Annotation
Algorithms are the heart of computer science, and the subject has countless practical applications as well as intellectual depth. This specialization is an introduction to algorithms for learners with at least a little programming experience. The specialization is rigorous but emphasizes the big picture and conceptual understanding over low-level implementation and mathematical details. After completing this specialization, you will be well-positioned to ace your technical interviews and speak fluently about algorithms with other programmers and computer scientists.

About the instructor: Tim Roughgarden has been a professor in the Computer Science Department at Stanford University since 2004. He has taught and published extensively on the subject of algorithms and their applications.

The primary topics in this part of the specialization are: asymptotic ("Big-oh") notation, sorting and searching, divide and conquer (master method, integer and matrix multiplication, closest pair), and randomized algorithms (QuickSort, contraction algorithm for min cuts).

Course program

1. Introduction; "big-oh" notation and asymptotic analysis.
   1.1. Why Study Algorithms?
   1.2. Integer Multiplication
   1.3. Karatsuba Multiplication
   1.4. Merge Sort: Motivation and Example
   1.5. Merge Sort: Pseudocode
   1.6. Merge Sort: Analysis
   1.7. Guiding Principles for Analysis of Algorithms
   1.8. The Gist
   1.9. Big-Oh Notation
   1.10. Basic Examples
   1.11. Big Omega and Theta

2. Divide-and-conquer basics; the master method for analyzing divide and conquer algorithms.
   2.1. O(n log n) Algorithm for Counting Inversions I
   2.2. O(n log n) Algorithm for Counting Inversions II
   2.3. Strassen's Subcubic Matrix Multiplication Algorithm
   2.4. O(n log n) Algorithm for Closest Pair I [Advanced - Optional]
   2.5. O(n log n) Algorithm for Closest Pair II [Advanced - Optional]
   2.6. Motivation
   2.7. Formal Statement

3. The QuickSort algorithm and its analysis; probability review.
   3.1. Quicksort: Overview
   3.2. Partitioning Around a Pivot
   3.3. Correctness of Quicksort [Review - Optional]
   3.4. Choosing a Good Pivot
3.5. Analysis I: A Decomposition Principle
3.6. Analysis II: The Key Insight
3.7. Analysis III: Final Calculations
3.8. Probability Review I
3.9. Probability Review II

4. Linear-time selection; graphs, cuts, and the contraction algorithm.
   4.1. Randomized Selection - Algorithm
   4.2. Randomized Selection - Analysis
   4.3. Deterministic Selection - Algorithm [Advanced - Optional]
   4.4. Deterministic Selection - Analysis I [Advanced - Optional]
   4.5. Deterministic Selection - Analysis II [Advanced - Optional]
   4.6. Omega(n log n) Lower Bound for Comparison-Based Sorting [Advanced - Optional]
   4.7. Graphs and Minimum Cuts
   4.8. Graph Representations
   4.9. Random Contraction Algorithm
   4.10. Analysis of Contraction Algorithm
   4.11. Counting Minimum Cuts

5. Breadth-first and depth-first search; computing strong components; applications.
   5.1. Graph Search - Overview
   5.2. Breadth-First Search (BFS): The Basics
   5.3. BFS and Shortest Paths
   5.4. BFS and Undirected Connectivity
   5.5. Depth-First Search (DFS): The Basics
   5.6. Topological Sort
   5.7. Computing Strong Components: The Algorithm
   5.8. Computing Strong Components: The Analysis

6. Dijkstra's shortest-path algorithm.
   6.1. Dijkstra's Shortest-Path Algorithm
   6.2. Dijkstra's Algorithm: Examples
   6.3. Correctness of Dijkstra's Algorithm
   6.4. Dijkstra's Algorithm: Implementation and Running Time

7. Heaps; balanced binary search trees.
   7.1. Heaps: Operations and Applications
   7.2. Heaps: Implementation Details [Advanced - Optional]
   7.3. Balanced Search Trees: Operations and Applications
   7.4. Binary Search Tree Basics, Part I
   7.5. Binary Search Tree Basics, Part II
   7.6. Red-Black Trees
   7.7. Rotations [Advanced - Optional]
   7.8. Insertion in a Red-Black Tree [Advanced]

8. Hashing; bloom filters.
   8.1. Hash Tables: Operations and Applications
   8.2. Hash Tables: Implementation Details, Part I
   8.3. Hash Tables: Implementation Details, Part II
   8.4. Pathological Data Sets and Universal Hashing Motivation
   8.5. Universal Hashing: Definition and Example [Advanced - Optional]
   8.6. Universal Hashing: Analysis of Chaining [Advanced - Optional]
   8.7. Hash Table Performance with Open Addressing [Advanced - Optional]
   8.8. Bloom Filters: The Basics
   8.9. Bloom Filters: Heuristic Analysis