

Chapter 4

Clustering - Definitions and Basic Algorithms

By Sarel Har-Peled, February 4, 2010^①

Do not read this story; turn the page quickly. The story may upset you. Anyhow, you probably know it already. It is a very disturbing story. Everyone knows it. The glory and the crime of Commander Suzdal have been told in a thousand different ways. Don't let yourself realize that the story is the truth.

It isn't. not at all. There's not a bit of truth to it. There is no such planet as Arachosia, no such people as klopts, no such world as Catland. These are all just imaginary, they didn't happen, forget about it, go away and read something else.

– The Crime and Glory of Commander Suzdal, Cordwainer Smith

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 interesting structure in the data. Go!”

The fuzziness arise naturally from the requirement that it would be “interesting”, as this is not well defined and depends on human perception which is sometime impossible to quantify clearly. Similarly, what is “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 bibliographical notes 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 defined between these items. While these items might be points in \mathbb{R}^d and the distance function is just the regular

^①This work is licensed under the Creative Commons Attribution-Noncommercial 3.0 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/3.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

Euclidean distance, it is sometime beneficial to consider the more abstract setting of a general metric space.

Definition 4.1.1 A *metric space* is a pair $(\mathcal{X}, \mathbf{d})$ where \mathcal{X} is a set and $\mathbf{d} : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty)$ 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 a *black-box access* to \mathbf{d} . Namely, given two points $\mathbf{p}, \mathbf{q} \in \mathcal{X}$, we assume that $\mathbf{d}_{\mathcal{M}}(\mathbf{p}, \mathbf{q})$ can be computed in constant time.

The input is a set of n points $\mathbf{P} \subseteq \mathcal{X}$. Given a set of centers \mathbf{C} , every point of \mathbf{P} is assigned to its nearest neighbor in \mathbf{C} . All the points of \mathbf{P} that are assigned to a center \bar{c} form the *cluster* of \bar{c} , denoted by

$$\text{cluster}(\mathbf{C}, \bar{c}) = \left\{ \mathbf{p} \in \mathbf{P} \mid \mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{c}) \leq \mathbf{d}(\mathbf{p}, \mathbf{C}) \right\}, \quad (4.1)$$

where $\mathbf{d}(\mathbf{p}, \mathbf{C}) = \min_{\bar{c} \in \mathbf{C}} \mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{c})$ denotes the *distance* of \mathbf{p} to the set \mathbf{C} . Namely, the center set \mathbf{C} partition \mathbf{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 $\mathbf{P} = \{\mathbf{p}_1, \dots, \mathbf{p}_n\}$, and consider the n dimensional point

$$\mathbf{P}_{\mathbf{C}} = \left(\mathbf{d}(\mathbf{p}_1, \mathbf{C}), \mathbf{d}(\mathbf{p}_2, \mathbf{C}), \dots, \mathbf{d}(\mathbf{p}_n, \mathbf{C}) \right).$$

The i th coordinate of the point $\mathbf{P}_{\mathbf{C}}$ is the cost of assigning \mathbf{p}_i to the closest center in \mathbf{C} .

4.2 On k -Center Clustering

In the *k -center clustering problem*, a set $\mathbf{P} \subseteq \mathcal{X}$ of n points is provided together with a parameter k . We would like to find k points $\mathbf{C} \subseteq \mathcal{X}$, such that the maximum distance of a point in \mathbf{P} to the 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 points. We would like to build k hospitals and minimize the maximum time it takes a patient to arrive to its closest hospital.

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

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

where $\mathbf{d}(\mathbf{p}, \mathbf{C}) = \min_{\bar{c} \in \mathbf{C}} \mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{c})$ denotes the *distance* of \mathbf{p} to the set \mathbf{C} . Note, that every point in a cluster is in distance at most $\|\mathbf{P}_{\mathbf{C}}\|_{\infty}$ from its respective center.

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

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

We will denote the set of centers realizing the optimal clustering by \mathbf{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 $\min_{\mathbf{C}} \max_{\mathbf{p} \in \mathbf{P}} \min_{\bar{c} \in \mathbf{C}} \mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{c})$.

It is known that the k -center clustering is **NP-HARD**, and it is in fact hard to approximate within a factor of 1.86 even in two dimensions. Surprisingly, there is a simple and elegant algorithm that achieves a 2-approximation.

4.2.1 The Greedy Clustering Algorithm

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

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

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

and the bottleneck point \bar{c}_i that realizes it. Next, we add \bar{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 add it to the current set of centers.

To make this algorithm slightly faster, observe that

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

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

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

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

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

Theorem 4.2.1 *Given a set of n points $\mathbf{P} \subseteq \mathcal{X}$, belonging to 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 \mathbf{P} ; namely, $\|\mathbf{P}_{\mathbf{K}}\|_{\infty} \leq 2\text{opt}_{\infty}$, where $\text{opt}_{\infty} = \text{opt}_{\infty}(\mathbf{P}, k)$. 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 $r_k = \|\mathbf{P}_{\mathbf{K}}\|_{\infty}$, and let \bar{c}_{k+1} be the point in \mathbf{P} realizing $r_k = \max_{\mathbf{p} \in \mathbf{P}} \mathbf{d}(\mathbf{p}, \mathbf{K})$. Let $\mathbf{C} = \mathbf{K} \cup \{\bar{c}_{k+1}\}$. Observe that by the definition of r_i , we have that $r_1 \geq r_2 \geq \dots \geq r_k$. Furthermore, for $i < j \leq k + 1$ we have that

$$\mathbf{d}_{\mathcal{M}}(\bar{c}_i, \bar{c}_j) \geq \mathbf{d}(\bar{c}_j, \mathbf{C}_{j-1}) = r_j \geq r_k.$$

Namely, the distance between the closest points in \mathbf{C} is at least r_k . Now, assume for the sake of contradiction that $r_k > 2\text{opt}_{\infty}(\mathbf{P}, k)$. Consider the optimal solution, that covers \mathbf{P} with k balls of radius opt_{∞} . By the triangle inequality, any two points inside such a ball are in distance at

most 2opt_∞ from each other. Thus, none of these balls can cover two points of $\mathbf{C} \subseteq \mathbf{P}$, since the minimum distance in \mathbf{C} is $> 2\text{opt}_\infty$. A contradiction, since then we need $k + 1 = |\mathbf{C}|$ balls of radius opt_∞ to cover \mathbf{P} . ■

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 a point $\mathbf{p} \in \mathbf{P}$, and let $\bar{\mathbf{c}}$ be the center it belongs to in C_{opt} . Also, let $\bar{\mathbf{g}}$ be the center of \mathbf{K} that is in cluster $(C_{\text{opt}}, \bar{\mathbf{c}})$. We have that $\mathbf{d}(\mathbf{p}, \bar{\mathbf{c}}) = \mathbf{d}(\mathbf{p}, C_{\text{opt}}) \leq \text{opt}_\infty = \text{opt}_\infty(\mathbf{P}, k)$. Similarly, observe that $\mathbf{d}(\bar{\mathbf{g}}, \bar{\mathbf{c}}) = \mathbf{d}(\bar{\mathbf{g}}, C_{\text{opt}}) \leq \text{opt}_\infty$. As such, by the triangle inequality, we have that $\mathbf{d}(\mathbf{p}, \bar{\mathbf{g}}) \leq \mathbf{d}(\mathbf{p}, \bar{\mathbf{c}}) + \mathbf{d}(\bar{\mathbf{c}}, \bar{\mathbf{g}}) \leq 2\text{opt}_\infty$.

By the pigeon hole principle, the only other possibility is that there are two centers $\bar{\mathbf{g}}$ and $\bar{\mathbf{h}}$ of \mathbf{K} that are both in cluster $(C_{\text{opt}}, \bar{\mathbf{c}})$, for some $\bar{\mathbf{c}} \in C_{\text{opt}}$. Assume, without loss of generality, that $\bar{\mathbf{h}}$ was added later to the center set \mathbf{K} by the algorithm **GreedyKCenter**, say in the i th iteration. But then, since **GreedyKCenter** always chooses the point furthest away from the current set of centers, we have that $\bar{\mathbf{c}} \in \mathbf{C}_{i-1}$ and

$$\|\mathbf{P}_{\mathbf{K}}\|_\infty \leq \|\mathbf{P}_{\mathbf{C}_{i-1}}\|_\infty = \mathbf{d}(\bar{\mathbf{h}}, \mathbf{C}_{i-1}) \leq \mathbf{d}(\bar{\mathbf{h}}, \bar{\mathbf{g}}) \leq \mathbf{d}(\bar{\mathbf{h}}, \bar{\mathbf{c}}) + \mathbf{d}(\bar{\mathbf{c}}, \bar{\mathbf{g}}) \leq 2\text{opt}_\infty. \quad \blacksquare$$

4.2.2 The greedy permutation

There is an interesting phenomena associated with **GreedyKCenter**. We can ran it till it exhausts all the points of \mathbf{P} (i.e., $k = n$). Then, this algorithm generates a permutation of \mathbf{P} ; that is $\mathbf{P} = \langle \bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2, \dots, \bar{\mathbf{c}}_n \rangle$. We will refer to \mathbf{P} as the *greedy permutation* of \mathbf{P} . There is also an associated sequence of radiuses $\langle r_1, r_2, \dots, r_n \rangle$, where all the points of \mathbf{P} are in distance at most r_i from the points of $\mathbf{C}_i = \langle \bar{\mathbf{c}}_1, \dots, \bar{\mathbf{c}}_i \rangle$.

Definition 4.2.2 A set $S \subseteq \mathbf{P}$ is a *r -packing* for \mathbf{P} if the following two properties hold.

- (i) **Covering property:** All the points of \mathbf{P} are in distance at most r from the points of S .
 - (ii) **Separation property:** For any pair point of points $\mathbf{p}, \mathbf{q} \in S$, we have that $\mathbf{d}_{\mathcal{M}}(\mathbf{p}, \mathbf{q}) \geq r$.
- (One can relax the separation property by requiring that the points of S would be at distance $\Omega(r)$ apart.)

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

Theorem 4.2.3 *Let \mathbf{P} be a set of n points in a finite metric space, and let its greedy permutation be $\langle \bar{\mathbf{c}}_1, \bar{\mathbf{c}}_2, \dots, \bar{\mathbf{c}}_n \rangle$ with the associated sequence of radiuses $\langle r_1, r_2, \dots, r_n \rangle$. For any i , we have that $\mathbf{C}_i = \langle \bar{\mathbf{c}}_1, \dots, \bar{\mathbf{c}}_i \rangle$ is a r_i -packing of \mathbf{P} .*

Proof: Note, that by construction $r_k = \mathbf{d}(\bar{\mathbf{c}}_k, \mathbf{C}_{k-1})$, for all $k = 1, \dots, n$. As such, for $j < k < i \leq n$, we have that $\mathbf{d}_{\mathcal{M}}(\bar{\mathbf{c}}_j, \bar{\mathbf{c}}_k) \geq r_k \geq r_i$, which implies the required separation property. The covering property follows by definition, see Eq. (4.2)_{p59}. ■

^②Mark 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 a 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.

4.3 On k -median clustering

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

Formally, given a set of centers C , the k -media clustering *price* of clustering P by C is denoted by

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

where $\mathbf{d}(p, C) = \min_{\bar{c} \in C} \mathbf{d}_{\mathcal{M}}(p, \bar{c})$ denotes the *distance* of p to the set C .

Formally, the *k -median problem* is to find a set C of k points, such that $\|P_C\|_1$ is minimized; namely,

$$\text{opt}_1(P, k) = \min_{C, |C|=k} \|P_C\|_1.$$

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 notations. Consider the set $U = \{P_C \mid C \in P^k\}$. Clearly, we have that $\text{opt}_{\infty}(P, k) = \min_{q \in U} \|q\|_{\infty}$, and $\text{opt}_1(P, k) = \min_{q \in U} \|q\|_1$.

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 . Since ℓ_1 and ℓ_{∞} are equal up to a factor equal to the dimension, we get the following.

Observation 4.3.1 For any point set P of n points and a parameter k , we have that $\text{opt}_{\infty}(P, k) \leq \text{opt}_1(P, k) \leq n \text{opt}_{\infty}(P, k)$.

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 $\text{opt}_1 = \text{opt}_1(P, k)$

4.3.1.1 The algorithm

A 2n-approximation. Observation 4.3.1 implies that if we compute a set of centers L using Theorem 4.2.1 then we have that

$$\|P_L\|_1 / 2n \leq \|P_L\|_{\infty} / 2 \leq \text{opt}_{\infty}(P, k) \leq \text{opt}_1 \leq \|P_L\|_1 \implies \text{opt}_1 \leq \|P_L\|_1 \leq 2n \text{opt}_1. \quad (4.3)$$

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 . 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 $|\mathbf{P}| |L_{\text{curr}}| = nk$ choices to consider, as we pick a center $\bar{c} \in L_{\text{curr}}$ to throw away and a new center to replace it by $\bar{o} \in (\mathbf{P} \setminus L_{\text{curr}})$. We consider the new candidate set of centers $\mathbf{K} \leftarrow (L_{\text{curr}} \setminus \{\bar{c}\}) \cup \{\bar{o}\}$. If $\|\mathbf{P}_{\mathbf{K}}\|_1 \leq (1 - \tau) \|\mathbf{P}_{L_{\text{curr}}}\|_1$ then the algorithm sets $L_{\text{curr}} \leftarrow \mathbf{K}$. 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

Note, that the running time of the algorithm is

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

by Eq. (4.3) and since $1/(1-\tau) \geq 1+\tau$. The final step follows since $1+\tau \leq \exp(\tau) \leq 1+2\tau$, for $\tau < 1/2$ as can be easily verified. 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 can not be too far from the true optimal solution.

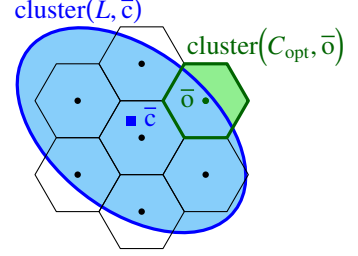
For the sake of simplicity of exposition, let us assume (for now) that the solution returned by the algorithm can not be improved (at all) by any swap, and let L be this set of centers. For a center $\bar{c} \in L$ and a point $\bar{o} \in \mathbf{P} \setminus L$, let $L - \bar{c} + \bar{o} = (L \setminus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

$$\forall \bar{c} \in L, \bar{o} \in \mathbf{P} \setminus L \quad 0 \leq \Delta(\bar{c}, \bar{o}) = \nu_1(L - \bar{c} + \bar{o}) - \nu_1(L). \quad (4.4)$$

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

From local clustering, to local clustering complying with the optimal clustering. One can modify the assignment of points to clusters in the locally optimal clustering so that the resulting clustering of P comply with the optimal partition; that is, every cluster in the optimal clustering would be contained in a single cluster of the modified local solution.

Furthermore, this modified local solution Π is not much more expensive. Now, in this partition there are many beneficial swaps (by making it into the optimal clustering). But these swaps can not be too profitable, since then they would have been profitable for the original local solution. This would imply that the local solution can not 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.

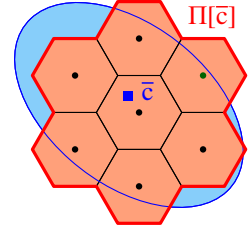


In the following, we denote by $\text{nn}(\mathbf{p}, X)$ the nearest-neighbor to \mathbf{p} in the set X .

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

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

Now, for any center $\bar{o} \in C_{\text{opt}}$, let $\text{nn}(\bar{o}, L)$ be its nearest neighbor in L , and observe that $\text{cluster}(C_{\text{opt}}, \bar{o}) \subseteq \Pi[\text{nn}(\bar{o}, L)]$ (see Eq. (4.1)_{p58}). The picture on the right shows the resulting modified cluster for the above example.



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

Lemma 4.3.2 *The increase in cost from moving from the clustering induced by L to the clustering of Π is bounded by $2 \|\mathbf{P}_{C_{\text{opt}}}\|_1$. That is $\sum_{\mathbf{p} \in P} \delta_{\mathbf{p}} \leq 2 \|\mathbf{P}_{C_{\text{opt}}}\|_1$.*

Proof: For a point $\mathbf{p} \in P$, let $\bar{c} = \text{nn}(\mathbf{p}, L)$ be its local center, $\bar{o} = \text{nn}(\mathbf{p}, C_{\text{opt}})$ be its optimal center, and let $\alpha(\mathbf{p}) = \text{nn}(\bar{o}, L)$ be its new assigned center in Π . Observe that $\mathbf{d}_{\mathcal{M}}(\mathbf{p}, \alpha(\mathbf{p})) = \mathbf{d}_{\mathcal{M}}(\bar{o}, \text{nn}(\bar{o}, L)) \leq \mathbf{d}_{\mathcal{M}}(\bar{o}, \bar{c})$. As such, by the triangle inequality, we have that

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

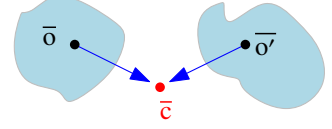
As such, $\delta_{\mathbf{p}} = \mathbf{d}_{\mathcal{M}}(\mathbf{p}, \alpha(\mathbf{p})) - \mathbf{d}(\mathbf{p}, L) \leq 2\mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{o}) + \mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{c}) - \mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{c}) = 2\mathbf{d}_{\mathcal{M}}(\mathbf{p}, \bar{o}) = 2\mathbf{d}(\mathbf{p}, C_{\text{opt}})$.

As such, $\sum_{\mathbf{p} \in P} \delta_{\mathbf{p}} \leq \sum_{\mathbf{p} \in P} 2\mathbf{d}(\mathbf{p}, C_{\text{opt}}) = 2 \|\mathbf{P}_{C_{\text{opt}}}\|_1$. ■

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 $\bar{c} \in L$ its *degree*, denoted by $\text{deg}(\bar{c})$, be the number of points of C_{opt} mapped to it by this nearest neighbor mapping.

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

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



Similarly, the points of cluster $\text{cluster}(C_{\text{opt}}, \bar{o})$ might be in trouble if we swap $\bar{c} \rightarrow \bar{o}'$.

Intuitively, since we now think about the local solution as being 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 \bar{c} from all its client in the local partition. Formally, the *ransom* of a center \bar{c} , that is a drifter, is $\text{ransom}(\bar{c}) = \sum_{p \in \text{cluster}(L, \bar{c})} \delta_p$. This is the price of reassigning all the points that are currently served by the drifter \bar{c} to some other center in L . Once this ransom is paid, \bar{c} serves nobody, and can be moved with no further charge.

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

$$\text{ransom}(\bar{c}) = \sum_{p \in \text{cluster}(L, \bar{c}) \setminus \Pi[\bar{c}]} \delta_p.$$

This is the increase in cost of reassigning the points of cluster (L, \bar{c}) when moving from the local clustering of L to the clustering of Π .

Observe that, by Lemma 4.3.2, we have that

$$\sum_{\bar{c} \in L} \text{ransom}(\bar{c}) \leq 2 \|\mathbf{P}_{C_{\text{opt}}}\|_1. \quad (4.5)$$

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

$$\text{opt}(\bar{o}) = \sum_{p \in \text{cluster}(C_{\text{opt}}, \bar{o})} \mathbf{d}(p, C_{\text{opt}}) \quad \text{and} \quad \text{local}(\bar{o}) = \sum_{p \in \text{cluster}(C_{\text{opt}}, \bar{o})} \mathbf{d}(p, L),$$

respectively.

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

$$0 \leq \Delta(\bar{c}, \bar{o}) \leq \text{ransom}(\bar{c}) - \text{local}(\bar{o}) + \text{opt}(\bar{o}) \quad \implies \quad \text{local}(\bar{o}) \leq \text{ransom}(\bar{c}) + \text{opt}(\bar{o}). \quad (4.6)$$

Indeed, \bar{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 cluster $(C_{\text{opt}}, \bar{o})$ instead of paying $\text{local}(\bar{o})$ are now paying (at most) $\text{opt}(\bar{o})$.

Eq. (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 the optimal center mapped to it, defines a natural swap which is relatively cheap. Putting all of these together would 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 $\text{nn}(\cdot, L)$. 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} , $k = |C_{\text{opt}}| = |L|$, and $\#_{\text{anchors}} = |A_{\text{opt}}|$. As such, we have that

$$\begin{aligned} |C_{\text{opt}}| &= |S_{\text{opt}}| + |A_{\text{opt}}| = \#_{\text{tyrants}} + \#_{\text{anchors}} + \#_{\text{drifters}} = |L| \\ \implies \#_{\text{drifters}} &= |S_{\text{opt}}| - \#_{\text{tyrants}} \geq |S_{\text{opt}}|/2, \end{aligned} \quad (4.7)$$

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

Lemma 4.3.3 *We have that* $\sum_{\bar{o} \in S_{\text{opt}}} \text{local}(\bar{o}) \leq 2 \sum_{\bar{c} \in \mathcal{D}} \text{ransom}(\bar{c}) + \sum_{\bar{o} \in S_{\text{opt}}} \text{opt}(\bar{o})$.

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

So, let \bar{c} be the drifter with the lowest $\text{ransom}(\bar{c})$. For any $\bar{o} \in S_{\text{opt}}$, we have that $\text{local}(\bar{o}) \leq \text{ransom}(\bar{c}) + \text{opt}(\bar{o})$, by Eq. (4.6). Summing over all such centers, we have that

$$\sum_{\bar{o} \in S_{\text{opt}}} \text{local}(\bar{o}) \leq |S_{\text{opt}}| \text{ransom}(\bar{c}) + \sum_{\bar{o} \in S_{\text{opt}}} \text{opt}(\bar{o}),$$

which is definitely smaller than the stated bound, since $|S_{\text{opt}}| \leq 2|\mathcal{D}|$, by Eq. (4.7)_{p65}. ■

Lemma 4.3.4 *We have that* $\sum_{\bar{o} \in A_{\text{opt}}} \text{local}(\bar{o}) \leq \sum_{\bar{o} \in A_{\text{opt}}} \text{ransom}(\text{nn}(\bar{o}, L)) + \sum_{\bar{o} \in A_{\text{opt}}} \text{opt}(\bar{o})$.

Proof: For a center $\bar{o} \in A_{\text{opt}}$, its anchor is $\bar{c} = \text{nn}(\bar{o}, L)$. Consider the swap $\bar{c} \rightarrow \bar{o}$.

We claim that Eq. (4.6) still holds in this settings. Indeed, moving \bar{c} to \bar{o} first requires \bar{c} to pay its ransom since the points of $\text{cluster}(L, \bar{c}) \setminus \text{cluster}(C_{\text{opt}}, \bar{o})$ needs to be reassigned. Now, the change in price of the points of $\text{cluster}(C_{\text{opt}}, \bar{o})$ can now be bounded by $-\text{local}(\bar{o}) + \text{opt}(\bar{o})$, as above.

As such, summing up Eq. (4.6) over all the centers in A_{opt} implies the claim. ■

Lemma 4.3.5 *Let L be the set of k centers computed by the local search algorithm. We have that* $\|P_L\|_1 \leq 5\text{opt}_1(P, k)$.

Proof: From the above two lemmas, we have that

$$\begin{aligned} \|P_L\|_1 &\leq \sum_{\bar{o} \in C_{\text{opt}}} \text{local}(\bar{o}) = \sum_{\bar{o} \in S_{\text{opt}}} \text{local}(\bar{o}) + \sum_{\bar{o} \in A_{\text{opt}}} \text{local}(\bar{o}) \\ &\leq 2 \sum_{\bar{c} \in \mathcal{D}} \text{ransom}(\bar{c}) + \sum_{\bar{o} \in S_{\text{opt}}} \text{opt}(\bar{o}) + \sum_{\bar{o} \in A_{\text{opt}}} \text{ransom}(\text{nn}(\bar{o}, L)) + \sum_{\bar{o} \in A_{\text{opt}}} \text{opt}(\bar{o}) \\ &\leq 2 \sum_{\bar{c} \in L} \text{ransom}(\bar{c}) + \sum_{\bar{o} \in C_{\text{opt}}} \text{opt}(\bar{o}) \leq 4 \|P_{C_{\text{opt}}}\|_1 + \|P_{C_{\text{opt}}}\|_1 = 5\text{opt}_1(P, k). \end{aligned}$$

by Eq. (4.5). ■

4.3.2.2 Removing the strict improvement assumption

In the above proof, we assumed that the current local minimum can not be improved by a swap. Of course, this might not hold for the algorithm solution, since the algorithm allows a swap only if it makes “significant” progress. In particular, Eq. (4.4) is in fact

$$\forall \bar{c} \in \mathbf{C}, \bar{o} \in \mathbf{P} \setminus \mathbf{C} \quad -\tau \|\mathbf{P}_{\mathbf{C}}\|_1 \leq \|\mathbf{P}_{\mathbf{C}-\bar{c}+\bar{o}}\|_1 - \|\mathbf{P}_{\mathbf{C}}\|_1. \quad (4.8)$$

To adapt the proof to use this modified inequalities, observe that the proof worked by adding up k inequalities defined by Eq. (4.4) and showing that the right side is bounded by $5 \|\mathbf{P}_{\mathbf{C}_{\text{opt}}}\|_1 - \|\mathbf{P}_{\mathbf{C}}\|_1$. Repeating the same argumentation on the modified inequalities, would yield

$$-\tau k \|\mathbf{P}_{\mathbf{C}}\|_1 \leq 5 \|\mathbf{P}_{\mathbf{C}_{\text{opt}}}\|_1 - \|\mathbf{P}_{\mathbf{C}}\|_1.$$

This implies $\|\mathbf{P}_{\mathbf{C}}\|_1 \leq 5 \text{opt}_1 / (1 - \tau k)$. For arbitrary $0 < \varepsilon < 1$, Setting $\tau = \varepsilon / 2k$ we have that $\|\mathbf{P}_{\mathbf{C}}\|_1 \leq 5(1 + \varepsilon k) \text{opt}_1$, since $1 / (1 - \tau k) \leq 1 + 2\tau k = 1 + \varepsilon$, for $\tau \leq 1/2k$. We summarize:

Theorem 4.3.6 *Let \mathbf{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 \mathbf{P} . The running time of the algorithm is $O(n^2 k^3 \frac{\log n}{\varepsilon})$.*

4.4 On k -means clustering

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

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

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

$$v_2^{\text{opt}}(\mathbf{P}, k) = \min_{\mathbf{C}, |\mathbf{C}|=k} \|\mathbf{P}_{\mathbf{C}}\|_2^2.$$

Local search also works for k -means and yields a constant factor approximation, we leave the proof of the following theorem to Exercise 4.6.4.

Theorem 4.4.1 *Let \mathbf{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 \mathbf{P} . The running time of the algorithm is $O(n^2 k^3 \frac{\log n}{\varepsilon})$.*

4.5 Bibliographical Notes

In this chapter we introduced the problem of clustering and showed some algorithms that achieve constant factor approximation. 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 rely 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, as 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 really desired).

***k*-center Clustering.** The algorithm **GreedyKCenter** was described 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 *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*-mean clustering are more interesting in the Euclidean settings where there is considerably more structure, and one can compute $(1 + \varepsilon)$ -approximation in polynomial time for fixed ε . We will return to this topic later.

Since *k*-median and *k*-means clustering can be easily be reduced to **Dominating Set** in a graph, this implies that both clustering problems are **NP-HARD** to solve exactly.

One can compute a similar permutation to the greedy permutation (for *k*-center clustering) also 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 4.6.3 is taken from there. For *k*-median clustering they present a constant factor approximation using linear programming relaxation, that also approximates the number of outliers. Recently, Chen [Che07] 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 4.5.1 *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 to 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. Problem for which local search was used include **Vertex Cover**, **Traveling Salesperson** and **Satisfiability**, and probably many more problems.

Provable cases where local search generates a guaranteed solution are less common and include facility location, k -median clustering [AGK⁺01], k -means clustering [KMN⁺04], weighted max cut, 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 4.6.1 (Another algorithm for k -center clustering.) [5 Points]

Consider the algorithm that given a point set P , and a parameter k , initially picks an arbitrary set $C \subseteq P$ of k points. Next, it computes the closet pair of points $\bar{c}, \bar{f} \in C$, and the point s realizing $\|P_C\|_\infty$. If $d(s, C) > d_M(\bar{c}, \bar{f})$ then the algorithm sets $C \leftarrow C - \bar{c} + \bar{f}$ and repeat this process till the condition no longer holds.

- (A) Prove that this algorithm outputs a k -center clustering of radius $\leq 2\text{opt}_\infty(P, k)$.
- (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 $O(1)$ -approximation to the optimal k -center clustering..

Exercise 4.6.2 (Handling outliers.) [10 Points]

Given a point set P , we would like to perform k -median clustering of it, when 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 ones maintain 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 4.6.3 (Handling outliers for k -center clustering.) [10 Points]

Given P , k and m , present an 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 what is the radius of the optimal solution.)

Exercise 4.6.4 (Local search for k -means clustering.) [10 Points]

Prove Theorem 4.4.1.

Bibliography

- [AGK⁺01] V. Arya, N. Garg, R. Khandekar, K. Munagala, and V. Pandit. Local search heuristic for k -median and facility location problems. In *Proc. 33rd Annu. ACM Sympos. Theory Comput.*, pages 21–29, 2001.
- [BVZ01] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001.
- [Che06] K. Chen. On k -median clustering in high dimensions. In *Proc. 17th ACM-SIAM Sympos. Discrete Algorithms*, pages 1177–1185, 2006.
- [Che07] K. Chen. A constant factor approximation algorithm for k -median with outliers. manuscript, 2007.
- [CKMN01] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan. Algorithms for facility location problems with outliers. In *Proc. 12th ACM-SIAM Sympos. Discrete Algorithms*, pages 642–651, 2001.
- [FG88] T. Feder and D. H. Greene. Optimal algorithms for approximate clustering. In *Proc. 20th Annu. ACM Sympos. Theory Comput.*, pages 434–444, 1988.
- [Gon85] T. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoret. Comput. Sci.*, 38:293–306, 1985.
- [GT00] A. Gupta and E. Tardos. A constant factor approximation algorithm for a class of classification problems. In *Proc. 32nd Annu. ACM Sympos. Theory Comput.*, pages 652–658, 2000.
- [GT08] A. Gupta and K. Tangwongsan. Simpler analyses of local search algorithms for facility location. *CoRR*, abs/0809.2554, 2008.

- [Ind99] P. Indyk. Sublinear time algorithms for metric space problems. In *Proc. 31st Annu. ACM Sympos. Theory Comput.*, pages 154–159, 1999.
- [Kle02] J. Kleinberg. An impossibility theorem for clustering. In *Neural Info. Proc. Sys.*, 2002.
- [KMN⁺04] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. A local search approximation algorithm for k -means clustering. *Comput. Geom. Theory Appl.*, 28:89–112, 2004.
- [KT06] J. Kleinberg and E. Tardos. *Algorithm design*. Addison-Wesley, 2006.
- [MP03] R. R. Mettu and C. G. Plaxton. The online median problem. *SIAM J. Comput.*, 32(3):816–832, 2003.