Final Exam Review Topics

The Final is 1-2:50 PM on Wednesday (5/8). Approximately, 30 % of the test will be comprehensive (chapters 1 - 5, Appendix B) and 70% will be from chapters 6 - 9 and 11.

The test will be closed-book, except for **three** 8.5"x11" sheets of paper containing notes (you may use both front and back of each sheet). You might want to read all of the test questions before answering any questions. This way you will be able to manage your time better.

Comprehensive Part (chapters 1-5, Appendix B) and chapter 6:

1) Algorithm Tracing: You should understand and be able to apply divide-and-conquer, dynamic programming, the greedy approach, backtracking, and best-first search branch-and-bound for problems discussed in class or on homework assignments.

2) Algorithm Design: You should understand and be able to apply divide-and-conquer, dynamic programming, the greedy approach, backtracking, and best-first search branch-and-bound to "new", simple problems.

3) Algorithm Analysis: You should be able to analyze recursive and non-recursive algorithms to determine their theta notation.

Problem-Solving Technique	**Problems**
Ch 1 Algorithms	Finding minimum/maximuim item in an array Sequential search
Divide-and-Conquer	Binary search Fibonacci Coin-change problem Quick sort Merge sort
Dynamic Programming	Coin-change problem Fibonacci Binomial Coefficient Traveling-Salesperson 0-1 Knapsack Problem
Backtracking	Coin-change problem 0-1 Knapsack Problem Sum-of-Subsets Problem Traveling-Salesperson
Branch-and-Bound	0-1 Knapsack Problem Traveling-Salesperson

Chapter 7: Computational Complexity of Sorting (Mainly sections: 7.1, 7.8, 7.9)
An understanding of the computational complexity argument for sorting using comparison of elements. The results of the analysis.
Understanding of Radix sort (don't waste your time memorizing the code, but understand how it works and its analysis)

Chapter 8: Computational Complexity of Searching (Mainly sections: 8.1, 8.4)
An understanding of the worst-case, computational complexity argument for searching using comparison of elements. The results of the analysis.
Understanding of the general concept of hashing.

Chapter 9. Introduction of the Theory of NP

Categories of Problems: P, intractable, and NP
Meaning of NP -- decision problems, polynomial-time nondeterministic algorithm
Meaning and usefulness of NP-Complete concept -- polynomial-time many-one reducible
Existence of NP-Complete problems -- CNF-Satisfiability and Cook's Thm. (Theory 9.2)
Using algorithm reduction/transformation to show a problem NP-Complete
Meaning of NP-hard problems
Handling NP-hard problems -- Backtracking & Branch-and-bound, poly-time algorithm for a restricted subclass of the problem, approximation algorithms
Approximation algorithm for TSP problems satisfying the triangle-inequality
Approximation algorithm for Bin-Packing problem

Chapter 11. Introduction to Parallel Algorithms

Parallel Architectures: Flynn's Categories - SISD, SIMD, MIMD
Address-Space Organization: Shared-Address-Space Architecture (UMA, NUMA), Message-Passing Architecture
Interconnection Networks: static (fully-connected, hypercube, 2D and 3D torus, fat-tree), dynamic (cross-bar switch), network bandwidth, bisection bandwidth
PRAM model: characteristics, R/W versions (EREW, ERCW, CREW, CRCW), algorithm extensions
CREW algorithms: find largest key in array, binomial coefficient (other dynamic programming problems), merge sort
CRCW algorithms: find largest key in array

For each of the following parallel computers, understand the general ideas of how they are programmed:

- Multi-core programming using pthreads: count of threes and TSP examples
- MPI Message passing examples with send/receive: count of threes, and TSP
- GPU programming: count of threes and TSP examples