CS5234: Combinatorial and Graph Algorithms

Problem Set 3

Due: September 6th, 6:30pm

Instructions.

- Start each problem on a separate page.
- Make sure your name is on each sheet of paper (and legible).
- Staple the pages together, and hand it in before class starts, or submit it on LumiNUS in the file area. Alternatively, if you submit it late, you can put it in the envelope next to my office door (and send me an e-mail).

Remember, that when a question asks for an algorithm, you should:

- First, give an overview of your answer. Think of this as the executive summary.
- Second, describe your algorithm in English, giving pseudocode if helpful.
- Third, give an example showing how your algorithm works. Draw a picture.

You may then give a proof of correctness, or explanation, of why your algorithm is correct, an analysis of the running time, and/or an analysis of the approximation ratio, depending on what the question is asking for.

Advice. Start the problem set early—some questions take time. Come talk to me about the questions. (Different students have different questions. Some have questions about how to write a good proof. Others need pointers of designing an algorithm. Still others want to understand the material from lecture more deeply before applying it to the problem sets.) I'm here for you to talk to.

Collaboration Policy. The submitted solution must be your own unique work. You may discuss your high-level approach and strategy with others, but you must then: (i) destroy any notes; (ii) spend 30 minutes on facebook or some other non-technical activity; (iii) write up the solution on your own; (iv) list all your collaborators. Similarly, you may use the internet to learn basic material, but do not search for answers to the problem set questions. You may not use any solutions that you find elsewhere, e.g. on the internet. Any similarity to other students' submissions will be treated as cheating.

Exercises and Review (*Do not submit.*)

Exercise 1. Chernoff Bounds are a closely related tool to Hoeffding Bounds. Here are two standard Chernoff Bounds: Given independent random variables $x_1, x_2, \ldots, x_n$ where each $x_i \in [0,1]$, let $X = \sum_{i=1}^{N} x_i$ and $\mu = \mathrm{E}[X]$. Choose $0 \leq \delta \leq 1$.

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\mu\delta^2/3}$$
$$\Pr[X \leq (1-\delta)\mu] \leq e^{-\mu\delta^2/2}$$

Assume you have independent random variables $y_1, y_2, \ldots, y_n$ where each $y_i \in [0, s]$ for some fixed constant s. Let $Y = \sum_{i=1}^{n} y_i$ and $\mu = \mathrm{E}[Y]$. Prove that for all $0 \leq \delta \leq 1$:

$$\Pr[Y \geq (1+\delta)\mu] \leq e^{-\mu\delta^2/(3s)}$$
$$\Pr[Y \leq (1-\delta)\mu] \leq e^{-\mu\delta^2/(2s)}$$

Exercise 2. Consider the following algorithm for estimating the number of edges in a connected graph $G = (V, E)$: Let x_i be the random variable representing the ith pair (u, v) selected, where

Algorithm 1: Edges$(G = (V, E), n, s)$

```
1 sum = 0
2 repeat s times
3     Choose a random u ∈ [1, n], v ∈ [1, n].
4     if there is an edge (u, v) ∈ E then sum = sum + 1
5 return (sum/s)·C(n, 2).
```

$x_i = 1$ if $(u, v) \in E$. Let $X = \sum_{i=1}^{s} x_i$. Notice that $\mathrm{E}[x_i] = m/\binom{n}{2}$ (where m is the actual number of edges in the graph), and $\mathrm{E}[X] = sm/\binom{n}{2}$. What happens if you try to apply a Chernoff Bound or a Hoeffding Bound to show that the result is a good estimate of the number of edges in the graph? Think about different types of graphs, i.e., both dense and sparse graphs.

Standard Problems (to be submitted)

Problem 1. Chanterelles.

Dr. Pac is an entrepreneur starting a new business culturing a new strain of chanterelles, a valuable type of mushroom. Her office is overflowing with trays full of mushrooms, as she attempts to determine the best conditions in which to grow mushrooms.

There are two key parameters that affect the growth: (1) humidity and (2) temperature. Unfortunately, every type of mushroom is a little different. You can assume (for the purpose of this problem) that there is some range of humidities $[H_1, H_2]$ and some range of temperatures $[T_1, T_2]$ such that Dr. Pac's mushrooms will grow only if the humidity *and* temperature are both within these ranges.

The goal of this problem is to help Dr. Pac determine the best range of temperatures and humidities for her mushrooms. To do this, you will design an experimental protocol (otherwise known as a "sampling algorithm") that examines various humidity/temperature pairs (h, t). Your algorithm may optionally choose any given (h, t) value; then Dr. Pac will run the experiment and inform your algorithm whether or not the mushrooms grew well.

Assume for simplicity that the decision is binary: either the mushrooms grow well or they grow poorly. Each value of h or t can range from 0 to n. (If you choose, you may assume that h and t are integers, but it may be easier to treat them as real numbers as it simplifies boundary conditions!)

When your algorithm completes, it should output two ranges: (h_1, h_2) and (t_1, t_2). Your goal is that the ranges output should be as close as possible to the real (H_1, H_2) and (T_1, T_2) ranges in which the mushroom grows well. More specifically, for some error parameter ϵ, your algorithm should guarantee:

1. *In range:* $H_1 \leq h_1 \leq h_2 \leq H_2$.
2. *In range:* $T_1 \leq t_1 \leq t_2 \leq T_2$.
3. *Limited error:* $(H_2 - H_1)(T_2 - T_1) - (h_2 - h_1)(t_2 - t_1) \leq \epsilon n^2$.

The first two conditions ensure that the mushrooms will grow in the range produced by your algorithm. The third condition defines an error metric: the error is equal to the number of (h, t) points where the mushrooms will grow, but that are not included in your ranges. As an example, imagine the mushrooms will grow at humidity levels $(10, 20)$ and temperature levels $(20, 25)$. Then your algorithm might return humidity levels $(12, 16)$ and temperature levels $(21, 24)$, leading to an error of $50 - 12 = 38$.

Throughout the problem, you may assume that $(H_2 - H_1)(T_2 - T_1)$ is at least $\epsilon n^2/2$. If it were not, then you could trivially return an empty set of ranges!

Your algorithm should return a correct answer with probability at least $2/3$.

Continued on the next page.

Problem 1.a. For a given error parameter ϵ and temperature and humidity ranges from 1 to n, give a sampling algorithm that solves this problem. You may assume that there is an unknown set of ranges (H_1, H_2) and (T_1, T_2) such that mushrooms will grow well at parameters (h, t) if and only if $H_1 \leq h \leq H_2$ and $T_1, \leq t \leq T_2$.

Problem 1.b. Prove that your algorithm is correct.

Problem 1.c. What is the query complexity of your algorithm?

Continued on the next page.

Alas, a problem! When Dr. Pac tries to use your protocol, she discovers a problem: sometimes, just by random chance, mushrooms still grow well even outside the optimal temperature/humidity range. (You can assume that within the temperature/humidity range, mushrooms *always* grow well.)

In fact, for some value of δ, when Dr. Pac runs an experiment, she observes the following: if $H_1 \leq h \leq H_2$ and $T_1 \leq t \leq T_2$, then the mushrooms always grow well; otherwise, the mushrooms grow well with probability δ. Assume that δ is smaller than ϵ, e.g., $\delta < \epsilon/48$ (for whatever constant value you prefer; 48 is only one possible example).

Dr Pac suggests the following experimental protocol:

- Sample s values (h, t) uniformly at random from the range $(0, n) \times (0, n)$.
- Let S be the set of (h, t) values in which the mushrooms grow well.
- Let H be the set of humidities in S. Sort the set H. Delete from H the $(\epsilon s/8) - 1$ points with the largest humidity and the $(\epsilon s/8) - 1$ points with the smallest humidity. Let h_1 be the point in H with the minimum humidity (among the remaining points). Let h_2 be the point in H with the maximum humidity (among the remaining points).
- Let T be the set of temperatures in S. Sort the set T. Delete from T the $(\epsilon s/8) - 1$ points with the largest temperatures and the $(\epsilon s/8) - 1$ points with the smallest temperatures. Let t_1 be the point in T with the minimum temperature (among the remaining points). Let t_2 be the point in T with the maximum temperature (among the remaining points).
- Return the ranges $(h_1, t_1), (h_2, t_2)$.

To solve the following problems, you may find that the Chernoff Bounds described in Exercise 1 are useful.

Problem 1.d. Prove that with probability at least 5/6, the ranges $(h_1, t_1) \times (h_2, t_2)$ are completely contained inside the ranges $(H_1, T_2) \times (H_2, T_2)$. (Hint: use Markov's Inequality.)

Problem 1.e. What is the query complexity of the algorithm? (That is, what is the value of s?). You may want to solve the next part first to determine the needed value of s.

Problem 1.f. Prove that with probability at least 2/3, the algorithm returns a correct answer (as defined earlier).