§1 Introduction

Qin Zhang
Brief self-introduction:

My name: Qin[Chin] Zhang

I have been a CS theoretician for 10+ years;

I mainly work on **Algorithms for Big Data**, in particular, data stream algorithms, sublinear algorithms, algorithms on distributed data, I/O-efficient algorithms, data structures

I have published in almost all major conferences/journal of theoretical computer science (STOC, FOCS, SODA, PODS, SPAA, ICALP, STACS; JACM, SICOMP, TALG, Algorithmica).

I write experimental papers too, and have published in the best database and machine learning venues (SIGMOD, NIPS).
“A procedure for solving a mathematical problem (as of finding the greatest common divisor) in a finite number of steps that frequently involves repetition of an operation.” — webster.com

“An algorithm is a finite, definite, effective procedure, with some input and some output.”

— Donald Knuth
Why study algorithms?

Algorithms are used everywhere, any time

**Internet.** Web search, packet routing ...

**Biology.** DNA similarity search, protein folding

**Multimedia.** MP3, JPG, face recognition ...

**Social networks.** Recommendations, advertisements ...

**Daily life helpers:** Google maps, auto translation ...

This course tries to introduce some basic concepts, techniques and tools for algorithm design. These may serve as building blocks for solving real-world problems
It is all about efficiency

We want to design algorithms that are time, space and communication efficient
C343 vs B403

- **C343**: Implementation and make use of classic data structures and algorithms.

```java
private static void sort(double[] a, int lo, int hi)
{
    if (hi <= lo) return;
    int lt = lo, gt = hi;
    int i = lo;
    while (i <= gt)
    {
        if (a[i] < a[lo]) exch(a, lt++, i++);
        else if (a[i] > a[lo]) exch(a, i, gt--);
        else i++;
    }
    sort(a, lo, lt - 1);
    sort(a, gt + 1, hi);
}
```

- **B403**: Design and analysis of algorithms.

\[
\sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{2}{j - i + 1} = 2 \sum_{i=1}^{N} \sum_{j=2}^{N-i+1} \frac{1}{j} \\
\leq 2N \sum_{j=1}^{N} \frac{1}{j} \\
\sim 2N \int_{x=1}^{N} \frac{1}{x} \, dx \\
= 2N \ln N
\]
Course schedule

0 : Introduction
   – Big-O notations, common running times

1 : Graph
   – BFS, DFS, DAG, topological sorting

2 : Greedy Algorithms
   – Interval scheduling, MST, shortest path

3 : Divide & Conquer
   – Mergesort, counting inversions, closest pair

4 : Dynamic Programming
   – Weighted interval scheduling, subset-sum, edit distance

5 : NP and Intractability
   – Polynomial reduction, NP-completeness, hard problems
Course schedule

0 : Introduction
   – Big-O notations, common running times

1 : Graph
   – BFS, DFS, DAG, topological sorting

2 : Greedy Algorithms
   – Interval scheduling, MST, shortest path

3 : Divide & Conquer
   – Mergesort, counting inversions, closest pair

4 : Dynamic Programming
   – Weighted interval scheduling, subset-sum, edit distance

5 : NP and Intractability
   – Polynomial reduction, NP-completeness, hard problems

If time permits, I will show you some advanced algorithm design techniques. E.g., approximation algorithms, randomized algorithms
Textbooks

- **Required textbook**
  - *Algorithm Design*
    by J. Kleinberg and E. Tardos
    Pearson Education
  
The book comes with slides:
  http://www.cs.princeton.edu/~wayne/kleinberg-tardos/
  (or Google “Algorithm Design slides”)

- **Supplementary Textbook**
  - *Introduction to algorithms*
    by T. Cormen, C. Leiserson, R. Rivest, C. Stein
    3rd edition. MIT
Instructors

- Instructor: Qin Zhang  
  Email: qzhangcs@indiana.edu  
  Office hours: Wed. 3-4pm @ Lindley 430A

- Associate Instructors:  
  - Erfan Sadeqi Azer  
  - Xiaomeng Ye  
  - (Helper) Haoyu Zhang  
  Office hours: posted on course website
Grading

- Assignments 20%: 6 written assignments
  
  **Due before class on the due date, in hard copy.** If you can, typeset in your favorite software and bring printed hard copy to class.
  
  If you are handwriting, make sure it is legible.
  
  No extensions or late homeworks will be granted.

- Project 15%: Solve 2 programming tasks on HackerRank website. See course website for details.

- Exams 65%: Mid-term (30%) and Final (35%).
Practice is very important to master algorithm design.

1. Subsections in the textbook that we do not cover in class
2. Solved exercises in the textbook
3. Other exercises in the textbook (do not appear in homeworks). Feel free to ask us questions if you meet any difficulty (email us the question number first so that we can get prepared).
4. Any other questions that you can find online – there are tons of algorithm design questions online. Again, feel free to ask us questions if you meet any difficulty.
Prerequisites

Participants must have a background in **discrete math** and **data structures**, and have taken

1. **C241** Discrete Structures for Computer Science
2. **C343** Data Structures
   

3. **MATH-M 216** "Analytic Geometry and Calculus II" (or **MATH-M 212** CALCULUS II)
Thank you!
Questions?