CS 174: Combinatorics and Discrete Probability Fall 2012

Homework 2

Due: Thursday, September 13, 2012 by **9:30am**

Instructions: *You should upload your homework solutions on bspace. You are strongly encouraged to type out your solutions using LaTeX. You may also want to consider using mathematical mode typing in some office suite if you are not familiar with LaTeX. If you must handwrite your homeworks, please write clearly and legibly. We will not grade homeworks that are unreadable. You are encouraged to work in groups of 2-4, but you* **must** *write solutions on your own. Please review the homework policy carefully on the class homepage.*

Note: You *must* justify all your answers. In particular, you will get no credit if you simply write the final answer without any explanation.

Problem 1. *(Exercise 1.23 from MU)* There may be several different min-cut sets in a graph. Using the analysis of the randomized min-cut algorithm, argue that there can be at most $n(n-1)/2$ distinct min-cut sets.

Problem 2. The goal of this exercise is to compute uniformly random sequences of length k of the integers $\{1, \ldots, n\}$ *without replacement*. If $k = n$, the sample is a permutation of $1, \ldots, n$.

(a) Assuming the existence of a constant-time generator that generates uniformly random *real* numbers in the interval $[0, 1]$, derive a $O(n \log(n))$-time algorithm for computing a random permutation of $1, \ldots, n$. (Hint: Think sorting.)

(b) Assuming the existence of a constant-time generator that generates uniformly random *integers* in the range $\{1, \ldots, m\}$ for any m, derive an $O(n)$-time algorithm that generates a random permutation of $1, \ldots, n$. (Hint: Start with a sorting algorithm that makes $O(n)$ data moves.)

(c) *(Optional, not graded)* Assuming the same random number generator as in (b) above, derive an $O(k)$ expected-time algorithm that generates a random sequence of length k from $1, \ldots, n$ without replacement.

Problem 3. *(Exercise 2.2 from MU)* A monkey types on a 26-letter keyboard that has lowercase letters only. Each letter is chosen independently and uniformly at random from the alphabet. If the monkey types 1,000,000 letters, what is the expected number of times the sequence "proof" appears?

Problem 4. *(Exercise 2.18 from MU)* The following approach is often called *reservoir sampling*. Suppose we have a sequence of items passing by one at a time. We want to maintain a sample of one item with the property that it is uniformly distributed over all the items that we have seen at each step. Moreover, we want to accomplish this without knowing the total number of items in advance or storing all of the items that we see.

Consider the following algorithm, which stores just one item in memory at all times. When the first item appears, it is stored in the memory. When the k^{th} item appears, it replaces the item in memory with probability $1/k$. Explain why this algorithm solves the problem.

Problem 5. *(Exercise 2.22 from MU)* Let $a_1, a_2, \ldots, a_n$ be a list of n distinct numbers. We say that a_i and a_j are *inverted* if $i < j$ but $a_i > a_j$. The *Bubblesort* sorting algorithms swaps pairwise adjacent inverted numbers in the list until there are no more inversions, so the list is in sorted order. Suppose that the input to Bubblesort is a random permutation, equally likely to be any of the $n!$ permutations of n distinct numbers. Determine the expected number of inversions that need to be corrected by Bubblesort.

Problem 6. *(Exercise 2.32 from MU)* You need a new staff assistant, and you have n people to interview. You want to hire the best candidate for the position. When you interview a candidate, you can give them a score, with the highest score being the best and no ties being possible. You interview the candidates one by one. Because of your company's hiring practices, after you interview the k^{th} candidate, you either offer the candidate the job before the next interview or you forever lose the chance to hire that candidate. We suppose the candidates are interviewed in a random order, chosen uniformly at random from all $n!$ possible orderings.

We consider the following strategy. First, interview m candidates but reject them all; these candidates give you an idea of how strong the field is. After the m^{th} candidate, hire the first candidate you interview who is better than all of the previous candidates you have interviewed.

(a) Let E be the event that we hire the best assistant, and let E_i be the event that i^{th} candidate is the best and we hire him. Determine $\Pr(E_i)$, and show that

$$\Pr(E) = \frac{m}{n} \sum_{j=m+1}^{n} \frac{1}{j-1}.$$

(b) Bound $\sum_{j=m+1}^{n}(1/(j-1))$ to obtain

$$\frac{m}{n}(\ln n - \ln m) \leq \Pr(E) \leq \frac{m}{n}(\ln(n-1) - \ln(m-1)).$$

(c) Show that $m(\ln n - \ln m)/n$ is maximized when $m = n/e$, and explain why this means that $\Pr(E) \geq 1/e$ for this choice of m.