BU CLA CS 530: Analysis of Algorithms

14 March 1996

Mid-Term Test

Reminder: This is a closed-book test. You are allowed one 8.5 by 11 crib sheet which you have to hand in with your test book.

Read carefully all questions before you start, organize your time, start with what you know best.

Problem 1 (15 points).

Determine whether each of the claims below is **true** or **false**. If it is true, give a short explanation (a couple of lines). If it is false, give a short counterexample.

(a) **Claim**: The worst-case running time of randomized quicksort on an input array of length $n \geq 1$ is at least $\Omega(n^2)$.

(b) **Claim**: The problem of determining an optimal order for multiplying a chain of matrices can be solved by a *greedy algorithm*, because it displays the optimal substructure and overlapping subproblems properties.

(c) Consider the problem of finding "all-pairs shortest paths" in a directed graph $G = (V, E)$ with weight function $w : E \to \mathbf{R}$.

Claim: Johnson's algorithm for this problem reduces it to $|V|$ invocations of Dijkstra's algorithm for finding "single-source shortest paths" in $G = (V, E)$ with an appropriately defined weight function $\hat{w} : E \to \mathbf{R}$.

(d) Partition a connected undirected graph $G = (V, E)$ with weight function $w : E \to \mathbf{R}$ into two disjoint components $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, where $V_1 \cup V_2 = V$ and $E_i = E \cap (V_i \times V_i)$ for $i = 1, 2$. Find the minimum spanning trees T_1 and T_2 for G_1 and G_2. Connect T_1 and T_2 by a smallest-weight edge between G_1 and G_2 — call the resulting spanning tree T.

Claim: T is a minimum spanning tree for G.

(e) **Claim**: In $\mathcal{O}(V + E)$ time, a matching in a bipartite graph $G = (V, E)$ can be tested to determine whether it is maximum.

Problem 2 (10 points).

Let $A[1\,..\,n]$ be an array of n integers. Let n be a power of 2. Write the pseudocode for, or describe precisely, an algorithm that finds the smallest and second smallest entries in A, such that two requirements are satisfied:

(1) The algorithm uses at most $n + \lceil \log n \rceil - 2$ comparisons.

(2) The algorithm does *not* use any memory space other than $A[1\,..\,n]$ to store the entries initially stored in it.

To get full credit both requirements must be satisfied.

Problem 3 (10 points).

Let n be an even positive integer. Design an efficient algorithm to partition an input array $A[1\,..\,n]$ of n integers into $n/2$ pairs satisfying the following property: For each pair, we compute the sum of its members, and if $s_1, s_2, \ldots, s_{n/2}$ are these $n/2$ sums, the algorithm finds a partition that *minimizes* the *maximum* sum.

Problem 4 (10 points).

Let $G = (V, E)$ be a directed graph with two weight functions $w_1 : E \to \mathbf{R}$ and $w_2 : V \to \mathbf{R}$, i.e. not only edges are assigned weights but so are vertices. Now the weight of a path $p = \langle v_0, v_1, \ldots, v_k \rangle$ is

$$w(p) = \sum_{i=1}^{k} w_1(v_{i-1}, v_i) \;+\; \sum_{i=0}^{k} w_2(v_i)$$

Design an efficient algorithm to find the shortest path between two vertices u and v, for all pairs $u, v \in V$. *Hint*: Consider a reduction to the standard formulation of the "all-pairs shortest paths" problem, or to $|V|$ invocations of the standard formulation of the "single-source shortest paths" problem, where there is only one weight function.

Created: 96.03.13