5633 Analysis of Algorithms – Spring 05

1/18/04

Preliminary Schedule

Date	Material
Tu 1/18	**Describing and analyzing algorithms (Ch. 1, 2.1, 2.2)** Algorithm description and analyzation; best case and worst case runtimes; insertion sort, incremental algorithm
Th 1/20	**Asymptotic notation (Ch. 3, Appendix A), analyzing algorithms** O, Ω, Θ, o, limit-theorem; runtime for code-snippets, harmonic number
Tu 1/25	**Heapsort (Ch. 6)** Abstract data types (ADT), priority queue, heap, heapsort, linear-time buildheap *Homework 1 assigned*
Th 1/27	**Divide-and-conquer (Ch. 2.3) and recurrences (Ch. 4.1, 4.2)** Divide-and-conquer, merge sort, binary search; Runtime recurrences. Solving recurrences with recursion tree; solving the recurrence with the substitution method (induction)
Tu 2/1	**Master theorem (Ch. 4.3), more divide-and-conquer (31.6 pages 879–880; Ch. 30 pages 822–824; 28.2)** Use of master theorem to solve recurrences. Repeated squaring for exponentiation, Fibonacci numbers, polynomial multiplication, Strassen's matrix multiplication. *Homework 1 due*
Th 2/3	**Quicksort 7.1–7.3; Randomized algorithms, random variables and expected values (Ch. C.3)** Quicksort, best-case and worst-case runtimes, randomized quicksort; Expected runtime analysis. Random variables, expected value. *Homework 2 assigned*
Tu 2/8	**Quicksort (Ch. 7.3,7.4)** Randomized quicksort, expected runtime analysis.
Th 2/10	**Sorting (Ch. 8.1, 8.2, 8.3)** Decision trees, lower $\Omega(n \log n)$ bound for comparison sorts, counting sort, radix sort *Homework 2 due*
Tu 2/15	**Test 1** Material until 2/8 (inclusive)
Th 2/17	**Order statistics (Ch. 9)** Order statistics (find i-th smallest element); Randomized selection, deterministic selection in linear time
Tu 2/22	**Hashing (Ch. 11; not 11.3.3 and not 11.5)** Direct-address tables, chaining, open addressing with linear probing, quadratic probing, double hashing. Hash functions *Homework 3 assigned*
Th 2/24	**Amortized analysis (Ch. 17.1, 17.2, 17.4)** Aggregate analysis (total runtime of n operations), accounting method (prepay for later operations); binary counter, dynamic tables
Tu 3/1	**Red-black trees (Ch. 13.1, 13.2, 13.3)** Red-black tree property, rotations, insertion; abstract data types, ADT dictionary *Homework 3 due*

Date	Material
Th 3/3	**B-trees (Ch. 18.1, 18.2)** k-ary search trees, B-tree def., height, insertion *Homework 4 assigned*
Tu 3/8	**Dynamic programming (Ch. 15.2, 15.3, 15.4)** Fibonacci, binomial coefficient, LCS: fill table, then construct solution from the table.
Th 3/10	**Dynamic programming (Ch. 15.2, 15.3, 15.4)** Matrix chain multiplication; general outline of dynamic programming: Optimal substructure (recurrence), overlapping subproblems, fill table bottom-up or by memoization. *Homework 4 due. Homework 5 assigned*
Tu 3/22	**Greedy algorithms (Ch. 16.2 pages 380 middle – 384; problem 16-1 on page 402)** Greedy algorithms (greedy-choice property, optimal substructure). Making change, fractional knapsack.
Th 3/24	**Minimum Spanning Trees (Ch. 23)** Prim (grows single tree), Kruskal (grows forest; uses union/find data structure) *Homework 5 due*
Tu 3/29	**Test 2** Material from 2/10 until 3/22 (inclusive)
Th 3/31	**Union-Find (Ch. 21.1, 21.2, 21.3)** Operations, list implementation, tree implementation, union-by-weight / union-by rank, path compression. Ackermann function, and inverse Ackermann function α. *Homework 6 assigned*
Tu 4/5	**Single-source shortest paths (Ch. 24 without 24.4)** Optimal substructure, triangle inequality, relaxation step; Dijkstra (only for non-negative edge weights), predecessor tree (shortest path tree); Bellman-Ford, detection of negative-weight cycles; Shortest paths in a DAG
Th 4/7	**All-Pairs Shortest Paths (Ch. 25.2)** Dynamic programming: Floyd-Warshall
Tu 4/12	**Maximum Flow (Ch. 26)** Flow networks; Max-flow min-cut, augmenting path, residual network *Homework 6 due*
Th 4/14	**Maximum Flow (Ch. 26)** Ford-Fulkerson, Edmonds-Karp *Homework 7 assigned*
Tu 4/19	**P and NP (Ch. 34)** Decision problems, definition of classes P and NP, polynomial-time reductions
Th 4/21	**P and NP (Ch. 34)** NP-hardness, NP-completeness; Show that problems are NP-complete by reducing from other problems; TSP, Clique, Independent Set, Vertex Cover, Hamilton Path, Hamilton Circuit
Tu 4/26	**Augmenting Data Structures (Ch. 14)** Augmenting red-black trees; Dynamic order statistics, interval trees *Homework 7 due*
Th 4/28	**Range Trees** Range trees, in 2 dimensions and in d dimensions; preprocessing time, query time.
Tu 5/3	**Test 3** Material from 3/24 until 4/26 (inclusive)