CPE 349 Midterm Study Guide

Vocabulary

- Graphs: directed, in-degree, out degree, weighted, path, acyclic, path length, depth, Euler path, Hamilton Path. source, sink, flow network, flow, flow value, cut, cut capacity, DFS and BFS (tree, back, forward, cross edges)
- Trees: leaf, height, spanning tree, minimal spanning tree
- Min Heap, Max Heap, sift down, sift up
- Exchange Argument, Stay ahead arguments to prove correctness of greedy algorithms

Sections from the text and topics from lecture and lab

Analysis knowledge

- Analysis framework – know steps and be able to apply
- Asymptotic notations and their properties, **Formal definitions** of O(), Ω() and Θ()
- Properties of logarithms, number of elements of n element set, permutations and combinations, summation formulas, arithmetic series, geometric series
- Recurrence relations and their solutions using back substitution and recursion tree, Be able to use Master Theorem – it will be provided if needed.

Brute force

- Traveling Salesperson and Knapsack problems
- Bread first search: Connectivity and BiPartite checking
- Depth first search; Pre and Post ordering of vertices

Basic data structure problems

- Topological Sort – both DFS and Source removal algorithms
- Recursive algorithms for generating subsets and permutations
- Partitioning algorithms, Heaps, Heap Sort (includes construction of heap)

Divide and conquer

- Selection problem
- Merge Sort, Quick Sort, Closest Pair (algorithm and proof of linearity of combine step)

Greedy approach

- Interval scheduling, proof of correctness, stay ahead argument
- Total Weighted Time to Completion – proof of correctness – concept of an exchange argument
- Prim's algorithm MST Cut property Proofs of correctness
- Dijkstra's Algorithm Shortest path proof of correctness
- Problem set on Greedy – proofs and analysis

Suggestions for Study:

- Review each algorithm by writing the pseudo code for it without referencing your notes. Repeat until know it. Avoid pure memorization. Think about what are the key steps and what implementation choices are important. For example, what type of loop, termination conditions etc.
- Do something similar for the definitions. What are the key things that define a complexity class. E.g. for $f(n) \in O(g(n))$, need two constants, one is to determine the input value beyond which the inequality holds. The other is the constant that multiplies the function g(n) so that it is larger than f(n). So use this to help you reconstruct the **formal mathematical definition**.
- Now review the labs, assignments, problems that were covered. Again the best way to do this is by self-testing. Can you solve them without referring to your notes.
- Start early, do not cram. Cramming may work for the exam but not for fixing the concepts in your long term memory. That means you will have to do it all over again for the final or for things like technical job interviews. Most importantly it will not become a new tool in your toolkit as a computer professional.