CSE 521: Design and Analysis of Algorithms **Fall 2019**

Problem Set 1

Deadline: Oct 11 (at 6:00 PM) in Canvas

Instructions

- You should think about each problem by yourself for at least an hour before choosing to collaborate with others.
- You are allowed to collaborate with fellow students taking the class in solving the problems (in groups of at most 2 people for each problem). But you **must** write your solution on your own.
- You are not allowed to search for answers or hints on the web. You are encouraged to contact the instructor or the TAs for a possible hint.
- You cannot collaborate on Extra credit problems
- Solutions typeset in LATEX are preferred.
- Feel free to use the Discussion Board or email the instructor or the TA if you have any questions or would like any clarifications about the problems.
- Please upload your solutions to Canvas. The solution to each problem **must** be uploaded separately.

---

1) In this problem you are supposed to implement Karger's min-cut algorithm. I have uploaded three input files to the course website. Each file contains the list of edge of a graph; note that the graphs may also have parallel edges. The label of each node is an integer.

   There are three inputs uploaded to the course website. For each input file you should output the size and the number of min cuts. Please upload your code to Canvas. You should also write the output of your program for each input in the designated "text box" of Problem 1 in Canvas.

2) a) Show how to construct a biased coin, which is 1 with probability p and 0 otherwise, using $O(1)$ random bits in expectation. [**Hint:** First show how to construct a biased coin using an arbitrary number of random bits. Then show that the expected number of bits examined is small.]

   b) Given $p_1, \ldots, p_n$ where $\sum_i p_i = 1$, show how to sample from $\{1, \ldots, n\}$ where i must be chosen with probability p_i, using $O(\log n)$ random bits in expectation.

   c) Show that the "in expectation" caveat is necessary: for example, one cannot sample uniformly over $\{1, 2, 3\}$ using $O(1)$ bits in the worst case.

3) Consider the following process for executing n jobs on n processors. In each round, every (remaining) job picks a processor uniformly and independently at random. The jobs that have no contention on the processors they picked get executed, and all other jobs *back off* and then try again. Jobs only take one round of time to execute, so in every round all the processors are available.

   For example, suppose we want to run 3 jobs on 3 processors. Suppose in round 1, jobs 1 and 2 choose the first processor and job 3 chooses the second processor. Then job 3 will be executed and jobs 1 and 2 back off. Suppose in round 2, job 1 chooses the third processor and job 2 chooses the first processor. Then both of them are executed and the process ends in 2 rounds.

   In this problem we almost show that the number of rounds until all jobs are finished is $O(\log \log n)$ with high probability.

a) Suppose less than $\sqrt{n}$ jobs are left at the beginning of some round. Show that for some constant $c > 0$, with probability at least c no job remains after this round.

b) In the first round we have n jobs. Show that the expected number of processors that are picked by no jobs is $n(1-1/n)^n$.

c) Suppose there are r jobs left at the beginning of some round. What is the expected number of processors that are matched to exactly one job? What is the expected number of jobs remaining to be completed after that round?

d) Suppose in each round the number of jobs completed is exactly equal to its expectation. Show that (under this false assumption) the number of rounds until all jobs are finished is $O(\log \log n)$.

e) In this part we almost justify the false assumption. Suppose there are r jobs left at the beginning of some round. Let E be the expected number of processors that are matched to exactly one job. Show that for any $k > 1$, the number of processors with exactly one matched job is in the interval $[E - k\sqrt{r}, E + k\sqrt{r}]$ with probability at least $1 - \exp(-\Omega(k^2))$. You can use the McDiarmid's inequality to prove the claim.

Theorem 1.1 (McDiarmid's inequality)**.** *Let $X_1, \ldots, X_n \in \mathcal{X}$ be independent random variables. Let $f : \mathcal{X}^n \to \mathbb{R}$. If for all $1 \le i \le n$ and for all $x_1, \ldots, x_n$ and $\tilde{x}_i$,*

$$|f(x_1, \ldots, x_n) - f(x_1, \ldots, x_{i-1}, \tilde{x}_i, x_{i+1}, \ldots, x_n)| \le c_i,$$

then

$$\mathbb{P}\left[|f(X_1, \ldots, X_n) - \mathbb{E}\left[f\right]| \ge \epsilon\right] \le 2 \exp\left(\frac{-2\epsilon^2}{\sum_{i=1}^n c_i^2}\right)$$

4) Consider an n-dimension hypercube as a network of parallel processors. The network has $N = 2^n$ processors where each processor is represented by an n bit string $x_0 x_1 \ldots x_{n-1}$ and two processors are connected by a wire if their bit representations differ in exactly one bit. We consider the permutation routing problem on such a network. Each processor x initially contains one packet p_x destined for some processor $d(x)$ in the network such that each processor is the destination of exactly one packet, i.e., $d(.)$ is a permutation. All communication between processors proceeds in a sequence of synchronous steps. At each time step each wire can transmit a single packet in each direction. So, in each step step, a processor can send at most one packet to each of its neighbors.

We want to design an algorithm to specify a route for each packet, i.e., a sequence of edges from the source to the destination. Note that a packet may have to wait for several steps at an intermediate node y because multiple packets may want to leave y through the same wire. The goal is to design an algorithm to route all packets in a small number of steps.

(a) Consider the following simple strategy called *bit-fixing*. To send a packet p_x from node x to the node $d(x)$, scan the bits of $d(x)$ from left to right, and compare them with the address of the current location of p_x, send p_x out of the current node along the edge corresponding to the left-most bit in which the current position and $d(x)$ differ. For example, in going from 1011 to 0000 in a 4-dimensional hypercube, the packet would pass through $0011, 0001$. Construct a permutation $d(.)$ such that bit-fixing strategy takes $\Omega(\sqrt{N}/n)$ steps.

Now, consider the following 2-phase simple strategy. Pick a uniformly random intermediate destination $\sigma(x)$ for each packet p_x. In the first phase use bit-fixing to send p_x to $\sigma(x)$. In the second phase send p_x from $\sigma(x)$ to $d(x)$. We prove that this routing strategy takes only $O(n^2)$ steps[1].

(b) Show that for each node y the expected number of packets that pass through y in the first phase is $O(n)$.

[1] We remark that it is also possible to prove that the 2-phase strategy takes only $O(n)$ steps but here we prove a weaker bound.

(c) Use the Bernstein's inequality to show that for each node y the number of packets that pass through y in the first phase is $O(n)$ with probability at least $1 - 1/N^2$.

Theorem 1.2 (Bernstein's inequality)**.** *Let $X_1, \ldots, X_n$ be independent Bernoulli random variables. Then*

$$\mathbb{P}\left[\sum_{i=1}^{n} X_i - \mathbb{E}\sum_{i=1}^{n} X_i > \epsilon\right] \leq \exp\left(\frac{-\frac{1}{2}\epsilon^2}{\sum \mathrm{Var}(X_i) + \epsilon/3}\right).$$

(d) Prove that the 2-phase strategy takes only $O(n^2)$ steps w.h.p..

5) **Extra Credit:** Say we have a plane with n seats and we have a sequence of n passengers $1, 2, \ldots, n$ who are going to board the plane in this order and suppose passenger i is supposed to sit at seat i. Say when 1 comes he chooses to sit at some arbitrary seat different from his own sit, 1. From now on, when passenger i boards, if her seat i is available she sits at i, otherwise she chooses sits at a uniformly random seat that is still available. What is the probability that passenger n sits at her seat n?