CS 4/56101
Design & Analysis of Algorithms

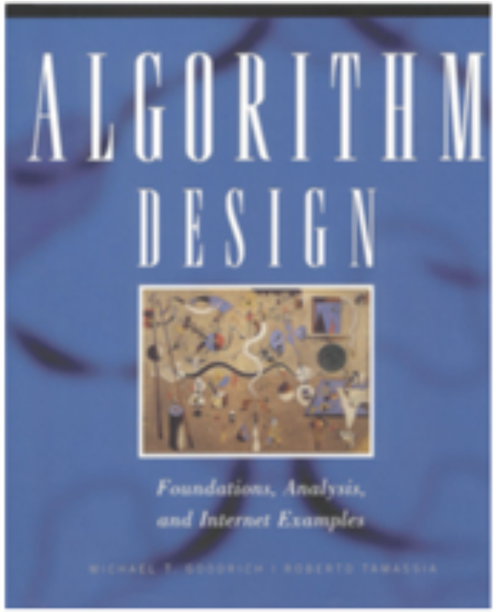
- **Course Website**:
  - http://www.cs.kent.edu/~hmichaud/daa-sum19/

- **Instructor**: Heather M. Guarnera
  - Office: MSB 352
  - Office Hours: by appointment
  - Email: hmichaud@kent.edu (**Piazza is best**)

Books

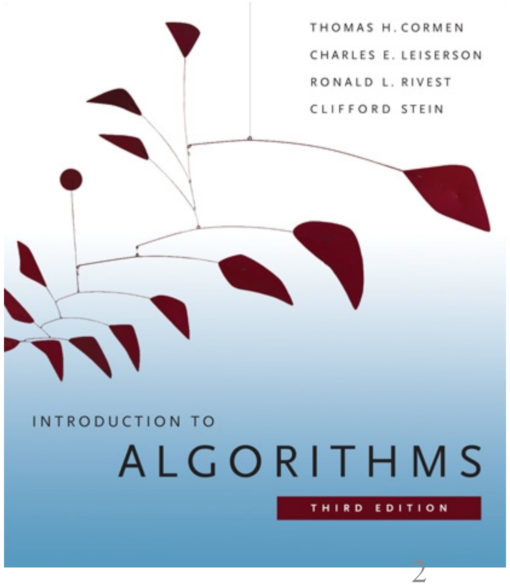
- Textbook:

Algorithm Design: Foundations, Analysis, and Internet Examples, by Michael T. Goodrich and Roberto Tamassia, 1st edition, Wiley, 2001



- An excellent reference:

Introduction to Algorithms, 3rd Edition, by T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, MIT, 2009.



Course Requirements

- Homework 35%
  - Good preparation for exams
  - Homework is weighted based on different problems

- Exams (closed book, no calculators, one sheet of notes)
  - Midterm 30% tentatively, Wed. July 10, half of class
  - Final 30% Fri. August 2, in class

- Participation 5%
  - Engagement in class and on Piazza

Other Syllabus Info

- Late Policy
  - Homework must be turned in by the end of class on the due-date.
  - Unexcused late homework is not accepted.
  - Missed exams and missed homework are only excused if absence was essential and can be fully documented.

- Registration Requirements
  - June 12: Official registration deadline
  - June 16: Last day to withdraw before a grade of "W" is assigned
  - July 14: Last day to withdraw with a grade of "W" assigned

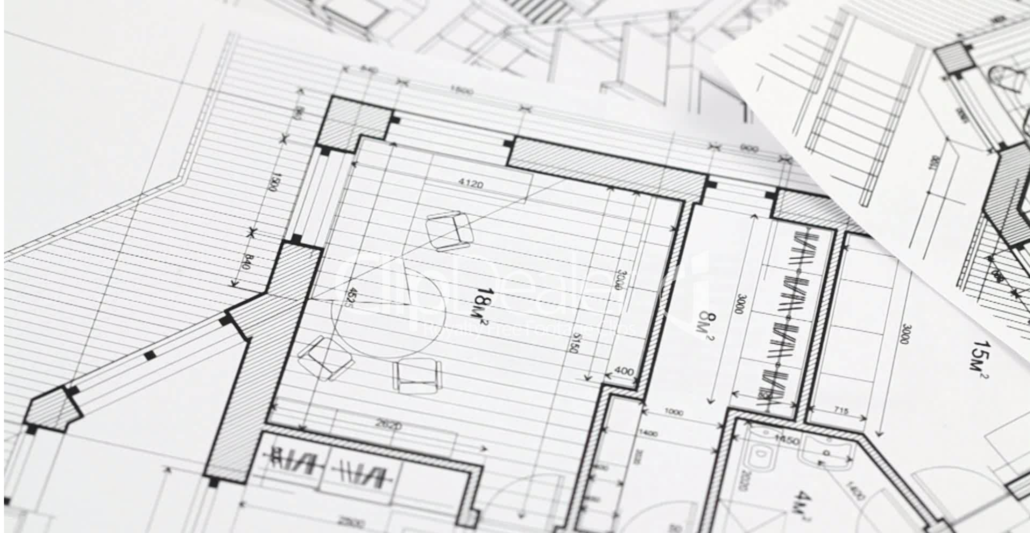
Example: Boss assigns a task

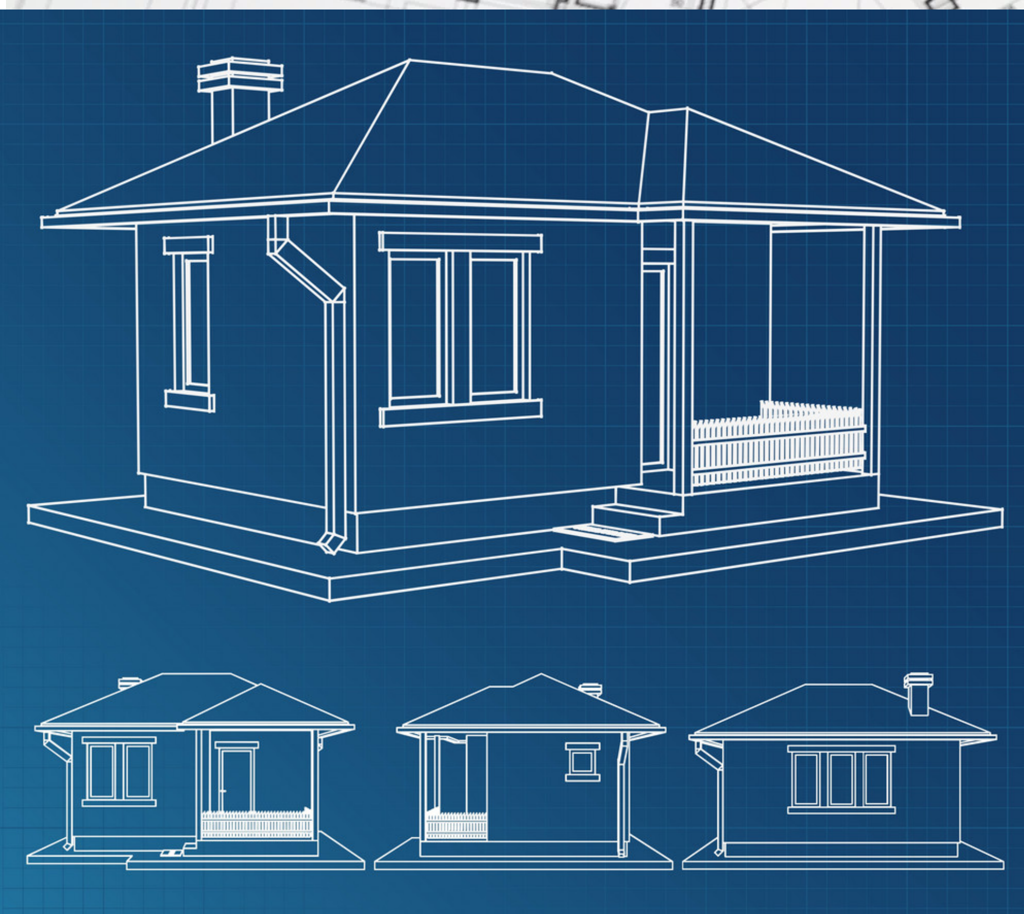
- Given today's prices of pork, grain, sawdust, etc…
- Given constraints on what constitutes a hotdog.
- Make the cheapest hotdog.

Every industry asks these questions.

- Mundane programmer: "Um? Tell me what to code."
- Better: "I learned an algorithm that will work."
- Best: "I can develop an algorithm for you."

How to do this?


18м²
8м²
15м²
4м²

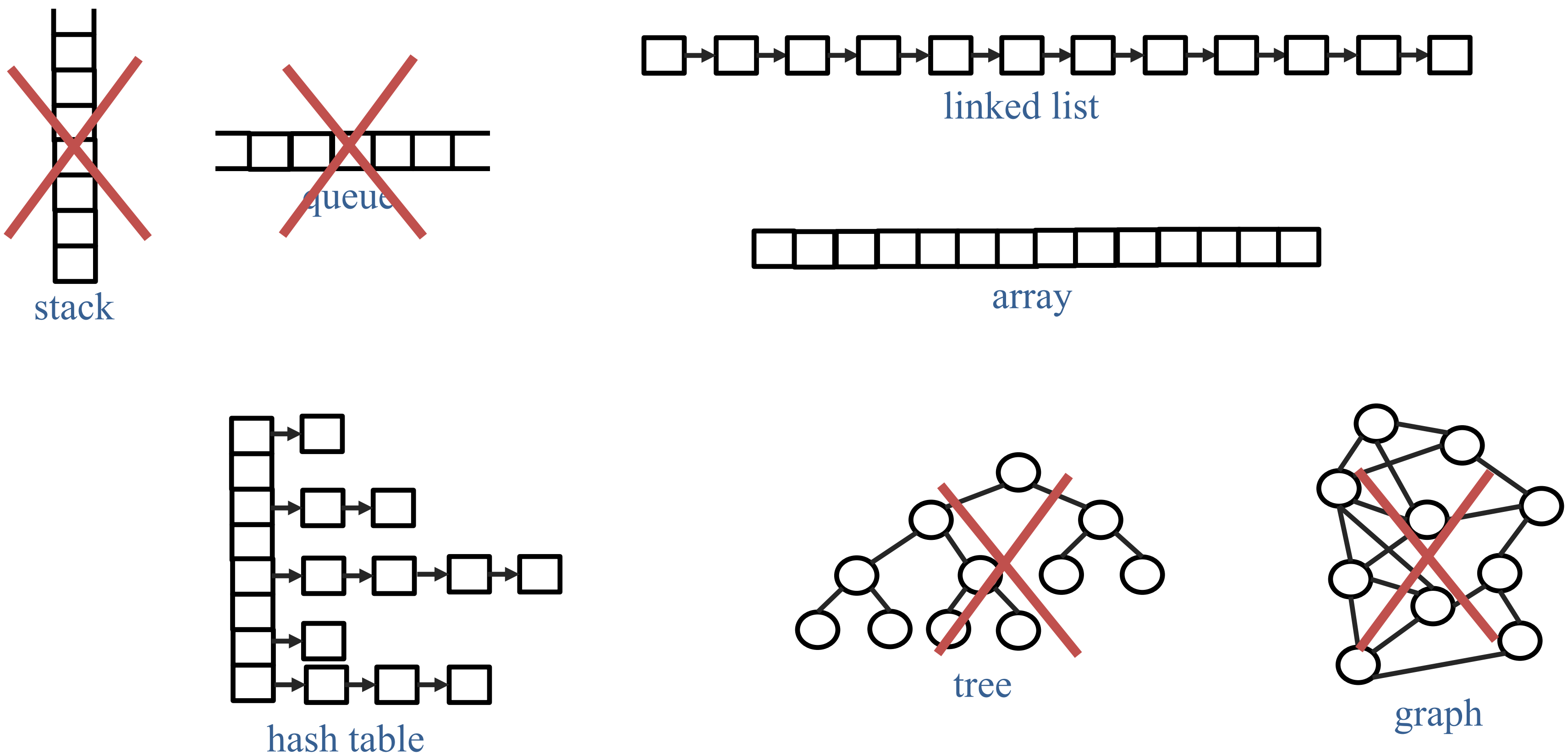



PETER RUPRECHT

Tools you need

Example: Design an inventory system which can quickly find an item.

- What data structure to use?



Tools you need

Example: Design an inventory system which can quickly find an item.

- What approach to take?

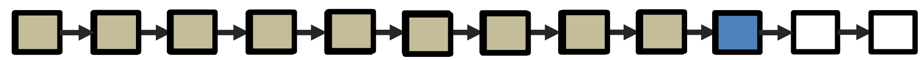
Brute force

Dynamic programming

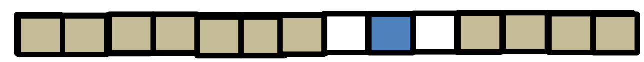
Divide and conquer

Greedy method

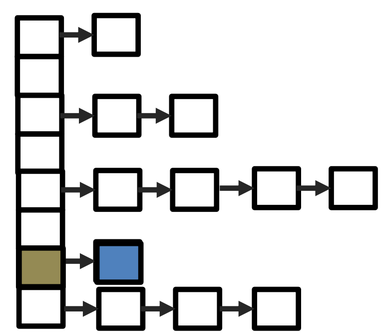
Prune and search



linked list



array



hash table

- Are there any existing algorithms that could be used/modified?

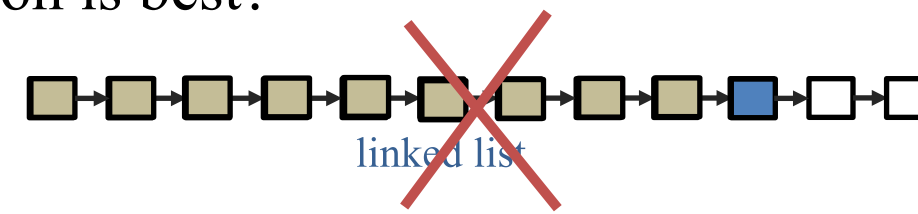
Tools you need

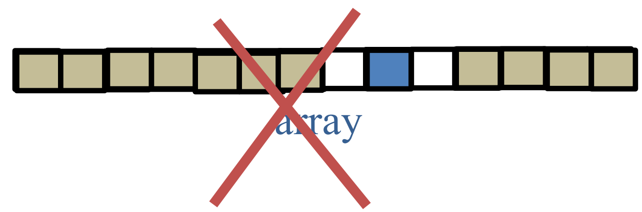
Example: Design an inventory system which can quickly find an item.

- How to determine which solution is best?
- Does it work as required?

  Rationalization

  Proof of correctness



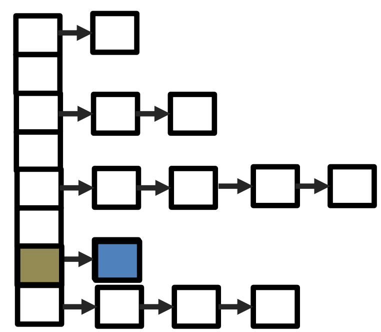


- How much memory is required? How long does it take?

  Big-oh notation

  Amortization

  Complexity analysis



hash table

Design & Analysis of Algorithms

- How to evaluate algorithms (correctness, complexity)
  - Notations and abstractions for describing algorithms
- Advanced data structures and their analysis
- Fundamental techniques to solve the vast array of unfamiliar problems that arise in a rapidly changing field
  - Up to date grasp of fundamental problems and solutions
  - Approaches to solve
- To think algorithmically like a ‘real’ computer scientist

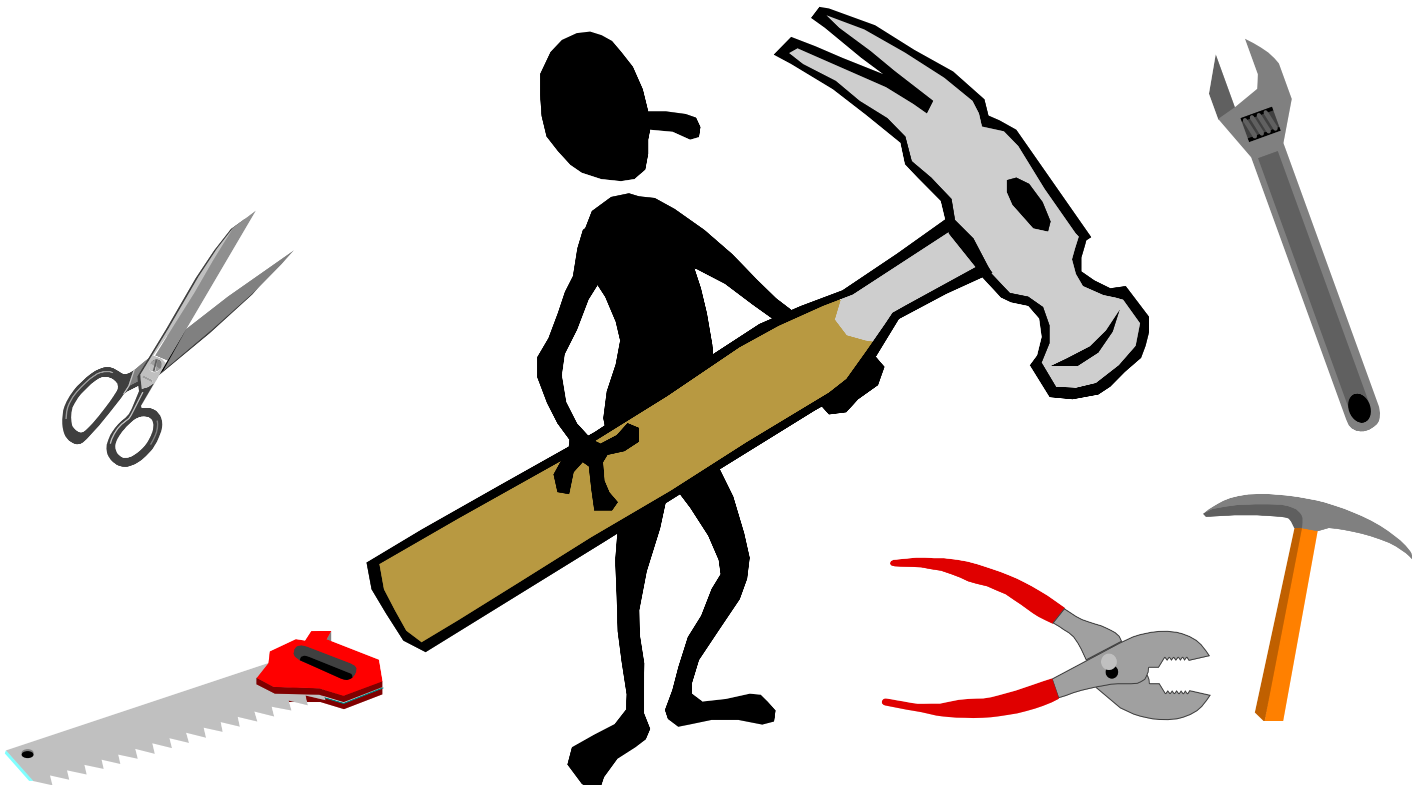
Course Content

- A list of algorithms
  - Learn the code
  - Trace them until you are convinced that they work
  - Implement them.

```
  class InsertionSortAlgorithm extends SortAlgorithm
{
   void sort(int a[]) throws Exception {
        for (int i = 1; i < a.length; i++) {
            int j = i;
            int B = a[i];
            while ((j > 0) && (a[j-1] > B)) {
                        a[j] = a[j-1];
                        j--;  }
            a[j] = B;
                }
}
```

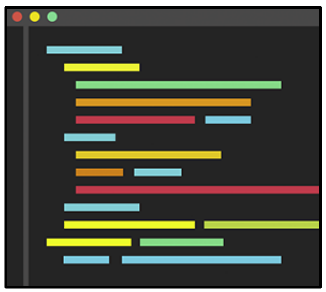
Course Content

- A survey of algorithmic design techniques
- Abstract thinking
- How to develop new algorithms for any problem that may arise



Start with some math

Time complexity as a function



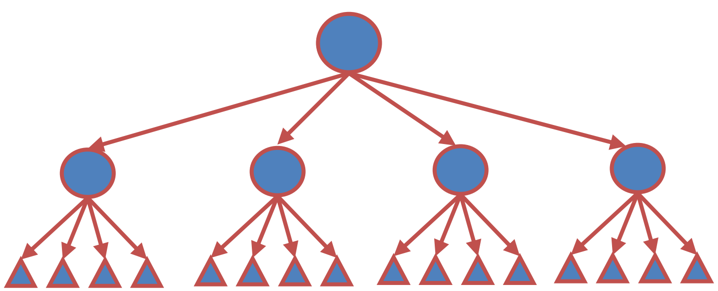
$t(n) = \Theta(n^2)$

Counting primitive operations

- Sequences and summations
- Linear functions
- Logarithmic and exponential functions

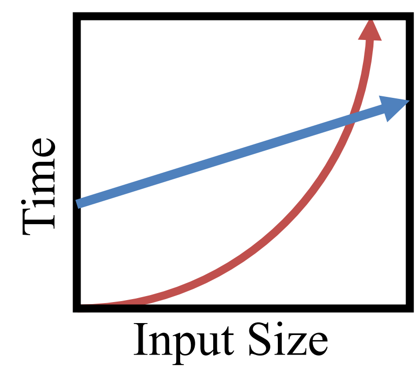
$$a + ar + ar^2 + ar^3 + \cdots + ar^{n-1} = \sum_{k=0}^{n-1} ar^k = a\left(\frac{1-r^n}{1-r}\right)$$

Recurrence Relations

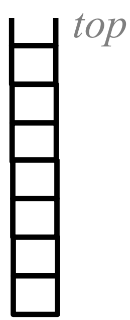


$T(n) = a\,T(n/b) + f(n)$

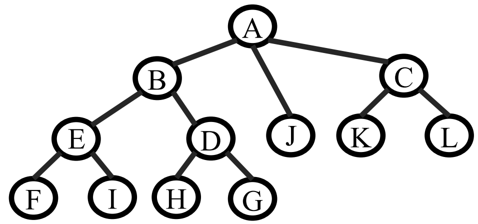
Classifying functions



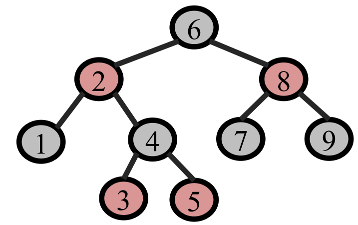
Data Structures



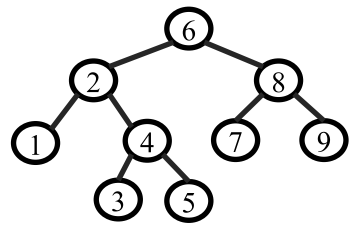
stack



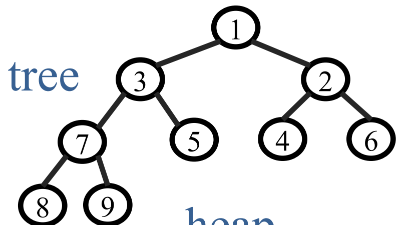
tree



red black tree



binary search tree

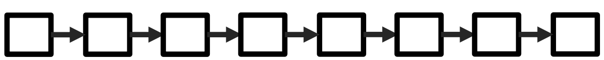


heap

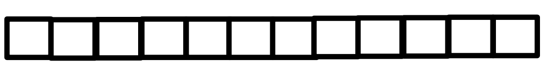
&

priority queues

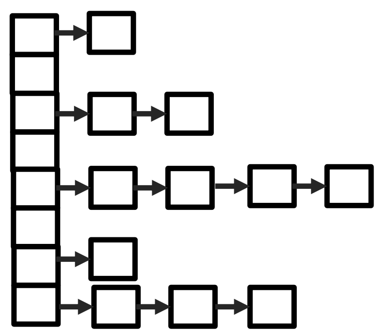
queue



linked list



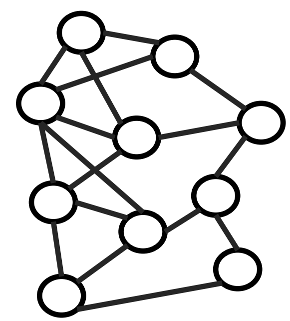
vector



hash table

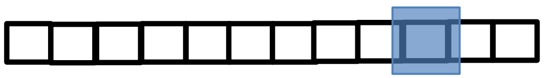
&

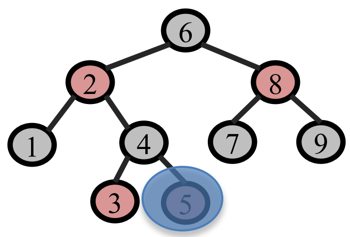
dictionaries

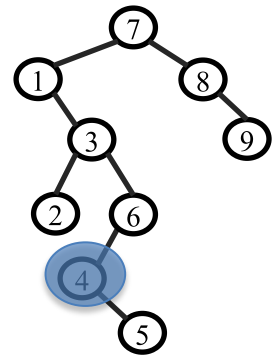


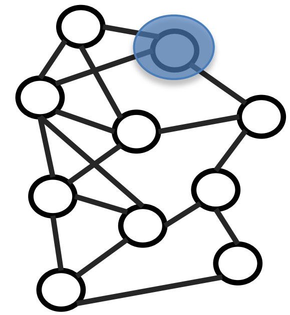
graph

Searching & Sorting









insertion sort

selection sort

heap sort

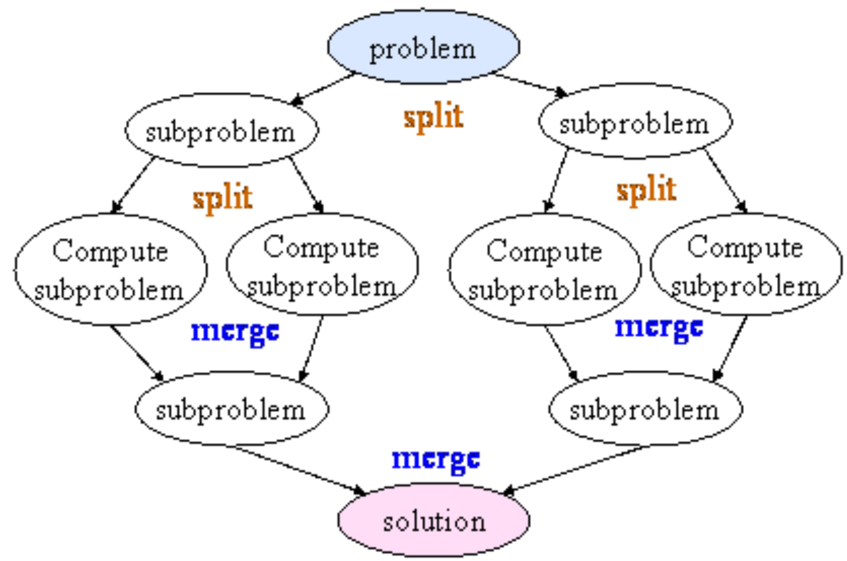
merge sort

quick sort

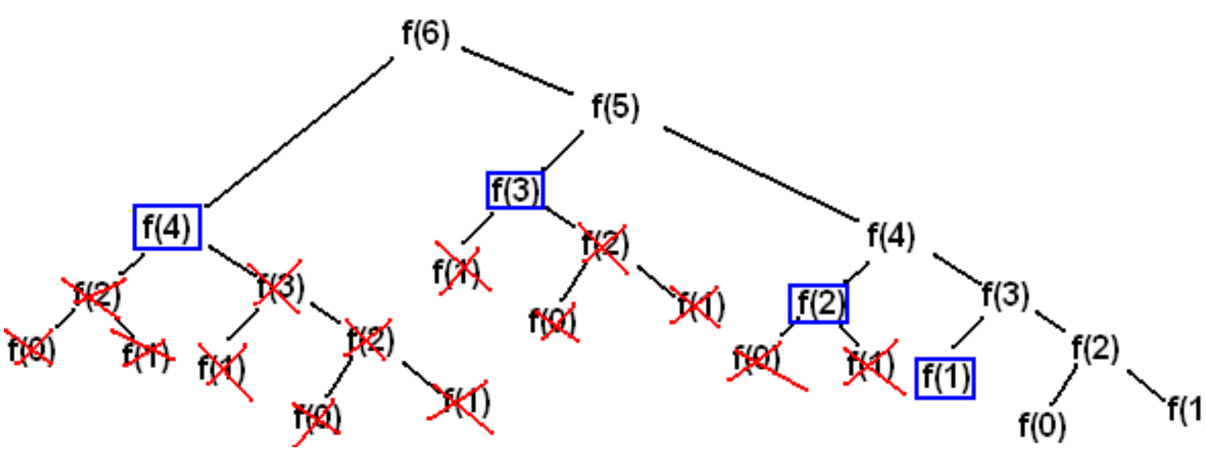
Fundamental Techniques

Greedy Algorithms

Divide and Conquer

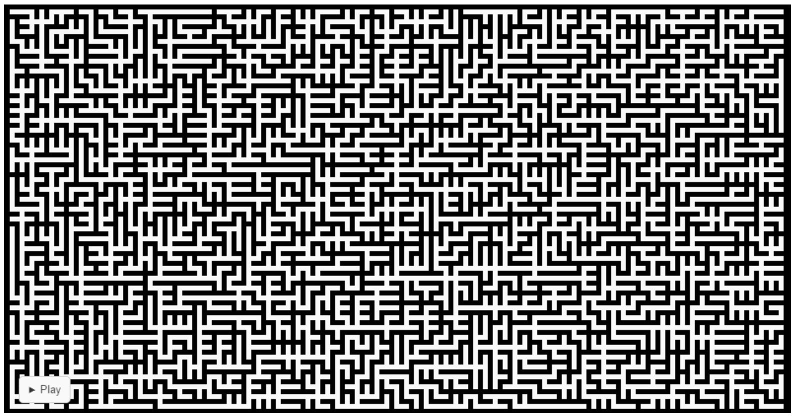


Dynamic Programming

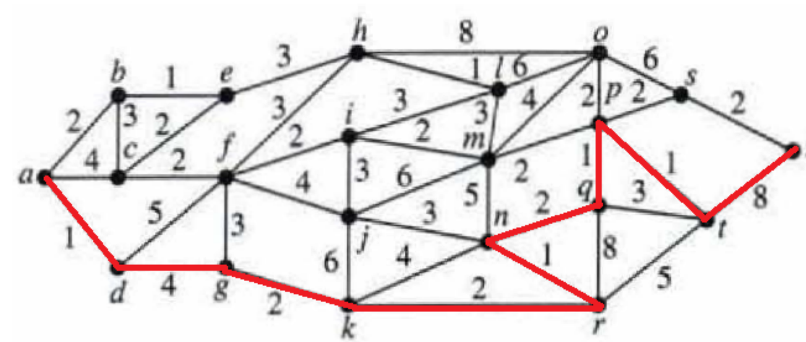


Graphs & Graph Algorithms

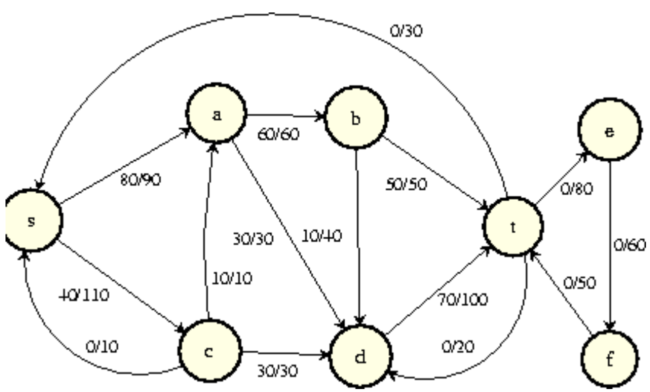
Graph search



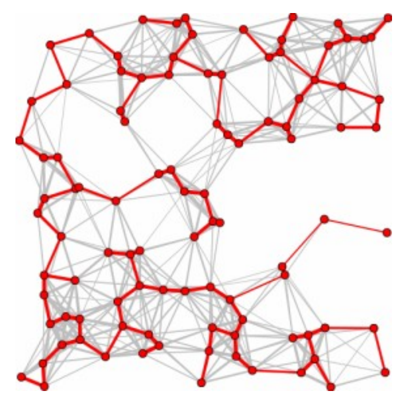
Shortest path



Network flow



Minimum Spanning Tree



Useful Learning Techniques

- You are expected to read ahead (before the lecture)
  – This will facilitate more productive discussion during class
- Guess at potential algorithms for solving a problem
  – Look for input instances where your algorithm is wrong
- Practice explaining
  – You'll be tested on your ability to explain material
- Ask questions
  – Why is it done this way and not that way?