E0 234 : Homework 1

Deadline : 27th January 2025, 11am

Instructions

- Please write your answers using LaTeX. Handwritten answers will not be accepted.
- You are forbidden from consulting the internet. You are strongly encouraged to work on the problems on your own.
- You may discuss these problems with another student. However, you must write your own solutions and must mention your collaborator's name. Otherwise, it will be considered as plagiarism.
- Academic dishonesty/plagiarism will be dealt with severe punishment.
- Late submissions are accepted only with prior approval (on or before the day of posting of HW) or a medical certificate.

1. *Inclusion-Exclusion using Indicator Functions.*
   Let $\mathbb{1}_A$ be the indicator random variable of the event A, i.e.,

$$\mathbb{1}_A(w) = \begin{cases} 1, & \text{if } \omega \in A, \\ 0, & \text{if } \omega \in A^c, \end{cases} \tag{1}$$

   for each ω in the sample space. Note that $\Pr[A] = \mathrm{E}[\mathbb{1}_A]$. Show the following:

   - $\mathbb{1}_A + \mathbb{1}_{A^c} = 1$;
   - $\mathbb{1}_{A \cap B} = \mathbb{1}_A \cdot \mathbb{1}_B$;
   - $\mathbb{1}_{A \cup B} = \mathbb{1}_A + \mathbb{1}_B - \mathbb{1}_A \cdot \mathbb{1}_B$;
   - More generally, if $B = \bigcup_{i=1}^n A_i$ then $\mathbb{1}_B = 1 - \prod_{i=1}^n (1 - \mathbb{1}_{A_i})$.

   Suppose $x_1, x_2, \ldots, x_n$ such that $0 \leq x_i \leq 1$; for $J \subseteq [n]$, let $x_J = \prod_{i \in J} x_i$. Let $p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^n (1 - x_i)$. Show the following:

$$\begin{aligned}
p(x_1, x_2, \ldots, x_n) &= \sum_{S \subseteq [n]} (-1)^{|S|} x_S; \\
p(x_1, x_2, \ldots, x_n) &\leq \sum_{S \subseteq [n], |S| \leq k} (-1)^{|S|} x_S & (k \text{ even}); \\
p(x_1, x_2, \ldots, x_n) &\geq \sum_{S \subseteq [n], |S| \leq k} (-1)^{|S|} x_S & (k \text{ odd}).
\end{aligned}$$

Conclude from these, using linearity of expectation, that

$$\Pr[\bigcup_{i=1}^{n} A_i] = \sum_i \Pr[A_i] - \sum_{i<j} \Pr[A_i \cap A_j] + \cdots + (-1)^{n+1} \Pr[A_1 \cap \cdots \cap A_n];$$

$$\Pr[\bigcup_{i=1}^{n} A_i] \geq \sum_i \Pr[A_i] - \sum_{i<j} \Pr[A_i \cap A_j] + \cdots + (-1)^{k+1} \sum_{i_1<i_2<\cdots<i_k} \Pr[A_{i_1} \cap \cdots \cap A_{i_k}], \quad (k \text{ even});$$

$$\Pr[\bigcup_{i=1}^{n} A_i] \leq \sum_i \Pr[A_i] - \sum_{i<j} \Pr[A_i \cap A_j] + \cdots + (-1)^{k+1} \sum_{i_1<i_2<\cdots<i_k} \Pr[A_{i_1} \cap \cdots \cap A_{i_k}], \quad (k \text{ odd}).$$

2. *Min-cuts with vertex merging.*
   Consider the randomized min-cut algorithm presented in the class. Suppose that at each step, instead of choosing a random edge for contraction, two vertices are chosen at random and are merged into a single vertex. Show that there exist graphs for which the probability that this algorithm finds a min-cut is exponentially small in the number of vertices.

3. *Improved Karger's Algorithm (M-U 1.25).* Consider the following algorithm for min cut. Starting with a graph with n vertices, first contract the graph down to k vertices using the randomized edge-contraction method discussed in class. Make copies of this reduced graph with k vertices, and now run the randomized min cut algorithm on this reduced graph ℓ times, independently. Determine the total number of edge contractions performed by this algorithm, and bound the probability of finding a min-cut in at least one of the ℓ runs.

   Optimize (as much as you can) values of k and ℓ for the variation above that maximize the probability of finding a minimum cut while using the same number of edge contractions as running the original Karger's algorithm twice.

4. *Randomized Quicksort.*
   Consider the following way to pick a random permutation π of the set of n (> 2) integers $\mathcal{I}_n := \{1, 2, \ldots, n\}$:

   - Run Randomized Quicksort.
   - Let T be the recursion tree corresponding to the execution of Randomized Quicksort.
   - Let π be the permutation induced by the level-order traversal of T (i.e., the nodes are visited in the increasing order of level numbers and in a left-to-right order within each level).

   Is π uniformly distributed over the space of all permutations of the elements in $\mathcal{I}_n$? Explain your answer.

Recommended practice problems (not for grading)

1. *Fixed Points in a Permutation.*
   A permutation on the numbers $[n] := \{1, 2, \ldots, n\}$ can be represented as a function $\pi : [n] \to [n]$, where $\pi(i)$ is the position of i in the ordering given by the permutation. A fixed point of a permutation $\pi : [n] \to [n]$ is an index x for which $\pi(x) = x$. We showed in the class that the expected number of fixed points of a permutation (chosen uniformly at random from the set of all permutations) is one. Now show that the number of fixed points converges to Poisson distribution with $\lambda = 1$.
   (*Hint.* Let X_n be the number of fixed points in a permutation. First, show that the number of permutations without fixed points (also called *derangements*) for n elements is $n!(\sum_{i=0}^{n} \frac{(-1)^i}{i!})$. Use this to show $\Pr[X_n = j] = \frac{1}{j!} \sum_{k=0}^{(n-j)} \frac{(-1)^k}{k!}$, and this converges to $\frac{1}{j!e}$.

2. *Splitting Graphs.*
In class, we gave a randomized algorithm that if a graph G has m edges then it contains a bipartite subgraph with at least $m/2$ edges. Can we *derandomize* it, i.e., give a deterministic algorithm with the same guarantee?

   Now we wish to prove a stronger guarantee that if the graph G has $2n$ vertices and m edges then it contains a bipartite subgraph with at least $mn/(2n-1)$ edges. If G has $2n+1$ vertices and m edges then it contains a bipartite subgraph with at least $m(n+1)/2n+1$ edges.

3. *Distinct Min Cuts.*
Consider the randomized min cut algorithm presented in the class. We showed that, for any graph G with n vertices, the probability that the algorithm finds a specific min cut C of G is at least $2/n(n-1)$.

   - What can we say about the maximum number of distinct min cuts that a graph G can have?
   - Give an example of a graph (with n vertices) with maximum number of distinct min cuts.
   - Use the randomized edge contraction algorithm to find all the global min cuts in any graph G.

4. *Min k-cut.*
Given an unweighted and undirected graph $G := (V, E)$, a 3-*cut* is a partition of V into three nonempty sets A, B, C. The size of the cut is the number of edges connecting vertices from different sets. Extend the randomized min cut algorithm presented in the class to give an algorithm to find a minimum 3-cut. Extend your answer to provide an algorithm for the case of minimum k-cuts for any constant integer $k \geq 3$ and show its probability of success.

5. *Concentration of Quicksort (M-U 4.21).*
We prove that the Randomized Quicksort algorithm sorts a set of n numbers in time $O(n \log n)$ with high probability. Consider the following view of Randomized Quicksort. Every point in the algorithm where it decides on a pivot element is called a node. Suppose the size of the set to be sorted at a particular node is s. The node is called good if the pivot element divides the set into two parts, each of size not exceeding $2s/3$. Otherwise, the node is called bad. The nodes can be thought of as forming a tree in which the root node has the whole set to be sorted and its children have the two sets formed after the first pivot step and so on.

   (a) Show that the number of good nodes on any path from the root to a leaf in this tree is not greater than $c \log_2 n$, where c is some positive constant.

   (b) Show that, with high probability (greater than $1 - 1/n^2$), the number of nodes on a given root to leaf path of the tree is not greater than $c' \log_2 n$, where c' is another constant.

   (c) Show that, with high probability (greater than $1 - 1/n$), the number of nodes on the longest root-to-leaf path is not greater than $c' \log_2 n$. (Hint: How many nodes are there in the tree?)

   (d) Use your answers to show that the running time of Quicksort is $O(n \log n)$ with probability at least $1 - 1/n$.

6. *Book:* **Mitzenmacher-Upfal (2nd edition): 2.13, 2.18, 2.20, 2.22, 2.23, 2.26, 2.32.**