CSE 521: Design and Analysis of Algorithms **Fall 2022**

Problem Set 1

Deadline: Oct 16 (at 11:59 PM) in gradescope

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. 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 TAs if you have any questions or would like any clarifications about the problems.
- Please upload your solutions to Gradescope.

In solving these assignments and any future assignment, feel free to use these approximations:

$$1-x \approx e^{-x}, \qquad \sqrt{1-x} \approx 1-x/2, \qquad n! \approx (n/e)^n, \qquad \left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{en}{k}\right)^k$$

1) Consider the following variant of Karger's algorithm: For $i = 1 \to n-4$, choose a uniformly random edge and contract. When we get to a graph with 4 (super) nodes output a uniformly random cut (note that a graph with 4 nodes has 7 many cuts).

   a) In this part implement the above algorithm and output the size of the mincut and **the probability** that it returns **a min-cut** of G within 0.01 error. Note that this probability may be much larger than what you can theoretically prove.

      I will 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. For example, given the following input you should output 1 (size of the mincut) and 0.23. This graph has 5 edges

```
1  3
3  4
4  6
6  3
6  7
```

      and nodes have labels $1, 3, 4, 6, 7$. It has a two minimum cuts $(\{1\}, \overline{\{1\}})$ and $(\{7\}, \overline{\{7\}})$ of size 1 so the probability that the algorithms finds one of these cuts is $8/35 \approx 0.23$.

      For each input file you should output the size of the mincut together with probability that algorithm returns a mincut. Please upload **your code** and **its output** for each input file to gradescope.

b) Suppose that the min cut of G is k and let $(S^*, \overline{S^*})$ be a cut of size $|E(S^*, \overline{S^*})| = 2k$. Prove that this algorithm outputs $(S^*, \overline{S^*})$ with probability at least $\Omega(1/n^4)$.

2) Given a graph $G = (V, E)$ such that $d(v) \geq k$ for all v. Design a (randomized) polynomial time algorithm that outputs a set S of size $|S| \leq O(\frac{n}{k} \log k)$ such that every vertex $v \in \overline{S}$ has at least one neighbor in S. Recall that u is a neighbor of v if $(u, v) \in E$.

3) Consider a sequence $p_1, \ldots, p_n$ of n distinct numbers. An increasing subsequence is a sequence $a_1 < a_2 < \cdots < a_k$ (for some $k \geq 1$) such that $p_{a_1} \leq \cdots \leq p_{a_k}$. For example, the longest increasing subsequence in $1, 3, 2, 4$ has length 3.

   (a) Let $f(n)$ be the expected length of longest increasing subsequence of n distinct numbers. Prove that $f(n) \geq f(n-1)$ for all $n \geq 2$.

   (b) Suppose that $p_n = i$. Prove that $f(n)$ conditioned on $p_n = i$ is at least $1 + f(i-1)$.

   (c) Write a recurrence relation for $f(n)$ and solve it to show that $f(n) \geq \sqrt{n}/c$ for some constant $c > 0$ (I can prove it for $c = 1$).

4) 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 $1 \leq i \leq n$ and for all $x_1, \ldots, x_n$ and $\tilde{x}_i$,*

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

   then

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

5) In the maximum cut problem we are given a graph $G = (V, E)$ we want to find $\max_{(S,\overline{S})} |E(S, \overline{S})|$. Unlike the mincut problem, this problem is NP-complete so we don't expect to find the optimum solution efficiently. Instead, we want to find an approximate solution.

   a) Show that the optimum solution of this problem is always at most $|E|$. Can you construct a graph G with $m = |E|$ edges (for any $m > 0$) such that the optimum of maxcut is equal to $|E|$?

   b) Design a randomized (polynomial-time) algorithm that outputs a (random) cut $(S, \overline{S})$ such that

   $$\mathbb{E}\left[|E(S, \overline{S})|\right] \geq |E|/2.$$

   So, such an algorithm gives a 2-approximation for max-cut (in expectation).

   c) Given a family $\mathcal{H}$ of pairwise independent of hash functions, design a randomize (polynomial-time) algorithm that uses only $O(\log n)$-many random bits and returns a cut $(S, \overline{S})$ such that

   $$\mathbb{E}\left[|E(S, \overline{S})|\right] \geq |E|/2.$$

   d) Turn the algorithm in part (c) to a deterministic algorithm (that uses no randomization).

6) **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?