Spring 2026, CS 583: Approximation Algorithms

Homework 4

Due: 04/30/2026 in Gradescope

- Each homework can be done in a group of size at most two. Only one homework needs to be submitted per group. However, we recommend that each of you think about the problems on your own first.
- Homework needs to be submitted in pdf format on Gradescope. See `https://courses.grainger.illinois.edu/cs374al1/fa2025/hw-pol.html` for more detailed instructions on Gradescope submissions.
- Follow academic integrity policies as laid out in student code. You can consult sources but cite all of them including discussions with other classmates and LLMs. Write in your own words. See the site mentioned in the preceding item for more detailed policies including specific instructions on using LLMs.
- Read through all the problems and think about them and how they relate to what we covered in the lectures. Do at least 3 problems with one problem on each of the following topics (i) cut problems (ii) SDP and (iii) network design.
- Please write clearly and concisely - clarity and brevity will be rewarded. Refer to known facts as necessary.

Problem 1. Consider an instance of the non-uniform Sparsest Cut problem on an undirected *cycle/ring* $G = (V, E)$ with edge capacities $c : E \to \mathbb{R}_+$. Note that in addition to the graph we are given demand pairs $(s_1, t_1), \ldots, (s_k, t_k)$ with non-negative demand values $D_1, \ldots, D_k$.

- Describe a simple combinatorial algorithm to find a sparsest cut in a cycle.
- Prove that any ring metric isometrically embeds into ℓ_1. And use this argue that the LP for sparsest cut has integrality gap 1 on cycles.

Problem 2. Given a graph $G = (V, E)$ with edge-weights $c : E \to \mathbb{R}_+$, you wish to partition G into $G_1 = G[V_1], G_2 = G[V_2], G_3 = G[V_3]$ such that $\lfloor |V|/3 \rfloor \leq |V_i| \leq \lceil |V|/3 \rceil$ for $1 \leq i \leq 3$, and the cost of the edges between the partitions is minimized. Using an α-approximation for the sparsest cut problem, give a pseudo-approximation for this problem where you partition the graph into 3 pieces $G[V_1'], G[V_2'], G[V_3']$ such that $|V|/c_2 \leq |V_i'| \leq |V|/c_1$ for some constants $1 < c_1 < c_2$ and the cost of the edges between the partitions is $O(\alpha)\textsc{opt}$. What constants c_1, c_2 can you guarantee? Note that c_1 and c_2 should be *constants*, independent of the graph size.

Problem 3. Problem 7.2 in Williamson-Shmoys book considers Multicut in trees and showed that the integrality gap is 2. Use this and and the result on tree embeddings to prove that the integrality gap of the natural LP for Multicut in general graphs is $O(\log n)$. Note that we already saw this directly but this problem is to illustrate the power of tree embeddings. This is same as Problem 15.5 in Williamson-Shmoys book.

Problem 4. Consider MAX-CUT with the additional constraint that specified pairs of vertices be on the same/opposite sides of the cut. Formally, we are given two sets of pairs of vertices, S_1 and S_2. The pairs in S_1 need to be separated, and those in S_2 need to be on the same side of the cut sought. Under these constraints, the problem is to find a maximum-weight cut.

1. Give an efficient algorithm to check if there is a *feasible* solution.
2. Assuming there is a feasible solution, give a strict quadratic program and vector program relaxation for this problem. Show how the algorithm for MAX-CUT we saw in class can be adapted to this problem while maintaining the same approximation ratio.

Problem 5. Problem 6.6 from the Williamson-Shmoys book. SDP for directed cut.

Problem 6. In the k-MST problem you are given an undirected edge-weighted graph $G = (V, E)$ with edge weights $c : E \to \mathbb{R}_+$ and an integer k. The goal is to find a tree $T = (V_T, E_T)$ in G of smallest edge weight ($\sum_{e \in E_T} c(e)$) such that $|V_T| \geq k$. Recall that in the Steiner tree problem, the input is an edge-weighted graph $G = (V, E)$ and a set of terminals $S \subseteq V$; the goal is to find a tree T of minimum edge-weight that connects (contains) all the terminals S.

Now consider the rooted k-Steiner tree problem. The input consists of an edge-weighted undirected graph $G = (V, E)$, a specified root vertex r and a set $S \subset V$ of terminals. The goal is to find a min-cost tree (V_T, E_T), a sub-graph of G, such that $r \in V_T$ and $|S \cap V_T| \geq k$. It is not hard to see that k-Steiner tree is a further generalization of k-MST but they are equivalent (do you see why?).

- Show that if there is an α-approximation for k-MST then there is an α-approximation for the Steiner tree problem.
- Obtain a randomized $O(\log n)$ approximation for k-Steiner tree problem via probabilistic embedding into tree metrics.

There is a 2-approximation for k-MST problem but it is rather involved.

Problem 7. We have mostly seen edge-weighted network design problems in undirected graphs. As we discussed in lecture, directed network design problems are hard to approximate. Undirected *node-weighted* network design problems fall in between. Here we consider node-weighted Steiner tree problem. The input is a graph $G = (V, E)$ with non-negative node weights $w : V \to \mathbb{R}_+$, and a set of k terminals $S \subseteq V$. The goal is to find a Steiner tree $T = (X, F)$ for the terminals (that is, $S \subseteq X$) with minimum node weight. We will assume that the weight of the terminals is 0 without loss of generality since every terminal has to be included in the tree. We saw that Steiner tree in edge-weighted graphs admits a simple 2-approximation via the MST heuristic, and improved approximations are also known. In contrast the node-weighted problem is harder. This problem will walk you through an approach to approximate it via Set Cover type ideas.

- Describe a reduction from Set Cover on m sets and n elements to node-weighted Steiner tree with $k = n$. Informally argue that an $\alpha(k)$-approximation for node-weighted Steiner tree implies an $\alpha(n)$-approximation for Set Cover. This should convince you that node-weighted Steiner tree does not admit a better than $\ln k$ approximation.

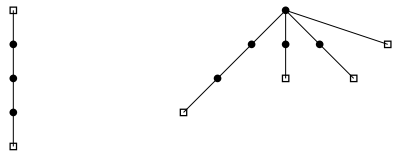

Figure 1: Examples of spiders with $k = 1$ and $k = 4$. Terminals are shown as a box and non-terminals as a circle.

- Given a set of terminals, a spider is a star-like tree with a center vertex c and k legs where each leg is a c-t path for some terminal t (the center itself can be a terminal). Thus, all the leaves are terminals if $k > 1$. If $k = 1$ then we require that the center to be a terminal and in this case the spider is simply a path connecting two terminals. See figure. Argue that any minimal Steiner tree (X, F) contains a collection of node-disjoint spiders that together include all but one terminal from S.
- The density of a spider is the ratio of the total weight of the nodes in the spider divided by the number of terminals in it. Describe a polynomial-time algorithm to find the best density spider.
- Consider the following algorithm. It finds a best density spider and contracts the nodes of the spider to the center which now becomes a terminal, and recurses on the residual instance. Formalize this algorithm and use Set Cover like analysis to argue that it yields a $O(\ln k)$ approximation.

Problem 8. Problem 11.3 from Williamson-Shmoys book.

Problem 9. Problem 11.5 from Williamson-Shmoys book.