CS 466/666 Algorithm Design and Analysis **Last Updated:** May 13 (10pm), 2018
Homework 1 **Due Date: May 25 (11pm), 2018**

You are allowed to discuss with others but not allowed to use any references except the course notes and the books "Probability and Computing" and "Randomized Algorithms". Please list your collaborators for each question. This will not affect your marks. In any case, you must write your own solutions.

There are totally 55 marks, and the full mark is 50. This homework is counted 8% of the course.

---

1. **Sampling**

   (5 marks) Suppose we have a long sequence of numbers coming one at a time. We would like to maintain a sample of one number with the property that it is uniformly distributed over all the items that we have seen at each step. We want to accomplish this without knowing the total number of items in advance or storing all of the items that we see.

   Prove that the following simple algorithm works. When the first item comes, it is stored in the memory. When the k-th item appears, it replaces the item in memory with probability $1/k$.

2. **Online Hiring**

   (10 marks) You need to hire a new staff. There are n applicants for this job. Assume that you will know how good they are (as a score) when you interview them, and the score for each applicant is different. So there is a unique candidate with the highest score, but you don't know that the applicant is the best when you interview him/her until you have interviewed all the applicants. The problem is that after you interview one applicant, you need to make an online decision to either give him/her an offer or forever lose the chance to hire that applicant. Suppose the applicants come in a random order (i.e. a uniformly random permutation), and you would like to come up with a strategy to hire the best applicant.

   Consider the following strategy. First, interview m applicants but reject them all. Then, after the m-th applicant, hire the first applicant you interview who is better than all of the previous applicants that you have interviewed.

   Let E be the event that you hire the best applicant. Let E_i be the event that the i-th applicant is the best and you hire him/her. Compute $\Pr(E_i)$ and show that

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

   Then, show that $\Pr(E) \geq \frac{m}{n}(\ln n - \ln m)$, and that $\Pr(E)$ can be arbitrarily close to $1/e$ for an appropriate choice of m when n tends to infinity.

3. **Minimum k-cut**

   (10 marks) Generalizing on the notion of a cut-set, we define a k-way cut-set in an undirected graph as a set of edges whose removal breaks the graph into k or more connected components. Show that the randomized contraction algorithm can be modified to find a minimum k-way cut-set in $n^{O(k)}$ time.

4. **Quicksort**

(10 marks) The expected runtime of the randomized quicksort algorithm is at most $2n \ln n$ steps where n is the number of elements to be sorted (see Section 2.5 of "Probability and Computing" for a proof). Prove that the probability that the actual runtime is more than say $100n \ln n$ is at most inverse polynomial in n. (Hint: Bound the probability that the recursion depth is large.)

5. **Coupon Collectors**

(10 marks) Now you know that it requires $\Theta(n \log n)$ coupons to collect n different types of coupons, with at least one coupon per type. You wonder whether it would be more efficient if a group of k people cooperate, such that each person buys cn coupons and exchange with each other so that each person has a complete set of coupons.

Prove the best bound you can on k to ensure that, with probability at least 0.9, each person only needs to buy at most $10n$ coupons.

6. **Graph Sparsification**

(10 marks) Given a complete graph G with n vertices where every edge is of the same weight, explain that the uniform sampling algorithm in L02 will succeed with high probability in giving a $(1 \pm \epsilon)$-cut approximator H of G with H having only $O((n \log n)/\epsilon^2)$ edges.

Also, provide a heuristic argument that when given the unweighted complete graph G as input, the uniform sampling algorithm requires $\Omega(n \log n)$ edges to produce any reasonable cut approximator H of G.