CS648A : Randomized Algorithms
Semester II, 2021-22, CSE, IIT Kanpur

Theoretical Assignment 3

Deadline : 11:55 PM, 15th April 2022.

Most Important guidelines

- It is only through the assignments that one learns the most about the algorithms and data structures. You are advised to refrain from searching for a solution on the net or from a notebook or from other fellow students. Remember - **Before cheating the instructor, you are cheating yourself**. The onus of learning from a course lies first on you. So act wisely while working on this assignment.
- Refrain from collaborating with the students of other groups. If any evidence is found that confirms copying, the penalty will be very harsh. Refer to the website at the link: https://cse.iitk.ac.in/pages/AntiCheatingPolicy.html regarding the departmental policy on cheating.

General guidelines

1. This assignment is to be done in groups of 2 students. You have to form groups on your own. You are strongly advised not to work alone.
2. The assignment consists of 3 problems. Each problem carries 50 marks.
3. **Naming the file**:
   The submission file has to be given a name that reflects the information about the type of the assignment, the number of the assignment, and the roll numbers of the 2 students of the group. If you are submitting the solution of Theoretical Assignment x, you should name the file as **Theor_x_Rollnumber1_Rollnumber2.pdf**.
4. **Each student of a group** has to upload the same submission file separately. Be careful during the submission of an assignment. Once submitted, it can not be re-submitted.
5. Deadline is strict. Make sure you upload the assignment well in time to avoid last minute rush.

1 How well did you internalize the Delay Sequences ?

Suppose there is an undirected graph $G = (V, E)$ on n vertices where degree of each vertex is d. Each vertex hosts a counter and a coin. Each coin, when tossed, gives heads with probability p. Let $Count(v)$ denotes the value of the counter hosted at vertex v at any moment of time. $Count(v)$ is initialized to 0 in the beginning for each $v \in V$. In each round, each vertex $v \in V$ tosses its coin and increments its counter value if the following conditions holds true.

- The outcome of coin tossed by the vertex v is heads.
- There is no neighboring counter of v with counter value less than $Count(v) - b$.

Assume that b and d are constants of integer values and $0 < p < 1$. Show that the all the counters will reach $\log n$ value with in $O(\log n)$ rounds with high probability.
Note: The constant hidden in big-Oh notation of $O(\log n)$ may be a function of b, d, and p; it is perfectly fine.

2 Randomization to break symmetry

A nontrivial task in many of the distributed algorithms is to break symmetry. Randomization usually turns out to be very powerful tool to accomplish this. The current problem should be able to convince you.

Consider a distributed network (V, E) consisting of $n = |V|$ nodes and degree of each node is d. A subset of nodes $S \subseteq V$ is said to be an independent set if for each $i, j \in S$, $(i, j) \notin E$. The aim is to compute an independent set S of large size. Here is the sketch of the distributed algorithm that computes such an independent set in a single round using $O(m)$ messages.

Algorithm 1: A distributed algorithm to compute independent set

Each node i tosses a coin which gives HEADS with probability p ;
Each node i sends the outcome of its coin toss to its neighbours and, in turn, receives their outcome as well;
Each node i adds itself to set S if ..... ;

1. Complete the algorithm by suitably writing the text in the dotted line.
2. Prove that the set S computed by the algorithm will indeed be an independent set.
3. Calculate the expected size of S in terms of n, d, p.
4. Find the value of p for which the expected size of S is maximum. Also state the corresponding value of the expected size of S.

3 Randomization to fool the adversary

Consider a communication network consisting of a central vertex v joined to n slave vertices $x_1, x_2, \ldots, x_n$ through communication links. We wish to ensure that v remains connected to at least one slave vertex. Keeping a communication link alive consumes a lot of power. So v may keep a link to only one slave vertex alive. However, as we all know, communication networks in real world are prone to failures. So we would like to keep multiple links alive. The question is : How to choose the right number and the right set of links which we keep alive so that v can communicate to at least one slave vertex even after failure of some links between v and the slave vertices? In precise words, the problem is modeled as follows.

There is a parameter k which denotes the maximum number of failed links on any day. Moreover $k \leq n/2$ always. For each of the next n^2 days, there will be a set of k links which will be failing. This set can be different for different days. Moreover, we have no apriori knowledge of these sets. We need an algorithm to pick sufficiently large number of links in the beginning itself so as to make sure that there is always an alive link between v and at least one slave vertex on each of the following n^2 days.

1. Design a deterministic algorithm for the problem. What is the number of links your algorithm will keep alive ?

2. Design a randomized algorithm for the problem with the guarantee that the node v has an alive link to at least one slave node on each of the n^2 days with probability at least $1 - 1/n^2$. What is the expected number of links your algorithm will keep alive ? Provide complete details of the analysis.