CS 466/666: Algorithm Design and Analysis	**University of Waterloo: Fall 2023**

Homework 4 Solutions

December 15, 2023

Problem 1. Recall the notion of a *spanner* from Lecture 15, namely, subgraphs that preserve the distances between pairs of vertices up to a *multiplicative* approximation. In this question, we consider the same problem, but this time, with an *additive* approximation guarantee.

We say that a subgraph $H = (V, E_H)$ of a graph $G = (V, E)$ is a **+2-additive spanner** iff for every vertices $u, v \in V$,

$$dist_G(u, v) \leqslant dist_H(u, v) \leqslant dist_G(u, v) + 2.$$

In this question, we will show that every undirected (unweighted) graph G admits a +2-additive spanner with $O(n\sqrt{n} \cdot \log n)$ edges.

(a) Suppose we sample each vertex in G independently and with probability $(10 \log n)/\sqrt{n}$ in a set S. Prove that with high probability every vertex with degree at least $\sqrt{n}$ has at least one neighbor in S. **(10 points)**

Solution. For any vertex $v \in V$,

$$\Pr(\text{no neighbor of } v \text{ is in } S) = \left(1 - \frac{10 \log n}{\sqrt{n}}\right)^{\deg(v)} \leqslant \exp\left(-10 \log n \cdot \frac{\deg(v)}{\sqrt{n}}\right).$$

Thus, if $\deg(v) \geqslant \sqrt{n}$, we obtain that

$$\Pr(\text{no neighbor of } v \text{ is in } S) \leqslant \exp(-10 \log n) \leqslant n^{-10}.$$

A union bound over at most n vertices proves the claim.

(b) Let S be a set as chosen in part (a). Let H be a subgraph of G that contains a BFS tree from every vertex in S, plus the set of all edges on vertices with degree $< \sqrt{n}$. Prove that H is a +2-spanner of G with $O(n\sqrt{n} \log n)$ edges with high probability. **(15 points)**

Solution. Each BFS tree has $n - 1$ edges and each vertex of degree $\leqslant \sqrt{n}$ contributes at most $\sqrt{n}$ edges to H, hence the bound on the size of H follows immediately.

We now prove H is a +2-spanner assuming the high probability event of part (a) happens. Fix any two vertices s and t and let P be a shortest path from s to t in G. If P contains no vertex of degree $\geqslant \sqrt{n}$, then all edges of P is in H also, hence making

$$dist_H(s, t) = dist_G(s, t).$$

Otherwise, let u be the vertex of degree $\geqslant \sqrt{n}$ on the path P (from s). We have the sub-path $s \to u$ in H. Moreover, by part (a), there is a neighbor $v \in N(u)$ of u that is in S. By the BFS tree we stored in H, we have the shortest path from v to t in H. Thus,

$$dist_H(s, t) \leqslant dist_H(s, u) + 1 + dist_H(v, t) = dist_G(s, u) + 1 + dist_G(v, t),$$

where the inequality is by triangle inequality (path s to t is $s \to u, v \to t$) and the equality is because we have the shortest path in G from s to u, as well as from v to t in H. Finally, we have that

$$dist_G(v, t) \leqslant 1 + dist_G(u, t)$$

again, by triangle inequality because we can from v to u and then from u to t in G (since (u, v) is an edge). As such,

$$dist_H(s,t) \leqslant 2 + dist_G(s,u) + dist_G(u,t) = 2 + dist_G(s,t)$$

as the shortest path from s to t consists of shortest path from s to u and from u to t (since u was on the shortest path). This proves H is a +2-spanner.

Problem 2. Suppose we have two players Alice and Bob, who have strings $x \in \{0,1\}^n$ and $y \in \{0,1\}^n$, respectively. The goal for us is to decide if Alice and Bob have the same string or not, i.e., is $x = y$ or not. The problem is that Alice and Bob cannot talk to each other and they can only toss their own random coins (as in there is no shared source of randomness between them). Design a solution where both Alice and Bob only send a single message of size

$$O(\sqrt{n}) \cdot \log^{O(1)}(n)$$

bits each to us *directly* and based on their messages, we can decide, with high probability, if $x = y$ or not.

(25 points)

Solution. Let P be a $2\sqrt{n} \times 2\sqrt{n}$ matrix filled up (in some arbitrary order) with the smallest $4n$ primes. This way, $P_{i,j}$ corresponds to a distinct prime number with $\Theta(\log n)$ bit-complexity for every $i, j \in [2\sqrt{n}]$. Notice that P is fixed and known to all parties. The protocol is as follows:

1. Alice samples $i \in [2\sqrt{n}]$ uniformly at random and privately and sends i along with $2\sqrt{n}$ numbers corresponding to taking $x \mod P_{i,j'}$ for every $j' \in [2\sqrt{n}]$ (we treat x as a number from 0 to $2^n - 1$).
2. Bob samples $j \in [2\sqrt{n}]$ uniformly at random and privately and sends j along with $2\sqrt{n}$ numbers corresponding to taking $y \mod P_{i',j}$ for every $i' \in [2\sqrt{n}]$ (we treat y as a number from 0 to $2^n - 1$).
3. Given messages of Alice and Bob, we find the *unique* prime $P_{i,j}$ where both players have send $x \mod P_{i,j}$ and $y \mod P_{i,j}$ and check if the two numbers are equal[1]. If so, we consider $x = y$ and otherwise, consider them different.

The proof of correctness is as follows. Firstly, $P_{i,j}$ is chosen uniformly at random from the first $4n$ primes. On the other hand, assuming $x \geqslant y$, $x - y$ is a number between 0 and $2^n - 1$ and thus it can have at most n prime factors. This means, the probability that $P_{i,j}$ is a prime factor of $x - y$ is at most $1/4$. Hence, the probability that the protocol outputs a wrong answer is at most $1/4$ and it uses $O(\sqrt{n} \cdot \log n)$ communication.

To make the protocol succeed with high probability, we simply repeat it $O(\log n)$ times independently and return the majority answer. By the majority trick we covered multiple times in the course, the new protocol succeeds with high probability while using $O(\sqrt{n} \cdot (\log n)^2)$ bits in total.

Problem 3. Consider the following linear program for the set cover problem with sets $S_1, \ldots, S_m$ from the universe $[n]$, which we studied in Lecture 10:

$$\begin{aligned} \min_{x \in \mathbb{R}^m} \quad & \sum_{i=1}^{m} x_i \\ \text{subject to} \quad & \sum_{S_i \ni e} x_i \geqslant 1 \quad \forall e \in [n] \\ & x_i \geqslant 0 \quad \forall i \in [m]. \end{aligned}$$

We use the MWU technique to design a $(1 + \varepsilon)$-approximation algorithm for this LP for a given $\varepsilon > 0$.

[1]Notice that Alice is sending an entire row of P and Bob is sending an entire column, so there is always a unique prime they both consider.

(a) For every element $e \in [n]$, maintain a weight w_e and let $W := \sum_{e\in[n]} w_e$. Consider this oracle LP:

$$\begin{aligned} \min_{x\in\mathbb{R}^m} \quad & \sum_{i=1}^{m} x_i \\ \text{subject to} \quad & \sum_{e\in[n]} w_e \cdot \sum_{S_i \ni e} x_i \geqslant W \\ & x_i \geqslant 0 \quad \forall i \in [m]. \end{aligned}$$

Design an algorithm for finding the optimum solution to this LP and prove that the value of this solution is always upper bounded by that of the original LP for set cover. **(10 points)**

Solution. We can re-write the LHS of the constraint as

$$\sum_{e\in[n]} w_e \cdot \sum_{S_i\ni e} x_i = \sum_{i=1}^{m} \underbrace{\sum_{e\in S_i} w_e}_{:=w(S_i)} = \sum_{i=1}^{m} w(S_i).$$

The optimum solution for this LP is to pick the set S_i with the smallest value of $w(S_i)$ and let $x_i = W/w(S_i)$ and $x_j = 0$ for all $i \neq j$. Since the constraint of the oracle LP is a convex combination of the constraints of the original LP, any feasible solution for the original LP is also feasible for this one. Thus, the optimum solution of the oracle LP, namely, $W/w(S_i)$ is at most as large as the optimum of the original LP.

(b) Consider the following update rule in the MWU for a given solution $x^{(t)}$ to the oracle LP of part (a) for weights $w^{(t)}$ at iteration $t \geqslant 1$. For any element $e \in [n]$, define $x^{(t)}_e := \sum_{S_i\ni e} x^{(t)}_i$ and update:

$$w^{(t+1)}_e \leftarrow (1 - \eta \cdot x^{(t)}_e) \cdot w^{(t)}_e,$$

for some $\eta > 0$ that you will need to choose later. Prove the following two equations after running the MWU algorithm with the above update rules for T iterations:

$$\begin{aligned} W^{(T+1)} &\leqslant \exp\left(-\eta \cdot T + \ln n\right) \\ w^{(T+1)}_e &\geqslant \exp\left(-\eta \cdot \sum_{t=1}^{T} x^{(t)}_e - \eta^2 \cdot \sum_{t=1}^{T} {x^{(t)}_e}^2\right). \end{aligned}$$

Note that you need to pick η properly to be able to prove the above bounds. **(20 points)**

Solution. We prove each equation separately.

The first equation. For every $t \geqslant 1$, we have,

$$\begin{aligned} W^{(t+1)} &= \sum_e w^{(t+1)}_e && \text{(by the definition of } W^{(t+1)}\text{)} \\ &= \sum_e (1 - \eta \cdot x^{(t)}_e) \cdot w^{(t)}_e && \text{(by the update rule of the algorithm)} \\ &= \sum_e w^{(t)}_e - \eta \cdot \left(\sum_e w^{(t)}_e \cdot x^{(t)}_e\right) && \text{(by rearranging the terms)} \\ &= W^{(t)} - \eta \cdot \left(\sum_e w^{(t)}_e \cdot x^{(t)}_e\right) && \text{(by the definition of } W^{(t)}\text{)} \\ &\leqslant W^{(t)} - \eta \cdot W^{(t)} && \text{(by the \underline{feasibility} of } x^{(t)} \text{ in the oracle LP)} \\ &= (1 - \eta) \cdot W^{(t)}. \end{aligned}$$

As a result, and since $W^{(1)} = n$, we obtain that after the T-th iteration:

$$W^{(T+1)} \leqslant (1-\eta)^T \cdot n \leqslant \exp\left(-\eta \cdot T + \ln n\right),$$

where we used the inequality $1 - x \leqslant e^{-x}$ for all $x \in (0, 1)$. Thus, for this part of the argument, we need η to be in $(0, 1)$.

The second equation. For every $e \in [n]$, we have,

$$\begin{aligned} w_e^{(T+1)} &= \prod_{t=1}^{T} \left(1 - \eta \cdot x_e^{(t)}\right) && \text{(by the update rule and since } w_e^{(1)} = 1\text{)} \\ &\geqslant \prod_{t=1}^{T} \exp\left(-\eta \cdot x_e^{(t)} - \eta^2 \cdot \left(x_e^{(t)}\right)^2\right), \end{aligned}$$

as long as

$$\eta < \frac{1}{2\rho_e} \qquad \text{where we define} \qquad \rho_e := \max_{t \geqslant 1} \quad x_e^{(t)}$$

as $1 - x \geqslant e^{-x-x^2}$ for $x \in (0, 1/2)$ derived from the Taylor expansion of $\ln(1-y) = -y - y^2/2 + O(y^3)$. Continuing, we have,

$$w_e^{(T+1)} \geqslant \exp\left(-\eta \cdot \sum_{t=1}^{T} x_e^{(t)} - \eta^2 \cdot \sum_{t=1}^{T} \left(x_e^{(t)}\right)^2\right) \qquad \text{(by using } e^{\alpha} \cdot e^{\beta} = e^{\alpha+\beta}\text{)}$$

concluding the proof.

(c) Use the previous two steps to design a polynomial time algorithm based on MWU that outputs a $(1+\varepsilon)$-approximation to the set cover LP. Remember to both bound the number of iterations of your MWU algorithm as well as the time that it takes to solve the oracle LP in each iteration. **(20 points)**

Solution. We run MWU for T iterations (to be determined soon) by solving the oracle as stated in the first part. Each step of solving the oracle can be done in $O(mn)$ time by computing the "weight" of each set by summing up the weight of all its elements. The updates can also be done in $O(mn)$ time by summing up the x-value of every set that covers a fixed element. Finally, we return

$$x^* := \frac{1}{(1-\varepsilon)} \cdot \sum_{t=1}^{T} x^{(t)}$$

as our final solution.

Suppose there exist an element e which is not covered by x^*. We thus have,

$$x_e^* < 1 \qquad \equiv \qquad \sum_{t=1}^{T} x_e^{(t)} < (1-\varepsilon) \cdot T.$$

Using this bound in the second equation of the previous part, we have,

$$\begin{aligned} w_e^{(T+1)} &\geqslant \exp\left(-\eta \cdot \sum_{t=1}^{T} x_e^{(t)} - \eta^2 \cdot \rho_e \cdot \sum_{t=1}^{T} x_e^{(t)}\right) && \text{(by the definition of } x_e^{(t)} \leqslant \rho_e \text{ for all } t \geqslant 1\text{)} \\ &\geqslant \exp\left(-\eta \cdot \sum_{t=1}^{T} x_e^{(t)} - \frac{\varepsilon}{2} \cdot \eta \cdot \sum_{t=1}^{T} x_e^{(t)}\right) \end{aligned}$$

as long as we set

$$\eta \leqslant \frac{\varepsilon}{2\rho_e}.$$

Continuing bounding the LHS, we get,

$$\begin{aligned}
&= \exp\left(-\eta \cdot (1+\frac{\varepsilon}{2}) \cdot \sum_{t=1}^{T} x_e^{(t)}\right) \\
&\geqslant \exp\left(-\eta \cdot (1+\frac{\varepsilon}{2}) \cdot (1-\varepsilon) \cdot T\right) \quad \text{(by the bound established earlier given } e \text{ is not covered)} \\
&\geqslant \exp\left(-\eta \cdot (1-\frac{\varepsilon}{4}) \cdot T\right). \qquad \text{(as } \varepsilon < 1/2\text{)}
\end{aligned}$$

Given that the weights are all positive and by using the bound on the first equation of the previous part, for the above to happen, we need to have,

$$\exp\left(-\eta \cdot (1-\frac{\varepsilon}{4}) \cdot T\right) \leqslant \exp\left(-\eta \cdot T + \ln n\right) \implies T \leqslant \frac{4\ln n}{\varepsilon \cdot \eta}.$$

Finally, to be able to obtain the above bounds, we can set

$$\eta = \frac{\varepsilon}{2n}$$

and argue that this satisfies all the required inequalities because $\rho_e \leqslant n$ for all $e \in [n]$. This is because $\rho_e = \max_t x_e^{(t)}$ is at most equal to the largest value returned for the oracle LP, which is upper bounded by the original LP, which itself is bounded by n.

All in all, this implies that after $T = O(n \ln n/\varepsilon^2)$ iterations, the resulting solution x^* will be feasible. Moreover, since every $x^{(t)}$ was at most as large as the optimum solution, we obtain that x^* is a $1/(1-\varepsilon)$-approximation which is a $(1+2\varepsilon)$-approximation for $\varepsilon < 1/2$. Re-scaling $\varepsilon \leftarrow \varepsilon/2$ in the above equations concludes the proof. Specifically, we have an algorithm with $O(mn^2 \ln n/\varepsilon^2)$ time for $(1+\varepsilon)$-approximation of the set cover LP.
