國立交通大學試題紙

九十七學年度第一次
博士班資格考

科目：演算法(A)　　　　日期：98 年 2 月 6 日　第 1 頁 共 1 頁

請"✓"明　✓不可看書　可看書

* 請將答案依題號順序寫入答案卷

答題時字跡需工整，否則不予計分。Write your answers legibly; otherwise you will get zero score.

1. (12%) Briefly describe Huffman's algorithm. Then, what is an optimal Huffman code for the following set of frequencies, based on the first 8 Fibonacci numbers?

   a:1 b:1 c:2 d:3 e:5 f:8 g:13 h:21

   Can you generalize your answer to find the optimal code when the frequencies are the first *n* Fibonacci numbers?
2. (12%) Describe an efficient dynamic programming algorithm for finding an optimal parenthesization of a matrix-chain product. Illustrate your algorithm by using an example of matrix-chain product whose sequence of dimensions is <5, 10, 3, 12, 5, 50, 6>.
3. (12%) In a department, there are *n* courses, *n* lecturers and *n* TAs. The basic course arrangement rules of the department are that each course needs exactly one lecture and one TA, and that each lecturer and TA has at most one class in a semester. Each lecturer has a list of courses that he/she can teach; and each TA also has a list of courses that he/she can serve as TA. Devise an algorithm to arrange the maximum number of courses and analyze its time complexity. Hint: reduce yours to another algorithm.
4. (14%) (a) Consider the *k*-clique problem: Given a positive integer *k* and a graph $G=(V, E)$, determine whether there exists some clique of size *k* in *G*. Prove that the *k*-clique problem is NP-complete. Hint: you can assume that SAT has been proved to be NP-complete. (b) Consider the max-clique problem: Given a graph *G*, find a maximum clique of *G*. Prove that the max-clique problem is NP-hard. Hint: use the result at (a).

國立交通大學試題紙

科目：演算法(B) 日期：98 年 2 月 6 日 第 1 頁 共 2 頁

請"✓"明 ✓不可看書 可看書

＊ 請將答案依題號順序寫入答案卷

答題時字跡需工整，否則不予計分。Write your answers legibly; otherwise you will get zero score.

1. Show that any comparison-based sorting algorithm that sorts n records needs at least $\Omega(n \log n)$ operations. *5%*
2. Suppose that a linear decision tree T can solve a computational problem A (A is not sorting problem). If we show that T can also solve the sorting problem. Is that true the minimum height of the T implies the lower bound of the sorting problem? Give me your reasons. *10%*
3. If we maintain a balance search tree, T, and we insert records into T. If T is out of balance after an insertion, we can rotate the tree to rebalance the tree. The cost of a single rotation could be high, but we know that for a sequence of n insertions, the amortized cost for a single rotation is constant. Suppose that we have a sequence of n insertions, then we inorder traverse the tree to produce a linear order of the n records. Note that the inorder traversal can be done in linear time and the amortized cost is constant for a single rotation, and there are at most n such rotations. So we sort n records in linear time. I beat the lower bound. Anything went wrong? Give me your reasons. *10%*
4. Show that building a min-heap can be done in linear time. *5%*
5. Union-Find operation implemented using forest representation. Suppose that we have a sequence of m Finds and n-1 Unions. Show that if only the heuristic "union by rank" is applied, the sequence of operations can be done in $O(m \lg n)$ time. And argue that this bound is tight (there are cases that this bound can be achieved). *Union by rank heuristic: To make a Union, the parent of the root of the tree with smaller rank points to the root of the tree with larger rank. If you don't feel comfortable with the "rank", you may use the "size of the tree" to replace the rank. 15%*

6. Delete the root from the following binary search tree. Draw the resulted tree. *5%*