CMPUT 675: Randomized Algorithms **Fall 2005**

Problem Set 1:

Due Date: Oct 11, 2005

You are allowed to discuss the problems with a small group of students, however you must not take any notes from your discussions. You must list the names of other students that you have discussed the problems with, or any other resources (other than the text book) that you have used. Also, please indicated which questions (if any) you had seen before or you knew the solution. These information will not affect your mark.

1. Suppose you have a biased coin with a fixed but unknown probability p of heads. How do you use this coin to generate unbiased random 0/1 bits?

2. You are given a stream of elements passing by one at a time. We have a memory that holds only k elements and we want to maintain a random sample of k elements seen so far at any given time. We don't know the total number of elements (i.e. when the stream is going to end); if the stream terminates our memory must contain a uniform sample of the elements. How do you do this?

3. Consider a uniform rooted tree of height h (i.e the longest root-to-leaf path has h edges), where every internal node as well as the root has 3 children. Suppose each leaf has a 0/1 value and each non-leaf node has a majority gate which returns the majority value of its children. The problem is to evaluate the value at the root, given values of the leaves. Let $n = 3^h$ be the number of leaves.

   (a) Show that for any deterministic algorithm, the adversary can force it to read all the n leaves.

   (b) Show that there is a non-deterministic algorithm that can compute the value at the root by reading $O(\sqrt{n})$ leaves.

   (c) Consider the randomized algorithm that, for every internal node, evaluates two subtrees rooted at two randomly chosen children of that node, and evaluates the third only if the first two answers disagree. Show that the expected number of leaves read by this algorithm is at most $O(n^{0.9})$.

4. Suppose that a fair coin is flipped n times. For $k > 0$ find an upper bound on the probability that there is a sequence of $\log n + k$ consecutive heads.

5. Show that the randomized quicksort algorithm runs in time $O(n \log n)$ with high probability. (Hint: prove that the number of pivots any particular element is compared with is at most $O(\log n)$ w.h.p.)

6. Consider the following version of occupancy problem. There are n bins and in the first round of process with throw n balls into these bins independently randomly (i.e. the destination of each ball is selected u.r.). After each round $i \geq 1$, we remove every non-empty bin and start

round $i+1$ by throwing n new balls into the remaining (empty) bins. This process terminates when all bins are removed (i.e. become non-empty).

(a) Prove that the expected number of rounds for this process is $O(\log^* n)$ where $\log^* n$ is the number of iterated log's we need to take until the result becomes ≤ 1 (e.g. $\log^* 2 = 1$, $\log^* 4 = 2$, $\log^* 16 = 3$, $\log^* 65536 = 4$, and $\log^* 2^{65536} = 5$).

(b) (optional) Prove that the number of rounds for this process is $O(\log^* n)$ w.h.p.

7. Consider the following game played by n players. Eacn player is to be fitted with a hat that is colored either Red or Blue each with probability $\frac{1}{2}$ independent of others; each person will be able to see the color of the hat of everyone else, but not the color of his own hat. Once the hats are placed, every player has the option of guessing the color of his own hat or remaining silent. At a signal, all the players simultaneously either guess the color of their hat or choose to remain silent. Players can agree on a strategy before the game starts but they cannot communicate with each other once the hats are placed on their heads (in particular they do not hear the decisions of others). The players win if at least one player guesses and all the guesses are correct (so even one wrong guess is enough to lose). Show a strategy for the players that achieves success probability of $1 - o(1)$ as n goes to infinity.

8. Consider the following variation of the above game. Again there are n players each is to be fitted with a red or blue hat. This time everybody must guess the color of his own hat and the goal is to maximize the number of correct guesses. However, there is no knowledge about the distribution of the hats to be placed (i.e. they are to be placed by an adversary which tries to foil our strategy). Clearly, if everybody guesses randomly red or blue with probability $\frac{1}{2}$ each, the expected number of correct guesses (regardless of how the hats are placed) is always $\frac{n}{2}$. Can you think of a deterministic algorithm that guarantees at least $\lfloor \frac{n}{2} \rfloor$ correct answers?