Design of Algorithms

Claudia Esteves & Jean-Bernard Hayet

Tue	9:30 a 10:50	G004
Thu	9:30 a 10:50	G004

Course Description

In this course, we will review the fundamentals in design and rigorous analysis of algorithms oriented toward the solution of data science problems. The first part of the course will be dedicated to reviewing data structures and algorithm paradigms; we will then focus on algorithms on graphs, searching problems, applications to data mining.

Prerequisites

Basic knowledge of discrete mathematics and linear algebra (vector spaces, bases, dimensions, matrices, linear applications, determinants, kernels).
At least one introductory programming course (control structures, conditionals, variables, functions).

Course Goals

On completion of the course, students will be able to

- Master fundamental concepts necessary to analyze algorithms in terms of complexity.
- Be able to read and reproduce proofs of correctness of algorithms and understand the different methodologies to elaborate them.
- Master the essential data structures and main algorithms for dealing with graphs, recursive problems and computational geometry.
- Discover the algorithmic nature of several problems involved in data science.
- Learn to implement rigorously and methodically an algorithm in a high-level programming language.

Course Content

Introduction	Introduction to the course (grading, bibliography, TA sessions, etc.).
Algorithms complexity	Asymptotic order of growth, Asymptotic notation, Big-O notation and properties.
Basic data structures	Stacks, queues and their implementation with deques in Python collections.
Graphs and trees	Graphs, Trees, Binary Trees, BSTs and their implementation in Python.

Binary search trees	Binary Search Trees (BST), and operations on BST. Balanced trees and applications to database indexing.
Heaps	Heaps, priority Queues, Heap sort. Find Median from Running Data Stream
Algorithms on graphs	Graphs in data science. Data structures for graphs. Traversal algorithms and applications. Shortest paths: Dijkstra; A*, Bellman-Ford. Minimum spanning trees: Kruskal's algorithm; Prim's algorithm; applications to clustering. The case of social network analysis: Basic concepts; Centrality measures. Spectral clustering and Page Rank.
Divide and conquer and dynamic programming	Problems with recursive solutions; DaD; examples. Dynamic programming.Examples.
Algorithms on large data sequences: Sequence alignment	Alignment through dynamic programming. Suffix arrays; Burrows-Wheeler Transform.
Searching in large datasets	The search problem - The Map ADT - Maps using sorted arrays and BSTs. Hashing: solving collisions; hash functions. Introduction to the hashing principle. Hashing functions - Handling collisions - Defining hashing functions for sequential data.
Searching in large datasets: Orthogonal Range Search	Searching with geometrical data - 1D range search with sorted arrays - 1D range search with BSTs - Range search in 2D: Grid based methods. Space partitioning trees. 2d-trees - interval trees.
Searching in large datasets: Similarity search	Similarity searching - VP trees. Approximate nearest neighbors.
Searching in large datasets: Locally sensitive hashing	Searching in large datasets: Locality Sensitive Hashing - Min-hashing - Applications.
Handling large datasets: Sketching and streaming	Motivation: Data streaming. A few examples. Hash-based sketching. Sampling; approximate quantiles

Bibliography

1. Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford, (2009). *Introduction to Algorithms,* 3rd Edition, MIT Press. [Main textbook]
2. Cormen, Thomas H. (2013). *Algorithms Unlocked*, MIT Press.
3. Kleinberg, Jon; Tardos, Eva (2005). *Algorithm Design*, Addison-Wesley.
4. Leskovec, Jure; Rajaraman, Anand; Ullman, Jeff (2014). *Mining Massive Datasets*, Cambridge University Press.
5. Hromkovic, Jurj (2005). Design and Analysis of Randomized Algorithms, Springer.

Support Sessions

2 hours a week with a teaching assistant

Grading

Two midterm exams (10% each), homework assignments (40%), final exam (20%), integrative project (20%)