Algorithm

Course Introduction

Instructor: Jae-Pil Heo (허재필)

About Instructor

- Jae-Pil Heo (허재필)
  - Assistant Professor
  - Dept. of Computer Science & Engineering (소프트웨어학과)
- Joined SKKU in 2017

- Visual Computing Lab @ SKKU
  - Mainly focuses on Computer Vision and Machine Learning
  - 3 Ph.D., 9 master, and 5 undergraduate students + me

- E-mail: jaepilheo@skku.edu
- Office: 85472 (4^{th} floor in Corporate Collaboration Bd.)

Acknowledgements

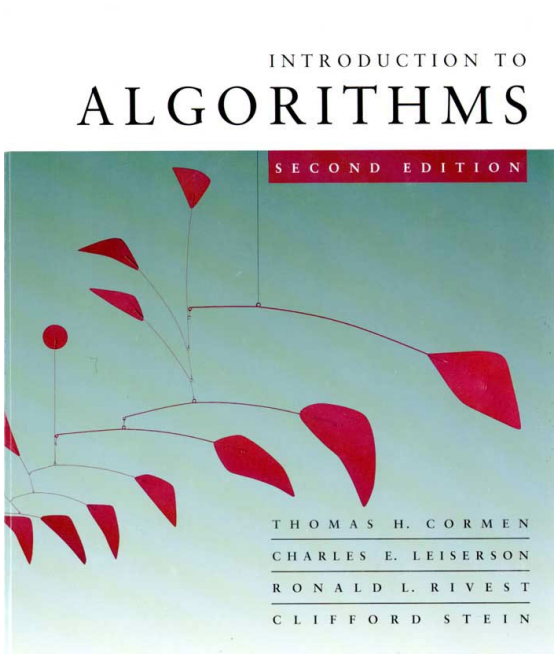
- The materials are built upon previous efforts of:
  - Prof. Douglas Wilhelm Harder (University of Waterloo)

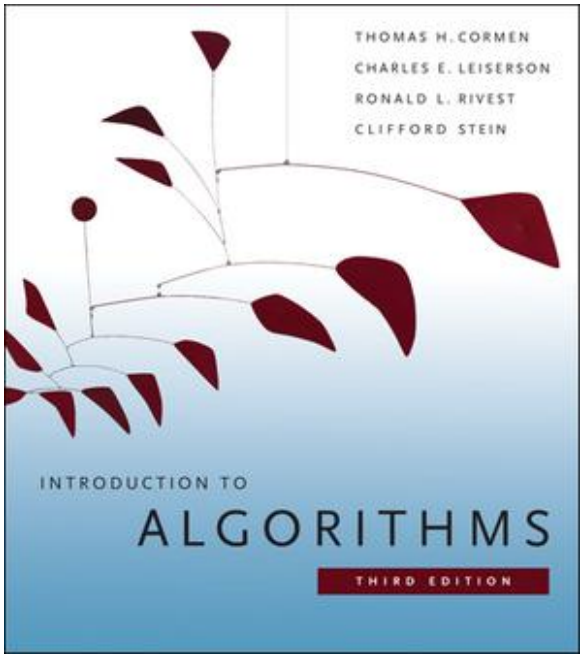
Prerequisites

- High-school math
- Programming skills
  - C-like programming language (C/C++)
- Data structures
- If you are unsure, consult the instructor at the end of this class.

Textbook

- Introduction to Algorithms, Third Edition
  - Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein
  - It is fine to refer the second edition.
  - The second edition is translated into Korean.





Course Objectives

- "How to think like a computer scientist"
  - A title of very famous book

- Computational thinking [Wikipedia]
  - Problem formulation (abstraction)
  - Solution expression (automation)
  - Solution execution and evaluation (analysis)

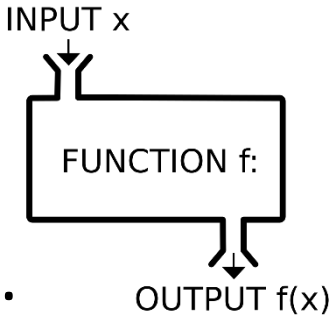
- Discuss key algorithms essential for solving problems with computers
  - Sorting, dynamic programming, greedy, graph algorithms,…

Course Objectives

- (Asymptotic) Algorithm analysis to evaluate solutions
  - Time/space complexities
- (Powerful) Programming skills
  - Translate your idea into languages that computers can understand
  - You should be a computer scientist who can implement whatever you understand.
- Most of CS guys including me agree with that
  - Algorithm is (one of) the most important subject(s) in Computer Science.

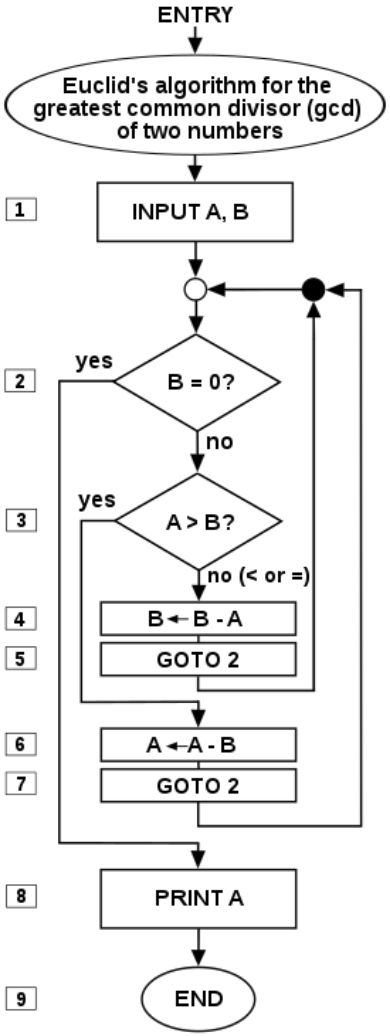
Algorithm

- An ***algorithm*** a sequence of computational steps that *transform the input into the output*.



- An ***instance*** of a problem consists of the *input* needed to compute a solution to the problem.

- A correct algorithm:
  - for every input instances, *halts* with the correct output



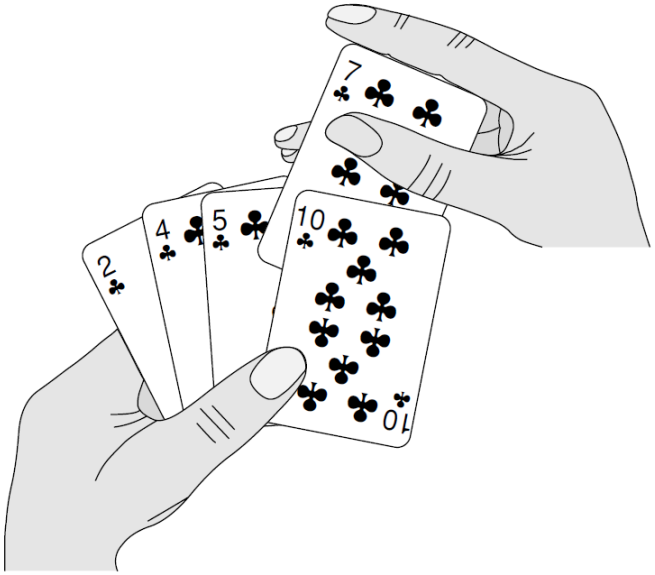
Figures from wikipedia

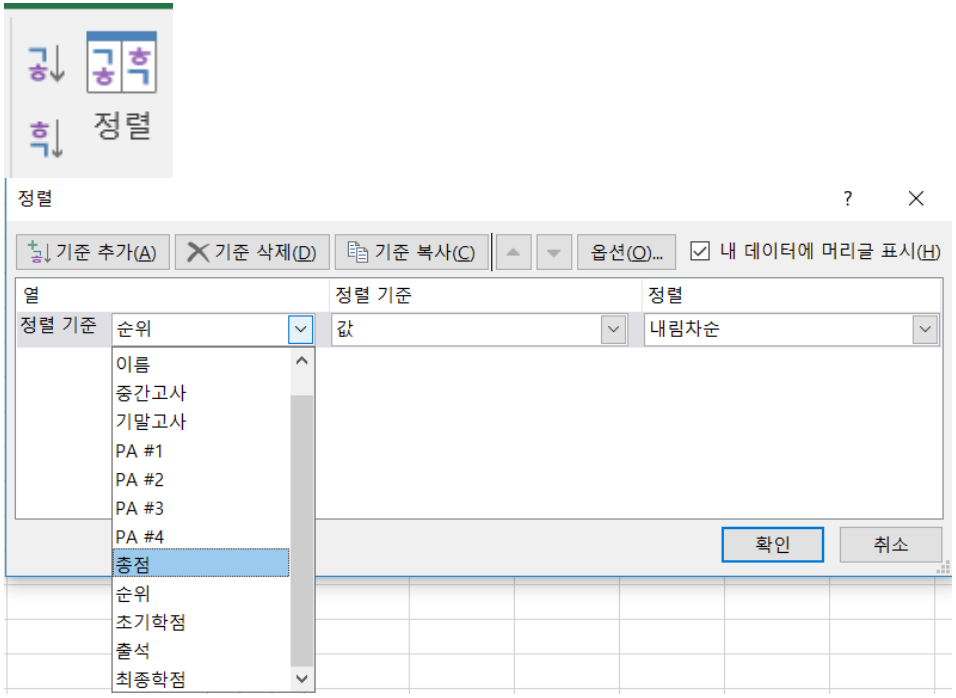
(Tentative) Topics

- Quick review of data structures
  - Lists, stacks, queues, trees, …
- Sorting algorithms
  - Insertion, bubble, heap, merge, quick, bucket, radix sorts
- Algorithm analysis
  - Growth of functions (asymptotic analysis), recurrences, amortized analysis
- Dynamic programming
- Greedy algorithms
- Graph algorithms
  - Minimum spanning trees, shortest paths, maximum flow, …
- NP-completeness
- Geometric algorithms
- Approximation algorithms
- …

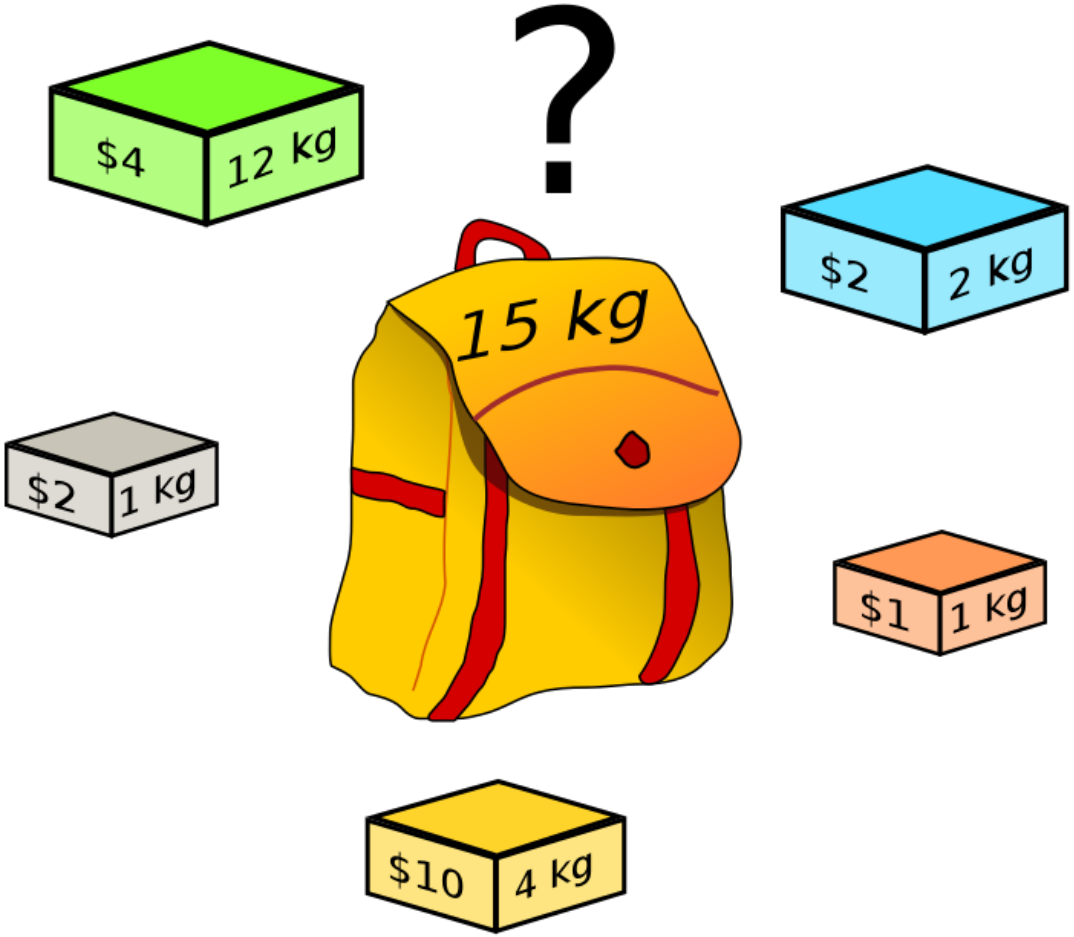
Sorting Problem

- Sorting problem
  - Input: A sequence of n numbers $< a_1, a_2, \ldots, a_n >$
  - Output: A permutation (reordering) of the input sequence, $< b_1, b_2, \ldots, b_n >$, such that $b_1 \leq b_2 \leq \cdots \leq b_n$
  - Instance: $< 7, 10, 4, 5, 2 >$





Knapsack Problem



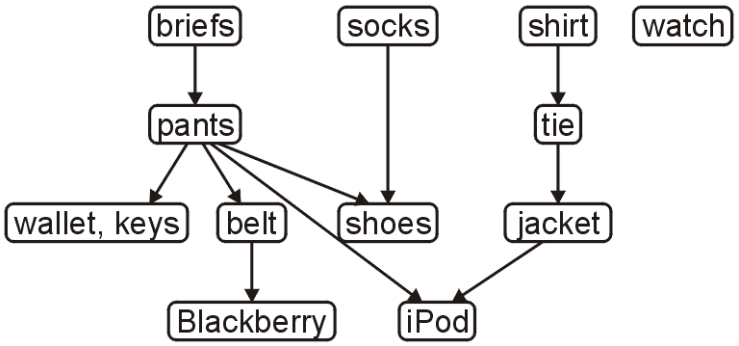
Figures from Wikipedia

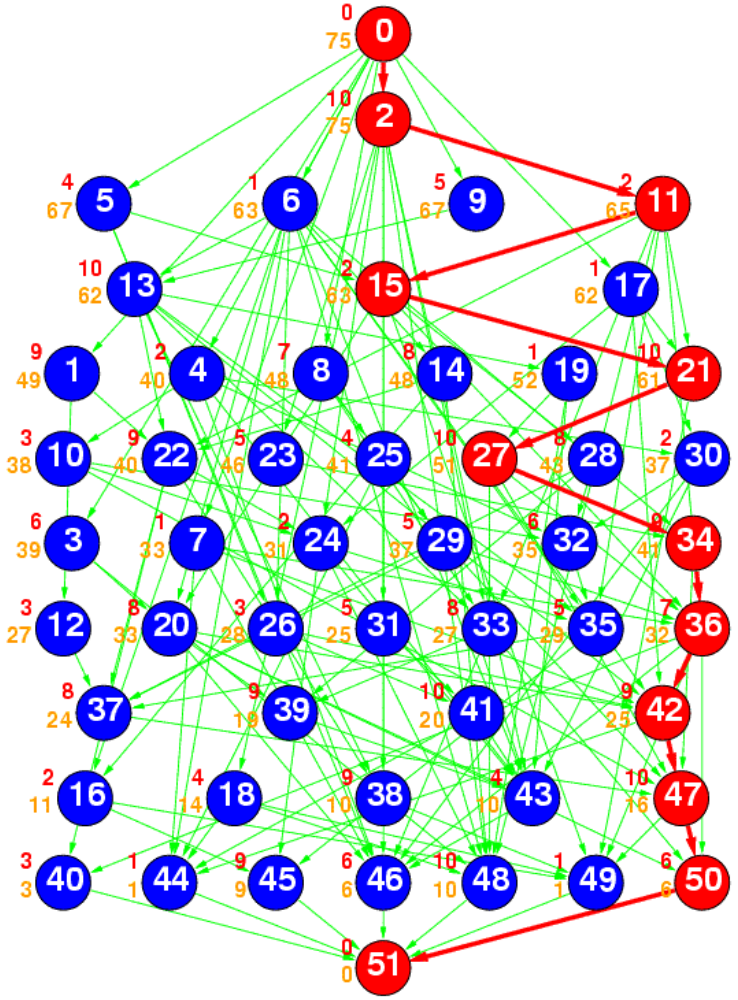
Edit Distance

		m	o	n	k	e	y
	0	1	2	3	4	5	6
m	1	0	1	2	3	4	5
o	2	1	0	1	2	3	4
n	3	2	1	0	1	2	3
e	4	3	2	1	1	1	2
y	5	4	3	2	2	2	1

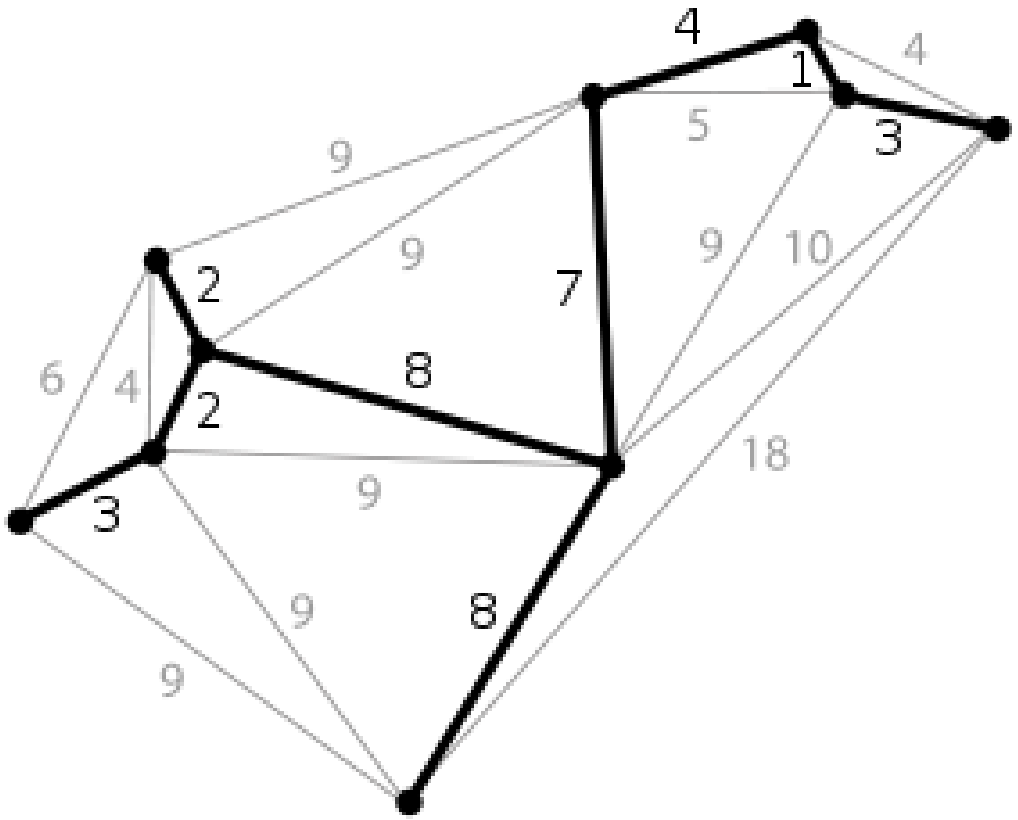
Figure from
https://vinayakgarg.wordpress.com/2012/12/10/edit-distance-using-dynamic-programming/

Topological Sort



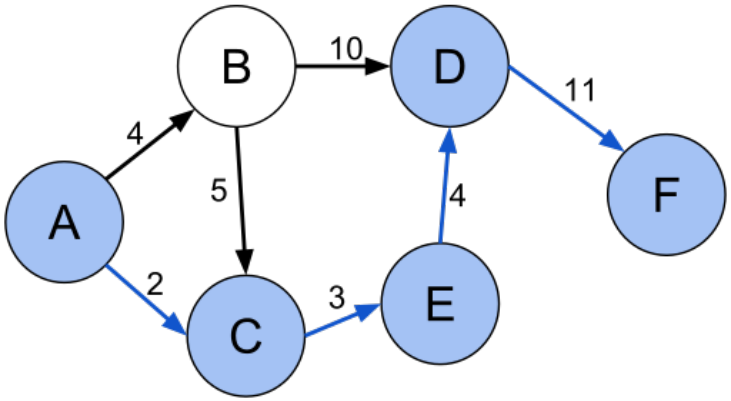


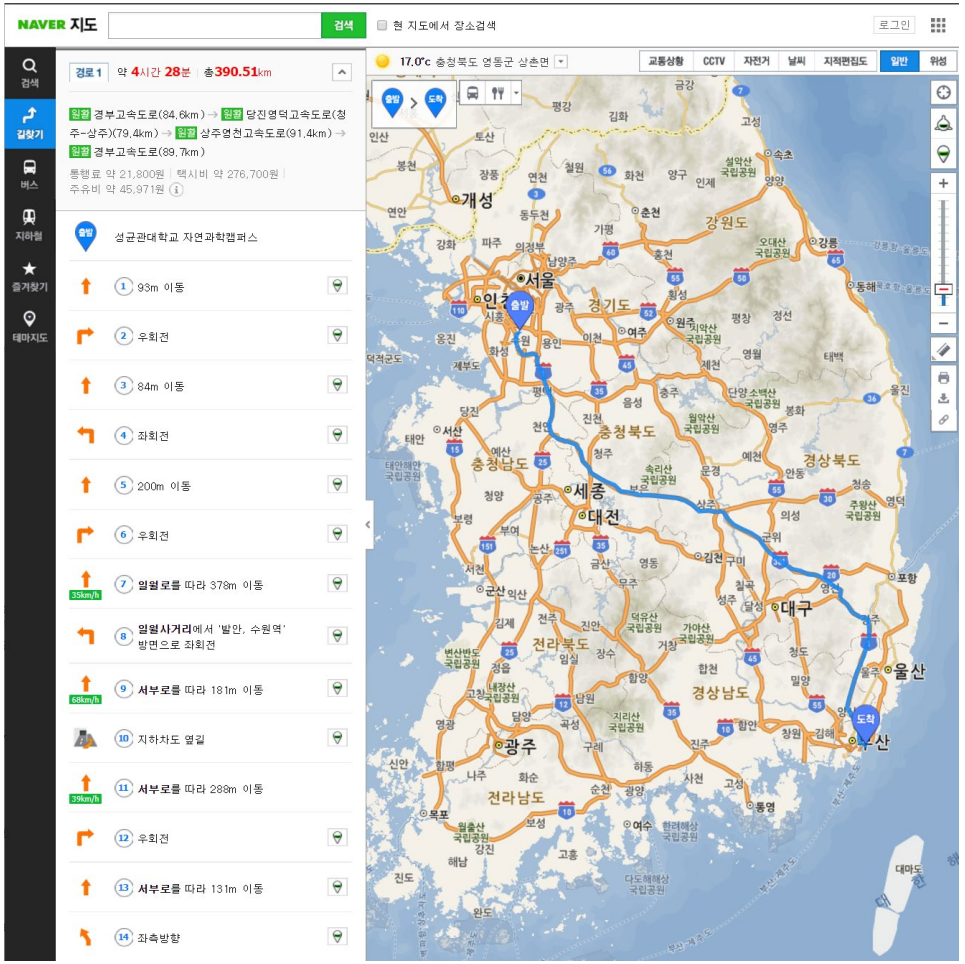
Minimum Spanning Trees (MST)



Figures from Wikipedia

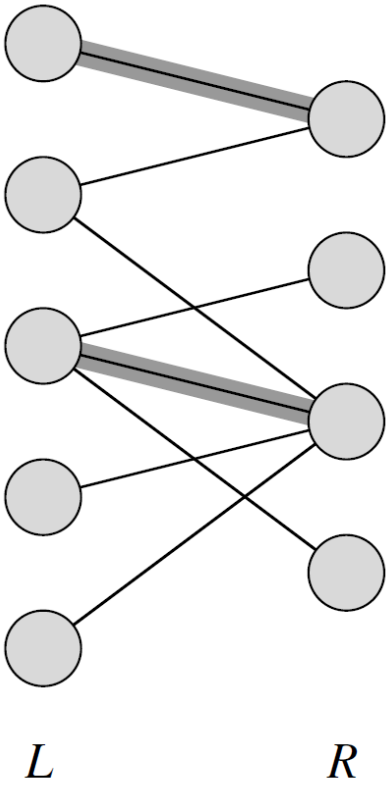
Shortest Path



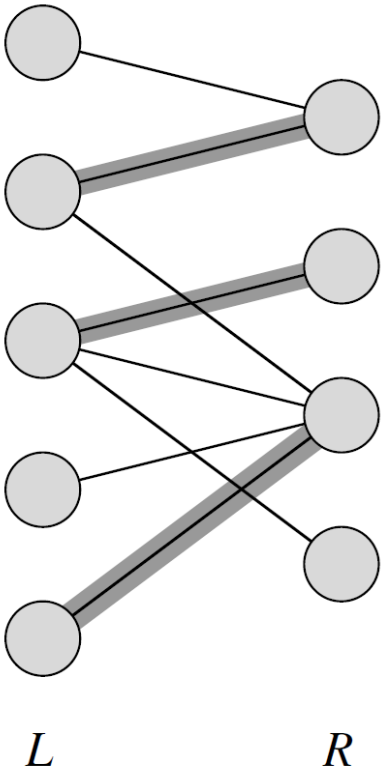


Figures from Wikipedia & NAVER

Network Flow

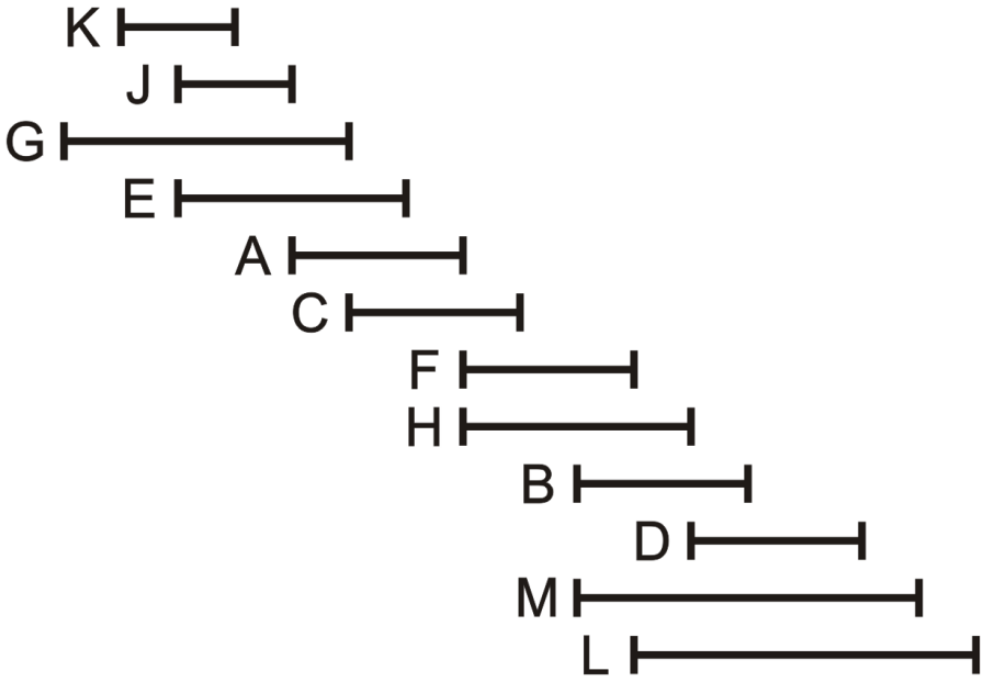


not maximum

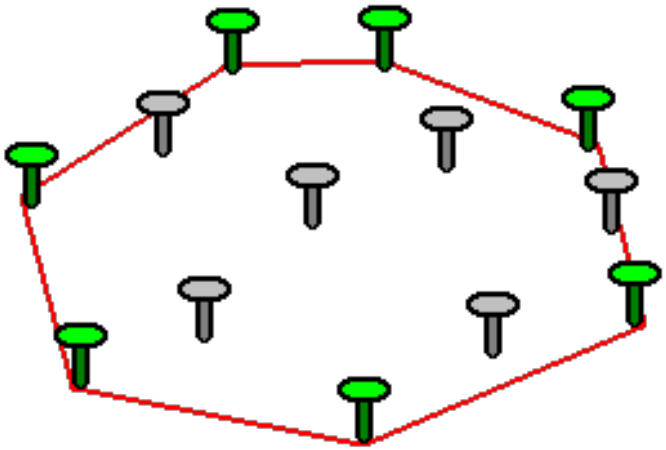


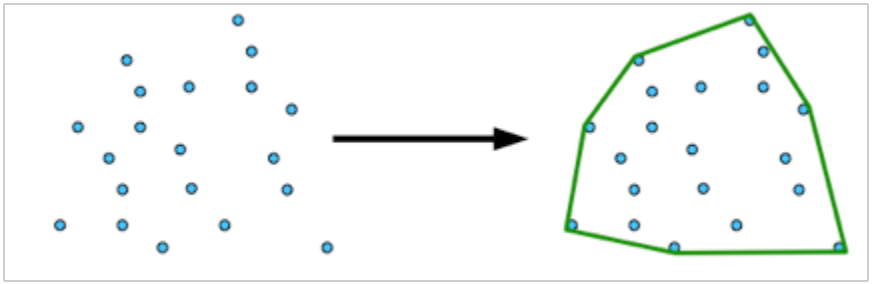
maximum

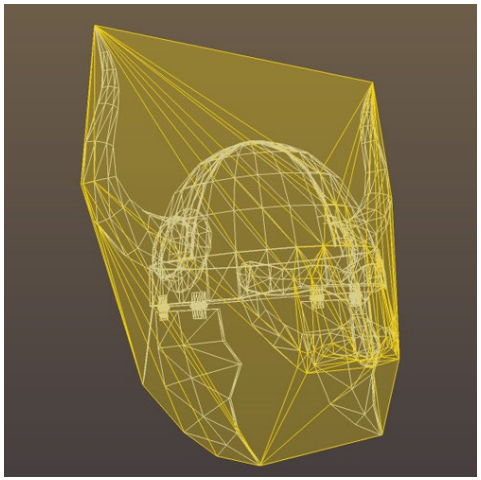
Interval Scheduling



Convex Hull







Figures from https://brilliant.org/wiki/convex-hull/

Grading

- Depending on the COVID-19 status, one of the following grading rules will be applied:
  - 1) Assignments 50% + Final Exam 50%
  - 2) Assignments 100%

- Assignments
  - (Mostly) Programming assignments
  - Late submission penalties for homework
    - Allowed to submit up to 48 hours after deadline
    - - 10% base penalty for any late submission
    - - 10% penalty for each 12 hours
    - Example:
      - 1s – 12H: -20%, 12H – 24H: -30%, …, 36H – 48H: -50%
  - Copy detection will be seriously performed.

Class Attendance Rule

- Every two absences → lower your grade (e.g., A+ → A)
  - Exemption for the first absence
  - Example
    - absence 3 – 4 times → 1 lower grade
    - absence 5 – 6 times → 2 lower grade
    - absence 7 – 8 times → 3 lower grade
    - ….. → F

Coding Sessions

- We will have coding sessions in class.
  - Instructor will perform implementation of algorithms covered in the class.
  - Codes implemented in the class will not be uploaded, since they could be used in the assignments.

Next Time

- Review: Mathematical Background
- Review: Data Structures

Any Question?