國立臺灣大學97學年度碩士班招生考試試題

1. (10 points) Complete the following table of running time $T(n)$ by specifying the letter for recurrence and the numeral for solution to that recurrence. Note that points will be deducted for wrong answers.

index	algorithm	recurrence	solution
(1)	merge sort (worst case)		
(2)	randomized quicksort (average case)		
(3)	randomized quicksort (worst case)		
(4)	selection (worst case)		
(5)	binary search (worst case)		

letter	recurrence	numeral	solution
a	$T(n) = 7T(n/2) + \Theta(n^2)$	1	$T(n) = \Theta(n)$
b	$T(n) = T(n/2) + \Theta(1)$	2	$T(n) = \Theta(\lg n)$
c	$T(n) = T(\lceil n/5 \rceil) + T(7n/10 + 6) + \Theta(n)$	3	$T(n) = \Theta(n^{\lg 7})$
d	$T(n) = 2T(n/2) + \Theta(n)$	4	$T(n) = \Theta(1)$
e	$T(n) = \frac{2}{n}\sum_{k=\lfloor n/2 \rfloor}^{n} T(k) + \Theta(n)$	5	$T(n) = \Theta(n^2)$
f	$T(n) = 2T(n/2) + \Theta(1)$	6	$T(n) = \Theta(n \lg n)$
g	$T(n) = \frac{2}{n}\sum_{k=0}^{n-1} T(k) + \Theta(n)$	7	$T(n) = \Theta(\sqrt{n})$
h	$T(n) = T(n-1) + \Theta(n)$	8	$T(n) = \Theta(n^3)$

2. (10 points) Consider inserting the keys 10, 22, 31, 4, 15, 28, 17, 88, 59 into a hash table of length $m = 11$ using open addressing with the primary hash function $h'(k) = k \bmod m$. Illustrate the result of inserting these keys using (a) linear probing, and (b) double hashing with $h_2(k) = 1 + (k \bmod (m-1))$.

3. (15 points) Consider the following scheduling problem for unit time tasks on a uniprocessor:

task i	1	2	3	4	5	6	7	8	9	10	11	12
deadline d_i	3	1	4	1	5	9	2	6	5	3	5	8
penalty w_i	15	12	10	10	10	10	9	7	6	4	3	1

Give an optimal schedule for these tasks, that minimizes the total penalty of those tasks that do not complete by their deadlines. Assume that the first scheduled task begins at time 0 and ends at time 1. What's the total penalty for your schedule?

4. (15 points) Naïve matrix multiplication of two $n \times n$ matrices, A and B, runs in $\Theta(n^3)$ time. Strassen's algorithm is a divide-and-conquer approach for matrix multiplication. By partitioning A and B into four $n/2 \times n/2$ matrices respectively, instead of 8 multiplications, Strassen showed a method that only requires 7 matrix multiplications and additional $\Theta(n^2)$ scalar operations to compute $A \times B$.

(a) (5 points) Show that Strassen's algorithm runs in $\Theta(n^{\lg 7})$ time.

(b) (5 points) Dr. Q proposes a method, called Algorithm Q, that can multiply two $c \times c$ matrices with m_c multiplications and $\Theta(n^2)$ scalar operations for a constant c. (For Strassen's algorithm, $c = 2$ and $m_2 = 7$.) Determine the asymptotic complexity of Algorithm Q for multiplying two $n \times n$ matrices in terms of n, c and m_c. You may assume that $m_c > c^2$.

見背面

(c) (5 points) When $m_3 < \alpha$, Algorithm Q is more efficient than Strassen's algorithm. What is α? (It is sufficient to express α in terms of logarithms. No need to calculate it out.)

5. (20 points) Let the input set X consist of n points on the 2-dimensional plane with integral coordinates.

   (a) The *nearest pair problem* is to identify two points in X whose Euclidean distance is minimum over all pairs in X. You are asked to prove or disprove that the nearest pair problem can be solved in $O(n \log n)$ time.

   (b) The *farthest pair problem* is to identify two points in X whose Euclidean distance is maximum over all pairs X. You are also asked to prove or disprove that the farthest pair problem can be solved in $O(n \log n)$ time.

6. (15 points) Consider a simple graph G with n nodes and m edges. A node subset S of G is a *node cover* of G if all edges of G are incident to at least one node of S. The problem of computing a minimum node cover for G is a well-known NP-complete problem. Now, you are asked to give an $O(n+m)$-time algorithm that outputs a node cover of G whose size is at most 2 times that of any node cover of G. (By the way, we usually call algorithms of this kind a 2-approximation). Besides giving your algorithm, you also have to prove that your algorithm indeed runs in $O(m+n)$ time and is a 2-approximation.

7. (15 points) Given two length-n binary strings A and B, consider the problem of computing a longest string C that is a substring of both A and B. You are asked to prove or disprove that this so-called *longest common substring problem* can be solved in $O(nlogn)$ time.

試題隨卷繳回