CSCI 393, Algorithm Design and Analysis

Spring 2020 Final Exam Topics

Textbook Sections

Chapter 1: Entire Chapter
Chapter 2: Introduction and Sections 2.1 – 2.6
Chapter 3: Entire Chapter
Chapter 4: Introduction and Sections 4.1, 4.2, 4.3
Chapter 7: Introduction and Section 7.1
Chapter 8: Introduction and Section 8.1
Chapter 9: Introduction and Section 9.1
Chapter 11: Introduction and Section 11.3

Topics Covered in Class

1. Greatest Common Divisor (GCD)
2. Complexity – Time vs. Space
    a. Input size
        i. Linear search, Best Case, Worst Case, Average Case
    b. Measuring time
        ii. Cannot use clock time (ms)
        iii. Use Basic Operations
    c. Common “Basic Operations” (constant time operations)
    d. Asymptotic Behavior
        i. Big-O
        ii. Ω
        iii. Θ
        iv. Little-o
        v. Little-ω
        vi. Properties of Order handout
        vii. Evaluation code segments
    e. Algorithm: Element Uniqueness
    f. Algorithm: Square Matrix Multiplication
    g. Algorithm: Number of Binary Digits to represent a positive decimal number
3. Brute Force
    a. Overview
    b. Selection Sort
    c. Bubble Sort
        i. Improvements on Bubble Sort
    d. Sequential Search
        ii. Improvements on Sequential Search
    e. String Matching
    f. Square Matrix Multiplication

   g. Closest Pairs
   h. Convex Hull
4. Exhaustive Search
   a. Traveling Salesman Problem
   b. Knapsack Problem
   c. Assignment Problem
5. Divide and Conquer
   a. General Plan
      i. Divide instance of problem into smaller instances of the same problem.
         1. Ideally all smaller instances should be about the same size
      ii. Solve the smaller instances
         1. Typically, by recursion
      iii. If necessary, combine the smaller solutions to get the solution to the original instance
6. Mergesort and its recurrence
7. Binary Search
8. QuickSort and its recurrence
   a. Analysis of QuickSort
9. Randomized Quicksort
   a. Analysis of Randomized Quicksort
10. Recurrences
   a. Guess and Test
   b. Explicit formula
   c. Recurrence and recurrence relation (just started)
   d. Master Method
11. Ways to speed up algorithms
   a. Pre-Computing
      i. Do pre-work in order to speed up running time.
      ii. Example – Searching
         1. Pre-sort the values so you can use Binary Search
      iii. Example – Computer f(x) for many values of x
         1. Make a table so you can look up f(x)
      iv. Example – Hash values for data items
         1. Store hash values along with the data items
   b. Use Additional Information
      i. To develop a specific solution, rather than the general solution.
      ii. Example – Counting Sort
12. Dynamic Programming
   a. For problems with a recursive relation
   b. Solve problem “bottom up” instead of “top down”
   c. Solve problem by:
      i. Solver smaller instances first
      ii. Save solutions to smaller instances in a table
      iii. Look up in the table when smaller solutions are needed
   d. Example – Fibonacci Numbers

    e. Example – Binomial Coefficient
13. Optimization Problems
    a. Introduction to optimization problems
    b. Dynamic Programming used for optimization problems
    c. Example – Coin Row Problem
14. Greedy Algorithms
    a. Principle of Optimization
    b. Need to prove algorithm returns optimal solution
    c. Example – Prim's algorithm
        i. Proof of correctness for Prim's algorithm
15. The question of P vs. NP
    a. The set P
    b. The set NP
    c. The set NP-Complete