

- Context-Free Grammars (CFGs) and Languages (CFLs)

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