Homework 2 (due 3/31)

DS-563 / CD-543 @ Boston University

Spring 2023

Before you start...

Collaboration policy: You may verbally collaborate on required homework problems, however, you must write your solutions independently. If you choose to collaborate on a problem, you are allowed to discuss it with at most 4 other students currently enrolled in the class.

The header of each assignment you submit must include the field "Collaborators:" with the names of the students with whom you have had discussions concerning your solutions. A failure to list collaborators may result in credit deduction.

You may use external resources such as textbooks, lecture notes, and videos to supplement your general understanding of the course topics. You may use references such as books and online resources for well known facts. However, you must always cite the source.

You may **not** look up answers to a homework assignment in the published literature or on the web. You may **not** share written work with anyone else.

Submitting: Solutions should be submitted via Gradescope (entry code: 4VYBJ6). You are allowed to submit your solutions both in handwriting or typed. If you decide to hand-write your solutions, make sure they are as readable as possible. If you decide to submit a typed version, we suggest using LaTeX.

Grading: Whenever we ask for an algorithm (or bound), you may receive partial credit if the algorithm is not sufficiently efficient (or the bound is not sufficiently tight).

Questions

First five questions are 33 points each. Submit solutions to *only three* of them. If you submit more, you may receive credit for an arbitrary subset of them. Last question is 10 points.

1. Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be arbitrary discrete distributions on $[n]$. For each $i \in [n]$, let p_i and q_i be the probabilities of drawing i from $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. Let $p = (p_1, \ldots, p_n)$ and $q = (q_1, \ldots, q_n)$. Hence, p and q are vectors in $\mathbb{R}^n$.

   For any subset S of $[n]$, we write $p(S)$ and $q(S)$ to denote the total probabilities of elements in S according to $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. That is, for any such S, $p(S) = \sum_{i \in S} p_i$ and $q(S) = \sum_{i \in S} q_i$.

   The total variation distance between $\mathcal{D}_1$ and $\mathcal{D}_2$ is defined as

   $$\mathrm{d_{TV}}(\mathcal{D}_1, \mathcal{D}_2) = \max_{S \subseteq [n]} |p(S) - q(S)|$$

Prove that:

(a) $\mathrm{d_{TV}}(\mathcal{D}_1, \mathcal{D}_2) = \max_{S \subseteq [n]} (p(S) - q(S))$
(b) $\mathrm{d_{TV}}(\mathcal{D}_1, \mathcal{D}_2) = -\min_{S \subseteq [n]} (p(S) - q(S))$
(c) $\mathrm{d_{TV}}(\mathcal{D}_1, \mathcal{D}_2) = \frac{1}{2}\|p - q\|_1$

2. Design a streaming algorithm that computes a large matching. Your input is a sequence of edges, each consisting of two vertex identifiers. Vertex identifiers are single words. No deletions are allowed. Your algorithm should compute a matching of size at least half the maximum matching size and should use $O(n)$ space, where n is the number of vertices.

3. Let $G = (V, E)$ be a weighted graph that is presented as a stream edge by edge (with no deletions). Compared to the previous question, each edge additionally has an associated positive integer, which describes its weight. Let $w_1 < w_2 < \cdots < w_k$ be possible weights of edges. We want to compute a matching in which the total weight of edges is at least $1/4$ times the maximum possible.

   We write E_i to denote the subset of edges of weight w_i and $E_{\geq i} = \bigcup_{j=i}^{k} E_j$ to denote the subset of edges of weight at least w_i. Let $\mathrm{MCM}(E')$ for $E' \subseteq E$ be the size of the maximum cardinality matching on E', i.e., the maximum number of edges in a matching selected from E' (note that we are ignoring here edge weights!).

   Consider the following streaming algorithm:

   - Ignoring weights of edges, for each $i \in [k]$, independently find a matching M_i of edges in $E_{\geq i}$ of size at least $\frac{1}{2}\,\mathrm{MCM}(E_{\geq i})$. This can be achieved, using the algorithm from the previous part of the homework with total space $O(kn)$.
   - $M \leftarrow \emptyset$
   - For $j = k, k-1, \ldots, 1$: add to M all edges in M_j that do not share an endpoint with any edge in M.
   - output M

   Prove the following, where M is the final matching produced by the algorithm:

   (a) For all $i \in [k]$, $|M \cap E_{\geq i}| \geq \frac{1}{4}\,\mathrm{MCM}(E_{\geq i})$.
   Hint: What is the size of M_i? Consider the iteration in which edges in M_i are being added to M (whenever possible). How many edges in M_i can a single edge already in M block from being added?

   (b) Let $M_\star$ be the matching of the maximum total weight of edges. Show a mapping from edges in $M_\star$ to edges in M such that at most four edges in $M_\star$ are mapped to the same edge in M and every edge is mapped to an edge of the same or higher weight.
   Hint: Construct the mapping by induction. First map edges in $M_\star \cap E_k$ to $M \cap E_{\geq k}$, then map edges in $M_\star \cap E_{k-1}$ to $M \cap E_{\geq k-1}$, then map edges in $M_\star \cap E_{k-2}$ to $M \cap E_{\geq k-2}$, and so on. Can you get stuck at some point by not being able to map at most 4 edges in $M_\star \cap E_{\geq i}$ to each edge in $M \cap E_{\geq i}$?

   (c) Show that the total weight of $M_\star$ is at most 4 times the total weight of M.

   (d) Explain why the total space used by the algorithm is $O(kn)$.

Note 1: Suppose that all weights are integers in $\{1, \ldots, W\}$. Then a $4(1+\epsilon)$ approximation can be computed in $O(\epsilon^{-1} n \log W)$ space by rounding all weights to powers of $(1+\epsilon)$.

Note 2: This algorithm was discovered by two undergraduate students after a line of research that had improved approximation constants from 6 to 4.911. More details on this after the homework is due. The bottom line is "everyone can contribute."

4. Recall the definition of the k–center clustering problem. For a given (multi)set S of points the goal is to find a set Q of at most k points such that

$$\max_{x \in S} \min_{q \in Q} \operatorname{dist}(x, q)$$

is as small as possible.

Let $\operatorname{opt}(S)$ be the cost of the optimal solution for this problem. well–known simple greedy algorithm[1] for this problem computes a 2–approximation (i.e., finds a solution of cost at most $2\operatorname{opt}(S)$) that is a *subset* of the input set. Let us denote the output of this algorithm using $\mathcal{A}(S)$.

Now, let your input (multi)set of points be $S = S_1 \cup S_2 \cup \ldots \cup S_k$. Consider the following algorithm that processes each S_i independently before combining the results of the computation:

- For each i, compute $Q_i = \mathcal{A}(S_i)$.
- Return $\mathcal{A}(\bigcup_{i=1}^{k} Q_i)$.

Prove that this algorithm produces a 4–approximation, i.e., finds a solution of cost at most $4\operatorname{opt}(S)$.

(Optional, no credit) Is this approximation factor optimal? Prove a better approximation guarantee or find a set in an arbitrary metric space on which this algorithm fails to produce an approximation factor better than 4.

5. In class, we said that sampling $O(1/\epsilon^2)$ random elements from a stream and returning their median gives an approximate median as well with high constant probability. Furthermore, one can take an approximate median of the sample and also achieve an approximate median of the input set. This implies that one can run the solution that we saw in class on a random subset of the stream and compute an approximate median with high constant probability, using only $O\left(\epsilon^{-1} \log^2(1/\epsilon)\right)$ words of space, assuming that each element can be stored using $O(1)$ words of space.

Let us now prove that this is the case. Assume the following constraints, which do not impact our application:

- all elements of the stream are distinct,
- $\epsilon \in (0, 1/10)$,
- n, the length of the stream, is at least $1000\epsilon^{-2}$.

Let $s_1 < s_2 < \cdots < s_n$ be a sorted version of the input stream, which we denote S. Suppose that you sample $\lceil C \cdot \epsilon^{-2} \rceil$ elements from S with replacement, for some sufficiently large constant C. Your task:

(a) Let $a = \lfloor \frac{n}{2}(1+\epsilon) \rfloor$. Prove that the fraction of samples that land in $\{s_1, s_2, \ldots, s_a\}$ is at least $\frac{1}{2}(1+\frac{\epsilon}{2})$ with probability 99/100.

[1] It's described for instance on the Wikipedia page for k–center clustering.

(b) Let $b = \lceil \frac{n}{2}(1-\epsilon) \rceil$. Prove that the fraction of samples that land in $\{s_b, s_{b+1}, \ldots, s_n\}$ is at least $\frac{1}{2}(1+\frac{\epsilon}{2})$ with probability 99/100.

(c) Conclude that an $(\epsilon/2)$–approximate median of the sample is an ϵ–approximate median of S with probability $98/100$.

Reminder: Recall that an element of a set S is its ϵ–approximate median if:

- it is greater than or equal to at least $\frac{|S|}{2}(1-\epsilon)$ elements of S
- and it is less than or equal to at least $\frac{|S|}{2}(1-\epsilon)$ elements of S

The question above was updated with fixed constants for simplicity. Previously, it said "Conclude that a $(c_1\epsilon)$–approximate median of the sample is a $(c_2\epsilon)$–approximate median of S with probability $98/100$ for some positive constants c_1 and c_2." Feel free to prove this version instead of the above one.

6. How much time (approximately) did you spend on this homework? Was is too easy/too hard?