Midterm Examination on Algorithms

Teacher: Biing-Feng Wang **April 13, 2004**

Remark: For each problem, you must justify your answers.

Problem 1: (5%, Growth of Functions)

Let $f(n)$, $g(n)$, and $h(n)$ be three functions such that $f(n)=\Theta(g(n))$ and $g(n)=\Theta(h(n))$. Prove that $f(n)=\Theta(h(n))$ by using the definition of Θ.

Problem 2: (10%, Recurrences)

Find an upper bound on recurrence $T(n)=T(n-a)+T(a)+n$, where $a\geq 1$ is a constant, by using the substitution method. (You may assume that $T(n)=1$ for $n\leq a$.)

Problem 3: (15%, Sorting) Consider the following algorithm.

```
STOOGE-SORT(A, i, j)
    if A[i] > A[j] then exchange(A[i], A[j])
    if i+1 ≥ j then return
    k ← ⌊(j-i+1)/3⌋
    STOOGE-SORT(A, i, j-k)      /* First two-thirds */
    STOOGE-SORT(A, i+k, j)      /* Last two-thirds */
    STOOGE-SORT(A, i, j-k)      /* First two-thirds */
```

(1) (8%) Argue that STOOGE-SORT(A, 1, n) correctly sorts the n numbers in $A[1..n]$.

(2) (3%) Give a recurrence $T(n)$ for the worst-case running-time of STOOGE-SORT.

(1) (4%) Give an upper bound on $T(n)$. (You can use any method.)

Problem 4: (10%, Sorting) Prove that the worst-case running time for a comparison sort is $\Omega(n\log n)$.

Problem 5: (15%, Binomial Heaps)

(1) (5%) Let $K=\{k_1, k_2, ..., k_n\}$ be a set of n keys. To construct a binomial heap H for K, one may call Make-Heap() first to get an empty heap H and then call Insert(H, k_i) for i=1, 2, ..., n. What is the time complexity of such a construction?

(2) (10%) Can you give a faster algorithm for constructing a binomial heap H for K? What is the time complexity?

Problem 6: (15%, Design of Algorithms) Suppose you are given a set $A=\{a_1, a_2, ..., a_n\}$ of n tasks, where a_i requires t_i units of processing time. You have one computer to run these n tasks, one at a time. Let c_i be the *completion time* of a_i, that is, the time at which a_i completes processing.

(1) (7%) Give an efficient algorithm that schedules the tasks so as to minimize the average completion time $(c_1+c_2+...+c_n)/n$. What is the time complexity?

(2) (8%) Argue that your algorithm in (1) is correct.

Problem 7: (10%, Design of Algorithms)

Give an algorithm that computes the total cost of an optimal triangulation of a given convex polygon $P=(p_0, p_1, ..., p_{n-1})$. The cost of a triangle $\Delta p_i p_j p_k$ is $w(\Delta p_i p_j p_k)$. What is the time complexity?

Problem 8: (10%, Design of Algorithms)

Describe an efficient algorithm that, given a set S of n distinct numbers, a number x, and a positive integer $k \leq n$, determines the k numbers in S that are closest to x. (The k numbers can be outputted in any order.) What is the time complexity?

Problem 9: (10%) This problem is to verify whether you had done homework by yourself. Please answer either of the following two problems. (If you answer both, only the one getting lesser score will be counted.)

(a) Give an algorithm to find the length of the longest monotonically decreasing subsequence of a sequence of n numbers. What is the time complexity?

(b) You are given an array of integers, where different integers may have different numbers of digits, but the total number of digits over all the integers in the array is n. Give an algorithm to sort the array. What is the time complexity?

Bonus: (10%)

(1) (5%) Use an example to explain the *left-child, right-sibling representation*?

(2) (5%) Explain *double hashing*.