CSE 521: Design and Analysis of Algorithms **Fall 2018**

Problem Set 1

Deadline: Oct 15 (at 12:00 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).
- You are not allowed to search for answers or hints on the web. You are encouraged to contact the instructors or the TA 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.

   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. There are four inputs uploaded to the course website. The last one, "b3.in" is very large but it has only 10 percent of the grade of this problem.

2) 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

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

3) Given a graph $G = (V, E)$ with $n = |V|$ vertices. Let k be the size of the minimum cut of G. In this problem we show that if k is large enough we can down sample G and preserve the size of its min-cut.

Let H be a subgraph of G defined as follows: For every edge e of G include e in H with probability $p = \frac{12 \ln n}{k\epsilon^2}$. If we include e in H we weight it by $1/p$.

We show that the (weighted) min-cut of H is at least $k(1-\epsilon)$ with probability at least $1 - 1/n$. Note that H has only $p|E|$ many edges (in expectation). So, H has significantly less edges than G if $k \gg \ln(n)/\epsilon^2$.

a) For a cut $(S, \overline{S})$, let

$$w_H(S, \overline{S}) = \frac{|H(S, \overline{S})|}{p}$$

be the sum of the weights of all edges of H across this cut. Show that for any set $S \subset V$,

$$\mathbb{E}\left[w_H(S, \overline{S})\right] = |E(S, \overline{S})|$$

b) Use Chernoff bound to show that for any set $S \subset V$,

$$\mathbb{P}\left[w_H(S, \overline{S}) < (1-\epsilon)|E(S, \overline{S})|\right] \le e^{-6 \ln n |E(S,\overline{S})|/k} = n^{-6|E(S,\overline{S})|/k}$$

c) Recall that in class we proved that each graph has at most $\binom{n}{2}$ many min-cuts. We say a cut of G is an α-min-cut if its size is at most α times the size of the min-cut, i.e., at most αk many edges. It follows by an extension of the Karger's algorithm that any graph G has at most $n^{2\alpha}$ α-min-cuts. Use union bound (and the latter fact) to show that for any integer $\alpha \ge 1$, with probability at least $1 - 1/n^2$, for all sets S where $\alpha k \le |E(S, \overline{S})| \le 2\alpha k$,

$$w_H(S, \overline{S}) \ge (1-\epsilon)|E(S, \overline{S})|.$$

d) Use another application of union bound to show that with probability at least $1 - 1/n$, for all sets $S \subset V$

$$w_H(S, \overline{S}) \ge (1-\epsilon)|E(S, \overline{S})|.$$

4) In lecture 4 we discussed the pairwise independent hash functions. We say that for a prime p we can generate a pairwise independent hash functions by choosing a, b independently from the interval $\{1, \ldots, p-1\}$ and using $ax + b$ as a random number. Suppose we generate t pseudo random numbers this way, $r_1, \ldots, r_t$ where $r_i = ai + b$. We want to say this set is far from being mutually independent. Consider the set $S = \{p/2, \ldots, p-1\}$ which has half of all elements. Prove that with probability at least $\Omega(1/t)$ none of the pseudo-random-numbers are in S. Note that if we had mutual independence this would have been $1/2^t$.

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?