CHAPTER 4

Clustering – Definitions and Basic Algorithms

In this chapter, we will initiate our discussion of ***clustering***. Clustering is one of the most fundamental computational tasks but, frustratingly, one of the fuzziest. It can be stated informally as: "Given data, find an interesting structure in the data. Go!"

The fuzziness arises naturally from the requirement that the clustering should be "interesting", which is not a well-defined concept and depends on human perception and hence is impossible to quantify clearly. Similarly, the meaning of "structure" is also open to debate. Nevertheless, clustering is inherent to many computational tasks like learning, searching, and data-mining.

Empirical study of clustering concentrates on trying various measures for the clustering and trying out various algorithms and heuristics to compute these clusterings. See the bibliographical notes in this chapter for some relevant references.

Here we will concentrate on some well-defined clustering tasks, including k-center clustering, k-median clustering, and k-means clustering, and some basic algorithms for these problems.

4.1. Preliminaries

A clustering problem is usually defined by a set of items, and a distance function between the items in this set. While these items might be points in $\mathbb{R}^d$ and the distance function just the regular Euclidean distance, it is sometime beneficial to consider the more abstract setting of a general metric space.

4.1.1. Metric spaces.

Definition 4.1. A ***metric space*** is a pair $(\mathcal{X}, \mathbf{d})$ where $\mathcal{X}$ is a set and $\mathbf{d} : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ is a ***metric*** satisfying the following axioms: (i) $\mathbf{d}_{\mathcal{M}}(x, y) = 0$ if and only if $x = y$, (ii) $\mathbf{d}_{\mathcal{M}}(x, y) = \mathbf{d}_{\mathcal{M}}(y, x)$, and (iii) $\mathbf{d}_{\mathcal{M}}(x, y) + \mathbf{d}_{\mathcal{M}}(y, z) \geq \mathbf{d}_{\mathcal{M}}(x, z)$ (triangle inequality).

For example, $\mathbb{R}^2$ with the regular Euclidean distance is a metric space. In the following, we assume that we are given ***black-box access*** to $\mathbf{d}_{\mathcal{M}}$. Namely, given two points $\mathsf{p}, \mathsf{q} \in \mathcal{X}$, we assume that $\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \mathsf{q})$ can be computed in constant time.

Another standard example for a finite metric space is a graph G with non-negative weights $\omega(\cdot)$ defined on its edges. Let $\mathbf{d}_{\mathsf{G}}(x, y)$ denote the shortest path (under the given weights) between any $x, y \in V(\mathsf{G})$. It is easy to verify that $\mathbf{d}_{\mathsf{G}}(\cdot, \cdot)$ is a metric. In fact, any ***finite metric*** (i.e., a metric defined over a finite set) can be represented by such a weighted graph.

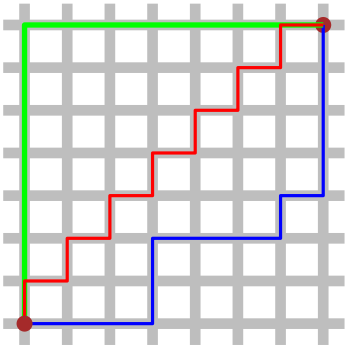
The ***L_p-norm*** defines the distance between two points $\mathsf{p}, \mathsf{q} \in \mathbb{R}^d$ as

$$\left\| \mathsf{p} - \mathsf{q} \right\|_p = \left(\sum_{i=1}^{d} |\mathsf{p}_i - \mathsf{q}_i|^p \right)^{1/p},$$

for $p \geq 1$. The ***L_2-norm*** is the regular Euclidean distance.

The ***L_1-norm***, also known as the ***Manhattan distance*** or ***taxicab distance***, is

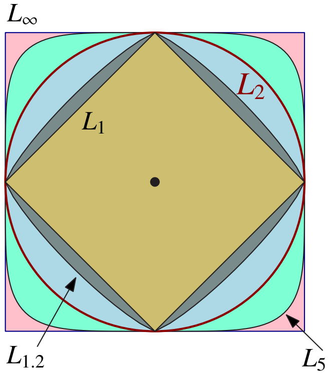
$$\left\|\mathsf{p}-\mathsf{q}\right\|_1 = \sum_{i=1}^{\mathsf{d}} \left|\mathsf{p}_i - \mathsf{q}_i\right|.$$



The L_1-norm distance between two points is the minimum path length that is axis parallel and connects the two points. For a uniform grid, it is the minimum number of grid edges (i.e., blocks in Manhattan) one has to travel between two grid points. In particular, the shortest path between two points is no longer unique; see the picture on the right. Of course, in the L_2-norm the shortest path between two points is the segment connecting the two points.

The ***L_∞-norm*** is

$$\left\|\mathsf{p}-\mathsf{q}\right\|_\infty = \lim_{p\to\infty} \left\|\mathsf{p}-\mathsf{q}\right\|_p = \max_{i=1}^{\mathsf{d}} \left|\mathsf{p}_i - \mathsf{q}_i\right|.$$



The triangle inequality holds for the L_p-norm, for any $p \geq 1$ (it is called the ***Minkowski inequality*** in this case). In particular, L_p is a metric for $p \geq 1$. Specifically, $\mathbb{R}^{\mathsf{d}}$ with any L_p-norm (i.e., $p \geq 1$) is another example of a metric space.

It is useful to consider the different unit balls of L_p for different value of p; see the figure on the right. The figure implies (and one can prove it formally) that for any point $\mathsf{p} \in \mathbb{R}^{\mathsf{d}}$, we have that $\|\mathsf{p}\|_p \leq \|\mathsf{p}\|_q$ if $p > q$.

Lemma 4.2. *For any* $\mathsf{p} \in \mathbb{R}^{\mathsf{d}}$, *we have that* $\|\mathsf{p}\|_1 / \sqrt{\mathsf{d}} \leq \|\mathsf{p}\|_2 \leq \|\mathsf{p}\|_1$.

Proof. Indeed, let $\mathsf{p} = (\mathsf{p}_1, \ldots, \mathsf{p}_{\mathsf{d}})$, and assume that $\mathsf{p}_i \geq 0$, for all i. It is easy to verify that for a constant α, the function $f(x) = x^2 + (\alpha - x)^2$ is minimized when $x = \alpha/2$. As such, setting $\alpha = \|\mathsf{p}\|_1 = \sum_{i=1}^{\mathsf{d}} |\mathsf{p}_i|$, we have, by symmetry and by the above observation on $f(x)$, that $\sum_{i=1}^{\mathsf{d}} \mathsf{p}_i^2$ is minimized under the condition $\|\mathsf{p}\|_1 = \alpha$, when all the coordinates of p are equal. As such, we have that $\|\mathsf{p}\|_2 \geq \sqrt{\mathsf{d}(\alpha/\mathsf{d})^2} = \|\mathsf{p}\|_1 / \sqrt{\mathsf{d}}$, implying the claim. ■

4.1.2. The clustering problem. There is a metric space $(\mathcal{X}, \mathbf{d})$ and the input is a set of n points $\mathsf{P} \subseteq \mathcal{X}$. Given a set of centers $\mathbf{C}$, every point of P is assigned to its nearest neighbor in $\mathbf{C}$. All the points of P that are assigned to a center $\bar{\mathsf{c}}$ form the ***cluster*** of $\bar{\mathsf{c}}$, denoted by

$$\text{cluster}(\mathbf{C}, \bar{\mathsf{c}}) = \left\{ \mathsf{p} \in \mathsf{P} \;\middle|\; \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \bar{\mathsf{c}}) = \mathbf{d}(\mathsf{p}, \mathbf{C}) \right\}, \tag{4.1}$$

where

$$\mathbf{d}(\mathsf{p}, \mathbf{C}) = \min_{\bar{\mathsf{c}} \in \mathbf{C}} \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \bar{\mathsf{c}})$$

denotes the ***distance*** of p to the set $\mathbf{C}$. Namely, the center set $\mathbf{C}$ partition P into clusters. This specific scheme of partitioning points by assigning them to their closest center (in a given set of centers) is known as a ***Voronoi partition***.

In particular, let $\mathsf{P} = \{\mathsf{p}_1, \ldots, \mathsf{p}_n\}$, and consider the n-dimensional point

$$\mathsf{P}_{\mathbf{C}} = \Big(\mathbf{d}(\mathsf{p}_1, \mathbf{C}), \mathbf{d}(\mathsf{p}_2, \mathbf{C}), \ldots, \mathbf{d}(\mathsf{p}_n, \mathbf{C})\Big).$$

The ith coordinate of the point $\mathsf{P}_{\mathbf{C}}$ is the distance (i.e., cost of assigning) of p_i to its closest center in $\mathbf{C}$.

4.2. On k-center clustering

In the ***k-center clustering problem***, a set $\mathsf{P} \subseteq \mathcal{X}$ of n points is provided together with a parameter k. We would like to find a set of k points, $\mathbf{C} \subseteq \mathsf{P}$, such that the maximum distance of a point in P to its closest point in $\mathbf{C}$ is minimized.

As a concrete example, consider the set of points to be a set of cities. Distances between points represent the time it takes to travel between the corresponding cities. We would like to build k hospitals and minimize the maximum time it takes a patient to arrive at her closest hospital. Naturally, we want to build the hospitals in the cities and not in the middle of nowhere.[1]

Formally, given a set of centers $\mathbf{C}$, the k-center clustering ***price*** of P by $\mathbf{C}$ is denoted by

$$\|\mathsf{P}_{\mathbf{C}}\|_\infty = \max_{\mathsf{p}\in\mathsf{P}} \mathbf{d}(\mathsf{p}, \mathbf{C}).$$

Note that every point in a cluster is within a distance at most $\|\mathsf{P}_{\mathbf{C}}\|_\infty$ from its respective center.

Formally, the ***k-center problem*** is to find a set $\mathbf{C}$ of k points, such that $\|\mathsf{P}_{\mathbf{C}}\|_\infty$ is minimized; namely,

$$\mathrm{opt}_\infty(\mathsf{P}, k) = \min_{\mathbf{C}\subseteq\mathsf{P}, |\mathbf{C}|=k} \|\mathsf{P}_{\mathbf{C}}\|_\infty.$$

We will denote the set of centers realizing the optimal clustering by C_{opt}. A more explicit definition (and somewhat more confusing) of the k-center clustering is to compute the set $\mathbf{C}$ of size k realizing $\min_{\mathbf{C}\subseteq\mathsf{P}} \max_{\mathsf{p}\in\mathsf{P}} \min_{\overline{\mathsf{c}}\in\mathbf{C}} \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}})$.

It is known that k-center clustering is NP-Hard, and it is in fact hard to approximate within a factor of 1.86, even for a point set in the plane (under the Euclidean distance). Surprisingly, there is a simple and elegant algorithm that achieves a 2-approximation.

Discrete vs. continuous clustering. If the input is a point set in $\mathbb{R}^{\mathsf{d}}$, the centers of the clustering are not necessarily restricted to be a subset of the input point, as they might be placed anywhere in $\mathbb{R}^{\mathsf{d}}$. Allowing this flexibility might further reduce the price of the clustering (by a constant factor). The variant where one is restricted to use the input points as centers is the ***discrete clustering*** problem. The version where centers might be placed anywhere in the given metric space is the ***continuous clustering*** version.

4.2.1. The greedy clustering algorithm. The algorithm **GreedyKCenter** starts by picking an arbitrary point, $\overline{\mathsf{c}}_1$, and setting $\mathbf{C}_1 = \{\overline{\mathsf{c}}_1\}$. Next, we compute for every point $\mathsf{p} \in \mathsf{P}$ its distance $d_1[\mathsf{p}]$ from $\overline{\mathsf{c}}_1$. Now, consider the point worst served by $\mathbf{C}_1$; this is the point realizing $r_1 = \max_{\mathsf{p}\in\mathsf{P}} d_1[\mathsf{p}]$. Let $\overline{\mathsf{c}}_2$ denote this point, and add it to the set $\mathbf{C}_1$, resulting in the set $\mathbf{C}_2$.

Specifically, in the ith iteration, we compute for each point $\mathsf{p} \in \mathsf{P}$ the quantity $d_{i-1}[\mathsf{p}] = \min_{\overline{\mathsf{c}}\in\mathbf{C}_{i-1}} \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}})$. We also compute the radius of the clustering

$$r_{i-1} = \left\|\mathsf{P}_{\mathbf{C}_{i-1}}\right\|_\infty = \max_{\mathsf{p}\in\mathsf{P}} d_{i-1}[\mathsf{p}] = \max_{\mathsf{p}\in\mathsf{P}} \mathbf{d}(\mathsf{p}, \mathbf{C}_{i-1}) \tag{4.2}$$

[1] Although, there are recorded cases in history of building universities in the middle of nowhere.

and the bottleneck point $\overline{\mathsf{c}}_i$ that realizes it. Next, we add $\overline{\mathsf{c}}_i$ to $\mathbf{C}_{i-1}$ to form the new set $\mathbf{C}_i$. We repeat this process k times.

Namely, the algorithm repeatedly picks the point furthest away from the current set of centers and adds it to this set.

To make this algorithm slightly faster, observe that

$$d_i[\mathsf{p}] = \mathbf{d}(\mathsf{p}, \mathbf{C}_i) = \min(\mathbf{d}(\mathsf{p}, \mathbf{C}_{i-1}), \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}_i)) = \min(d_{i-1}[\mathsf{p}], \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}_i)).$$

In particular, if we maintain for each point $\mathsf{p} \in \mathsf{P}$ a single variable $d[\mathsf{p}]$ with its current distance to its closest center in the current center set, then the above formula boils down to

$$d[\mathsf{p}] \leftarrow \min(d[\mathsf{p}], \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}_i)).$$

Namely, the above algorithm can be implemented using $O(n)$ space, where $n = |\mathsf{P}|$. The ith iteration of choosing the ith center takes $O(n)$ time. Thus, overall, this approximation algorithm takes $O(nk)$ time.

A ***ball*** of radius r around a point $\mathsf{p} \in \mathsf{P}$ is the set of points in P with distance at most r from p; namely, $\mathbf{b}(\mathsf{p}, r) = \left\{ \mathsf{q} \in \mathsf{P} \;\middle|\; \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \mathsf{q}) \leq r \right\}$. Thus, the k-center problem can be interpreted as the problem of covering the points of P using k balls of minimum (maximum) radius.

Theorem 4.3. *Given a set of n points P in a metric space $(\mathcal{X}, \mathbf{d})$, the algorithm* **GreedyKCenter** *computes a set $\mathbf{K}$ of k centers, such that $\mathbf{K}$ is a 2-approximation to the optimal k-center clustering of P; namely, $\|\mathsf{P}_{\mathbf{K}}\|_\infty \leq 2\mathrm{opt}_\infty$, where $\mathrm{opt}_\infty = \mathrm{opt}_\infty(\mathsf{P}, k)$ is the price of the optimal clustering. The algorithm takes $O(nk)$ time.*

Proof. The running time follows by the above description, so we concern ourselves only with the approximation quality.

By definition, we have $r_k = \|\mathsf{P}_{\mathbf{K}}\|_\infty$, and let $\overline{\mathsf{c}}_{k+1}$ be the point in P realizing $r_k = \max_{\mathsf{p} \in \mathsf{P}} \mathbf{d}(\mathsf{p}, \mathbf{K})$. Let $\mathbf{C} = \mathbf{K} \cup \{\overline{\mathsf{c}}_{k+1}\}$. Observe that by the definition of r_i (see (4.2)), we have that $r_1 \geq r_2 \geq \ldots \geq r_k$. Furthermore, for $i < j \leq k+1$ we have that

$$\mathbf{d}_{\mathcal{M}}\!\left(\overline{\mathsf{c}}_i, \overline{\mathsf{c}}_j\right) \geq \mathbf{d}_{\mathcal{M}}\!\left(\overline{\mathsf{c}}_j, \mathbf{C}_{j-1}\right) = r_{j-1} \geq r_k.$$

Namely, the distance between any pair of points in $\mathbf{C}$ is at least r_k. Now, assume for the sake of contradiction that $r_k > 2\mathrm{opt}_\infty(\mathsf{P}, k)$. Consider the optimal solution that covers P with k balls of radius opt_∞. By the triangle inequality, any two points inside such a ball are within a distance at most $2\mathrm{opt}_\infty$ from each other. Thus, none of these balls can cover two points of $\mathbf{C} \subseteq \mathsf{P}$, since the minimum distance between members of $\mathbf{C}$ is $> 2\mathrm{opt}_\infty$. As such, the optimal cover by k balls of radius opt_∞ cannot cover $\mathbf{C}$ (and thus P), as $|\mathbf{C}| = k + 1$, a contradiction. ■

In the spirit of never trusting a claim that has only a single proof, we provide an alternative proof.[②]

Alternative proof. If every cluster of C_{opt} contains exactly one point of $\mathbf{K}$, then the claim follows. Indeed, consider any point $\mathsf{p} \in \mathsf{P}$, and let $\overline{\mathsf{c}}$ be the center it belongs to in C_{opt}. Also, let $\overline{\mathsf{g}}$ be the center of $\mathbf{K}$ that is in $\mathrm{cluster}\!\left(C_{\mathrm{opt}}, \overline{\mathsf{c}}\right)$. We have that $\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}) = \mathbf{d}\!\left(\mathsf{p}, C_{\mathrm{opt}}\right) \leq \mathrm{opt}_\infty = \mathrm{opt}_\infty(\mathsf{P}, k)$. Similarly, observe that $\mathbf{d}_{\mathcal{M}}(\overline{\mathsf{g}}, \overline{\mathsf{c}}) = \mathbf{d}\!\left(\overline{\mathsf{g}}, C_{\mathrm{opt}}\right) \leq \mathrm{opt}_\infty$. As such, by the triangle inequality, we have that $\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{g}}) \leq \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}) + \mathbf{d}_{\mathcal{M}}(\overline{\mathsf{c}}, \overline{\mathsf{g}}) \leq 2\mathrm{opt}_\infty$.

[②] Mark Twain is credited with saying that "I don't give a damn for a man that can only spell a word one way." However, there seems to be some doubt if he really said that, which brings us to the conclusion of never trusting a quote if it is credited only to a single person.

By the pigeon hole principle, the only other possibility is that there are at least two centers $\overline{g}$ and $\overline{h}$ of $\mathbf{K}$ that are both in cluster$\left(C_{\mathrm{opt}}, \overline{c}\right)$, for some $\overline{c} \in C_{\mathrm{opt}}$. Assume, without loss of generality, that $\overline{h}$ was added later than $\overline{g}$ to the center set $\mathbf{K}$ by the algorithm **GreedyKCenter**, say in the ith iteration. But then, since **GreedyKCenter** always chooses the point furthest away from the current set of centers, we have that

$$\|\mathsf{P}_{\mathbf{K}}\|_\infty \leq \left\|\mathsf{P}_{\mathbf{C}_{i-1}}\right\|_\infty = \mathbf{d}\left(\overline{h}, \mathbf{C}_{i-1}\right) \leq \mathbf{d}_{\mathcal{M}}\left(\overline{h}, \overline{g}\right) \leq \mathbf{d}_{\mathcal{M}}\left(\overline{h}, \overline{c}\right) + \mathbf{d}_{\mathcal{M}}(\overline{c}, \overline{g}) \leq 2\mathrm{opt}_\infty. \qquad \blacksquare$$

4.2.2. The greedy permutation. There is an interesting phenomena associated with **GreedyKCenter**. If we run it till it exhausts all the points of P (i.e., $k = n$), then this algorithm generates a permutation of P; that is, $\langle \mathsf{P} \rangle = \langle \overline{c}_1, \overline{c}_2, \ldots, \overline{c}_n \rangle$. We will refer to $\langle \mathsf{P} \rangle$ as the ***greedy permutation*** of P. There is also an associated sequence of radii $\langle r_1, r_2, \ldots, r_n \rangle$, where all the points of P are within a distance at most r_i from the points of $\mathbf{C}_i = \langle \overline{c}_1, \ldots, \overline{c}_i \rangle$.

Definition 4.4. A set $S \subseteq \mathsf{P}$ is an ***r-packing*** for P if the following two properties hold.

(i) ***Covering property***: All the points of P are within a distance at most r from the points of S.

(ii) ***Separation property***: For any pair of points $\mathsf{p}, \mathsf{q} \in S$, we have that $\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \mathsf{q}) \geq r$.

(One can relax the separation property by requiring that the points of S be at a distance $\Omega(r)$ apart.)

Intuitively, an r-packing of a point set P is a compact representation of P in the resolution r. Surprisingly, the greedy permutation of P provides us with such a representation for all resolutions.

Theorem 4.5. *Let P be a set of n points in a finite metric space, and let its greedy permutation be $\langle \overline{c}_1, \overline{c}_2, \ldots, \overline{c}_n \rangle$ with the associated sequence of radii $\langle r_1, r_2, \ldots, r_n \rangle$. For any i, we have that $\mathbf{C}_i = \langle \overline{c}_1, \ldots, \overline{c}_i \rangle$ is an r_i-packing of P.*

Proof. Note that by construction $r_{k-1} = \mathbf{d}(\overline{c}_k, \mathbf{C}_{k-1})$, for all $k = 2, \ldots, n$. As such, for $j < k \leq i \leq n$, we have that $\mathbf{d}_{\mathcal{M}}\left(\overline{c}_j, \overline{c}_k\right) \geq \mathbf{d}(\overline{c}_k, \mathbf{C}_{k-1}) = r_{k-1} \geq r_i$, since $r_1, r_2, \ldots, r_n$ is a monotonically non-increasing sequence. This implies the required separation property.

The covering property follows by definition; see $(4.2)_{\mathrm{p49}}$. $\blacksquare$

4.3. On k-median clustering

In the ***k-median clustering problem***, a set $\mathsf{P} \subseteq \mathcal{X}$ is provided together with a parameter k. We would like to find a set of k points, $\mathbf{C} \subseteq \mathsf{P}$, such that the sum of the distances of points of P to their closest point in $\mathbf{C}$ is minimized.

Formally, given a set of centers $\mathbf{C}$, the k-median clustering ***price*** of clustering P by $\mathbf{C}$ is denoted by

$$\|\mathsf{P}_{\mathbf{C}}\|_1 = \sum_{\mathsf{p} \in \mathsf{P}} \mathbf{d}(\mathsf{p}, \mathbf{C}).$$

Formally, the ***k-median problem*** is to find a set $\mathbf{C}$ of k points, such that $\|\mathsf{P}_{\mathbf{C}}\|_1$ is minimized; namely,

$$\mathrm{opt}_1(\mathsf{P}, k) = \min_{\mathbf{C} \subseteq \mathsf{P}, |\mathbf{C}| = k} \|\mathsf{P}_{\mathbf{C}}\|_1.$$

We will denote the set of centers realizing the optimal clustering by C_{opt}.

There is a simple and elegant constant factor approximation algorithm for k-median clustering using ***local search*** (its analysis however is painful).

A note on notation. Consider the set $U = \left\{ \mathsf{P}_{\mathbf{C}} \;\middle|\; \mathbf{C} \in \mathsf{P}^k \right\}$. Clearly, we have that $\mathrm{opt}_\infty(\mathsf{P}, k) = \min_{\mathsf{q} \in U} \|\mathsf{q}\|_\infty$ and $\mathrm{opt}_1(\mathsf{P}, k) = \min_{\mathsf{q} \in U} \|\mathsf{q}\|_1$.

Namely, k-center clustering under this interpretation is just finding the point minimizing the L_∞-norm in a set U of points in n dimensions. Similarly, the k-median problem is to find the point minimizing the L_1-norm in the set U.

Claim 4.6. *For any point set* P *of* n *points and a parameter* k*, we have that* $\mathrm{opt}_\infty(\mathsf{P}, k) \leq \mathrm{opt}_1(\mathsf{P}, k) \leq n\, \mathrm{opt}_\infty(\mathsf{P}, k)$.

Proof. For any point $\mathsf{p} \in \mathbb{R}^n$, we have that $\|\mathsf{p}\|_\infty = \max_{i=1}^n |\mathsf{p}_i| \leq \sum_{i=1}^n |\mathsf{p}_i| = \|\mathsf{p}\|_1$ and $\|\mathsf{p}\|_1 = \sum_{i=1}^n |\mathsf{p}_i| \leq \sum_{i=1}^n \max_{j=1}^n \left|\mathsf{p}_j\right| \leq n \|\mathsf{p}\|_\infty$.

Let $\mathbf{C}$ be the set of k points realizing $\mathrm{opt}_1(\mathsf{P}, k)$; that is, $\mathrm{opt}_1(\mathsf{P}, k) = \|\mathsf{P}_{\mathbf{C}}\|_1$. We have that $\mathrm{opt}_\infty(\mathsf{P}, k) \leq \|\mathsf{P}_{\mathbf{C}}\|_\infty \leq \|\mathsf{P}_{\mathbf{C}}\|_1 = \mathrm{opt}_1(\mathsf{P}, k)$. Similarly, if $\mathbf{K}$ is the set realizing $\mathrm{opt}_\infty(\mathsf{P}, k)$, then $\mathrm{opt}_1(\mathsf{P}, k) = \|\mathsf{P}_{\mathbf{C}}\|_1 \leq \|\mathsf{P}_{\mathbf{K}}\|_1 \leq n \|\mathsf{P}_{\mathbf{K}}\|_\infty = n \cdot \mathrm{opt}_\infty(\mathsf{P}, k)$. ■

4.3.1. Approximation algorithm – local search. We are given a set P of n points and a parameter k. In the following, let C_{opt} denote the set of centers realizing the optimal solution, and let $\mathrm{opt}_1 = \mathrm{opt}_1(\mathsf{P}, k)$.

4.3.1.1. *The algorithm.*

A $2n$-approximation. The algorithm starts by computing a set of k centers L using Theorem 4.3. Claim 4.6 implies that

$$\begin{aligned} &\|\mathsf{P}_L\|_1 / 2n \leq \|\mathsf{P}_L\|_\infty / 2 \leq \mathrm{opt}_\infty(\mathsf{P}, k) \leq \mathrm{opt}_1 \leq \|\mathsf{P}_L\|_1 \\ \implies \quad &\mathrm{opt}_1 \leq \|\mathsf{P}_L\|_1 \leq 2n\, \mathrm{opt}_1. \end{aligned} \tag{4.3}$$

Namely, L is a $2n$-approximation to the optimal solution.

Improving it. Let $0 < \tau < 1$ be a parameter to be determined shortly. The local search algorithm **algLocalSearchKMed** initially sets the current set of centers L_{curr} to be L, the set of centers computed above. Next, at each iteration it checks if the current solution L_{curr} can be improved by replacing one of the centers in it by a center from the outside. We will refer to such an operation as a ***swap***. There are at most $|\mathsf{P}|\,|L_{\mathrm{curr}}| = nk$ choices to consider, as we pick a center $\overline{\mathsf{c}} \in L_{\mathrm{curr}}$ to throw away and a new center to replace it by $\overline{\mathsf{o}} \in (\mathsf{P} \setminus L_{\mathrm{curr}})$. We consider the new candidate set of centers $\mathbf{K} \leftarrow (L_{\mathrm{curr}} \setminus \{\overline{\mathsf{c}}\}) \cup \{\overline{\mathsf{o}}\}$. If $\|\mathsf{P}_{\mathbf{K}}\|_1 \leq (1 - \tau) \left\|\mathsf{P}_{L_{\mathrm{curr}}}\right\|_1$, then the algorithm sets $L_{\mathrm{curr}} \leftarrow \mathbf{K}$. The algorithm continues iterating in this fashion over all possible swaps.

The algorithm **algLocalSearchKMed** stops when there is no swap that would improve the current solution by a factor of (at least) $(1 - \tau)$. The final content of the set L_{curr} is the required constant factor approximation.

4.3.1.2. *Running time.* An iteration requires checking $O(nk)$ swaps (i.e., $n - k$ candidates to be swapped in and k candidates to be swapped out). Computing the price of every such swap, done naively, requires computing the distance of every point to its nearest center, and that takes $O(nk)$ time per swap. As such, overall, each iteration takes $O\Big((nk)^2\Big)$ time.

Since $1/(1 - \tau) \geq 1 + \tau$, the running time of the algorithm is

$$O\left((nk)^2 \log_{1/(1-\tau)} \frac{\|\mathsf{P}_L\|_1}{\mathrm{opt}_1}\right) = O\Big((nk)^2 \log_{1+\tau} 2n\Big) = O\left((nk)^2 \frac{\log n}{\tau}\right),$$

by (4.3) and Lemma $28.10_{\mathrm{p}348}$. Thus, if τ is polynomially small, then the running time would be polynomial.

4.3.2. Proof of quality of approximation. We claim that the above algorithm provides a constant factor approximation for the optimal k-median clustering.

4.3.2.1. *Definitions and intuition.* Intuitively, since the local search got stuck in a locally optimal solution, it cannot be too far from the true optimal solution.

For the sake of simplicity of exposition, let us assume (for now) that the solution returned by the algorithm cannot be improved (at all) by any swap, and let L be this set of centers. For a center $\overline{c} \in L$ and a point $\overline{o} \in \mathsf{P} \setminus L$, let $L - \overline{c} + \overline{o} = (L \setminus \{\overline{c}\}) \cup \{\overline{o}\}$ denote the set of centers resulting from applying the swap $\overline{c} \to \overline{o}$ to L. We are assuming that there is no beneficial swap; that is,

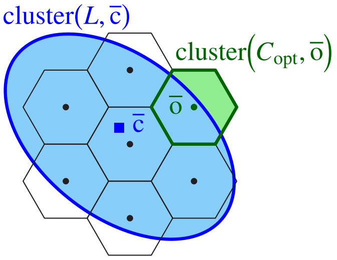
$$(4.4) \qquad \forall \overline{c} \in L, \overline{o} \in \mathsf{P} \setminus L \qquad 0 \leq \Delta(\overline{c}, \overline{o}) = \nu_1(L - \overline{c} + \overline{o}) - \nu_1(L),$$

where $\nu_1(X) = \|\mathsf{P}_X\|_1$.

Equation (4.4) provides us with a large family of inequalities that all hold together. Each inequality is represented by a swap $\overline{c} \to \overline{o}$. We would like to combine these inequalities such that they will imply that $5 \left\|\mathsf{P}_{C_{\mathrm{opt}}}\right\|_1 \geq \|\mathsf{P}_L\|_1$, namely, that the local search algorithm provides a constant factor approximation to optimal clustering. This idea seems to be somewhat mysterious (or even impossible), but hopefully it will become clearer shortly.

From local clustering to local clustering complying with the optimal clustering. The first hurdle in the analysis is that a cluster of the optimal solution $\mathrm{cluster}\left(C_{\mathrm{opt}}, \overline{o}\right)$, for $\overline{o} \in C_{\mathrm{opt}}$, might intersect a large number of clusters in the local clustering (i.e., clusters of the form $\mathrm{cluster}(L, \overline{c})$ for $\overline{c} \in L$).

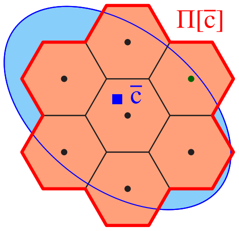
Fortunately, one can modify the assignment of points to clusters in the locally optimal clustering so that the resulting clustering of P complies with the optimal partition and the price of the clustering increases only moderately; that is, every cluster in the optimal clustering would be contained in a single cluster of the modified local solution. In particular, now an optimal cluster would intersect only a single cluster in the modified local solution.


cluster$(L, \overline{c})$
cluster$\left(C_{\mathrm{opt}}, \overline{o}\right)$
$\overline{o}$
$\overline{c}$

Furthermore, this modified local solution Π is not much more expensive. Now, in this modified partition there are many beneficial swaps (by making it into the optimal clustering). But these swaps cannot be too profitable, since then they would have been profitable for the original local solution. This would imply that the local solution cannot be too expensive. The picture on the right depicts a local cluster and the optimal clusters in its vicinity such that their centers are contained inside it.

In the following, we denote by $\mathsf{nn}(\mathsf{p}, X)$ the nearest neighbor to p in the set X.

For a point $\mathsf{p} \in \mathsf{P}$, let $\overline{o}_{\mathsf{p}} = \mathsf{nn}\left(\mathsf{p}, C_{\mathrm{opt}}\right)$ be its optimal center, and let $\alpha(\mathsf{p}) = \mathsf{nn}\left(\overline{o}_{\mathsf{p}}, L\right)$ be the center p should use if p follows its optimal center's assignment. Let Π be the modified partition of P by the function $\alpha(\cdot)$.



That is, for $\overline{c} \in L$, its cluster in Π, denoted by $\Pi[\overline{c}]$, is the set of all points $\mathsf{p} \in \mathsf{P}$ such that $\alpha(\mathsf{p}) = \overline{c}$.

Now, for any center $\overline{o} \in C_{\mathrm{opt}}$, let $\mathsf{nn}(\overline{o}, L)$ be its nearest neighbor in L, and observe that $\mathrm{cluster}\left(C_{\mathrm{opt}}, \overline{o}\right) \subseteq \Pi\left[\mathsf{nn}(\overline{o}, L)\right]$ (see $(4.1)_{\mathrm{p48}}$). The picture on the right shows the resulting modified cluster for the above example.

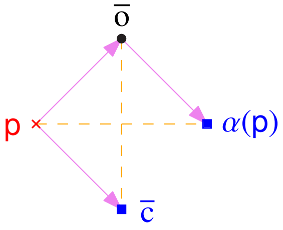
Let δ_{p} denote the price of this reassignment for the point p; that is, $\delta_{\mathsf{p}} = \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \alpha(\mathsf{p})) - \mathbf{d}(\mathsf{p}, L)$. Note that if p does not get reassigned, then $\delta_{\mathsf{p}} = 0$ and otherwise $\delta_{\mathsf{p}} \geq 0$, since $\alpha(\mathsf{p}) \in L$ and $\mathbf{d}(\mathsf{p}, L) = \min_{\overline{\mathsf{c}} \in L} \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}})$.

Lemma 4.7. *The increase in cost from moving from the clustering induced by L to the clustering of* Π *is bounded by* $2 \left\| \mathsf{P}_{C_{\mathrm{opt}}} \right\|_1$. *That is,* $\sum_{\mathsf{p} \in \mathsf{P}} \delta_{\mathsf{p}} \leq 2 \left\| \mathsf{P}_{C_{\mathrm{opt}}} \right\|_1$.

Proof. For a point $\mathsf{p} \in \mathsf{P}$, let $\overline{\mathsf{c}} = \mathsf{nn}(\mathsf{p}, L)$ be its local center, let $\overline{\mathsf{o}} = \mathsf{nn}\left(\mathsf{p}, C_{\mathrm{opt}}\right)$ be its optimal center, and let $\alpha(\mathsf{p}) = \mathsf{nn}(\overline{\mathsf{o}}, L)$ be its new assigned center in Π. Observe that $\mathbf{d}_{\mathcal{M}}(\overline{\mathsf{o}}, \alpha(\mathsf{p})) = \mathbf{d}_{\mathcal{M}}(\overline{\mathsf{o}}, \mathsf{nn}(\overline{\mathsf{o}}, L)) \leq \mathbf{d}_{\mathcal{M}}(\overline{\mathsf{o}}, \overline{\mathsf{c}})$.

As such, by the triangle inequality, we have that

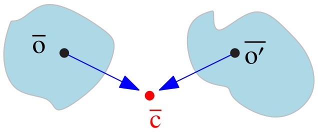
$$\begin{aligned}
\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \alpha(\mathsf{p})) &\leq \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{o}}) + \mathbf{d}_{\mathcal{M}}(\overline{\mathsf{o}}, \alpha(\mathsf{p})) \leq \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{o}}) + \mathbf{d}_{\mathcal{M}}(\overline{\mathsf{o}}, \overline{\mathsf{c}}) \\
&\leq \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{o}}) + (\mathbf{d}_{\mathcal{M}}(\overline{\mathsf{o}}, \mathsf{p}) + \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}})) \\
&= 2\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{o}}) + \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}).
\end{aligned}$$



Finally, $\delta_{\mathsf{p}} = \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \alpha(\mathsf{p})) - \mathbf{d}(\mathsf{p}, L) \leq 2\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{o}}) + \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}) - \mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{c}}) = 2\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \overline{\mathsf{o}}) = 2\mathbf{d}\left(\mathsf{p}, C_{\mathrm{opt}}\right)$. As such, $\sum_{\mathsf{p} \in \mathsf{P}} \delta_{\mathsf{p}} \leq \sum_{\mathsf{p} \in \mathsf{P}} 2\mathbf{d}\left(\mathsf{p}, C_{\mathrm{opt}}\right) = 2 \left\| \mathsf{P}_{C_{\mathrm{opt}}} \right\|_1$. ■

Drifters, anchors, and tyrants. A center of L that does not serve any center of C_{opt} (i.e., its cluster in Π is empty) is a ***drifter***. Formally, we map each center of C_{opt} to its nearest neighbor in L, and for a center $\overline{\mathsf{c}} \in L$ its ***degree***, denoted by $\deg(\overline{\mathsf{c}})$, is the number of points of C_{opt} mapped to it by this nearest neighbor mapping.

As such, a center $\overline{\mathsf{c}} \in L$ is a ***drifter*** if $\deg(\overline{\mathsf{c}}) = 0$, an ***anchor*** if $\deg(\overline{\mathsf{c}}) = 1$, and a ***tyrant*** if $\deg(\overline{\mathsf{c}}) > 1$. Observe that if $\overline{\mathsf{c}}$ is a drifter, then $\Pi[\overline{\mathsf{c}}] = \emptyset$.



The reader should not take these names too seriously, but observe that centers that are tyrants cannot easily move around and are bad candidates for swaps. Indeed, consider the situation depicted in the figure on the right. Here the center $\overline{\mathsf{c}}$ serves points of P that belong to two optimal clusters $\overline{\mathsf{o}}$ and $\overline{\mathsf{o}'}$, such that $\overline{\mathsf{c}} = \mathsf{nn}(\overline{\mathsf{o}}, L) = \mathsf{nn}\left(\overline{\mathsf{o}'}, L\right)$. If we swap $\overline{\mathsf{c}} \to \overline{\mathsf{o}}$, then the points in the cluster $\mathrm{cluster}\left(C_{\mathrm{opt}}, \overline{\mathsf{o}'}\right)$ might find themselves very far from any center in $L - \overline{\mathsf{c}} + \overline{\mathsf{o}}$. Similarly, the points of $\mathrm{cluster}\left(C_{\mathrm{opt}}, \overline{\mathsf{o}}\right)$ might be in trouble if we swap $\overline{\mathsf{c}} \to \overline{\mathsf{o}'}$.

Intuitively, since we shifted our thinking from the local solution to the partition Π, a drifter center is not being used by the clustering, and we can reassign it so that it decreases the price of the clustering.

That is, since moving from the local clustering of L to Π is relatively cheap, we can free a drifter $\overline{\mathsf{c}}$ from all its clients in the local partition. Formally, the ***ransom*** of a drifter center $\overline{\mathsf{c}}$ is $\mathrm{ransom}(\overline{\mathsf{c}}) = \sum_{\mathsf{p} \in \mathrm{cluster}(L, \overline{\mathsf{c}})} \delta_{\mathsf{p}}$. This is the price of reassigning all the points that are currently served by the drifter $\overline{\mathsf{c}}$ to the center in L serving their optimal center. Once this ransom is paid, $\overline{\mathsf{c}}$ serves nobody and can be moved with no further charge.

More generally, the ***ransom*** of any center $\overline{\mathsf{c}} \in L$ is

$$\mathrm{ransom}(\overline{\mathsf{c}}) = \sum_{\mathsf{p} \in \mathrm{cluster}(L, \overline{\mathsf{c}}) \setminus \Pi[\overline{\mathsf{c}}]} \delta_{\mathsf{p}}.$$

Note that for a drifter $\Pi[\overline{\mathsf{c}}] = \emptyset$ and $\mathrm{cluster}(L, \overline{\mathsf{c}}) = \mathrm{cluster}(L, \overline{\mathsf{c}}) \setminus \Pi[\overline{\mathsf{c}}]$, and in general, the points of $\mathrm{cluster}(L, \overline{\mathsf{c}}) \setminus \Pi[\overline{\mathsf{c}}]$ are exactly the points of $\mathrm{cluster}(L, \overline{\mathsf{c}})$ being reassigned.

Hence, $\text{ransom}(\overline{c})$ is the increase in cost of reassigning the points of $\text{cluster}(L, \overline{c})$ when moving from the local clustering of L to the clustering of Π.

Observe that, by Lemma 4.7, we have that

$$\sum_{\overline{c} \in L} \text{ransom}(\overline{c}) \leq 2 \left\| \mathsf{P}_{C_{\text{opt}}} \right\|_1 . \tag{4.5}$$

For $\overline{o} \in C_{\text{opt}}$, the ***optimal price*** and ***local price*** of $\text{cluster}\left(C_{\text{opt}}, \overline{o}\right)$ are

$$\text{opt}\,(\overline{o}) = \sum_{\mathsf{p} \in \text{cluster}(C_{\text{opt}}, \overline{o})} \mathbf{d}\big(\mathsf{p}, C_{\text{opt}}\big) \qquad \text{and} \qquad \text{local}(\overline{o}) = \sum_{\mathsf{p} \in \text{cluster}(C_{\text{opt}}, \overline{o})} \mathbf{d}(\mathsf{p}, L)\,,$$

respectively.

Lemma 4.8. *If* $\overline{c} \in L$ *is a drifter and* $\overline{o}$ *is any center of* C_{opt}*, then* $\text{local}(\overline{o}) \leq \text{ransom}(\overline{c}) + \text{opt}(\overline{o})$.

Proof. Since $\overline{c} \in L$ is a drifter, we can swap it with any center in $\overline{o} \in C_{\text{opt}}$. Since L is a locally optimal solution, we have that the change in the cost caused by the swap $\overline{c} \to \overline{o}$ is

$$\begin{aligned} 0 \leq \Delta(\overline{c}, \overline{o}) &\leq \text{ransom}(\overline{c}) - \text{local}(\overline{o}) + \text{opt}(\overline{o}) \\ \implies \quad &\text{local}(\overline{o}) \leq \text{ransom}(\overline{c}) + \text{opt}(\overline{o})\,. \end{aligned} \tag{4.6}$$

Indeed, $\overline{c}$ pays its ransom so that all the clients using it are now assigned to some other centers of L. Now, all the points of $\text{cluster}\left(C_{\text{opt}}, \overline{o}\right)$ instead of paying $\text{local}(\overline{o})$ are now paying (at most) $\text{opt}(\overline{o})$. (We might pay less for a point $\mathsf{p} \in \text{cluster}\left(C_{\text{opt}}, \overline{o}\right)$ if it is closer to $L - \overline{c} + \overline{o}$ than to $\overline{o}$.) ■

Equation (4.6) provides us with a glimmer of hope that we can bound the price of the local clustering. We next argue that if there are many tyrants, then there must also be many drifters. In particular, with these drifters we can bound the price of the local clustering cost of the optimal clusters assigned to tyrants. Also, we argue that an anchor and its associated optimal center define a natural swap which is relatively cheap. Putting all of these together will imply the desired claim.

There are many drifters. Let S_{opt} (resp. A_{opt}) be the set of all the centers of C_{opt} that are assigned to tyrants (resp. anchors) by $\mathsf{nn}(\cdot, L)$. Observe that $S_{\text{opt}} \cup A_{\text{opt}} = C_{\text{opt}}$. Let $\mathcal{D}$ be the set of drifters in L.

Observe that every tyrant has at least two followers in C_{opt}; that is, $\left|S_{\text{opt}}\right| \geq 2\#_{\text{tyrants}}$. Also, $k = \left|C_{\text{opt}}\right| = |L|$ and $\#_{\text{anchors}} = \left|A_{\text{opt}}\right|$. As such, we have that

$$\begin{aligned} &\#_{\text{tyrants}} + \#_{\text{anchors}} + \#_{\text{drifters}} = |L| = \left|C_{\text{opt}}\right| = \left|S_{\text{opt}}\right| + \left|A_{\text{opt}}\right| \\ &\implies \#_{\text{drifters}} = \left|S_{\text{opt}}\right| + \left|A_{\text{opt}}\right| - \#_{\text{anchors}} - \#_{\text{tyrants}} = \left|S_{\text{opt}}\right| - \#_{\text{tyrants}} \geq \left|S_{\text{opt}}\right|/2. \end{aligned} \tag{4.7}$$

Namely, $2\#_{\text{drifters}} \geq \left|S_{\text{opt}}\right|$.

Lemma 4.9. *We have that* $\displaystyle\sum_{\overline{o} \in S_{\text{opt}}} \text{local}(\overline{o}) \leq 2 \sum_{\overline{c} \in \mathcal{D}} \text{ransom}(\overline{c}) + \sum_{\overline{o} \in S_{\text{opt}}} \text{opt}(\overline{o}).$

Proof. If $\left|S_{\text{opt}}\right| = 0$, then the statement holds trivially.

So assume $\left|S_{\text{opt}}\right| > 0$ and let $\overline{c}$ be the drifter with the lowest $\text{ransom}(\overline{c})$. For any $\overline{o} \in S_{\text{opt}}$, we have that $\text{local}(\overline{o}) \leq \text{ransom}(\overline{c}) + \text{opt}(\overline{o})$, by (4.6). Summing over all such centers, we have that

$$\sum_{\overline{o} \in S_{\mathrm{opt}}} \mathrm{local}(\overline{o}) \le \left|S_{\mathrm{opt}}\right| \mathrm{ransom}(\overline{c}) + \sum_{\overline{o} \in S_{\mathrm{opt}}} \mathrm{opt}(\overline{o}),$$

which is definitely smaller than the stated bound, since $\left|S_{\mathrm{opt}}\right| \le 2\left|\mathcal{D}\right|$, by (4.7). ■

Lemma 4.10. *We have that* $\sum_{\overline{o} \in \mathrm{A}_{\mathrm{opt}}} \mathrm{local}(\overline{o}) \le \sum_{\overline{o} \in \mathrm{A}_{\mathrm{opt}}} \mathrm{ransom}(\mathsf{nn}(\overline{o}, L)) + \sum_{\overline{o} \in \mathrm{A}_{\mathrm{opt}}} \mathrm{opt}(\overline{o})$.

Proof. For a center $\overline{o} \in \mathrm{A}_{\mathrm{opt}}$, its anchor is $\overline{c} = \mathsf{nn}(\overline{o}, L)$. Consider the swap $\overline{c} \to \overline{o}$, and the increase in clustering cost as we move from L to $L - \overline{c} + \overline{o}$.

We claim that $\mathrm{local}(\overline{o}) \le \mathrm{ransom}(\overline{c}) + \mathrm{opt}(\overline{o})$ (i.e., (4.6)) holds in this setting. The points for which their clustering is negatively affected (i.e., their clustering price might increase) by the swap are in the set $\mathrm{cluster}(L, \overline{c}) \cup \mathrm{cluster}\left(C_{\mathrm{opt}}, \overline{o}\right)$, and we split this set into two disjoint sets $X = \mathrm{cluster}(L, \overline{c}) \setminus \mathrm{cluster}\left(C_{\mathrm{opt}}, \overline{o}\right)$ and $Y = \mathrm{cluster}\left(C_{\mathrm{opt}}, \overline{o}\right)$.

The increase in price by reassigning the points of X to some other center in L is exactly the ransom of $\overline{c}$. Now, the points of Y might get reassigned to $\overline{o}$, and the change in price of the points of Y can now be bounded by $-\mathrm{local}(\overline{o}) + \mathrm{opt}(\overline{o})$, as was argued in the proof of Lemma 4.8.

Note that it might be that points outside $X \cup Y$ get reassigned to $\overline{o}$ in the clustering induced by $L - \overline{c} + \overline{o}$. However, such reassignment only further reduce the price of the swap. As such, we have that $0 \le \Delta(\overline{c}, \overline{o}) \le \mathrm{ransom}(\overline{c}) - \mathrm{local}(\overline{o}) + \mathrm{opt}(\overline{o})$. As such, summing up the inequality $\mathrm{local}(\overline{o}) \le \mathrm{ransom}(\overline{c}) + \mathrm{opt}(\overline{o})$ over all the centers in $\mathrm{A}_{\mathrm{opt}}$ implies the claim. ■

Lemma 4.11. *Let L be the set of k centers computed by the local search algorithm. We have that* $\|\mathsf{P}_L\|_1 \le 5\mathrm{opt}_1(\mathsf{P}, k)$.

Proof. From the above two lemmas, we have that

$$\begin{aligned}
\|\mathsf{P}_L\|_1 &= \sum_{\overline{o} \in C_{\mathrm{opt}}} \mathrm{local}(\overline{o}) = \sum_{\overline{o} \in S_{\mathrm{opt}}} \mathrm{local}(\overline{o}) + \sum_{\overline{o} \in \mathrm{A}_{\mathrm{opt}}} \mathrm{local}(\overline{o}) \\
&\le 2 \sum_{\overline{c} \in \mathcal{D}} \mathrm{ransom}(\overline{c}) + \sum_{\overline{o} \in S_{\mathrm{opt}}} \mathrm{opt}(\overline{o}) + \sum_{\overline{o} \in \mathrm{A}_{\mathrm{opt}}} \mathrm{ransom}(\mathsf{nn}(\overline{o}, L)) + \sum_{\overline{o} \in \mathrm{A}_{\mathrm{opt}}} \mathrm{opt}(\overline{o}) \\
&\le 2 \sum_{\overline{c} \in L} \mathrm{ransom}(\overline{c}) + \sum_{\overline{o} \in C_{\mathrm{opt}}} \mathrm{opt}(\overline{o}) \le 4\left\|\mathsf{P}_{C_{\mathrm{opt}}}\right\|_1 + \left\|\mathsf{P}_{C_{\mathrm{opt}}}\right\|_1 = 5\mathrm{opt}_1(\mathsf{P}, k),
\end{aligned}$$

by (4.5). ■

4.3.2.2. *Removing the strict improvement assumption.* In the above proof, we assumed that the current local minimum cannot be improved by a swap. Of course, this might not hold for the **algLocalSearchKMed** solution, since the algorithm allows a swap only if it makes "significant" progress. In particular, (4.4) is in fact

$$(4.8) \qquad \forall \overline{c} \in L, \overline{o} \in \mathsf{P} \setminus L, \qquad -\tau \|\mathsf{P}_L\|_1 \le \|\mathsf{P}_{L-\overline{c}+\overline{o}}\|_1 - \|\mathsf{P}_L\|_1.$$

To adapt the proof to use this modified inequality, observe that the proof worked by adding up k inequalities defined by (4.4) and getting the inequality $0 \le 5\left\|\mathsf{P}_{C_{\mathrm{opt}}}\right\|_1 - \|\mathsf{P}_L\|_1$. Repeating the same argumentation on the modified inequalities, which is tedious but straightforward, yields

$$-\tau k \|\mathsf{P}_L\|_1 \le 5\left\|\mathsf{P}_{C_{\mathrm{opt}}}\right\|_1 - \|\mathsf{P}_L\|_1.$$

This implies $\|\mathsf{P}_L\|_1 \leq 5 \left\|\mathsf{P}_{C_{\mathrm{opt}}}\right\|_1 / (1 - \tau k)$. For arbitrary $0 < \varepsilon < 1$, setting $\tau = \varepsilon / 10k$, we have that $\|\mathsf{P}_L\|_1 \leq 5(1 + \varepsilon/5)\mathrm{opt}_1$, since $1/(1 - \tau k) \leq 1 + 2\tau k = 1 + \varepsilon/5$, for $\tau \leq 1/10k$. We summarize:

Theorem 4.12. *Let* P *be a set of n points in a metric space. For* $0 < \varepsilon < 1$*, one can compute a* $(5 + \varepsilon)$*-approximation to the optimal k-median clustering of* P*. The running time of the algorithm is* $O\!\left(n^2 k^3 \frac{\log n}{\varepsilon}\right)$.

4.4. On k-means clustering

In the ***k*-means clustering problem**, a set $\mathsf{P} \subseteq \mathcal{X}$ is provided together with a parameter k. We would like to find a set of k points $\mathbf{C} \subseteq \mathsf{P}$, such that the sum of squared distances of all the points of P to their closest point in $\mathbf{C}$ is minimized.

Formally, given a set of centers $\mathbf{C}$, the k-center clustering ***price*** of clustering P by $\mathbf{C}$ is denoted by

$$\|\mathsf{P}_{\mathbf{C}}\|_2^2 = \sum_{\mathsf{p} \in \mathsf{P}} (\mathbf{d}_{\mathcal{M}}(\mathsf{p}, \mathbf{C}))^2 ,$$

and the ***k*-means problem** is to find a set $\mathbf{C}$ of k points, such that $\|\mathsf{P}_{\mathbf{C}}\|_2^2$ is minimized; namely,

$$\mathrm{opt}_2(\mathsf{P}, k) = \min_{\mathbf{C}, |\mathbf{C}| = k} \|\mathsf{P}_{\mathbf{C}}\|_2^2 .$$

Local search also works for k-means and yields a constant factor approximation. We leave the proof of the following theorem to Exercise 4.4.

Theorem 4.13. *Let* P *be a set of n points in a metric space. For* $0 < \varepsilon < 1$*, one can compute a* $(25 + \varepsilon)$*-approximation to the optimal k-means clustering of* P*. The running time of the algorithm is* $O\!\left(n^2 k^3 \frac{\log n}{\varepsilon}\right)$.

4.5. Bibliographical notes

In this chapter we introduced the problem of clustering and showed some algorithms that achieve constant factor approximations. A lot more is known about these problems including faster and better clustering algorithms, but to discuss them, we need more advanced tools than what we currently have at hand.

Clustering is widely researched. Unfortunately, a large fraction of the work on this topic relies on heuristics or experimental studies. The inherent problem seems to be the lack of a universal definition of what is a good clustering. This depends on the application at hand, which is rarely clearly defined. In particular, no clustering algorithm can achieve all desired properties together; see the work by Kleinberg [**Kle02**] (although it is unclear if all these desired properties are indeed natural or even really desired).

***k*-center clustering.** The algorithm **GreedyKCenter** is by Gonzalez [**Gon85**], but it was probably known before, as the notion of r-packing is much older. The hardness of approximating k-center clustering was shown by Feder and Greene [**FG88**].

***k*-median/means clustering.** The analysis of the local search algorithm is due to Arya et al. [**AGK⁺01**]. Our presentation however follows the simpler proof of Gupta and Tangwongsan [**GT08**]. The extension to k-means is due to Kanungo et al. [**KMN⁺04**]. The extension is not completely trivial since the triangle inequality no longer holds. However, some approximate version of the triangle inequality does hold. Instead of performing a single swap, one can decide to do p swaps simultaneously. Thus, the running time deteriorates since there are more possibilities to check. This improves the approximation constant for the k-median (resp., k-means) to $(3 + 2/p)$ (resp. $(3 + 2/p)^2$). Unfortunately, this is (essentially) tight in the worst case. See [**AGK+01, KMN+04**] for details.

The k-median and k-means clustering are more interesting in Euclidean settings where there is considerably more structure, and one can compute a $(1+\varepsilon)$-approximation in linear time for fixed ε and k and d; see [**HM04**].

Since k-median and k-means clustering can be used to solve the dominating set in a graph, this implies that both clustering problems are NP-Hard to solve exactly.

One can also compute a permutation similar to the greedy permutation (for k-center clustering) for k-median clustering. See the work by Mettu and Plaxton [**MP03**].

Handling outliers. The problem of handling outliers is still not well understood. See the work of Charikar et al. [**CKMN01**] for some relevant results. In particular, for k-center clustering they get a constant factor approximation, and Exercise 4.3 is taken from there. For k-median clustering they present a constant factor approximation using a linear programming relaxation that also approximates the number of outliers. Recently, Chen [**Che08**] provided a constant factor approximation algorithm by extending the work of Charikar et al. The problem of finding a simple algorithm with simple analysis for k-median clustering with outliers is still open, as Chen's work is quite involved.

Open Problem 4.14. Get a *simple* constant factor k-median clustering algorithm that runs in polynomial time and uses exactly m outliers. Alternatively, solve this problem in the case where P is a set of n points in the plane. (The emphasize here is that the analysis of the algorithm should be simple.)

Bi-criteria approximation. All clustering algorithms tend to become considerably easier if one allows trade-off in the number of clusters. In particular, one can compute a constant factor approximation to the optimal k-median/means clustering using $O(k)$ centers in $O(nk)$ time. The algorithm succeeds with constant probability. See the work by Indyk [**Ind99**] and Chen [**Che06**] and references therein.

Facility location. All the problems mentioned here fall into the family of facility location problems. There are numerous variants. The more specific ***facility location*** problem is a variant of k-median clustering where the number of clusters is not specified, but instead one has to pay to open a facility in a certain location. Local search also works for this variant.

Local search. As mentioned above, ***local search*** also works for k-means clustering [**AGK+01**]. A collection of some basic problems for which local search works is described in the book by Kleinberg and Tardos [**KT06**]. Local search is a widely used heuristic for attacking NP-Hard problems. The idea is usually to start from a solution and try to locally improve it. Here, one defines a neighborhood of the current solution, and one tries to move to the best solution in this neighborhood. In this sense, local search can be thought of as a hill-climbing/EM (expectation maximization) algorithm. Problems for which local search was used include vertex cover, traveling salesperson, and satisfiability, and probably many other problems.

Provable cases where local search generates a guaranteed solution are less common and include facility location, k-median clustering [**AGK+01**], weighted max cut, k-means [**KMN+04**], the metric labeling problem with the truncated linear metric [**GT00**], and image segmentation [**BVZ01**]. See [**KT06**] for more references and a nice discussion of the connection of local search to the ***Metropolis algorithm*** and ***simulated annealing***.

4.6. Exercises

Exercise 4.1 (Another algorithm for k-center clustering). Consider the algorithm that, given a point set P and a parameter k, initially picks an arbitrary set $\mathbf{C} \subseteq \mathsf{P}$ of k points. Next, it computes the closest pair of points $\overline{\mathsf{c}}, \overline{\mathsf{f}} \in \mathbf{C}$ and the point s realizing $\|\mathsf{P}_{\mathbf{C}}\|_{\infty}$. If $\mathbf{d}(\mathsf{s}, \mathbf{C}) > \mathbf{d}_{\mathcal{M}}\left(\overline{\mathsf{c}}, \overline{\mathsf{f}}\right)$, then the algorithm sets $\mathbf{C} \leftarrow \mathbf{C} - \overline{\mathsf{c}} + \mathsf{s}$ and repeats this process till the condition no longer holds.

(A) Prove that this algorithm outputs a k-center clustering of radius $\leq 2\mathrm{opt}_{\infty}(\mathsf{P}, k)$.
(B) What is the running time of this algorithm?
(C) If one is willing to trade off the approximation quality of this algorithm, it can be made faster. In particular, suggest a variant of this algorithm that in $O(k)$ iterations computes an $O(1)$-approximation to the optimal k-center clustering.

Exercise 4.2 (Handling outliers). Given a point set P, we would like to perform a k-median clustering of it, where we are allowed to ignore m of the points. These m points are ***outliers*** which we would like to ignore since they represent irrelevant data. Unfortunately, we do not know the m outliers in advance. It is natural to conjecture that one can perform a local search for the optimal solution. Here one maintains a set of k centers and a set of m outliers. At every point in time the algorithm moves one of the centers or the outliers if it improves the solution.

Show that local search does not work for this problem; namely, the approximation factor is not a constant.

Exercise 4.3 (Handling outliers for k-center clustering). Given P, k, and m, present a polynomial time algorithm that computes a constant factor approximation to the optimal k-center clustering of P with m outliers. (Hint: Assume first that you know the radius of the optimal solution.)

Exercise 4.4 (Local search for k-means clustering). Prove Theorem 4.13.