TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS – 2017

TRIMESTER 2

COMP 361

DESIGN AND ANALYSIS OF ALGORITHMS

Time Allowed: TWO HOURS

CLOSED BOOK

Permitted materials: Only silent non-programmable calculators or silent programmable calculators with their memories cleared are permitted in this examination.

Non-electronic foreign language to English dictionaries are permitted.

Instructions: The exam will be marked out of 120 marks.

Attempt all questions.

Questions	**Marks**
1. Definitions	[20]
2. Greedy Algorithms	[30]
3. Dynamic Programming	[35]
4. Approximation and Probabilistic	[20]
5. Guest Lectures	[15]

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

Asymptotic notation:

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

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:

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

Matrix Multiplication: When multiplying $m \times n$ matrix A by $n \times p$ matrix B one gets an $m \times p$ matrix C where the *row by column* strategy is followed: element $c_{i,j} = a_{i,1} \times b_{1,j} + a_{i,2} \times b_{2,j} + \ldots + a_{i,n} \times b_{n,j}$.

1. Definitions **(20 marks)**

   (a) **(10 marks)** State the template for a typical Divide and Conquer algorithm and then state a typical proof structure for the correctness of a Divide and Conquer algorithm as described in the lectures.

   (b) **(10 marks)** Define complexity classes P, NP, NPC, and $NP\text{-}Hard$ and draw a set diagram showing how they will relate to each other if $P = NP$, and separately if $P \neq NP$.

2. Greedy Algorithms **(30 marks)**

   Your friends are starting a security company that needs to obtain licenses for n different pieces of cryptographic software. Due to regulations, they can only obtain these licenses at the rate of at most one per month.

   Each license is currently selling for a price of \$100. However, they are all becoming more expensive according to exponential growth curves: in particular, the cost of license j increases by a factor of $r_j > 1$ each month, where r_j is a given parameter. This means that if license j is purchased t months from now, it will cost $100 * {r_j}^t$. We will assume that all the price growth rates are distinct; that is, $r_i \neq r_j$ for licenses $i \neq j$ (even though they start at the same price of \$100).

   The question is: *Given that the company can only buy at most one license a month, in which order should it buy the licenses so that the total amount of money it spends is as small as possible?*

   (a) **(15 marks)** Give an algorithm that takes the n rates of price growth $r_1, r_2, \ldots, r_n$, and computes an order in which to buy the licenses so that the total amount of money spent is minimized. The running time of your algorithm should be polynomial in n.

   (b) **(10 marks)** Prove that it is correct.

   (c) **(5 marks)** State and prove its [polynomial] time complexity.

3. Dynamic Programming **(35 marks)**

You're trying to run a large computing job in which you need to simulate a physical system for as many discrete steps as you can. The lab you're working in has two large supercomputers (which we'll call A and B) which are capable of processing this job. However, you're not one of the high-priority users of these supercomputers, so at any given point in time, you're only able to use as many spare cycles as these machines have available.

Here's the problem you face. Your job can only run on one of the machines in any given minute. Over each of the next n minutes, you have a "profile" of how much processing power is available on each machine. In minute i, you would be able to run $a_i > 0$ steps of the simulation if your job is on machine A, and $b_i > 0$ steps of the simulation if your job is on machine B. You also have the ability to move your job from one machine to the other; but doing this costs you a minute of time in which no processing is done on your job.

So, given a sequence of n minutes, a plan is specified by a choice of A, B, or "*move*" for each minute, with the property that choices A and B cannot appear in consecutive minutes — they have to be separated by "*move*". For example, if your job is on machine A in minute i, and you want to switch to machine B, then your choice for minute $i+1$ must be *move*, and then your choice for minute $i+2$ can be B. The value of a plan is the total number of steps that you manage to execute over the n minutes: so it's the sum of a_i over all minutes in which the job is on A, plus the sum of b_i over all minutes in which the job is on B.

The problem. Given values $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$, find a plan of maximum value. Note that your plan can start with either of the machines A or B in minute 1.

Example. Suppose $n = 4$, and the values of a_i and b_i are given by the following table.

	Minute 1	**Minute 2**	**Minute 3**	**Minute 4**
A	10	1	1	10
B	5	1	20	20

Then the plan of maximum value would be to choose A for minute 1, then *move* for minute 2, and then B for minutes 3 and 4. The value of this plan would be $10 + 0 + 20 + 20 = 50$.

(a) **(10 marks)** Show that the following algorithm does not correctly solve this problem, by giving an instance on which it does not return the correct answer.

(Question 3 continued on next page)

In your example, say what the correct answer is and also what the algorithm below finds.

```
In minute 1, choose the machine achieving the larger of a_1, b_1
Set i = 2
While i ≤ n
      What was the choice in minute i − 1?
      If A:
      If b_{i+1} > a_i + a_{i+1} then
            Choose move in minute i and B in minute i + 1
            Proceed to iteration i + 2
      Else
            Choose A in minute i
            Proceed to iteration i + 1
      Endif
      If B:  behave as above with roles of A and B reversed
EndWhile
```

(b) **(15 marks)** Give an efficient algorithm that takes values for $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$ and returns the value of an optimal plan.

(c) **(10 marks)** Prove that your algorithm is correct.

4. Approximation and Probabilistic **(20 marks)**

Consider a country in which 100,000 people vote in an election. There are only two candidates on the ballot: a Democratic candidate (denoted H) and a Republican candidate (denoted T). As it happens, this county is heavily Democratic, so 80,000 people go to the polls with the intention of voting for H, and 20,000 go to the polls with the intention of voting for T.

However, the layout of the ballot is a little confusing, so each voter, independently and with probability $\frac{1}{100}$, votes for the wrong candidate — that is, the one that he or she didn't intend to vote for. (Remember that in this election, there are only two candidates on the ballot.)

Let X denote the random variable equal to the number of votes received by the Democratic candidate H, when the voting is conducted with this process of error. Determine the expected value of X, and give an explanation of your derivation of this value.

5. Guest Lectures **(15 marks)**

(a) **(5 marks)** Describe what is *Kolmogorov Complexity* as presented in the guest lecture and how it can be used in computer graphics?

(b) **(5 marks)** Explain what is meant by the term *Deep Learning*?

(c) **(5 marks)** Explain what is meant by the term *Blockchain*?

* * * * * * * * * * * * * * *