Randomized Algorithms

Introduction

Joseph Chuang-Chieh Lin
Dept. CSIE, Tamkang University

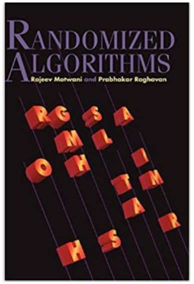
Self Introduction

- **Ph.D.**: CSIE, National Chung Cheng University, 2011.
  - DAAD-NSC Sandwich Program (2007−2008).
  - Dissertation supervisors: Maw-Shang Chang & Peter Rossmanith (RWTH Aachen)

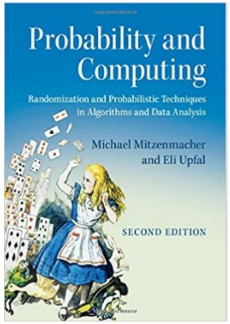
- **Postdoc** in Genomics Research Center, Academia Sinica (2011−2014).
- **Postdoc** in Institute of Information Science, Academia Sinica (2014−2018).
- **Quantitative Analyst (intern)** of Point72/Cubist Systematic Strategies (2018−2019).
- **Quantitative Analyst** of Seth Technologies Inc. (2020−2021/01).

Textbooks and Materials

- Textbooks:
  - *Randomized Algorithms*. Motwani, R. and Raghavan, P., 1995. Cambridge University Press.



  - *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. M. Mitzenmacher and E. Upfal, 2005.



- Other materials:
  - Prepared slides.

Prerequisites

- Basic undergraduate courses in
  - Algorithms
  - Data structures
  - Probability theory
  - Discrete mathematics

- Motivation.

- Curiosity.

Topics

- Examples of Probability Paradoxes
- Las Vegas and Monte Carlo
- Randomized Quicksort
- Chernoff Bounds
- The Stable Marriage Problem
- The Coupon Collector's Problem
- The Secretary Problem
- Random Graphs
- Random Treaps
- Markov Chains (Optional)
- Monte Carlo Simulation (Optional)

Grading Policy

- Assignments (x 10, 50%)
- Programming Project bonus (x *n*, 5%)
- Final Team Report (50%)
  - Peer-Grading Mechanism.
  - Election Game Equilibrium.
  - Presentation for other selected topics.

Traditional deterministic algorithms

Randomized algorithms

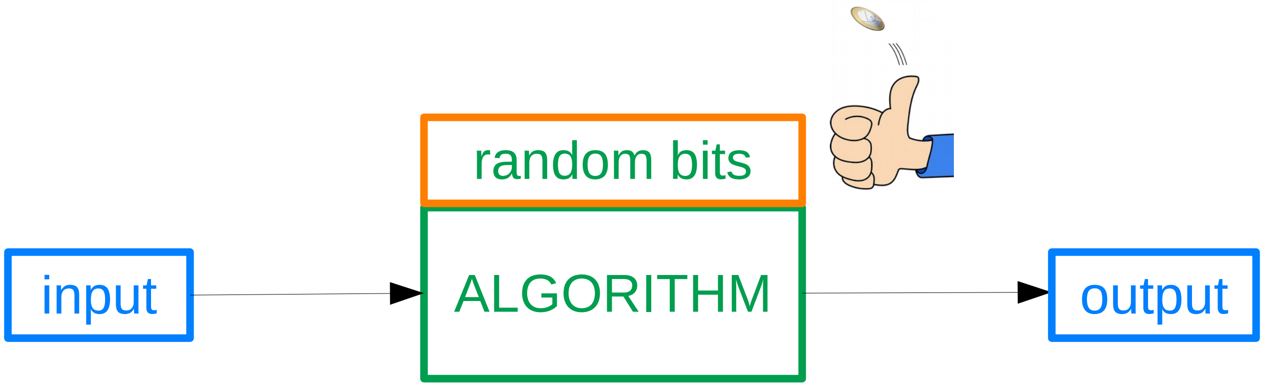


figure freely downloaded from https://pixabay.com/illustrations/coin-flipping-coin-hand-flip-flick-5822271/

Why?

- Randomized algorithms are
  - often much ***simpler*** than the best known deterministic ones.
  - often much ***more efficient*** (faster or using less space) than the best known deterministic ones.

- Sometimes ideas from the randomized algorithms lead to good deterministic algorithms.

Comparisons

- It's different from the *average-case* analysis of deterministic algorithms.
  - e.g., expected running time of a deterministic algorithm on input sampled from a distribution.

- In most scenarios, it's NOT a heuristic algorithm.
  - The accuracy is guaranteed, or
  - The running time is guaranteed.

An illustrating example:

- **Problem:** find a grade-‘*A*’ student in a class of n students where half of them get ‘A’.

- What is the time complexity for the best deterministic algorithm?
  - I mean, in the “worst case”.

A randomized algorithm (from Wikipedia)

findingA_LV(array *L*, *n*)

begin

  repeat

    Randomly select one element out of *n* elements.

  until '*A*' is found

end

Assignment: Prove that the expected number of iterations is $\lim_{n\to\infty} \sum_{i=1}^{n} \frac{i}{2^i} \leq 2.$

A randomized algorithm (from Wikipedia)

findingA_MC(array L, n, k)
begin
    $i \leftarrow 0$
    repeat
        Randomly select one element out of n elements.
        $i \leftarrow i + 1$
    until $i = k$ or 'A' is found
end

After k iterations, $\Pr[\text{find } A] = 1 - (1/2)^k$.

Birthday problem (paradox)

- There are n randomly chosen people in a room.
- How ***large*** should n be such that there is at least one pair of them having the same birthday (mm/dd)?
- By the pigeonhole principle, n = 367? or 366?

- Let us consider this problem in the other way around.

  How ***large*** should n be such that there is at least one pair of them having the same birthday (mm/dd) with probability ≥ 0.5?

Birthday problem (paradox)

- n people: $x_1, x_2, \ldots, x_n$
- Event i: some pair of $x_1, x_2, \ldots, x_i$ have the same birthday.
- $\Pr[\text{Event2}] = 1 - \frac{364}{365}$
- $\Pr[\text{Event3}] = 1 - \frac{364}{365} \cdot \frac{363}{365}$
- …
- $\Pr[\text{Event23}] = 1 - \frac{364}{365} \cdot \frac{363}{365} \cdot \ldots \frac{343}{365} \approx 0.507297.$
- 23 is much less than 366 or 367.