Final EXAMINATION April 2006

DURATION: 3 HOURS No. of Students 70

Department Name & Course Number: Computer Science COMP4804
Course Instructor(s): P. Morin

AUTHORIZED MEMORANDA: None	Closed Book Examination. No Aids

Students **must** count the number of pages in this examination question paper before beginning to write, and report any discrepancy to a proctor. This question paper has 4 pages.

This examination question paper **may** be taken from the examination room.

In addition to this question paper, students require:			
	an examination booklet	[yes]	no
	a scantron sheet	yes	[no]

Remember Markov's Inequality: For a non-negative random variable X

$$\Pr\{X \geq t\mathrm{E}[X]\} \leq 1/t$$

and Chernoff's Bounds: For a binomial(p, n) random variable B,

$$\Pr\{B \geq (1+\epsilon)np\} \leq e^{-\epsilon^2 np/3} \quad \text{and} \quad \Pr\{B \leq (1-\epsilon)np\} \leq e^{-\epsilon^2 np/2} \ .$$

1 [5 marks] Expectation of a Product

Let X and Y be two *independent* random variables, i.e.,

$$\Pr\{X = x \cap Y = y\} = \Pr\{X = x\} \times \Pr\{Y = y\} \tag{1}$$

for any x and y. Prove that $\mathrm{E}[X \times Y] = \mathrm{E}[X] \times \mathrm{E}[Y]$. Here's what the proof looks like

$$\begin{aligned} \mathrm{E}[X \times Y] &= \sum_x \sum_y xy \Pr\{X = x \cap Y = y\} \\ &= \cdots \\ &= \mathrm{E}[X] \times \mathrm{E}[Y] \ . \end{aligned}$$

Complete the proof by filling in the $\cdots$.

2 [10 marks] Universal Hashing

Suppose we store n distinct elements $x_1, \ldots, x_n$ using an array of m lists $L_1, \ldots, L_m$ in the following way. For each x_i we have a *hash value* $h(x_i) \in \{1, \ldots, m\}$ and we store x_i in the list $L_{h(x_i)}$. The hash funcion h is chosen in such a way that, for $x \neq y$,

$$\Pr\{h(x) = h(y)\} \leq 1/m \ .$$

1. Consider a value $x \notin \{x_1, \ldots, x_n\}$. Use indicator variables to find the best upper bound possible on the expected size of the list $L_{h(x)}$. That is, prove that $\mathrm{E}[|L_{h(x)}|] \leq \mathit{blah}$ for the appropriate value *blah*

2. Consider a value $x_i \in \{x_1, \ldots, x_n\}$. Give the best upper bound you can on $\mathrm{E}[|L_{h(x_i)}|]$.

3 [15 marks] A Recursive Algorithm on Lists

Consider a subroutine that operates on a list. If we give the subroutine a list of length n then it runs in $O(n)$ time and, once it is complete, the input list has expected size $n/2$. We want to repeatedly call the subroutine until the list has size $O(1)$.

1. We say that a call to the subroutine is a *success* if it reduces the size of the list by a factor of at least 3/4 ($|L'| \leq 3|L|/4$). Give a lower bound on the probability that a particular call to the subroutine is a success.

2. Let X_k denote the number of successes that occur during a sequence of k consecutive calls. Give a lower bound on $\mathrm{E}[X_k]$.

3. Give an upper bound on the probability that the algorithm requires more than $c \log_{4/3} n$ calls to the subroutine.

4 [10 marks] Dynamically Growing and Shrinking Arrays

Consider the following memory-efficient implementation of a stack as an array A. Initially, we allocate an array A of size 1. Let n denote the size (number of elements on the stack) at some point in time.

If the user pushes an element on to the stack but the array is already full (because it has size n, i.e, $|A| = n$) then we allocate a new array A' of size $\lceil 3n/2 \rceil$, copy A into A', and set $A \leftarrow A'$.

Conversely, if the user pops an element from the stack, but the stack has size $2n$ ($|A| = 2n$) then we allocate a new array A' of size $\lceil 3n/2 \rceil$, copy the first n elements of A into A' and then set $A \leftarrow A'$.

1. Define a non-negative potential function, $\Phi(n, |A|)$, that will allow you to prove that both push and pop take $O(1)$ amortized time. (Hint: Having $|A| = 3n/2$ is good, but having $|A| = n$ or $|A| = 2n$ is bad.)

2. Using your potential function, show that push takes $O(1)$ amortized time.

3. Using your potential function, show that pop takes $O(1)$ amortized time.

5 [5 marks] Hardness of Truck Packing

Imagine you work at a shipping center and a truck is leaving from your shipping center to the one in Medicine Hat, Alberta. The truck has a maximum weight limit of W. You have n boxes of varying weights $w_1, \ldots, w_n$ that need to be shipped to Medicine Hat and would like to load the truck down as heavily as possible without exceeding its maximum weight limit. Explain why this problem is NP-hard by showing that, if you could solve this truck packing problem, you could solve one of the NP-hard problems discussed in class.

6 [20 marks] NP-Completeness of 3-Hitting-Set

Let $S = \{(a_i, b_i, c_i) : 1 \leq i \leq n\}$ be a set of n triples with $a_i, b_i, c_i \in \{1, \ldots, m\}$ and let k be a positive integer. The 3-Hitting-Set problem asks us to find a set X of k items such that, for every $1 \leq i \leq n$ at least one of a_i, b_i or c_i is in X. For example, if we are given the triples:

$$S = \{(1, 2, 3), (2, 3, 4), (1, 5, 6), (2, 7, 8)\}$$

then $X = \{1, 2\}$ is a hitting set of size 2 since every triple contains either a 1 or a 2.

1. Show that 3-Hitting-Set is NP-complete.

2. Give an $O(3^k n)$ time algorithm to solve 3-Hitting-Set.

3. Give a polynomial time greedy 3-approximation algorithm for 3-Hitting-Set. Be sure to analyze both the running time and approximation factor of the algorithm.

4. Using linear programming we can find $x_1, \ldots, x_m$, with each $0 \leq x_i \leq 1$, such that

$$x_{a_i} + x_{b_i} + x_{c_i} \geq 1 \tag{2}$$

for all $1 \leq i \leq n$ and

$$\sum_{i=1}^{m} x_i$$

is minimized.

Describe how, starting with $x_1, \ldots, x_m$, to find $x'_1, \ldots, x'_m$ that also satisfy (2), such that each x'_i is either 0 or 1, and

$$\sum_{i=1}^{m} x'_i \leq 3 \sum_{i=1}^{m} x_i \ .$$

7 [5 bonus marks] Bonus Algorithms Question

Save this question for when you've answered everything else. Let $A = \langle A_1, \ldots, A_n \rangle$ be a sequence of n distinct real numbers. Give an $O(n \log n)$ time algorithm that finds a subsequence $A_{i_1}, \ldots, A_{i_{\lceil \sqrt{n} \rceil}}$ of length $\lceil \sqrt{n} \rceil$ that is either strictly increasing ($A_{i_j} < A_{i_{j+1}}$) or stricting decreasing ($A_{i_j} > A_{i_{j+1}}$). For example, in the sequence

$$A = \langle 9, 4, \underline{2}, 6, \underline{5}, 1, 3, 8, \underline{7} \rangle$$

the subsequence $\langle A_3, A_5, A_9 \rangle = \langle 2, 5, 7 \rangle$ is strictly increasing.