國立交通大學試題紙

九十九學年度第一次
博士班資格考

科目：演算法(A)　　日期：100 年 1 月 27 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. 17%　Interval graphs

Given a set of n time intervals $\{[s_i, f_i) \mid i = 1$ to n}, we construct a graph, called interval graph, as follows:

Corresponding to each time interval $[s_i, f_i)$, there is a node v_i, $i = 1$ to n.

There is an edge between nodes v_i and v_j if and only if the corresponding intervals $[s_i, f_i)$ and $[s_j, f_j)$ overlap, i.e. $[s_i, f_i) \cap [s_j, f_j) \neq \phi$.

(1). Propose a polynomial time greedy algorithm to find a maximum independent set in this interval graph. (Describe your algorithm briefly, and state the time complexity.)

(2)Consider the vertex cover problem in interval graphs. Is it possible to have a polynomial time algorithm to find a minimum vertex cover in the above interval graph? If yes, describe the algorithm. If not, prove it.

2. 16%　Consider the maximum-flow problem, and max-flow min-cut theorem.

(1) Prove:　the value of a maximum flow≤ the capacity of a minimum cut.

(2) Prove that, in fact,

the value of a maximum flow= the capacity of a minimum cut

You have to write some mathematical expression to support your proof.

You may use special terms and notations, or draw some simple graphs to illustrate your proof idea.

國立交通大學試題紙

九十九學年度第一次
博士班資格考

科目：演算法(A)　　日期： 100 年 1 月 27 日

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.

3. Assume the basic knowledge of NP-completeness: satisfiability , hamiltonian-cycle, vertex-cover, subset-sum problems are NP-complete. And assume $P \neq NP$

(1)4% Determine whether each of the following two problems (A) and (B) is in P or is NP-hard

(2) 13% If it is in P, describe the algorithm briefly, and state the time complexity. If it is NP-hard, prove your answer.

(A) Find ρ-approximation solution for the vertex-cover problem, for some $\rho > 1$.

(B) Find ρ-approximation solution for the traveling-salesman problem, for some $\rho > 1$.

國立交通大學試題紙

九十九學年度第一次
博士班資格考

科目：演算法(B)　　日期：100年1月27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. (16%)
   (a) $T(n) = 4\,T(n/2) + n^2$
   (b) $T(n) = 4\,T(n/2) + n^2 \log n$
   (c) $T(n) = 4\,T(n/2) + n^2 / \log^2 n$
   (d) $T(n) = 4\,T(n/2) + n \log n$
   (e) $T(n) = 1\times n + 2\times(n-1) + \ldots + (n-1)\times 2 + n\times 1$
   (f) $T(n) = 1\times k^2 + 2\times(k-1)^2 + 4\times(k-2)^2 + \ldots + (n/2)\times 1^2$, if $n=2^k$
   (g) $T(n) = n/n + n/(n+1) + n/(n+2) + \ldots + n/n^2$
   (h) $T(n) = \log 1 + \log 2 + \log 3 + \ldots + \log n + \log (n+1) \ldots + \log n^2$

   For each of the above $T(n)$, derive the corresponding $f(n)$ with $T(n) = \Theta(f(n))$. Hint: $f(n)$ must be one of the following: $\Theta(n)$, $\Theta(n \log n)$, $\Theta(n \log^2 n)$, $\Theta(n^2)$, $\Theta(n^2 \log n)$, $\Theta(n^2 \log^2 n)$, $\Theta(n^3)$, $\Theta(n^3 \log n)$ and $\Theta(n^3 \log^2 n)$.
2. (12%) (a) Given n integers $(a_1, a_2, \ldots, a_n)$ ranging from 1 to n^2, design a $O(n)$ algorithm to sort the n items and show its time complexity.
   (b) Prove that any comparison sort algorithm requires $\Omega(n \log n)$ comparisons in the worst case. The proof seems contradictory to (a). Explain the reason.
3. (12%) Design data structures, called Vector, which require the following operations (no deletion for simplicity)
   (1) `find(key)`: returns the data with `key`;
   (2) `insert(key, data)`: insert `data` with `key` into the Vector.

   Design the data structure such that
   (a) the time complexity for each `find` operation is O(1),
   (b) the amortized time complexity for each `insert` operation is O(1), and
   (c) the space complexity is always O(n) if there are n characters in the string buffer.

   And explain why (a), (b), and (c) are satisfied in your data structure.
4. (10%) In bioinformatics, there is a sequence comparison problem, defined as follows. Assume to score +1 for a match, -1 for a mismatch, and –3 for an insertion or deletion; then, derive the matching with the highest score. Briefly describe the algorithm and illustrate your algorithm by the following two sequences, CGATCGGCAT and CAATGTGAATC.