EC 504 – Advanced Data Structures and Algorithms

Spring, 2025

Lecture Time

M-W 2:30 PM – 4:15 PM, CDS B62

Staff Information

Instructor: Prof. David Castañón
Office: PHO 434
Email: dac@bu.edu (please use EC504 in the subject line when emailing)
Office hours: Tuesday 3-4pm, PHO 434 or by appointment

GST: Seyda Guzelhan (seyda@bu.edu)
Office hours: TBD

Course Description

This course will focus on data structures and algorithm design. As a graduate course in this area, the course will focus more on analysis and theoretical aspects rather than implementation details. Students are expected to be able to design and analyze new algorithms for problems based on the techniques discussed in class. Homeworks will be a mix of pen-and-paper exercises and implementation exercises.

There will also be a larger project component done in groups

Prerequisites

For undergraduates, EC330 (Applied Algorithms and Data Structures) is required. It is assumed that all students have taken a course in basic data structures and algorithms and are familiar with the following: sorting, asymptotic analysis, recurrence relations, basic data structures such as linked lists, stacks and queues, hash tables, binary search trees, and heaps. Discrete math skills are crucial for the algorithmic techniques discussed in class, particularly induction proofs.

Webpage

Announcements, course material, readings, and an updated schedule will be posted to the course Blackboard page on learn.bu.edu. Interactive discussions and questions will be conducted through Piazza.com. A link to the Piazza discussion page is included in the blackboard site.

Textbooks

Required: Cormen, Leiserson, Rivest, and Stein. *Introduction to Algorithms*. MIT Press, 2009 (3rd Edition). This book is exhaustive, and covers nearly all of the topics in the course (and many more that we will not have time to cover). It is an excellent reference to own. There is a recent 4th edition that covers overlapping topics, but the 3rd edition has better overlap with our topics.

Useful: Kleinberg and Tardos. *Algorithm Design*. Addison-Wesley, 2005. This book will be on reserve at the Science and Engineering Library. This book has more extensive coverage of algorithms, but does not focus on data structures.

Homework

There will be 6-8 written homeworks during the course, and 4 software homeworks. Written homeworks will typically include an on-line component, in the form of a Blackboard quiz, that will be auto-graded on the spot, and an off-line part with answers that must be submitted as pdf files on Gradescope. A Gradescope link will be provided from our course home page in learn.bu.edu. If you must turn in a off-line homework late, there will be a 25% penalty deducted for a one-day late written HW. No credit will be given for HW that is more than one day late.

Software

As part of this course, you will have an account on our virtual lab, the Shared Computing Clusters (SCC) at the Massachusetts Green High Performance Computing Center (MGHPCC) (see http://www.bu.edu/tech/services/research/computation/scc/)
This is modern HPC center. As best practice, you can use your laptop and the environment that fits your style and then the final code is run and benchmarked on the SCC using Linux.

For our software homework, we will provide data in text files and main programs in C++. If you wish to use a different language for your homeworks, you can do so by translating the provided main programs into equivalent main programs in your chosen language, and doing your implementations in your chosen language. As long as you generate the output files in correct format, we will accept the solution as appropriate.

Software HW will be submitted in your directories at scc1.bu.edu. Instructions will be provided to indicate what needs to be turned in, where it needs to be turned in, and in what format.
.
Important notes on homework:

When a question asks to design an algorithm, you need to provide a description of your solution with pseudo-code as well as a proof of correctness and an analysis of the running time. The clearest way to explain an algorithm is to use English, with some notation. Solutions that consist only of pseudo-code tend to be indecipherable by anyone except the author. These solutions also tend to have inaccuracies and are often incorrect. We reserve the right to deduct a significant number of points for solutions that consist only of pseudo-code with no explanation.

Homeworks can be done in groups (and in fact this is highly encouraged), but homework exercises need to be written up individually. Also, write down the names of any students with whom you collaborated. You must be able to fully explain your answers on demand, if necessary. You may not use resources

Exams

There will be two exams: a midterm exam in-class (date TBD) and a final exam (date TBD). The exams will be closed-book but some notes will be allowed.

Grade Breakdown

Midterm: 20% Final: 20%
Written Homework: 20% Software Homework: 20%
Project (including presentation): 20%

Accommodations for Students with Documented Disabilities

If you are a student with a disability or believe you might have a disability that requires accommodations, requests for accommodations must be made in a timely fashion to Disability & Access Services, 25 Buick St, Suite 300, Boston, MA 02215; 617-353-3658 (Voice/TTY). Students seeking academic accommodations must submit appropriate medical documentation and com

CourseTopics

This is a tentative ordered syllabus discussing what we will cover in class. Deviations will occur, depending on class progress

1. Review: Algorithm Analysis (CLRS 2-4)
    a. Asymptotic complexity: definitions, analysis, masters' theorem
    b. Recursions; application: order statistics
    c. Analysis of simple data structures (lists, stacks, queues)
2. Sorting: Some classical and modern approaches (CLRS 4,9, Notes on TimSort)
3. Review: Trees and their Properties (CLRS App. B)
4. Efficient Search in one-dimension
    a. Balanced Search Trees (Red-Black, Splay, B+ Trees) (CLRS 12,13,18,20)
    b. Priority queues (Binary, Binomial, Fibonacci,Rank-pairing Heaps) (CLRS 6,19, notes)
    c. Advanced hash tables (CLRS 9, Notes)
    d. Tries and String Matching: KMP algorithm, Aho-Corasick Tries (CLRS 32,notes)
5. Disjoint Sets (CLRS 21)
6. Graphs and Network Optimization 1
    a. Simple graph algorithms: traversals, topological sorts (CLRS 22)
    b. Minimum Spanning Trees:Kruskal, Prim's, Boruvka's algorithms (CLRS 23, notes)
    c. Other greedy algorithms: scheduling problems
7. Network optimization 2
    a. Single Source Shortest Paths: Dijkstra's algorithm , Bellman-Ford variations (CLRS 24)
    b. All-Pairs: Floyd Warshall, Johnson (CLRS 25)
    c. Single source, single destination: A* search (notes)
8. Network optimization 3
    a. Max-flow: Ford-Fulkerson, Edmonds-Karp, Preflow-push (CLRS 26, notes)
9. Dynamic Programming
    a. Weighted interval sum, knapsack, other applications
10. Advanced topics
    a. n-dimensional search: Quad-trees, k-d trees, R-trees (Notes)
    b. Complexity Theory (CLRS 34)
    c. Approximation Algorithms - Knapsack, Traveling Salesperson, … (CLRS 35)
    d. Other topics (games, etc) (Notes)