Weekplan: Approximation Algorithms II

Inge Li Gørtz

References and Reading


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

![Example Graph](image)

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

\[
\max_{1 \leq i \leq k, u, v \in V_i} c(u, v).
\]

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 \( d : E \to \mathbb{Q}^+ \) 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 \( |C| \leq k \) 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 $S \neq \emptyset$
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 **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.

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

   5.2 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 $G$.

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