國立台灣海洋大學資訊工程學系博士班

97學年度第二學期博士班資格考命題卷（筆試）

科目：演算法
命題教授：林清池老師
日期：2009/01/12

1. Please briefly describe the following standard techniques which are commonly used in algorithm designs. (20%)

   (a) Greedy Method
   (b) Divide and Conquer
   (c) Dynamic Programming
   (d) Randomization

2. Let $f(n)$ and $g(n)$ be real-valued functions. The asymptotic notations, O, Ω, and Θ, are defined as follows.

   - $f(n)$ is *asymptotically upper bounded by* $g(n)$, denoted by $f(n) = O(g(n))$, *iff* there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n)$ for all $n > n_0$.
   - $f(n)$ is *asymptotically lower bounded by* $g(n)$, denoted by $f(n) = \Omega(g(n))$, *iff* there exists positive constants c and n_0 such that $f(n) \geq c \cdot g(n)$ for all $n > n_0$.
   - $f(n)$ is *asymptotically equivalent to* $g(n)$, denoted by $f(n) = \Theta(g(n))$, *iff* there exists positive constants c_1, c_2 and n_0 such that $c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$ whenever $n > n_0$.

   Please answer the following questions according to the definitions given above. (20%)

   (a) Asymptotically upper bound the function $f(n) = n^2 - 2n + 1$ by the O notation. Justify your answer by demonstrating the constants.

   (b) Asymptotically lower bound the function $f(n) = n^2 - 2n + 1$ by the Ω notation. Justify your answer by demonstrating the constants.

   (c) Show that "$f(n) = \Theta(n)$" *if and only if* "$f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$".

   (d) What's wrong with the following argument?

   $$\sum_{1 \leq k \leq n} k \cdot n = \sum_{1 \leq k \leq n} O(n) = n \cdot O(n) = O(n^2)$$

   (In fact, we have $\sum_{1 \leq k \leq n} k \cdot n = n \sum_{1 \leq k \leq n} k = n \cdot \frac{n}{2}(n+1) = O(n^3)$.)

3. Given a sequence $A = \{a_1, a_2, \ldots, a_n\}$, the *longest increasing subsequence problem* is to find an increasing subsequence of A with the longest length. (12%)

   (a) Find a longest increasing subsequence of $\{3, 7, 5, 9, 2, 6, 4\}$.

   (b) Describe an algorithm that solves the longest increasing subsequence problem in $O(n^2)$ time.

4. Please describe briefly the following sorting algorithms along with their time complexities. (12%)

   (a) Insertion sort

   (b) Quick sort

5. Let $A = \{a_1, a_2, \ldots, a_n\}$ be an array of n numbers. The *Stooge sort* algorithm proposed by Professors Howard et al., is as follows.

   $\textsc{Stooge-sort}(A,\ i,\ j)$

   - **If** $i + 20 > j$, **then** perform insertion sort on $A(i \ldots j)$ **and return**
   - $k \longleftarrow \lfloor (j - i + 1)/3 \rfloor$
   - $\textsc{Stooge-sort}(A,\ i,\ j - k)$ // stooge-sort on the first two-thirds
   - $\textsc{Stooge-sort}(A,\ i + k,\ j)$ // stooge-sort on the last two-thirds
   - $\textsc{Stooge-sort}(A,\ i,\ j - k)$ // stooge-sort on the first two-thirds again

   Based on the algorithm given above, please answer the following questions. (16%)

   (a) Argue that $\textsc{Stooge-sort}(A,\ 1,\ n)$ sorts the array A correctly.

   (b) Analyze the time complexity of $\textsc{Stooge-sort}$. Also compare it with the time complexities of insertion sort and quick sort. Do you think $\textsc{Stooge-sort}$ is good enough for practical usage?

6. Let $G = (V, E)$ be an undirected graph, n be the number of vertices, and $v_0 \in V$ be a vertex. (20%)

   (a) Please give an algorithm that solves the single-source shortest distance problem. That is, an algorithm that computes the distances from a given vertex, say v_0, to all other vertices.

   (b) Briefly analyze the correctness and time complexity of your algorithm.

   (c) When the given graph G is restricted to a tree, describe an algorithm that solves the single-source shortest distance problem from v_0 to all other vertices in $O(n)$ time.

   (d) When the given graph G is restricted to a tree, describe an algorithm that supports constant time queries to the distance between any pair of vertices in G by performing an $O(n)$ time pre-processing. You may use any well-developed algorithms as your subroutines to this problem.