CSE 331 Algorithms and Complexity

Xiangyu Guo

xiangyug@buffalo.edu

(I) Syllabus:

- Course webpage
  - ▷ https://www.acsu.buffalo.edu/~xiangyug/teaching/cse331-summer20/index.html
  - Contain schedule, HW, handouts
- Piazza (for Q&A)
  - ▷ https://piazza.com/buffalo/summer2020/cse331
  - Also provided on the course webpage
- UB Learns (for submitting HWs)
  - You shall be enrolled automatically

- Lectures (see details on Piazza)
  - Recorded video uploaded to UBox
  - UBLearns → Panopt.
  - Lecture time meeting (on Zoom)
    ▷ Mon, Wed, Fri. 9:00 – 10:00 am
  - Office hour (on Zoom)
    ▷ Fri. 10:00 – 11:00 am. (tentative)

Expected background

- Math
  - ▷ Read & Write proofs, Reasoning and induction
  - ▷ Basic calculus (limits, differentiation, and integration) and basic linear algebra (matrix, linear equation system)
- Programming
  - ▷ Recursion
  - ▷ Data structure : linked list, stack, queue, binary tree
  - ▷ PL : C/C++, Java, or Python

CSE 250

What you'll learn

- ▷ Meta algorithm design techniques
  - Backtracking
  - Greedy
  - Divide-and-Conquer (DaC)
  - Dynamic programming (DP)
- ▷ How to analyze algs rigorously
  - Correctness — The alg **always** work as expected.
  - Running time (efficiency) — Asymptotic notation (Big-O analysis)
- ▷ NP-completeness
  - "Life is just hard."

Reference

▷ Textbook (reading is recommended but not required)

- [KT] Algorithm Design, by *Jon Kleinberg* and *Eva Tardos*
  - A classic, commonly used in previous alg courses of UB
- [DPV] Algorithms , by *S. Dasgupta*, *C.H. Papadimitriou*, and *U.V. Vazirani*
  - An unified treatment of algs & DS.
- [Eric] Algorithms , by *Jeff Erickson*
  - My personal favorite.

see the course website for more !

Grading

▷ 40% 4–5 theory HW {
- pencils & papers
- scan & upload PDF files to UBLearn.

▷ 10% 1 Programming HW

- Allowed PL: C/C++, Java, or Python (recommended)

▷ 20% 1 Midterm (Jun 26, tentative)

- Take-home or online (not decided yet)

▷ 30% Final (Aug 03, tentative)

- Form as above

Collaboration Policy

For HWs

▷ You are allowed to

- Use course material (Textbook & handouts)
- Ask me for hints
- Collaborate with classmates
  - Explicitly give them credits
  - write down sols in your own words

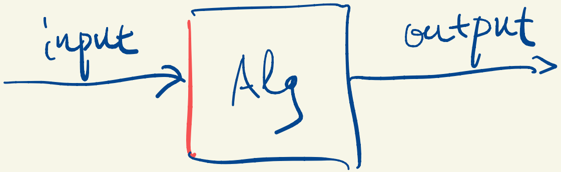
▷ You are NOT allowed to

- Copy other students' sols.
- Google or ask questions online for sols.

(II) What is an algoriehm?

▷ classical def:

Donald Knuth: An algorithm is a finite, definite effective procedure, with some input and some output.



- Computational prob. : relation between input/output.

  e.g. relation: output is 2x times the input.

▷ 3 major properties:

- well-defined: one can conduct every step mechanically
- Correct: $\forall$ input, produce correct output.
- Efficient: runs fast.

Example : Greatest Common Divison (GCD)

▷ Problem def :

Input : two integers $a, b > 0$

Output : the GCD of a and b

GCD:

$\max t$

s.t. $a \bmod t = 0$

$b \bmod t = 0$

▷ Example : Input : 210 , 270

Output : 30

▷ Alg. : Euclidean's Alg

- Assume $a > b$

  $GCD(a, b) = a$ if $b = 0$ else. $GCD(b, a \bmod b)$.

- $GCD(270, 210) = GCD(210, 60) = GCD(60, 30) = GCD(30, 0)$

  $\Rightarrow$ output 30.

Exmp: Sorting

Prob Def:

Input: seq of n numbers $a_1, a_2, \cdots, a_n$

Output: a permutation $a_1', a_2', \cdots, a_n'$ s.t.

$$a_1' \le a_2' \le \cdots \le a_n'$$

Exmp: Input: 53, 12, 15, 0, 4, 97, 22

Output: 0, 4, 12, 15, 22, 53, 97.

Alg: Insertion Sort, Merge Sort, Quick Sort, . . .

Non-Exmp: Be-A-Millionare-Never-Pay-Tax

Alg: 1. Get a million $

2. Don't report tax until IRS come to you

→ tell them "I forgot"

- Not well-defined:

  Step 1 — unclear how to conduct

- Not correct:

  Step 2 — You may get arrested.

- Not efficient:

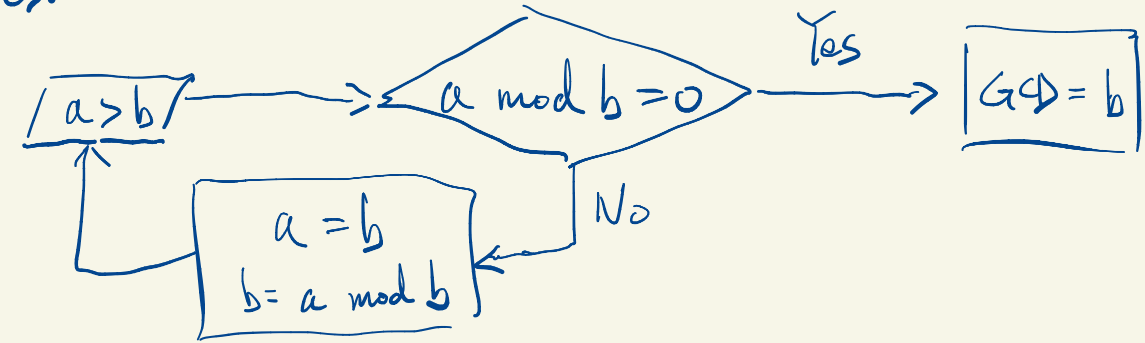
  Step 1 — May take a life time.

Describe Alg

▷ Take GCD as example

- plain english: choose the minimum of the two inputs a and b, say $b<a$, then compute GCD of b and a mod b output a if b is 0.

- flow charts.



✓ · Pseudo-code.

```
GCD(a, b)
  while b > 0
      (a, b) ← (b, a mod b)
  return a
```

- Actual PL.

Analysis of Algorithms

▷ Correctness

- Need rigorous math proof
- Advanced (not covered in this course)
  - CSE 631 : "not far from correct" (approx alg)
  - CSE 632 : "correct with some probability" (randomized alg)

▷ Efficiency.

- Asymptotic behavior:
  - How running time scales with input size.
  - Sorting n numbers ⟶ time T
    Sorting 10·n numbers ⟶ time ?

▷ Won't consider engineering side.