# Lecture 1 – Mathematical Preliminaries

## COSE215: Theory of Computation

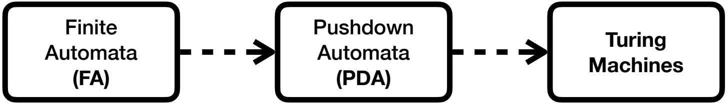
Jihyeok Park

PLRG

2025 Spring

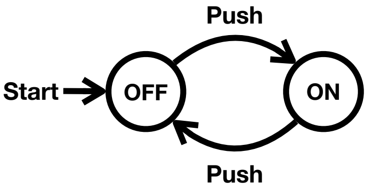
# Roadmap: Towards Turing Machine

A Turing machine is a specific kind of **automaton**.




- **Part 1: Finite Automata (FA)**
  - Regular Expressions (REs)
  - Regular Languages (RLs)
  - Applications: text search, etc.
- **Part 2: Pushdown Automata (PDA)**
  - Context-Free Grammars (CFGs)
  - Context-Free Languages (CFLs)
  - Applications: programming languages, natural language processing, etc.
- **Part 3: Turing Machines (TMs)**
  - Lambda Calculus (LC)
  - Recursively Enumerable Languages (RELs)
  - Undecidability and Intractability

# Recall




**Theorem**

*The current state is* OFF *if and only if the button is pushed* **even** *times.*

- Is it possible to prove it?

Let's learn **mathematical background and notation.**

# Contents

# Contents

# Notations in Logics

PLRG

| Notation | Description |
|---|---|
| $A, B$ | arbitrary **statements**. |
| $P(x)$ | a **predicate** is a statement having variables (e.g., $x$). |
| $A \wedge B$ | the **conjunction** of $A$ and $B$. (i.e., "$A$ and $B$"). |
| $A \vee B$ | the **disjunction** of $A$ and $B$. (i.e., "$A$ or $B$"). |
| $\neg A$ | the **negation** of $A$. (i.e., "not $A$"). |

$$(\textbf{De Morgan's Laws}) = \begin{cases} \neg(A \wedge B) = \neg A \vee \neg B \\ \neg(A \vee B) = \neg A \wedge \neg B \end{cases}$$

(**Truth Table**) =

| $A$ | $B$ | $\neg(A \wedge B)$ | $\neg A \vee \neg B$ |
|---|---|---|---|
| $T$ | $T$ | $F$ | $F$ |
| $T$ | $F$ | $T$ | $T$ |
| $F$ | $T$ | $T$ | $T$ |
| $F$ | $F$ | $T$ | $T$ |

# Notations in Logics

| Notation | Description |
|---|---|
| $A \Rightarrow B$ | the **implication** of $A$ and $B$<br>(i.e., "if $A$ then $B$" or "$A$ implies $B$")<br>(i.e., $\neg A \vee B$). |
| $A \Leftrightarrow B$ | $A$ **if and only if (iff)** $B$<br>(i.e., $A \Rightarrow B \wedge B \Rightarrow A$). |
| $\forall x \in X.\ P(x)$ | the **universal quantifier**<br>(i.e., "for all $x$ in $X$, $P(x)$ holds"). |
| $\exists x \in X.\ P(x)$ | the **existential quantifier**<br>(i.e., "there exists $x$ in $X$ such that $P(x)$ holds"). |

# Notations in Set Theory

PLRG

- A **set** is a collection of elements. For example,
  - (Integers) $= \mathbb{Z} = \{\cdots, -2, -1, 0, 1, 2, \cdots\}$
  - (Natural Numbers) $= \mathbb{N} = \{0, 1, 2, \cdots\}$
  - (Squares of $\mathbb{N}$) $= \{x^2 \mid x \in \mathbb{N}\} = \{0, 1, 4, 9, 16, 25, 36, \cdots\}$
  - (Even Numbers) $= \{x \in \mathbb{N} \mid x \equiv 0 \pmod 2\} = \{0, 2, 4, 6, 8, \cdots\}$

$$\textit{where} \qquad a \equiv b \pmod n \iff \exists k \in \mathbb{Z}.\ a = b + kn$$

  (i.e., $a$ is **congruent** to $b$ **modulo** $n$)
- The **empty set** is denoted by $\varnothing$.
- The **cardinality** of a set $X$ is denoted by $|X|$.
- $x$ is an **element** of a set $X$ is denoted by $x \in X$.
- $X$ is a **subset** of $Y$ is denoted by $X \subseteq Y$.

$$X \subseteq Y \iff \forall x \in X.\ x \in Y$$

- $X$ is a **proper subset** of $Y$ is denoted by $X \subset Y$.

$$X \subset Y \iff X \subseteq Y \land X \neq Y$$

# Notations in Set Theory

- The **union** of sets

$$\begin{array}{ll} X \cup Y & = \{x \mid x \in X \vee x \in Y\} \\ \bigcup \mathcal{C} & = X_1 \cup X_2 \cup \cdots \cup X_n = \{x \mid \exists X \in \mathcal{C}.\ x \in X\} \end{array}$$

  where $\mathcal{C} = \{X_1, X_2, \cdots, X_n\}$.

- The **intersection** of sets

$$\begin{array}{ll} X \cap Y & = \{x \mid x \in X \wedge x \in Y\} \\ \bigcap \mathcal{C} & = X_1 \cap X_2 \cap \cdots \cap X_n = \{x \mid \forall X \in \mathcal{C}.\ x \in X\} \end{array}$$

  where $\mathcal{C} = \{X_1, X_2, \cdots, X_n\}$.

- The **difference** of sets

$$X \setminus Y = \{x \mid x \in X \wedge x \notin Y\}$$

# Notations in Set Theory

- The **complement** of a set $X$ is denoted by $\overline{X}$.

$$\overline{X} = \{x \mid x \in U \land x \notin X\}$$

where $U$ is the **universal set**.

- The **power set** of a set $X$ is denoted by $2^X$ or $\mathcal{P}(X)$.

$$2^X = \mathcal{P}(X) = \{Y \mid Y \subseteq X\}$$

- The **Cartesian product** of sets $X$ and $Y$ is denoted by $X \times Y$.

$$X \times Y = \{(x, y) \mid x \in X \land y \in Y\}$$

# Contents

# Inductions on Integers

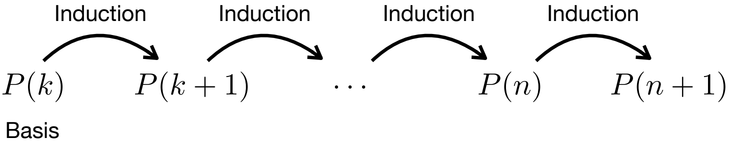
**Definition (Inductions on Integers)**

Let $P(n)$ be a predicate on integers, and if

- **(Basis Case)** $P(k)$ holds where $k$ is an integer, and
- **(Induction Case)** for all integer $n \geq k$, $P(n) \Rightarrow P(n+1)$,

then $P(i)$ holds for all $i \geq k$.

$P(n)$ is called **induction hypothesis** (I.H.).



# Inductions on Integers – Example 1

**Example**

Prove that $\forall n \geq 0.\ \sum_{i=0}^{n} i = \frac{n(n+1)}{2}$.

**Proof)**

- (Basis Case): $0 = 0(0+1)/2$ $\square$
- (Induction Case): Assume that it holds for $n$ (I.H.)

  Then, let's prove it for $n+1$:

$$\begin{aligned} \sum_{i=0}^{n+1} i &= (n+1) + \sum_{i=0}^{n} i \\ &= (n+1) + \frac{n(n+1)}{2} \qquad (\because I.H.) \\ &= \frac{(n+1)(n+2)}{2} \quad \square \end{aligned}$$

# Inductions on Integers – Example 2

PLRG

## Example

Prove that $\forall n \geq 0.\ \sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$.

**Proof)**

- (Basis Case): $0^2 = 0(0+1)(2*0+1)/6 \quad \square$
- (Induction Case): Assume that it holds for $n$ (I.H.).

  Then, let's prove it for $n+1$:

$$
\begin{aligned}
\sum_{i=0}^{n+1} i^2 &= (n+1)^2 + \sum_{i=0}^{n} i^2 \\
&= (n+1)^2 + \frac{n(n+1)(2n+1)}{6} \qquad (\because I.H.) \\
&= \frac{(n+1)(n+2)(2(n+1)+1)}{6} \quad \square
\end{aligned}
$$

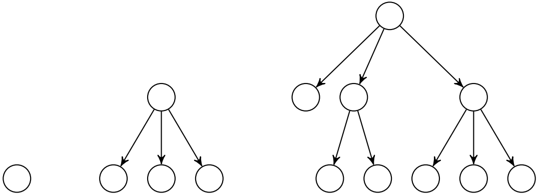
# Structural Inductions – Inductive Definitions

In CS, we often define somethings as **inductively-defined sets**.
For example, we can define **trees** as follows:

## Example (Inductive Definition of Trees)

A **tree** is defined as follows:

- **(Basis Case)** A single **node** $N$ is a tree.
- **(Induction Case)** If $T_1, \cdots, T_n$ are trees, then a graph defined with a new root node $N$ and edges from $N$ to $T_1, \cdots, T_n$ is a tree.



# Structural Inductions – Inductive Definitions

Another example is a set of **arithmetic expressions**:

**Example (Inductive Definition of Arithmetic Expressions)**

An **arithmetic expression** is defined as follows:

- **(Basis Case)** A **number** or a **variable** is an arithmetic expression.
- **(Induction Case)** If $E$ and $F$ are arithmetic expressions, then so are $E$+$F$, $E*F$, and ($E$).

| | | |
|---|---|---|
| `42` | `x` | `x + y` |
| `42 * x` | `(x)` | `(x * y) * z` |
| `(2 + x) * y` | `x * (x * y)` | `((((x))))` |

# Structural Inductions

### Definition (Structural Inductions)

Let $P(x)$ be a predicate on a **inductively-defined set** $X$, and if

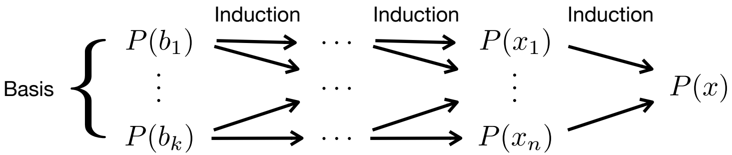
- **(Basis Case)** $P(b_1), \cdots, P(b_k)$ hold for all basis cases $b_1, \cdots, b_k$.
- **(Induction Case)** for all $x \in X$,

$$P(x_1) \wedge \cdots \wedge P(x_n) \Rightarrow P(x)$$

  where $x_1, \cdots, x_n$ are the **sub-structures** of $x$.

then $P(x)$ holds for all $x \in X$.

$P(x_1), \cdots, P(x_n)$ are called **induction hypotheses**.



**Example**

Prove that for all tree $T$, the number of nodes in $T$ is equal to the number of edges in $T$ plus one.

**Proof)** Let $N(T)$ be the number of node and $E(T)$ be the number of edges in $T$. Let's prove $\forall T.\ N(T) = E(T) + 1$.

- (Basis Case): $N(T) = 1$ and $E(T) = 0$. □
- (Induction Case): Assume that it holds for $T_1, \cdots, T_n$ (I.H.). Then,

$$\begin{aligned} N(T) &= 1 + \sum_{i=1}^{n} N(T_i) \\ &= 1 + \sum_{i=1}^{n} (E(T_i) + 1) \qquad (\because I.H.) \\ &= 1 + n + \sum_{i=1}^{n} E(T_i) \\ &= 1 + E(T) \quad \square \end{aligned}$$

# Structural Inductions – Example 2

PLRG

**Example**

Prove that for all arithmetic expression $E$, the number of left parentheses in $E$ is equal to the number of right parentheses in $E$.

**Proof)** Let $L(E)$ be the number of left parentheses and $R(E)$ be the number of right parentheses in $E$. Let's prove $\forall E.\ L(E) = R(E)$.

- (Basis Case): $L(E) = R(E) = 0$ for numbers and variables. $\square$
- (Induction Case): Assume that it holds for $E$ and $F$ (I.H.). Then,

$$
\begin{aligned}
L(E\texttt{+}F) &= L(E) + L(F) = R(E) + R(F) \qquad (\because I.H.) \\
&= R(E\texttt{+}F) \quad \square \\
L(E\texttt{*}F) &= L(E) + L(F) = R(E) + R(F) \qquad (\because I.H.) \\
&= R(E\texttt{*}F) \quad \square \\
L(\texttt{(}E\texttt{)}) &= L(E) + 1 = R(E) + 1 \qquad (\because I.H.) \\
&= R(\texttt{(}E\texttt{)}) \quad \square
\end{aligned}
$$

# Mutual Inductions

Sometimes, we need to prove multiple predicates simultaneously.

**Definition (Mutual Inductions)**

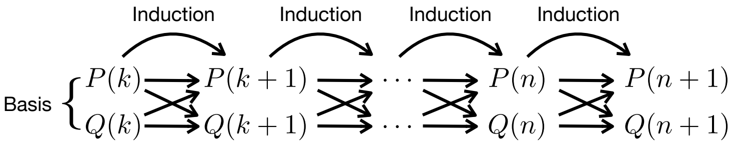
Let $P(x)$ and $Q(x)$ are predicates on integers, and if

- **(Basis Case)** $P(k)$ and $Q(k)$ hold where $k$ is an integer, and
- **(Induction Case)** for all $n \geq k$,

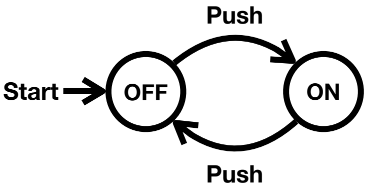
$$P(n) \land Q(n) \Rightarrow P(n+1) \land Q(n+1)$$

then $P(i)$ and $Q(i)$ hold for all $i \geq k$.

$P(n)$ and $Q(n)$ are called **induction hypotheses**.



# Mutual Inductions – Example




**Theorem**

*The current state is* OFF *if and only if the button is pushed* **even** *times.*

It is difficult to prove it with only one predicate.

Let's prove it with two predicates:

**Theorem**

*The current state is* OFF *if and only if the button is pushed* **even** *times, and the current state is* ON *if and only if the button is pushed* **odd** *times.*

# Mutual Inductions – Example

**Theorem**

*The current state is* OFF *if and only if the button is pushed* **even** *times, and the current state is* ON *if and only if the button is pushed* **odd** *times.*

**Proof)** Let $S(i)$ be the current state after $i$ times of pushing. Let's prove

$$\forall i \geq 0.\ S(i) = \text{OFF} \iff i \equiv 0 \pmod 2 \qquad (P)$$

$$\forall i \geq 0.\ S(i) = \text{ON} \iff i \equiv 1 \pmod 2 \qquad (Q)$$

- (Basis Case): Known facts: $S(0) = \text{OFF}$ and $0 \equiv 0 \pmod 2$
  - $(P, \Rightarrow)$: $S(0) = \text{OFF} \Rightarrow 0 \equiv 0 \pmod 2$ because $0 \equiv 0 \pmod 2$
  - $(P, \Leftarrow)$: $S(0) = \text{OFF} \Leftarrow 0 \equiv 0 \pmod 2$ because $S(0) = \text{OFF}$
  - $(Q, \Rightarrow)$: $S(0) = \text{ON} \Rightarrow 0 \equiv 1 \pmod 2$ because $S(0) \neq \text{ON}$
  - $(Q, \Leftarrow)$: $S(0) = \text{ON} \Leftarrow 0 \equiv 1 \pmod 2$ because $0 \not\equiv 1 \pmod 2$

# Mutual Inductions – Example

- (Induction Case): Assume that it holds for $n$ (I.H.):

$$S(n) = \text{OFF} \iff n \equiv 0 \pmod 2 \qquad (P - I.H.)$$
$$S(n) = \text{ON} \iff n \equiv 1 \pmod 2 \qquad (Q - I.H.)$$

  - $(P, \Leftrightarrow)$:

$$\begin{aligned} S(n+1) = \text{OFF} &\iff S(n) = \text{ON} \\ &\iff n \equiv 1 \pmod 2 \qquad (\because Q - I.H.) \\ &\iff n + 1 \equiv 0 \pmod 2 \end{aligned}$$

  - $(Q, \Leftrightarrow)$:

$$\begin{aligned} S(n+1) = \text{ON} &\iff S(n) = \text{OFF} \\ &\iff n \equiv 0 \pmod 2 \qquad (\because P - I.H.) \\ &\iff n + 1 \equiv 1 \pmod 2 \end{aligned}$$

# Contents

# Symbols & Words

- We first define a finite and non-empty set of **symbols** $\Sigma$.
- A **word** $w \in \Sigma^*$ is a sequence of symbols.
  - $\Sigma = \{0, 1\}$ – binary symbols.

$$\epsilon, 0, 1, 00, 01, 10010, \cdots \in \Sigma^*$$

  - $\Sigma = \{\text{a}, \text{b}, \cdots, \text{z}\}$ – lowercase letters.

$$\epsilon, \text{a}, \text{b}, \text{abc}, \text{hello}, \text{cs}, \text{students}, \cdots \in \Sigma^*$$

  - $\Sigma = \{a \mid a \text{ is an Unicode character}\}$ – Unicode characters.

$$\epsilon, \text{안녕하세요}, \text{こんにちは}, \bigstar\blacksquare\blacktriangle\oplus, \cdots \in \Sigma^*$$

# Symbols & Words

| Notation | Description |
|---|---|
| $\epsilon$ | the **empty word**. |
| $w_1 w_2$ | the **concatenation** of $w_1$ and $w_2$.<br>($w_1$ is a **prefix** of $w_1 w_2$ and $w_2$ is a **suffix** of $w_1 w_2$) |
| $w^R$ | the **reverse** of $w$. |
| $\lvert w \rvert$ | the **length** of $w$. |
| $\Sigma^k$ | the set of all words of length $k$. |
| $\Sigma^*$ | the set of all words (the **Kleene star**).<br>(i.e., $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \cdots = \bigcup_{k \geq 0} \Sigma^k$) |
| $\Sigma^+$ | the set of all words except $\epsilon$ (the **Kleene plus**).<br>(i.e., $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \cdots = \bigcup_{k \geq 1} \Sigma^k$) |

# Languages

A **language** $L \subseteq \Sigma^*$ is a specific set of words.

When $\Sigma = \{0, 1\}$, we can define the following languages:

- $L = \{\epsilon, 0, 1\}$ – the empty word, zero, and one.
- $L = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \cdots\}$ – all binary words.
- $L = \{0^n 1^n \mid n \geq 0\}$ – equal number of consecutive zeros and ones.
- $L = \{10, 11, 101, 111, 1011, \cdots\}$ – ???

# Languages – Operations

- The **union**, **intersection**, and **difference** of languages:

$$L_1 \cup L_2 \qquad L_1 \cap L_2 \qquad L_1 \setminus L_2$$

- The **reverse** of a language:

$$L^R = \{w^R \mid w \in L\}$$

- The **complement** of a language:

$$\overline{L} = \Sigma^* \setminus L$$

- The **concatenation** of languages:

$$L_1 L_2 = \{w_1 w_2 \mid w_1 \in L_1 \land w_2 \in L_2\}$$

# Languages – Operations

- The **power** of a language defined inductively:

$$\begin{aligned} L^0 &= \{\epsilon\} \\ L^n &= L^{n-1}L \qquad (n \geq 1) \end{aligned}$$

- The **Kleene star** of a language:

$$L^* = L^0 \cup L^1 \cup L^2 \cup \cdots = \bigcup_{n \geq 0} L^n$$

- The **Kleene plus** of a language:

$$L^+ = L^1 \cup L^2 \cup L^3 \cup \cdots = \bigcup_{n \geq 1} L^n$$

- For any language $L$, are the following true?
  - $\epsilon \in L^*$ – Yes. Because $\epsilon \in L^0 \subseteq L^*$
  - $\epsilon \notin L^+$ – No. If $\epsilon \in L$, then $\epsilon \in L = L^1 \subseteq L^+$

# Summary

1. Mathematical Notations
   - Notations in Logics
   - Notations in Set Theory

2. Inductive Proofs
   - Inductions on Integers
   - Structural Inductions
   - Mutual Inductions

3. Notations in Languages
   - Symbols & Words
   - Languages

# Next Lecture

PLRG

- Basic Introduction of Scala

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