TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS — 2007

MID-YEAR

COMP 303
Design and Analysis of Algorithms

Time Allowed: 3 Hours

Instructions: There are 5 questions of varying weights.
Available marks total 180, so allow about one minute per mark.
Answer all questions.

Calculators and other electronic devices are **not** permitted.
Printed foreign language dictionaries **are** permitted.

The following definitions are provided for your convenience. You may find it useful to tear off this front page of the paper.

Asymptotic notation:

$$\begin{aligned} O(g(n)) &= \{f(n) \mid (\exists c)(\mathbf{aa}\, n)[0 \leq f(n) \leq c.g(n)]\} \\ \Omega(g(n)) &= \{f(n) \mid (\exists c)(\mathbf{aa}\, n)[f(n) \geq c.g(n) \geq 0]\} \\ \Theta(g(n)) &= \{f(n) \mid (\exists c, d)(\mathbf{aa}\, n)[0 \leq c.g(n) \leq f(n) \leq d.g(n)]\} \end{aligned}$$

Master Theorem: Let $T(n)$ be defined by the recurrence $T(n) = aT(n/b) + f(n)$.
Let $\alpha = \log_b a$.

1. If $(\exists \epsilon > 0)[f(n) \in O(n^{\alpha-\epsilon})]$ then $T(n) \in \Theta(n^{\alpha})$.
2. If $f(n) \in \Theta(n^{\alpha})$ then $T(n) \in \Theta(n^{\alpha} \log n)$.
3. If $(\exists \epsilon > 0)[f(n) \in \Omega(n^{\alpha+\epsilon})]$ and $(\exists c < 1)(\mathbf{aa}\, n)[a.f(n/b) \leq c.f(n)]$ then $T(n) \in \Theta(f(n))$.

Logarithms:

$$\log_a x = y \text{ if and only if } a^y = x$$

THIS PAGE INTENTIONALLY LEFT BLANK

Question 1. [36 marks]

(a) By using O, Ω, or Θ as appropriate, give the best possible asymptotic bound for:

(i) the time complexity of the linear search algorithm on an unsorted list

(ii) the time complexity of the linear search algorithm on a sorted list

(iii) the time complexity of the binary search algorithm on a sorted list

(iv) the time complexity of the search problem on an unsorted list

(v) the time complexity of the search problem on a sorted list

Justify each of your answers. [10 marks]

(b) Using the definitions for O, Ω, and Θ given on the front page of this paper, show that:

(i) $n^2 + n \in O(n^3)$

(ii) $\frac{1}{2}n(n+1) \in \Theta(n^2)$

(iii) if $f(n) \in O(g(n))$ then $g(n) \in \Omega(f(n))$ [12 marks]

(c) You are given two problems A and B, and told that A is NP-complete. How would you:

(i) show that B is NP-Hard?

(ii) show that B is NP-Complete? [6 marks]

(d) For each of the following recurrence relations, give the asymptotic complexity (Θ bound) of $T(n)$. Justify your answers using the Master Theorem (given on page 1 of the paper).

(i) $T(n) = \begin{cases} 1, & \text{if } n = 0 \\ 4T(\lceil \frac{n}{2} \rceil) + n^2, & \text{otherwise} \end{cases}$

(ii) $T(n) = \begin{cases} 1, & \text{if } n = 0 \\ 8T(\lceil \frac{n}{2} \rceil) + n^2 \log n, & \text{otherwise} \end{cases}$ [8 marks]

Question 2. [40 marks]

(a) Write pseudocode to define the basic structure of a typical divide-and-conquer algorithm. Explain the components of your algorithm. [6 marks]

(b) Give a simple divide and conquer algorithm for finding values of both the largest and smallest elements in an array of n numbers. [12 marks]

(c) Give the general structure of the proof of correctness of a divide and conquer algorithm, and use it to show your algorithm from part (b) is correct. [9 marks]

(d) Give a recurrence relation for the number of key comparisons performed by your algorithm. [4 marks]

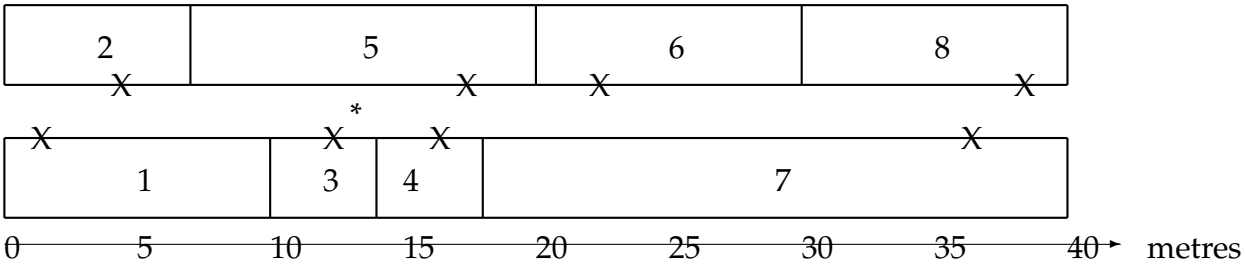
(e) Give a Θ bound on your recurrence relation from part (d). [3 marks]

(f) Briefly describe two other algorithms for the same problem, comparing their asymptotic efficiencies with the one you calculated in part (e). [6 marks]

Question 3. [30 marks]

(a) You are a programmer working for a firm that specializes in the installation of fire extinguishers in office buildings. Regulations require that the maximum distance from any office door to the nearest fire extinguisher is ten metres; but because fire extingishers are expensive, the company wishes to minimize the number used in any given corridor. You may assume that offices are laid out along straight corridors, so that for the purposes of this problem, a layout may be described as an ordered sequence of points on a number line.

For example, the diagram below shows eight offices (numbered 1–8); each door is marked with X. The layout may be described by the sequence $\langle 1, 4, 12, 16, 17, 22, 36, 38 \rangle$.



An extinguisher at position 13 (marked * on the diagram) could service offices 2, 3, 4, 5, and 6, but no others (we assume the width of the corridor is negligible).

(i) Show that the problem has the **optimal substructure** property. Be sure to define carefully what it means for one problem instance to be a subproblem of another.

(ii) Design a **greedy** algorithm that will return positions for the minimum number of extinguishers that can service all the offices on a corridor.

(iii) Outline an argument that your algorithm is correct. [18 marks]

(b) Below are two algorithms for finding a minimal spanning tree $G' = (V', E')$ of a graph $G = (V, E)$.

Prim(V, E) returns (V', E')
  $V' \leftarrow \{\}$
  $E' \leftarrow \{\}$
  while $V' \neq V$
    $e \leftarrow$ shortest edge $x \rightarrow y$
      such that $x \in V'$ but $y \notin V'$
    $E' \leftarrow E' \cup \{e\}$
    $V' \leftarrow V' \cup \{y\}$

Kruskal(V, E) returns (V', E')
  $V' \leftarrow V$
  $E' \leftarrow \{\}$
  while $\#E' < \#V - 1$
    $e \leftarrow$ shortest edge $x \rightarrow y$ in E
    $E \leftarrow E - \{e\}$
    if no path from x to y in E'
      $E' \leftarrow E' \cup \{e\}$

(i) Briefly explain how Kruskal's algorithm may be efficiently implemented using a **Union-Find** data structure to manage disjoint, non-empty sets of nodes.

(ii) Let $n = \#V$ be the number of nodes, and $a = \#E$ be the number of edges, in the graph G. Give asymptotic running times for the two algorithms: justify your answer. [12 marks]

Question 4. [48 marks]

The senior partner in a large firm of criminal lawyers is trying to plan her case-load for the next week. As the senior partner, she gets to choose any cases she wants from the large number available: she will choose the cases that maximize her income while not taking longer in total than the number of hours she wishes to work.

Our experienced lawyer has developed several specialties over the years, and she knows exactly how long those cases will take and what they will earn. She knows, for example, that defending a drug dealer takes 5 hours and earns $1000; defending a jewel thief takes 12 hours and earns $1800; while defending a poor student who has been arrested for busking without a licence takes 2 hours and earns $200. The lawyer does not want to work more than 24 hours in the week (she needs two afternoons free for golf). Currently awaiting trial in the police cells are 3 drug dealers, 4 jewel thieves, and 6 buskers.

The above information is summarized in the following table, in which the cases are numbered $i = 1, 2, 3$; $t[i]$ gives the time per case i; $e[i]$ gives the earnings per case i; $k[i]$ gives the number of cases i available; and T is the total number of hours available:

Case	i	$t[i]$ (hours)	$e[i]$ (dollars)	$k[i]$	$e[i]/t[i]$
Drug dealers	1	5	1000	3	200
Jewel thieves	2	12	1800	4	150
Buskers	3	2	200	6	100

$T = 24$

Our lawyer (who did not study COMP303) thinks she can choose the optimal case-load using a greedy algorithm:

> **Greedy:** Look at each case in decreasing order of earnings per hour, taking the case only if it can be completed in the available time.

(a) Define **acceptable solution** and **correct solution** for the general problem: that is, assuming that the values n and T and the arrays t, e, and k are inputs. [4 marks]

(b) Show that the greedy algorithm does **not** give the optimal solution for this problem instance. [4 marks]

(c) Show that the general problem has the **optimal substructure** property. [8 marks]

(d) Describe an appropriate **dynamic programming** algorithm for solving the general problem. [8 marks]

(e) Briefly outline a proof that your algorithm is correct. [4 marks]

(f) State the asymptotic complexity of your algorithm. **Justify your answer.** [4 marks]

(g) Another approach to the lawyer's problem is to use a **backtracking search**. Describe a backtracking search algorithm for the general problem. You may find it useful to illustrate your algorithm using the example problem instance above. [8 marks]

(h) Describe a related problem that gives an upper bound on the value of a solution to the lawyer's problem, and explain how solutions to the related problem may be used to improve the efficiency of the backtracking search. [8 marks]

Question 5. [26 marks]

In the travelling salesman problem, we are given a complete edge-weighted undirected graph $G = (V, E)$ that associates with each edge $(u, v, c) \in E$ a positive integer cost c. We must find a Hamiltonian cycle of G that has minimum cost.

(a) Describe the problem formally: state signature, precondition, acceptability criterion, and objective function. Make sure your answer carefully defines what it means for a graph to be complete, the constraint on the edge weights, and what it means for the answer to be a minimum-cost Hamiltonian cycle. [9 marks]

(b) How do you know there is a solution for every instance of this problem? [2 marks]

(c) Describe a polynomial-time algorithm that will find a Hamiltonian cycle (not necessarily a minimum-cost cycle) in the graph. [3 marks]

(d) Describe a backtracking search algorithm that will solve the travelling salesman problem. [4 marks]

(e) Now, suppose that in addition to the assumptions above, we know that G is Euclidean.

(i) Amend your precondition from (a) to formally define this new information.

(ii) Give an efficient algorithm that uses approximation to find a reasonable (though not necessarily optimal) near-solution for the amended problem.

(iii) Show that the cost of the cycle found by your algorithm in (i) is no greater than twice the cost of the optimal cycle. [8 marks]

****