Lecture 0 – Course Overview

COSE215: Theory of Computation

Jihyeok Park

PLRG

2024 Spring

Course Information

- **Instructor:** Jihyeok Park (박지혁)
  - **Position:** Assistant Professor in CS, Korea University
  - **Expertise:** Programming Languages, Software Analysis
  - **Office hours:** 14:00–16:00, Tuesdays (appointment by e-mail)
  - **Office:** 609A, Science Library Bldg
  - **Email:** jihyeok_park@korea.ac.kr
- **Class:** COSE215 - 01 (English)
- **Lectures:** 13:30–14:45, Mondays and Wednesdays @ 604 Woojung Hall of Informatics (우정정보관 604호)
- **Homepage:** `https://plrg.korea.ac.kr/courses/cose215/`
- **Teaching Assistant:** **Jungyeom Kim (김준겸)**
  - **Email**: kimjg1119@korea.ac.kr

Schedule

Weak	Contents
1	Basic Concepts
2	Deterministic Finite Automata (DFA)
3	Nondeterministic Finite Automata (NFA)
4	Regular Expressions and Languages
5	Properties of Regular Languages
6	Context-Free Grammars and Languages
7	Parse Trees and Ambiguity
8	**Midterm Exam (Apr. 24 - Wed.)**
9	Pushdown Automata
10	Deterministic Pushdown Automata
11	Properties of Context-Free Languages
12	Turing Machines (TMs)
13	Extensions of Turing Machines
14	Undecidability
15	P, NP, and NP-Completeness
16	**Final Exam (Jun. 19 - Wed.)**

Grading

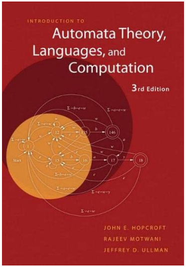
- **5–7 Homework Assignments: 30%**
  - Programming assignments in Scala (submission in **Blackboard**)
  - You can utilize or refer to any other materials (e.g., ChatGPT), but you **MUST** write your **OWN** solution.
  - Cheating is strictly prohibited. Cheating will get you an F.
- **Midterm exam: 30%**
  - April 24 (Wed.) 13:30 – 14:45 (in class, 75 min.)
- **Final exam: 30%**
  - June 19 (Wed.) 13:30 – 14:45 (in class, 75 min.)
- **Attendance: 10%**
  - Please use **Blackboard** to attend the class.

Course Materials

- **Self-contained lecture notes.**

https://plrg.korea.ac.kr/courses/cose215/

- Reference:



John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman. Introduction to automata theory, languages, and computation. Third edition.

Goal of This Course

PLRG

- What is the *mathematical model* of computers?

Turing Machine!

Let's learn **Turing Machine**

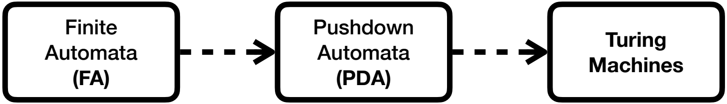
- Is it possible to solve *every problem* using computers?

No!

Let's learn **Undecidability** and **Intractability**

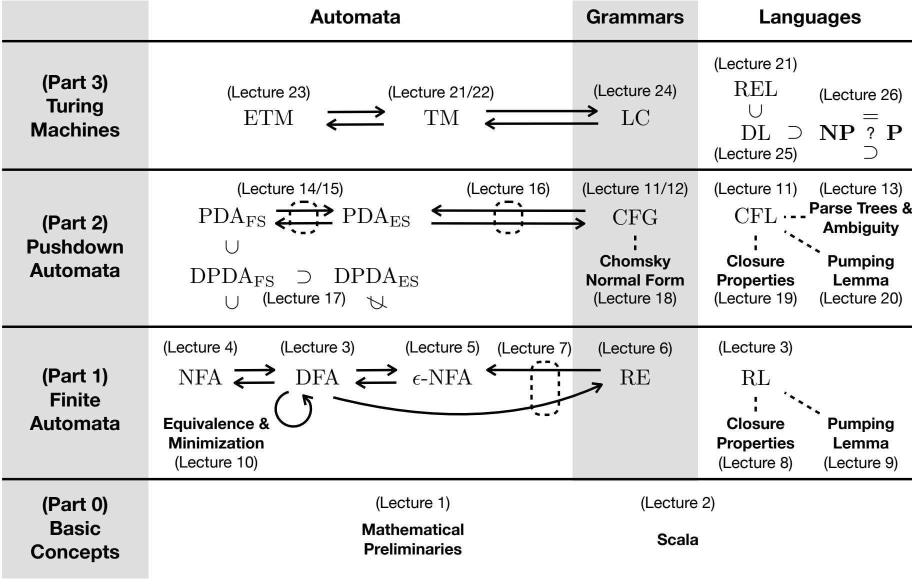
Roadmap: Towards Turing Machine

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

Roadmap: Towards Turing Machine

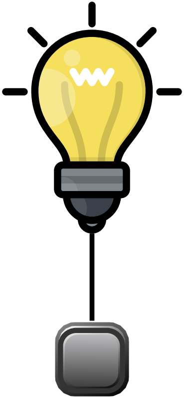


Introduction of Automata

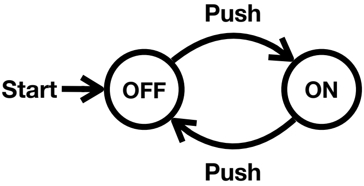
A Turing machine is a specific kind of **automaton**.

Then, what is an **automaton**?

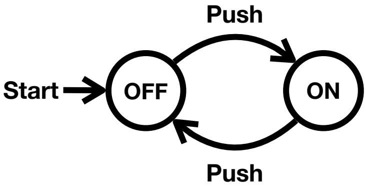
For example,



$\Longrightarrow$



Introduction of Automata



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.**

- Is it possible to implement the automaton?

Let's learn **Scala** as an implementation language.

Next Lecture

- Mathematical Preliminaries

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