Syllabus

General Info

Course CS225 Algorithmic Analysis

Instructor Charilaos Skiadas (skiadas at hanover dot edu)

Term Winter 2017-2018

Office SCH 121C / LYN 108

Office Hours MWF 10-11 in SCH 121C, T 12-2 in LYN 108, and by appointment

Book *Introduction to the design and analysis of algorithms*, by A. Levitin, 3rd edition

Websites for notes[1].

Class times MWF 2:40pm-3:50pm in LYN120A.

Course Description

Algorithmic Analysis is the in-depth study of algorithms and their efficiency. Algorithms are ubiquitous in Computer Science; along with Data Structures they constitute the fundamental blocks of any programming activity, namely:

1. How do we represent our application's data/information? (data structures)
2. How do we process this information to achieve the application's goals? (algorithms)
3. How do we assess and improve the efficiency of this processing? (algorithmic analysis)

As a very simple example, imagine that you need to process 1 million records with a certain algorithm. With algorithmic analysis you may be able to determine that this algorithm will take at least 10 years to run, and therefore you will have to find some other way around the problem and will not need to spend time actually coding this algorithm.

Algorithmic analysis may also help you determine how an algorithm's running time may grow with the size of the input: If it takes us 10 seconds to process these $1,000$ records, how long will it take us to process $10,000$ records? The answer really depends on the algorithm we use and its efficiency, exactly the kind of work that algorithmic analysis focuses on.

In this class we will build on your knowledge of basic data structures as well as build on it. Arrays, linked lists and binary trees are the fundamental building blocks for

[1] skiadas.github.io/AlgorithmsCourse/site/

more complicated data structures such as heaps and hash tables, and we will spend considerable time studying these applications.

Along the way we will also have an introduction to the Java programming language. Java is a mainstream language with relatively simple fundamental building blocks and strong support for object-oriented programming. The programming assignments will explore various algorithmic techniques while at the same time building your comfort with Java.

One of the fundamental goals of the class is to introduce you to a bevy of problem-solving techniques, including:

- brute force methods
- depth-first and breadth-first graph searches
- decrease-and-conquer techniques (including insertion sort and topological sort)
- divide-and-conquer techniques (including mergesort and quicksort)
- transform-and-conquer (for example instance-simplification via presorting or representation-change in 2-3-trees)
- dynamic programming and other space-time tradeoffs
- greedy algorithms
- backtracking and branch-and-bound techniques for algorithmically-hard problems

Lastly, the course introduces you to the limitations of algorithmic power, and the study of the so-called NP-complete problems, which are not believed to be solvable in a polynomial amount of time. These topics will be further explored in our follow-up course, Theory of Computation.

Course Components

Reading Notes

On the website you will find a schedule[2] with links to documents for each class day. In those documents you will find notes for the day's lesson, and reading assignments.

Class Attendance and Participation

You are expected to attend every class meeting. You are only allowed to miss 3 classes without excuse, and every absence beyond that will result in 1 percentage point removed from your final grade. There are very few reasons that would qualify as an excuse for an absence. During each class day you will be working on and submitting activity sheets, which will be scored as your class participation grade.

[2] http://skiadas.github.io/AlgorithmsCourse/site/schedule.html

Homework Assignments

There will be regular homework assignments about once a week. These assignments will expect you to answer theoretical questions and/or design algorithms for certain problems. Homework assignments are 10% of your final grade.

Programming Assignments

There will be regular programming assignments about once every two weeks. The assignments are there to help you practice your understanding of some of the concepts, as well as to give you some familiarity with programming in Java. Programming assignments are 20% of your final grade.

Exams

There will be two midterms, tentatively scheduled for Friday, February 9th and Friday, March 16, and a final during finals week. **You have to be here for the exams**. If you have conflicts with these days, let me know as soon as possible. Do not plan your vacation before you are aware of the finals schedule.

Getting Help

- You should never hesitate to ask me questions. I will never think any less of anyone for asking a question. Stop by my office hours or just email me your question, which has the great benefit of forcing you to write it down in clear terms, which often helps you understand it better.
- You are allowed, and in fact encouraged, to work together and help each other regarding the notes and the theory. You can also discuss general questions about the programming assignments and the homework assignments. But I expect you to work on the programming assignments and the homework assignments on your own.

Grading

Your final grade depends on class attendance, homework, project, quizzes, midterms and the final, as follows:

Component	Percent
Participation	10%
Homework	10%
Programming	20%
Worst Midterm	15%
Middle Midterm	20%

Component	Percent
Best Midterm	25%

This gives a number up to 100, which is then converted to a letter grade based roughly on the following correspondence:

Letter grade	Percentage Range
A, A-	90%-100%
B+, B, B-	80%-90%
C+, C, C-	70%-80%
D+, D, D-	60%-70%
F	0%-60%