CST207
DESIGN AND ANALYSIS OF ALGORITHMS

Lecture 1: Introduction to Algorithm

Lecturer: Dr. Yang Lu

Email: luyang@xmu.edu.my

Office: A1-432

Office hour: TBD

Lecturer Information

Yang Lu (Jason), Ph.D.

- Experience:
  - 2020~Present: Assistant Professor, School of Electrical and Computer Engineering, Xiamen University Malaysia.
  - 2019~Present: Assistant Professor, Department of Computer Science, School of Informatics, Xiamen University.
  - 2018~2019: Research Intern, Tencent.
  - 2014~2019: Ph.D. student, Department of Computer Science, Hong Kong Baptist University.
  - 2012~2014: Master student, Department of Software Engineering, University of Macau.
  - 2008~2012: Bachelor student, Department of Software Engineering, University of Macau.
- Research interests:
  - Artificial intelligence, machine learning.
- Personal website:
  - https://jasonyanglu.github.io/

Consultation Hours

- Office location: A1-432
- Consultation Hours:
  - TBD
- For other time slot:
  - Send email to confirm my presence first.
  - Or take your luck to knock the door (I will be in my office for most of the time in this semester).

Class Rules

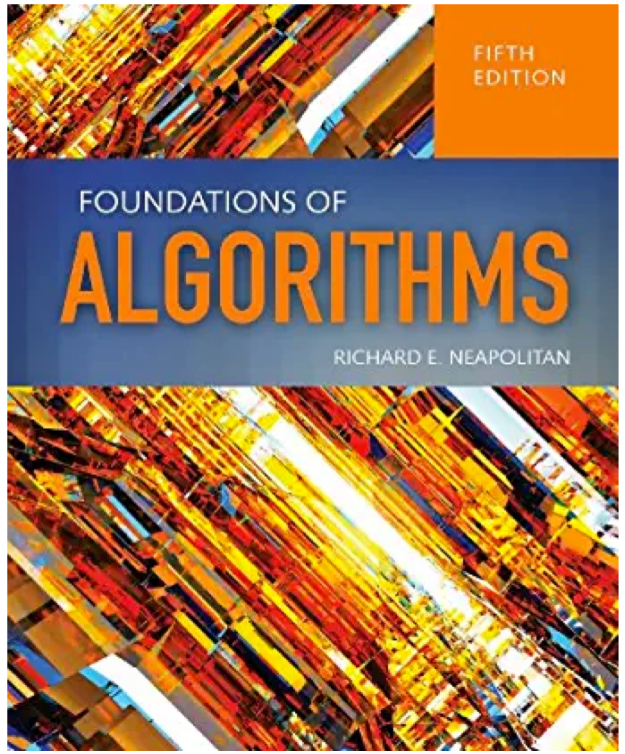
- You can do anything except:
  - Make noises (chatting, singing…)
  - Catch others' eyes (dancing, push-ups…)
  - Eat food with strong smell (instant noodles, durian…)
- Feel free to interrupt me if you have questions (no need to raise hand).
- According to the university policy, taking attendance is needed.
  - **Important: you are required to have an 80% attendance to be able to seat for the final exam.**

Course Assessment

- Temporary according to the situation:
  - Midterm exam: 20%
  - Final exam: 30%
  - Individual assignment: 20%
  - Individual project: 30%
- **Important: cheating and plagiarism will get no marks.**

Textbook

- Richard E. Neapolitan, Foundations of Algorithms (5th Edition), Jones & Bartlett Learning, 2014



Why Study Algorithms?

- Internet
  - Web search, packet routing, distributed file sharing, ...
- Artificial Intelligence
  - Autopilot vehicle, computer vision, machine translations, …
- Computers
  - Circuit layout, file system, compilers, ...
- Computer graphics
  - Movies, video games, virtual reality, ...
- Security
  - Cell phones, e-commerce, voting machines, ...
- Multimedia
  - MP3, JPG, DivX, HDTV, face recognition, ...
- Social networks
  - Recommendations, news feeds, advertisements, ...
- Physics
  - Aerodynamics simulation, particle collision simulation, ...
- Biology
  - Human genome project, protein folding, ...
- ⋮

Why Study Algorithms?

- Algorithm engineer is rich!

Job title	Average salary per year in US (2020)
Algorithm Engineer	**$135,272**
Database Engineer	**$119,281**
Software Engineer	**$108,058**
System Engineer	**$102,266**
Network Engineer	**$96,220**
IT Project Manager	**$95,712**
Web Developer	**$75,671**

Data source: https://www.indeed.com/salaries?from=gnav-title-webapp

What is an Algorithm

- Example: Find the name "Lisa Barber" in the phone book
  - Strategy 1
    - Starting with the first name "Anderson Aaron".
    - If it is not matched, check the next one.
    - Until find "Lisa Barber".
  - Strategy 2
    - Slide your finger along the Alphabet bar to "B".
    - Check the current surname is before or after "Ba" in alphabetical order.
    - Find surname "Barber".
    - Look for the first name "Lisa".

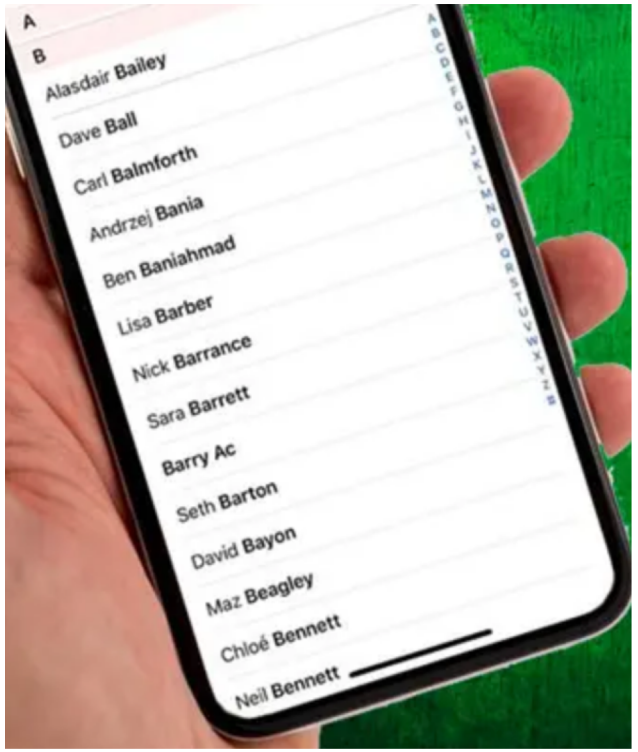


Image source: https://messagingapplab.com/wp-content/uploads/2019/04/Knowing-Who-Has-My-Number-in-Their-Contact-List.jpg.webp

What is an Algorithm

- Example with more mathematics: Compute the greatest common divisor of two integers
  - Problem:

    Find $\gcd(m, n)$, the largest integer that divides both and evenly, not both zero integers m and n.
  - Instances:
    - $\gcd(60, 24) = 12,$
    - $\gcd(60, 0) = 0.$
  - Three different algorithms:
    - Consecutive integer checking algorithm.
    - Euclid's algorithm.
    - Middle-school algorithm.

Consecutive Integer Checking Algorithm

- Step 1: Assign the value of $\min\{m, n\}$ to t.
- Step 2: Divide m by t. If the remainder is 0, go to Step 3; otherwise, go to Step 4.
- Step 3: Divide n by t. If the remainder is 0, return and stop; otherwise, go to Step 4.
- Step 4: Decrease t by 1 and go to Step 2.

Remark: Does not work correctly when one of its input numbers is 0

Middle-School Algorithm

- Step 1: Find the prime factorization of m.
- Step 2: Find the prime factorization of n.
- Step 3: Find all the common prime factors.
- Step 4: Compute the product of all the common prime factors and return it as $\gcd(m, n)$.

Euclid's Algorithm

- Step 1: If $n = 0$, return and stop; otherwise go to Step 2.
- Step 2: Divide m by n and assign the value of the remainder to r.
- Step 3: Assign the value of n to m and the value of r to n . Go to Step 1.

Euclid's algorithm is based on repeated application of equality:

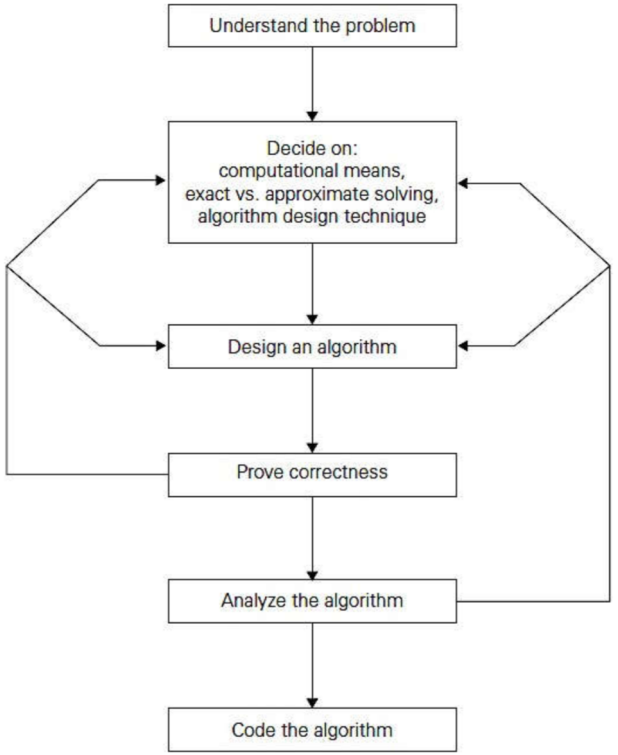
$$\gcd(m, n) = \gcd(n, m \bmod n)$$

until $n \bmod m$ is equal to 0

Definitions

- Problem: A question to which we seek an answer.
  - Find $\gcd(m, n)$.
- Algorithm: A step-by-step procedure applying a technique for solving the problem.
  - Consecutive integer checking algorithm.
  - Euclid's algorithm.
  - Middle-school algorithm.
  - …
- Parameters: Variables that are not assigned specific values in the statement of the problem.
  - m, n.
- Instance: Specific assignment of values to the parameters.
  - $m = 60, n = 24$.
- Solution: The answer to the problem in that instance.
  - 12

Process of Analysis and Design of Algorithm



Algorithm Design Techniques and Strategies

- In this course, you are going to learn:
  - Recursive algorithms
  - Divide-and-conquer algorithms
  - Dynamic programming
  - Greedy algorithms
  - Graph algorithms
  - Backtracking
  - Branch and bound
- They are not specific algorithm, but a algorithm family.
- The algorithms in the same family share the same general idea to solve problems.

Analysis of Algorithms

- How good is an algorithm? Is correctness of an algorithm enough?
- Time efficiency
  - Indicates how fast an algorithm runs.
  - How many CPU cycles needed.
- Space efficiency
  - Amount of memory units required by an algorithm.
- Does there exist a better algorithm?
- How to compare algorithms?

Important Problem Types

- Sorting: Rearrange the items of a given list in nondecreasing order.
- Searching: Finding a given value, called a search key, in a given set or multiset.
- String processing: e.g., matching of text strings, find subsequences.
- Graph problems: e.g., modelling transportation, communication, social and economic networks (graph traversal algorithms, shortest path algorithms).
- Combinatorial problems: e.g., resource scheduling for wireless communications.
- Geometric problems: e.g., closest-pair problem, convex-hull problem for computer graphics and robotic vision.
- Numerical problems: Solving equations and systems of equations, computing definite integrals, evaluating functions.

Pseudocode

- In the lecture, I will follow the textbook to use C++-like pseudocode.
- In assignments and exams, you can use pseudocode based on any language (Pascal, Java, Python), as long as it can clearly show your algorithm steps even if somebody is not familiar with the language.

C++-Like Pseudocode

Data Type	Meaning
index	An integer variable used as an index
number	A variable that could be defined as integer (int) or real (float)

Operator	C++ symbol
and	&&
or	||
not	!

Comparison	C++ symbol
x = y	x == y
x ≠ y	x != y
x ≤ y	x <= y
x ≥ y	x >= y

C++-Like Pseudocode

- Some conventions:
  - Simply use A[][] to represent 2-d array variable in function argument.
  - if (low <= x && x <= high) can be written as if (low ≤ x ≤ high).
  - for (i = 1; i <= n; i++) can be written as repeat (n times) {}.
- Use short sentence to describe simple instruction:
  - if (x is in A[]) {}.
  - exchange S[i] and S[j].

Pseudocode Example

- Exchange Sort
  - Problem: sort n keys in nondecreasing order.
  - Inputs: positive integer n, array of keys S indexed from 1 to n.
  - Outputs: the array S containing the keys in nondecreasing order.
- The items for search and sort is commonly called *keys*. Their type is called *keytype* in pseudocode.

```
void exchangesort (int n, keytype S[])
{
    index i, j;
    for (i=1; i<=n; i++)
        for (j=i+1; j<=n; j++)
            if (S[j] < S[i])
                exchange S[i] and S[j];
}
```

Pseudocode Example

- Matrix multiplication
  - Problem: determine the product of two $n \times n$ matrices.
  - Inputs: a positive integer n, two-dimensional arrays of numbers A and B, each of which has both its rows and columns indexed from 1 to n.
  - Outputs: a two-dimensional array of numbers C, which has both its rows and columns indexed from 1 to n, containing the product of A and B.
- Recall that if we have two 2×2 matrices

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \text{ and } B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix},$$

their product $C = A \times B$ is given by

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j}.$$

```
void matrixmult(int n,
                const number A[][],
                const number B[][],
                      number C[][])
{
    index i, j, k;

    for (i=1; i<=n; i++)
        for (j=1; j<=n; j++){
            C[i][j] = 0;
            for (k=1; k<=n; k++)
                C[i][j] = C[i][j] + A[i][k] * B[k][j];
                //C[i][j] += A[i][k] * B[k][j];
        }

}
```

References

- Richard E. Neapolitan, Foundations of Algorithms (5th Edition).
  - Chapter 1

Acknowledgement: Thankfully acknowledge slide contents shared by Prof. Xuemin Hong