UNIVERSITY OF EDINBURGH

FACULTY OF SCIENCE AND ENGINEERING

COMPUTER SCIENCE THREE

ALGORITHMS AND DATA STRUCTURES

Tuesday 12 September 2000

9:30 to 11:00

Examiners:

Prof. K. Turner (External)

Prof. G. Brebner (Chairman)

INSTRUCTIONS TO CANDIDATES

Answer any TWO questions.

All questions carry equal weight.

1. (a) Recall that the procedure Randomised-Partition is used as part of the randomised version of the Quicksort algorithm. Like Partition, Randomised-Partition splits the input array into two parts. Denoting the input for the procedure by the sub-array $A[p \ldots r]$ the procedure rearranges the elements in A and returns an index q such that $A[p \ldots q]$ and $A[q+1 \ldots r]$ are both non-empty and such that each element in $A[p \ldots q]$ is smaller than or equal to each element in $A[q+1 \ldots r]$.

i. Explain in detail how Randomised-Partition achieves the above. If you find it more convenient you may write pseudo-code for it but this is not required; if you do, you should still explain the general idea. [*5 marks*]

ii. Recall the definition of the selection problem: on input array A of size n and index $1 \leq k \leq n$, output i such that $A[i]$ is the k'th smallest element of A. For this problem you may assume that all elements of A are distinct (i.e. there are no repetitions).

Describe a divide and conquer randomised algorithm for the selection problem based on Randomised-Partition with expected running time of $O(n)$. You should describe the algorithm, explain why it is correct and derive a recurrence giving an upper bound on the expected running time (but you do not have to solve the recurrence)

[**Note:** You may find the following property of Randomised-Partition useful. Denote the size of the low part of the output from partition by L and the size of the input array by n so that $L = q-p+1$ and $n = r-p+1$. If all elements in A are distinct then for Randomised-Partition

$$\begin{aligned} Pr[L=1] &= 2/n \\ Pr[L=i] &= 1/n \text{ for } 2 \leq i \leq n-1 \end{aligned}$$

] [*10 marks*]

(b) For n distinct elements $x_1, \ldots, x_n$ with positive weights $w_1, \ldots, w_n$ such that $\sum_{i=1}^{n} w_i = 1$ the weighted median is the element x_k satisfying $\sum_{x_j < x_k} w_j \leq 1/2$ and $\sum_{x_j > x_k} w_j \leq 1/2$. Note that x_k itself is not included in either sum. One can show that if all weights are bounded by $1/2$ then a weighted median exists. You may assume that this is satisfied below.

i. Argue that the median of $x_1, \ldots, x_n$ is the weighted median of the x_i with weights $w_i = 1/n$ for all i. You may assume in this part that n is odd. [*2 marks*]

ii. Show how to compute the weighted median of n elements in $O(n \log n)$ worst case time by using sorting. You should describe an algorithm, explain why it is correct and analyse its running time. [*8 marks*]

2. (a) Briefly explain the operations, $\textsc{Make-Set}(x)$, $\textsc{Find-Set}(x)$, and $\textsc{Union}(x, y)$ for disjoint sets, where x, y denote objects. *[4 marks]*

   (b) Describe an implementation of a data structure for disjoint sets such that any sequence of m operations, of which n are $\textsc{Make-Set}(\cdot)$ operations, takes time $O(m + n \log n)$.

   You should describe the representation and how each operation is performed. You do *not* have to derive the time bound. *[6 marks]*

   (c) This part of the question is concerned with minimum spanning trees on a connected graph $G = (V, E)$. Professor MiSTy has recently been able to establish a new fact about minimum spanning trees:

   Fact 1: Let e be a maximum weight edge on some cycle in G. There is a minimum spanning tree of $G' = (V, E \setminus \{e\})$ that is also a minimum spanning tree of G.

      i. Suggest a new algorithm for finding minimum spanning trees based on Fact 1 and analyse its running time. You may use without proof the fact that DFS can be used to find a cycle on G in time $O(|V| + |E|)$. *[5 marks]*

      ii. Prove Fact 1. (Hint: Try to see what happens if a minimum spanning tree does include e. Could you modify it to get another one that does not include e ?) *[10 marks]*

3. For this question let $G = (V, E)$ be a flow network with capacity function c, source vertex s, and sink vertex t.

   (a) A flow in G is a real valued function $f : V \times V \to R$ which satisfies the tree properties known as *capacity constraint*, *skew symmetry*, and *flow conservation*. Define each of these properties. *[3 marks]*

   (b) Recall that for subsets X, Y of V we defined the implicit summation notation by $f(X, Y) = \sum_{x \in X} \sum_{y \in Y} f(x, y)$. Show that for all X, Y, $f(X, Y) = -f(Y, X)$, and conclude that for all X, $f(X, X) = 0$. *[3 marks]*

   (c) Recall that using the summation notation the flow is defined as $|f| = f(s, V)$. Show that $|f| = f(V, t)$. Make sure to justify steps in your proof.

   [**Note:** You may want to use the properties from the previous part as well as the following one: If $X \cap Y = \emptyset$ then $f(X \cup Y, Z) = f(X, Z) + f(Y, Z)$ and $f(Z, X \cup Y) = f(Z, X) + f(Z, Y)$.] *[5 marks]*

   (d) Suppose that G has integer capacities and we are given an integral maximum flow in G (that is, for each edge e, $f(e)$ is a natural number).

      i. Suppose that the capacity of one edge (u, v) is increased by 1. Give an $O(|V| + |E|)$ algorithm to update the maximum flow. You should describe the algorithm and analyse its running time. *[5 marks]*

      ii. For this part you may assume in addition that G has no cycles. That is, G is a directed acyclic graph (DAG). Suppose that the capacity of one edge (u, v) is decreased by 1. Give an $O(|V| + |E|)$ algorithm to update the maximum flow. You should describe the algorithm and analyse its running time.

      [**Hint:** The restriction to DAG simplifies the problem. Try to patch the flow in G to cope with the decrease in capacity. Your construction may use the fact that G has no cycles.] *[9 marks]*