CS:5360 Fall 2018 Homework 1

Due: Thu, 9/6

Notes: (a) It is possible that solutions to some of these problems are available to you via textbooks on randomized algorithms or on-line lecture notes, etc. If you use any such sources, please acknowledge these in your homework fully *and* present your solutions in your own words. You will benefit most from the homework, if you seriously attempt each problem on your own first, before seeking other sources. (b) As mentioned in the syllabus, it is okay to form groups of two in solving and submitting homework solutions. But, my advice from (b) still applies: you will benefit most from the homework, if you seriously attempt each problem on your own first, before seeking help from your group partner. (c) Discussing these problems with any of your classmates is okay, provided you and your classmates are not being too specific about solutions. In any case, make sure that you take no written material away from these discussions *and* (as in (b)) you present your solutions in your own words. When discussing homework with classmates please be aware of guidelines on "Academic Dishonesty" as mentioned in the course syllabus.

1. Show that any problem that is in RP and in $coRP$ has a Las Vegas algorithm that runs in polynomial time.

2. Bowl I contains 3 red chips and 7 blue chips. Bowl II contains 6 red chips and 4 blue chips. A bowl is selected (uniformly) at random and then one chip is drawn (again, uniformly at random) from the selected bowl. (a) Compute the probability that this chip is red. (b) Conditioned on the hypothesis that the chip is red, find the conditional probability that it is drawn from Bowl II.

3. Let $0 < \epsilon_2 < \epsilon_1 < 1$. Consider a Monte Carlo algorithm that gives the correct solution to a problem with probability at least $1 - \epsilon_1$, regardless of the input. How many independent executions of this algorithm suffice to raise the probability of obtain a correct solution to at least $1 - \epsilon_2$, regardless of the input? Show your work.

4. Describe an algorithm that takes as input a positive integer n and returns a random permutation of $I = \{1, 2, \ldots, n\}$, chosen uniformly at random from all $n!$ permutations of I. The algorithm should use, as its source of randomness, a procedure that returns a random bit in $O(1)$ time. Assume that the two bits, 0 and 1, are equally likely to be returned by a call to the procedure and that each call returns a bit that is independent of past history. Your algorithm is required to run in $O(n \log n)$ time.

   Your answer should consist of a brief description of the algorithm, followed by a brief analysis that shows that the algorithm runs in $O(n \log n)$ time and produces every permutation of I with equal probability.

5. Suppose that at every iteration of Karger's min-cut algorithm, instead of choosing a random edge for contraction, we choose two vertices u and v at random (i.e., uniformly at random from all unordered vertex pairs) and contract the u-v pair. Show that there are graph examples on which the probability that this modified algorithm finds a min-cut is exponentially small. More precisely, show that there is a family $\mathcal{F}$ of graphs such that for every n large enough, $\mathcal{F}$ contains a graph G_n with n vertices such that the modified Karger's algorithm finds a min-cut on G_n with probability at most $1/a^{n/b}$ for constants $a > 1$ and $b \geq 1$.