Weekplan: Approximation Algorithms II

Inge Li Gørtz

References and Reading

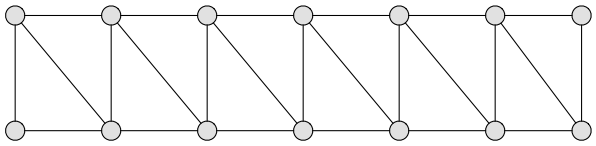
[1] Algorithm Design, Kleinberg and Tardos, Addison-Wesley, section 11.2, 11.3.

[2] A unified approach to approximation algorithms for bottleneck problems, D. S. Hochbaum and D. B. Shmoys, Journal of the ACM, Volume 33 Issue 3, 1986.

[3] The Design of Approximation Algorithms, Williamson and Shmoys, Cambridge Press, section 2.1.

We expect you to read [1] in detail before the lecture. [3] is alternative reading. [2] provides background on the k-center problem.

1 $[w]$ **k-center** Run both k-center algorithms on the example below with $k = 4$. All edges have length 1.



2 The k-supplier problem The *k-supplier* problem is similar to the k-center problem, but the vertices are partitioned into *suppliers* $F \subseteq V$ and *customers* $C \subseteq V$. The goal is to find k suppliers such that the maximum distance from a customer to a supplier is minimized. Give a 3-approximation algorithm for the k-suppliers problem.

3 Metric k-clustering Give an 2-approximation algorithm for the following problem.

Let $G = (V, E)$ be a complete undirected graph with edge costs satisfying the triangle inequality, and let k be a positive integer. The problem is to partition V into sets $V_1, \ldots, V_k$ so as to minimize the costliest edge between two vertices in the same set, i.e., minimize

$$\max_{1 \le i \le k,\, u,v \in V_i} c(u, v).$$

4 Priority k-center Consider the following variant of the k-center problem, where the vertices have priorities: Each vertex have a priority, and we want to find a set of k centers so that the maximum *prioritized* distance of a vertex to its closest center is minimized. That is, the higher value priority a vertex has, the closer it should be to a center.

Formally, in the *prioritized k-center problem* we are given a complete graph $G = (V, E)$ with a cost function on the edges $d : E \to Q^+$ satisfying the triangle inequality, a priority function on vertices: $p : V \to \mathbb{R}^+$, and a positive integer k. The problem is to find a set of centers $C \subseteq V$ with $|C| \le k$ minimizing

$$r(C) = \max_{v \in V} p(v) \cdot d(v, C),$$

where

$$d(v, C) = \min_{u \in C} d(v, u).$$

The following algorithm for the prioritized k-center problem assumes we know the optimal radius r.

Algorithm 1 Prority-Center (G, r)

1: Set $S = V$ and $C = \emptyset$.
2: **while** $S \neq \emptyset$ **do**
3: Select the heaviest vertex $v \in S$ (the vertex with highest priority)
4: Set $C = C \cup \{v\}$
5: Remove all vertices u from S with $p(u) \cdot d(u, v) \leq 2r$ from v.
6: **end while**
7: Return C

4.1 Assume we know the optimum covering radius r. Let C be the set of centers computed by the algorithm $kCenter(G, r)$, let C^* be the set of optimal centers (each vertex is assigned to its closest center).

(a) Consider an iteration of the algorithm and let v be the vertex chosen in this iteration. Let c^* be the center v is assigned to in the optimal solution. Let $z \in S$ be a vertex assigned to c^* in the optimal solution. Show that $p(z) \cdot d(z, v) \leq 2r$.

(b) Show that at most one vertex from each cluster from C^* belongs to C.

4.2 Prove that Algorithm 1 is a 2-approximation algorithm for the prioritized k-center problem (assuming that we know the optimal covering radius r).

5 Representative set of proteins (exercise 2 in [1]) At a lecture in a computational biology conference one of us attended a few years ago, a well-known protein chemist talked about the idea of building a "representative set" for a large collection of protein molecules whose properties we don't understand. The idea would be to intensively study the proteins in the representative set and thereby learn (by inference) about all the proteins in the full collection. To be useful, the representative set must have two properties.

- It should be relatively small, so that it will not be too expensive to study it.
- Every protein in the full collection should be "similar" to some protein in the representative set. (In this way, it truly provides some information about all the proteins.)

More concretely, there is a large set P of proteins. We define similarity on proteins by a distance function d: Given two proteins p and q, it returns a number $d(p, q) \geq 0$. In fact, the function $d(\cdot, \cdot)$ most typically used is the so-called sequence alignment measure. We will assume that the distance function satisfy the triangle inequality. There is a predefined distance cut-off Δ that's specified as part of the input to the problem; two proteins p and q are deemed to be "similar"to one another if and only if $d(p, q) \leq \Delta$. We say that a subset of P is a *representative set* if, for every protein p, there is a protein q in the subset that is similar to it—that is, for which $d(p, q) \leq \Delta$. Our goal is to find a representative set that is as small as possible.

5.1 Give a polynomial-time algorithm that approximates the minimum representative set to within a factor of $O(\log n)$. Specifically, your algorithm should have the following property: If the minimum possible size of a representative set is s^*, your algorithm should return a representative set of size at most $O(s^* \log n)$.

5.2 Note the close similarity between this problem and the Center Selection Problem—a problem for which we considered approximation algorithms in Section 11.2. Why doesn't the algorithm described there solve the current problem?

6 Vertex cover A *vertex cover* in a graph $G = (V, E)$ is a subset of vertices $S \subseteq V$ so that each edge has at least one end in S. In the *cardinality vertex cover problem* the goal is to find a vertex cover of the input graph $G = (V, E)$ of minimum size.

A *matching* in a graph $G = (V, E)$ is subset of edges $M \subseteq E$ so that no two edges of M hare an endpoint. A *maximal matching* is a matching that is maximal under inclusion. That is, adding any edge from E to the maximal matching will cause two edges in M to share an endpoint.

6.1 Show that a maximal matching can be computed in polynomial time.

6.2 Show that the size of a maximal matching in a graph G is a lower bound on the size of the minimum vertex cover in G.

6.3 Give a 2-approximation algorithm for cardinality vertex cover.

6.4 What is the relation between the (cardinality) vertex cover problem and the (cardinality) set cover problem?