CS60003 Algorithm Design and Analysis, Autumn 2010–11

Mid-Semester Examination

Maximum marks: 50 | September 18, 2010 (AN) | Total time: 2 hours

---

Roll no: _____ **Name:** _____

[*Write your answers in the question paper itself. Be brief and precise. Answer <u>all</u> questions.*]

1. Let S, T_1, T_2 be strings of lengths n, m_1, m_2 with $m_1 + m_2 \leqslant n$. Your task is to locate whether the pattern $T_1 * T_2$ (that is, T_1 followed by zero or more symbols followed by T_2) can be found in S.

(a) Show by means of an example that there can be $\Theta(n^2)$ different matches of $T_1 * T_2$ in S. **(5)**

(b) Supply an $\mathrm{O}(n^2)$-time algorithm to compute all matches of $T_1 * T_2$ in S. **(5)**

(c) Supply an $O(n)$-time algorithm to decide whether there is any match of the pattern $T_1 * T_2$ in S. **(5)**

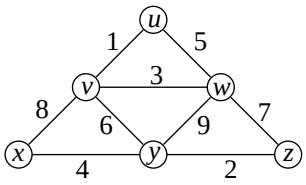
2. Prof. Avarice proposes an algorithm to compute the minimum spanning tree in a connected undirected graph $G = (V, E)$ with a cost $c(e)$ associated with each edge $e \in E$. Your task is to assist Prof. Avarice by supplying an efficient algorithmic implementation of his idea.

(a) Propose an efficient algorithm that, given an edge $e \in E$, determines whether or not e belongs to a cycle in G. What is the running time of your algorithm? **(5)**

(b) The MST algorithm of Prof. Avarice goes as follows.

1. Sort E in non-increasing order of the edge costs. Store this sorted list in T.
2. So long as G contains more than $|V| - 1$ edges, repeat Steps 3–5:
3. Pick the edge e from T with largest cost.
4. If e belongs to a cycle in G, remove e from E.
5. Remove e from T.
6. Output the reduced graph (V, E).

Demonstrate how Prof. Avarice's algorithm works on the following graph. **(5)**



(c) Describe an efficient implementation of Prof. Avarice's algorithm. What is the running time of your implementation? **(5)**

(d) Although Prof. Avarice's algorithm may be poorer than Prim's and Kruskal's MST algorithms in terms of running time, a more potent danger awaits you. Prove or disprove: Prof. Avarice's algorithm always outputs a minimum spanning tree of a connected graph. **(5)**

3. We want to merge n sorted lists $L_1, L_2, \ldots, L_n$ of sizes $l_1, l_2, \ldots, l_n$, respectively. Suppose that at any point of time, we are allowed to merge only two sorted lists, that is, simultaneously merging t sorted lists for $t \geqslant 3$ is not allowed. (For example, the sorted lists may be residing in files in a palmtop computer which has very little memory and allows only three opened file pointers at any time.) The effort associated with merging two sorted lists of sizes u and v is taken as $u + v$. Your task is to select the merging sequence in such a way that the total effort of merging the given n lists is minimized.

(a) Suppose that $n = 4$ lists are given with respective sizes $10, 20, 30, 40$. Find the efforts of the following two ways of merging L_1, L_2, L_3, L_4: **(5)**

$$\text{merge}(\text{merge}(L_1, L_2), \text{merge}(L_3, L_4)) \text{ and } \text{merge}(\text{merge}(\text{merge}(L_1, L_2), L_3), L_4).$$

(b) Describe an $O(n \log n)$-time algorithm to compute a way of merging the input lists with minimum possible effort. The input consists only of the lengths $l_1, l_2, \ldots, l_n$ of the lists. Your algorithm should produce only an optimal merging strategy. It does not have to merge the lists. However, you should supply an optimality proof for your algorithm. **(10)**