CS 580 Algorithm Design, Analysis, And Implementation
Fall 2021 Vassilis Zikas HW 6

Due Monday December 6 at 11:59 p.m.

Important Note: *You are **only allowed one late day** for this assignment i.e. last date to submit the assignment (with late penalty) is 11:59 pm Tuesday, December 7. Any submission made after 11:59 pm Monday, December 6 and before 11:59 pm Tuesday, December 7 will receive a 10% reduction in the grade.* **No submissions will be accepted/graded after 11:59 pm Tuesday, Dec. 7!**

1. (30 points) Consider the following clustering problem known as the k-center problem. The input is an undirected, complete graph $G = (V,E)$ with a distance metric $d(.,.)$ on the vertex set and an integer k. In this problem, the goal is to partition the graph's vertex set into k clusters each of which has a cluster center. We aim to *minimize the maximum distance* of a vertex to its cluster's center, call this distance the cost of the clustering.

   We assume that the distance metric has the following properties: symmetric $\forall i,j\ d(i,j) = d(j,i)$, follows triangle inequality $\forall i,j,k\ d(i,j) \leq d(i,k) + d(k,j)$, and $\forall i, d(i,i) = 0$.

   Consider the following greedy strategy to build a set S of k cluster centers. First, we pick one vertex arbitarily and include it in S. While size of S is less than k, we pick the vertex $v \in V$ that maximizes the following $\min_{j \in S} d(v,j)$, and add it to S. That is, we start with arbitrary vertex v_1. Then we pick the vertex v_2 that has the maximum distance from v_1. Then we pick the vertex v_3 that is the furthest from v_1 and v_2, i.e. that maximizes the minimum distance from v_1 and v_2, and so on. For example, consider the instance of k-center problem given below with $k = 3$ and the distances between any pair of points is the Euclidean distance between them. The optimal solution is $\{1^*, 2^*, 3^*\}$ but the greedy algorithm outputs $\{1,2,3\}$.

   Prove that the above algorithm returns a clustering (i.e. a set of cluster centres) of cost atmost two times the cost of any optimal clustering (optimal is the one that minimizes the maximum distance of any vertex to it's cluster center).

2. (40 points) Suppose that A is an algorithm that outputs k cluster centres such that cost of clustering is at most

1.5 times the optimal cost for any instance of the k-center problem (defined in the previous question). Prove that existence of A implies a polynomial time algorithm for 3-SAT.

Hint: It might be easier to prove that the existence of A implies a polynomial time algorithm for some other known NP-Hard problem. You can now use this to get an algorithm for 3-SAT.

3. (30 points) Consider the MAX-CUT problem defined as follows. The input is an undirected graph $G : (V, E)$ and the goal is to find a cut of maximum size. In other words, the goal is to find a partition of the vertex set into two parts U and $W = V \setminus U$ such that number of edges with one end point in U and other other in W is maximized.

   Consider the following algorithm: for each vertex $v \in V$, include it in U independently with probability $1/2$. Prove that we obtain a randomized $1/2$-approximation algorithm for the MAX-CUT problem.

   Recall that a randomized algorithm for a maximisation problem has approximation ratio $\alpha(n) \leq 1$, if for any instance of size n, the *expected* cost C of the returned solution and the optimal cost C^* satisfy $C \geq C^* \alpha$.

4. (2 points) Have you assigned pages in Gradescope?

5. **Bonus Problem.** (30 points, 10 each) Suppose that you are given an algorithm A that can analyze whether a person has a disease called Senioritis-580 by analyzing their DNA (assume that a person's DNA is a length n string). However, A is not always correct. Consider the following scenarios:

   (a) If a person has Senioritis-580 , then A outputs YES (has Senioritis-580) with probability $4/5$. However, if the person does not have Senioritis-580 , then A always outputs NO (does not have Senioritis-580). Give a polynomial time algorithm that has the following properties. For $p = 1 - 1/poly(n)$ where $poly(n)$ is a polynomial in n, if the person has Senioritis-580 , then it should output YES with probability atleast p. If the person does not have Senioritis-580 , then it should output NO with probability atleast p.

   (b) Now, suppose that if a person has Senioritis-580 , then A outputs YES with probability $q = 4/5$. However, if the person does not have Senioritis-580 , then A outputs NO with probability $q = 4/5$. Give a polynomial time algorithm that has the following properties. For $p = 1 - 1/poly(n)$ where $poly(n)$ is a polynomial in n, if the person has Senioritis-580 , then it should output YES with probability atleast p. If the person does not have Senioritis-580 , then it should output NO with probability atleast p.

   (c) In the previous scenario, prove or disprove: For all values of $q > 1/2$, there exists an algorithm that runs in polynomial time such that it is correct with probability atleast $3/4$ (i.e. $p = 3/4$).

   Assume that A runs in $O(n)$ time. Any polynomial time solution will be accepted for full credit in the above questions.