CS 5150/6150
Advanced Algorithms

Aditya Bhaskara
University of Utah
Welcome

• First year (MS/PhD)

• UG?

• Most recent algorithms class?

• Discrete math/probability/calculus?
Algorithms?

• Backbone of every computation
Goals of this course

• How to design “efficient” algorithms (general principles)

• Analyze the run time, memory usage, etc of an algorithm, and prove correctness

• Understand limits of efficient algorithms
Logistics

Course homepage:
http://theory.cs.utah.edu/fall18/algorithms/ (video links, notes, ...)

Canvas for HWs, discussion, announcements

Teaching Assistants:
• Maheshakya Wijewardena
• Hong Xu
• Amin Mohammadi
• Sravan Kumar Neerati
Grading

• Homeworks: best 5 out of 6 – 60%

• Project – 20%

• Final Exam – 20%
All HWs must be prepared in LaTeX/Markdown, submitted as pdf !!
LaTeX

**Warning**: if you’ve not used LaTeX, start HW early!!

- Submit HW0 on Friday
Collaboration

- Discussion allowed on HWs, but **list all** collaborators for every problem

- Cite all online sources

- **Write solutions yourselves**

- Official policy:  
  https://www.cs.utah.edu/academic-misconduct/
Pre-requisites
Basic data structures/algorithms

• Arrays, lists, binary search trees, ...

• Big-Oh (don’t care about constants, lower order terms) e.g. $T(N) = 2N^2 + 8N + 42$

• Run time analysis

Why do we not care about constants?
Graphs

• Basics of graphs – breadth/depth first search, shortest paths
Question

Given array of N integers


**Question:** how quickly can you find if a given number x is in the array?

*What if the numbers are not sorted? equal elements?*
Homework example

Question 5: Finding $A[i] = i$  

This is from Dasgupta/Papadimitriou/Vazirani, Exercise 2.17

You are given a sorted array of $n$ elements. Determine if there's an $i$ such that $A[i] = i$. 

[10]
Describing algorithms

```cpp
bool sorted(vector<int>& A) {
    for (int i=0; i<N-1; i++)
        if (A[i] > A[i+1]) return false;
    return true;
}

void sort(vector<int>& A) {
    while (!sorted(A)) {
        for (int i=0; i<N; i++) {
            if (A[i] > A[i+1]) swap(A, i, i+1);
        }
    }
}
```
Describing algorithms ...

input: array A[0, ..., N-1]

procedure sort(A):
    while A is not sorted do:
        for i from 0... N-1:
            if (A[i] > A[i+1]), swap them;

*Use text, not code; assume basic sub-routines...*
Describing algorithms ...

input: graph G = (V, E)

procedure Euler(G):
    find odd degree nodes, set ‘cur’ to one of them
    while (there exists an unused edge ‘e’ out of ‘cur’):
        mark ‘e’ as used
        set ‘cur’ = other end point of ‘e’
    if all edges in G are used, return YES, else return NO
Analysis exercise

input: array A[0, ..., N-1]

procedure sort(A):
    while A is not sorted do:
        for i from 0... N-1:
            if (A[i] > A[i+1]), swap them;
Analysis exercise

• What is the worst case run time?

• Is this bound tight?

• Are there inputs where it runs faster?

• Is it “typically” slow?
Course outline
Next class

• Recap: basic data structures
  • binary search trees
  • graphs and adjacency lists

• Recurrence relations, analyzing running times
  • HW 1 will be out by the weekend