Randomized Algorithms

— P, NP, RP, PP, ZPP, BPP, . . .

Joseph Chuang-Chieh Lin

Department of Computer Science & Information Engineering,
Tamkang University

Fall 2023

Outline

1 RAMs & Turing Machines

2 Complexity Classes

Outline

1. RAMs & Turing Machines
2. Complexity Classes

RAM (Random Access Machine)

- RAM is a model of computation used when describing and analyzing algorithms.
- A machine can perform operations involving registers and main memory.
- The *unit-cost* RAM: each instruction can be performed in one time step.
    - Too powerful; no known polynomial time simulation of this type of model by Turing machines.
- The *log-cost* RAM: each instruction requires time proportional to the logarithm of the size of its operands.

Turing Machine

A physical Turing machine (with finite amount of tape).

Deterministic Turing Machine

A deterministic Turing machine is a quadruple $M = (S, \Sigma, \delta, s)$.

- S: a finite set of states (s: the initial state)
- Σ: A finite set of symbols (including special symbols BLANK and FIRST).
- δ: the transition function.
  - $S \times \Sigma \mapsto (S \cup \{\text{HALT, YES, NO}\}) \times \Sigma \times \{\leftarrow, \rightarrow, \text{STAY}\}$.
  - HALT, YES, NO: The three halting states not in S.

Turing Machine (Input & Tape)

- The input to the TM: written on a tape.
- The TM, as an algorithm, may read from and write on this tape.
- Assume that HALT, YES, NO as well as the symbols $\leftarrow, \rightarrow$, and STAY are not in $S \cup \Sigma$.
- The TM begins in the initial state s with its cursor at the first symbol FIRST of input x.
- The input is a string of $(\Sigma \setminus \{\text{BLANK}, \text{FIRST}\})^*$.
  - The left-most BLANK on the tape: the end of the input string.

Turing Machine (Transition)

- The transition function δ: can be thought as a *program*.
- In each step, the TM reads the symbol α pointed by the cursor;
- Based on α and the current state, choose:
  - a next state;
  - a symbol β to be overwritten on α;
  - a cursor motion direction from $\{\leftarrow, \rightarrow, \mathsf{STAY}\}$.
- The cursor never falls off the left end of the input: FIRST.
- The BLANK symbol can be overwritten.

Turing Machine (Acceptance & Reject)

- The TM has accepted the input x: if the TM halts in the YES state.
- The TM has rejected the input x: if the TM halts in the NO state.
- State HALT: for the computation of functions whose range is not Boolean (output of the function is written on the tape).

Probabilistic Turing Machine

A probabilistic Turing machine is a Turing machine augmented with the ability to generate an unbiased coin flip in one step.

- This corresponds to a randomized algorithm.

Outline

1 RAMs & Turing Machines

2 Complexity Classes

SAT

An instance of satisfiability (SAT):

$$(x_1 \wedge \neg x_2 \wedge x_4) \vee (\neg x_3 \wedge \neg x_4 \wedge x_5) \vee (\neg x_1 \wedge x_2 \wedge x_4 \wedge \neg x_5)$$

- $x_1, x_2, \ldots$: variables
- $\neg x_1, x_2$: literals
- $(\cdots)$: clauses

Language Recognition Problems

Language Recognition Problems

Any decision problem can be treated as a language recognition problem.

- Σ^*: the set of all possible strings over Σ.
- $|S|$: length of string s.

A language $L \subseteq \Sigma^*$ is any collection of strings over Σ.

A Language Recognition Problem

Decide whether a given string $x \in \Sigma^*$ belongs to L.

Complexity Class

A collection of languages all of whose recognition problems can be solved under prescribed bounds on the computational resources.

P & ***NP***

P

The class ***P*** consists of all languages L which has a polynomial time algorithm A s.t. for any input $x \in \Sigma^*$,

- $x \in L \Rightarrow A(x)$ accepts;
- $x \notin L \Rightarrow A(x)$ rejects.

NP

The class ***NP*** consists of all languages L which has a polynomial time algorithm A s.t. for any input $x \in \Sigma^*$,

- $x \in L \Rightarrow \exists y \in \Sigma^*,\ A(x, y)$ accepts for $|y| \leq \text{poly}(|x|)$;
- $x \notin L \Rightarrow \forall y \in \Sigma^*,\ A(x, y)$ rejects..

A Useful, Alternative Viewpoint

The class $\boldsymbol{P}$ consists of all language L such that for any $x \in L$, a proof of $x \in L$ (represented by the string y) can be found and verified in polynomial time.

The class $\boldsymbol{NP}$ consists of all language L such that for any $x \in L$, a proof of $x \in L$ (represented by the string y) can be verified in polynomial time.

Obviously,

$$\boldsymbol{P} \quad \boldsymbol{NP}.$$

A Useful, Alternative Viewpoint

The class $\boldsymbol{P}$ consists of all language L such that for any $x \in L$, a proof of $x \in L$ (represented by the string y) can be found and verified in polynomial time.

The class $\boldsymbol{NP}$ consists of all language L such that for any $x \in L$, a proof of $x \in L$ (represented by the string y) can be verified in polynomial time.

Obviously,

$$\boldsymbol{P} \subseteq \boldsymbol{NP}.$$

Complementary Classes

For any complexity class $\mathcal{C}$, the complementary class co-$\mathcal{C}$ is the set of languages whose complement is in $\mathcal{C}$. That is,

$$\text{co-}\mathcal{C} = \{L \mid \bar{L} \in \mathcal{C}\}.$$

Examples: co-$\boldsymbol{P}$ & co-$\boldsymbol{NP}$

P

The class co-$\boldsymbol{P}$ consists of all languages L which has a polynomial time algorithm A s.t. for any input $x \in \Sigma^*$,

- $x \notin L \Rightarrow A(x)$ accepts;
- $x \in L \Rightarrow A(x)$ rejects.

NP

The class $\boldsymbol{NP}$ consists of all languages L which has a polynomial time algorithm A s.t. for any input $x \in \Sigma^*$,

- $x \notin L \Rightarrow \exists y \in \Sigma^*$, $A(x, y)$ accepts for $|y| \leq \text{poly}(|x|)$;
- $x \in L \Rightarrow \forall y \in \Sigma^*$, $A(x, y)$ rejects..

Open Questions: $\boldsymbol{P} = \boldsymbol{NP} \cap \text{co-}\boldsymbol{NP}$? $\boldsymbol{NP} = \text{co-}\boldsymbol{NP}$?

Similarly, ...

EXP & *NEXP*

EXP

The class ***EXP*** consists of all languages L which has an exponential time algorithm A s.t. for any input $x \in \Sigma^*$,

- $x \in L \Rightarrow A(x)$ accepts;
- $x \notin L \Rightarrow A(x)$ rejects.

NEXP

The class ***NEXP*** consists of all languages L which has an exponential time algorithm A s.t. for any input $x \in \Sigma^*$,

- $x \in L \Rightarrow \exists y \in \Sigma^*,\ A(x, y)$ accepts for $|y| \leq \text{poly}(|x|)$;
- $x \notin L \Rightarrow \forall y \in \Sigma^*,\ A(x, y)$ rejects..

A Useful, Alternative Viewpoint

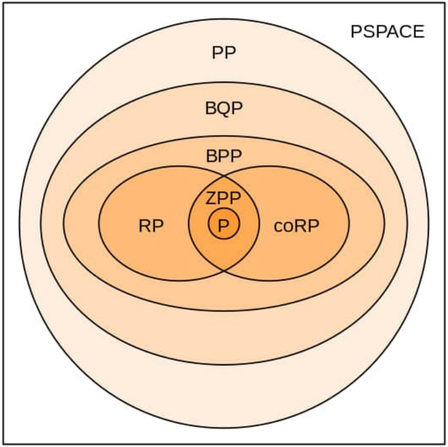
The class ***EXP*** consists of all language L such that for any $x \in L$, a proof of $x \in L$ (represented by the string y) can be found and verified in exponential time.

The class ***NEXP*** consists of all language L such that for any $x \in L$, a proof of $x \in L$ (represented by the string y) can be verified in exponential time.

Obviously,

$$\boldsymbol{EXP} \subseteq \boldsymbol{NEXP}.$$

Randomized Complexity Classes



Discussions