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

EXAMINATIONS — 2005

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:

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

continued...

THIS PAGE INTENTIONALLY LEFT BLANK

Question 1. [36 marks]

(a) [9 marks] Explain the definitions of O, Ω and Θ given on page 1 of this paper. State under what circumstances each of the three operators is used.

(b) [5 marks] Explain how the **principle of invariance** justifies our use of asymptotic notation in the analysis of algorithms.

(c) [6 marks] Using the definitions above, show that:

$$\Theta(\log_2 n) = \Theta(\log_4 n)$$

(d) [16 marks] For each of the following recurrence relations, give the asymptotic complexity (Θ bound) of $T(n)$. Justify your answers using either induction or the Master Theorem (given on page 1 of the paper), as appropriate.

(i) $T(n) = \begin{cases} 1, & \text{if } n = 0 \\ T(n-1) + n, & \text{otherwise} \end{cases}$

(ii) $T(n) = \begin{cases} 1, & \text{if } n = 0 \\ 8T(\lceil \frac{n}{2} \rceil) + n^3, & \text{otherwise} \end{cases}$

(iii) $T(n) = \begin{cases} 1, & \text{if } n = 0 \\ 9T(\lceil \frac{n}{3} \rceil) + n^3, & \text{otherwise} \end{cases}$

Question 2. [32 marks]

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

(b) [4 marks] Divide-and-conquer algorithms deal with subproblems. What *two* properties must these subproblems have for a divide-and-conquer algorithm to be efficient?

(c) [22 marks] Assume you are given two sorted lists of length n:

$$\begin{aligned} u &= \langle u_1, u_2, \ldots, u_n \rangle \\ v &= \langle v_1, v_2, \ldots, v_n \rangle \end{aligned}$$

(i) Devise an efficient divide and conquer algorithm for finding the median of the combined lists: that is, to find the nth largest of the $2n$ elements. Do **not** merge the two lists.

Hint The median of a single sorted list can be determined in constant time. Let u_m be the median of u, and v_m be the median of v. What can you say about the median of the combined lists if $u_m = v_m$? if $u_m < v_m$? if $u_m > v_m$?

(ii) Outline an argument that your algorithm is correct.

Question 3. [30 marks]

During a typical day, a student must schedule many activities (lectures, study, assignments, meals, sleeping, recreation). Most days, it isn't possible to complete all the activities planned for the day.

Suppose on a particular day you have n activities, each with a fixed starting time s_i and finishing time f_i. In general, the times for some activities will overlap, so it won't be possible to schedule all activities, but you want to do as many of them as possible.

The following algorithm is intended to solve this scheduling problem, returning the largest set M of activities that can be completed.

```
schedule($s, f$)
  $M \leftarrow \{1\}$
  $j \leftarrow 1$
  for $i \leftarrow 2$ to $n$
    if $s_i \geq f_j$ then
      $M \leftarrow M \cup \{i\}$
      $j \leftarrow i$
```

(a) [6 marks] Which of the algorithm design techniques discussed in the course does this algorithm use? Explain why.

(b) [6 marks] What assumption about the input is required for this algorithm to be correct?

(c) [12 marks] Sketch a proof of correctness for this algorithm. You do not need to complete the proof in detail, but should indicate how each correctness criterion for this class of algorithms apply to this problem.

(d) [6 marks] What is the asymptotic complexity for this algorithm? State any assumptions you make about the elementary operations.

Question 4. [40 marks]

You are working for Transit, the organization charged with providing funding for New Zealand's transport systems. Naturally, you want to allocate your limited budget (4 billion dollars) to the regions in a way that maximises the benefit to all New Zealanders. Of course, that benefit is rather hard to quantify.

This being election year, the Government has made your job rather easier by "requesting" that you use a rather simple measure of benefit: the number of votes to be won in any region.

The table below summarizes the costs of the first-priority roading project for each of the four regions Auckland (1), Waikato (2), Wellington (3), and Canterbury (4). Like the politicians, you may assume that nobody else matters. For each region $i \in 1..4$, the table indicates the cost $c[i]$ of the project (in millions of dollars); the number of votes $v[i]$ to be won in the region (in thousands); and, to save you the trouble of calculation, the ratio of votes to dollars (in votes per million dollars). Also shown is the total money available, M.

i	Region	$c[i]$	$v[i]$	$v[i]/c[i]$
		(\$mill)	(000)	(votes per \$mill)
1	Auckland	2000	800	400
2	Waikato	800	200	250
3	Wellinton	1000	350	350
4	Canterbury	1200	360	300

$M = 4000$ (in millions of dollars).

The problem is to determine the allocation of funding, given the data from a table such as the above. Assume the inputs for the problem are n (the number of regions), $c[1..n]$, $v[1..n]$, and M. Assume the output is an array $f[1..n]$ of integer values 0 or 1 ($f[i] = 1$ if region i should be funded, and $f[i] = 0$ otherwise).

(a) [6 marks] Define **acceptable solution** and **correct solution** for the problem.

(b) [4 marks] You initially try to solve the problem using a greedy algorithm: look at each region in decreasing order of votes per million dollars, funding it only if it can be done with the available money.

Show that the greedy algorithm does **not** give the optimal solution for the problem instance in the table above.

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

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

(e) [8 marks] Outline an algorithm for solving the problem by **backtracking search**. You may find it useful to explain your algorithm using the problem instance in the table above.

(f) [6 marks] Describe how one may **prune** and **bound** the search tree to make backtracking search more efficient.

Question 5. [42 marks]

(a) [15 marks] Consider the problem of searching a sequence for a specified value.

Inputs:

- an **unordered** sequence $s = \langle s_1, s_2, \ldots, s_n \rangle$ of integers, of length n;
- an integer x

Output:

- some index i such that $s_i = x$, if there is one; otherwise zero.

The problem may be solved using a **linear search**.

(i) Define a linear search algorithm to satisfy the specification above.

(ii) Give an exact asymptotic (Θ) bound for the **best case** complexity of linear search, and explain how the best case arises.

(iii) Give an exact asymptotic (Θ) bound for the **worst case** complexity of linear search, and explain why there can be no algorithm that solves this problem in asymptotically better worst-case time.

(b) [5 marks] Explain why *polynomial reductions* are of interest in analysing the complexity of problems.

(c) [8 marks] Describe, in informal terms, the complexity classes $\mathcal{P}$, $\mathcal{NP}$, $\mathcal{NP}$-**Complete**, $\mathcal{NP}$-**Hard**.

(d) [14 marks] For each of the following properties, state whether the property is known to be true; known to be false; unknown but probably true; unknown but probably false. Justify each answer.

(i) $O(n^2) \subseteq O(n^3)$
(ii) $\Theta(n^2) \subseteq \Theta(n^3)$
(iii) $O(n^2) \subseteq \mathcal{P}$
(iv) $\Omega(n^2) \subseteq \mathcal{P}$
(v) $\mathcal{P} \subseteq \mathcal{NP}$
(vi) $\mathcal{NP}$-**Complete** $\subset \mathcal{NP}$
(vii) $\mathcal{NP} \subseteq \mathcal{NP}$-**Hard**

****