

科目：演算法(A)

日期：97年7月25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. 15%

(a) Consider the ordered searching problem. (See below.)

For any comparison searching algorithm, the number of comparisons has a worst case lower bound.

Write the (best) lower bound, and prove your answer.

Is there any algorithm achieve this lower bound? 5%

The ordered searching problem: Given a sorted sequence, $a_1 < a_2 < a_3 < \dots < a_{n-1} < a_n$,and given an x , find an a_i such that $x = a_i$.

(b) For any comparison based sorting algorithm, the number of comparisons has a worst case lower bound.

Write the (best) lower bound without proof. 1%

(c) Consider the problem of sorting n numbers.Suppose that, among these n input numbers, there are only $O(\log n)$ distinct ones.Show that there is a comparison based sorting algorithm to sort these n numbers in $O(n \log \log n)$ time. 7%

(d) Compare the results in part (b) and (c), and explain why. 2%

2. 15%

(a) Use Dynamic programming technique to find an optimal solution for the 0-1 knapsack problem

(See the definition below. Assume that all the data, v_i , w_i and W , are positive integers.)

(b) Is the algorithm a polynomial time algorithm? Explain briefly.

(Define the object function, write the recursive relation, give the initial condition,

illustrate the table, indicate where the answer is,

and estimate the time complexity and space complexity.)

0-1 knapsack problem :

There are n items, the i th item is worth v_i dollars and weights w_i pounds for $i=1$ to n .A person wants to take as much valuable a load as possible but he can carry at most W pounds in his knapsack.

Each item must either be taken or left behind. He cannot take a fraction of an item or take an item more than 0.

Assume that all the data, v_i , w_i and W , are positive integers.

3. 8%

Consider the amortized analysis of an algorithm.

Explain the basic idea of the potential method. Write some mathematical expressions to explain it.

(You may use those special terms used in this method.)

Explain how one can find a worst case upper bound of the time complexity using the potential method.)

國立交通大學試題紙

九十六學年度第二次
博士班資格考

科目：演算法(A)

日期：97年7月25日 第2頁共2頁

4. 12%

Given the basic knowledge of the text book, answer and prove the following problems.

- (a) Is the longest-simple-cycle problem in P or NP-complete (NP-hard)? 1%
- (b) Prove briefly your answer to part (a). 5%
- (c) Is the shortest-simple-cycle problem in P or NP-complete (NP-hard)? 1%
- (d) Prove briefly your answer to part (c). 5%

The longest-simple-cycle problem:

Determine a simple cycle (no repeated vertices) of maximum length in a graph.

The shortest-simple-cycle problem:

Determine a simple cycle (no repeated vertices) of minimum length in a graph.

科目：演算法(B)

日期：97年7月25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%) Let $T(n) = \Theta(f(n))$. Assume that $T(n)$ is constant for sufficiently small n . Derive $f(n)$ in the simplest formula for each of the following $T(n)$.

$$T(n) = 2T(n/2) + n \log n.$$

$$T(n) = 2T(n/2) + n.$$

$$T(n) = 2T(n/2) + n / \log n.$$

$$T(n) = T(n/2) + T(n/3) + T(n/6) + n.$$

$$T(n) = T(3n/4) + T(n/5) + n.$$

$$T(n) = n^{1/3}T(n^{2/3}) + n.$$

2.(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?

3.(16%) Describe an $O(|V|^2 \log |V| + |V| |E|)$ -time algorithm to solve the all-pairs shortest-paths problem for weighted directed graphs (V, E) , where negative-weight cycles are allowed. Hint: use the Bellman-Ford algorithm, Dijkstra's algorithm, and the reweighting technique.

4.(10%) For the maximum flow problem, describe an $O(|V| |E|^2)$ -time algorithm, where $|V|$ is the number of vertices and $|E|$ is the number of edges in the graph (V, E) . Note: need to derive the time complexity.