# Analysis of Complete-Linkage, Design of Algorithms for the Separated $k$-Clustering and Euclidean $k$-MSR Problems

Inaugural-Dissertation

zur Erlangung des Doktorgrades der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Julian Wargalla
aus Troisdorf

Düsseldorf, Juni 2024

aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Berichterstatter:

1. Prof. Dr. Melanie Schmidt

2. Prof. Dr. Heiko Röglin

Tag der mündlichen Prüfung: 23.07.2024

# Abstract

Cluster analysis is a major field within (exploratory) data analysis. Its goal is to classify objects according to degrees of proximity or (dis)similarity in such a way that the resulting clusters are relevant to whatever problem is at hand. In this dissertation, we establish new theoretical results for several clustering algorithms and the objectives that evaluate their output.

After a general introduction, in which we broadly outline the tasks of cluster analysis and introduce the main objective functions used to evaluate solutions, we turn to the Complete-Linkage algorithm in Chapter 2, a major algorithm for computing hierarchical clusterings. We improve the previously best-known level-by-level lower bounds for its approximation ratio on general metrics from $\Omega(\log k)$, established by Dasgupta and Long, to $\Omega(k)$ and report the first non-trivial upper bounds. For $\mathbf{CL}^{\text{rad}}$, which greedily optimizes the $k$-center objective, we match the lower bound by showing that the algorithm always computes $O(k)$-approximations. For $\mathbf{CL}^{\text{diam}}$, which greedily optimizes the $k$-diameter objective and which is much harder to analyze, we prove an upper bound of $O(k^2)$.

In Chapter 3, we analyze how well various important objectives, such as $k$-center, $k$-diameter, $k$-median, $k$-means, and $k$-diameter can be combined with the $k$-separation objective. The former ensure that objects within the same cluster are not too dissimilar, whereas the latter ensures that objects within different clusters are not too similar. Since no solutions have to exist that approximate two objectives simultaneously, we instead consider approximations of the Pareto sets of the corresponding bi-objective clustering problems.

In Chapter 4, we study the Euclidean $k$-center PTAS for constant $k$ established by Bădoiu et al. and modify it in such a way that it yields a PTAS for the Euclidean $k$-MSR problem with constant $k$ and logarithmically many outliers. Although similar results have recently been published, this is, to the best of our knowledge, the first algorithm to explicitly do this.

Chapter 5, a relatively short chapter, then concludes the dissertation by showcasing some preliminary results for the connected $k$-center problem.

# Contents

# Chapter 1

# Introduction

## 1.1 The Problem(s) of Cluster Analysis

Let us begin by trying to circumscribe, very roughly at first, the problem(s) central to cluster analysis. Cluster analysis as a whole belongs to the field of (exploratory) data analysis and refers to "the process of classifying objects into subsets that have meaning in the context of a particular problem."[1] It is the "formal study of algorithms and methods"[2] that try to make 'visible' or uncover underlying structures or patterns of organization that are relevant for whatever problem is under consideration. Cluster analysis is not concerned with individual objects themselves but rather with their relations. As the term 'cluster' already indicates, the sought-after partitionings derive from degrees of proximity between the considered objects: generally speaking, similar objects should belong to the same cluster, whereas dissimilar objects should belong to different clusters. "All the real knowledge which we possess, depends on methods by which we distinguish the similar from the dissimilar."[3] Trying to draw such distinctions and thereby distribute beings into classes has a long history that can be traced back all the way to the taxonomical considerations of Aristotle, even if the setting is very different. Relations of similarity or dissimilarity should not be taken as absolute in any way but rather concern measurements that have been conducted in a particular context. They might be implicitly part of the data set and derive from distances between points representing objects in some (geo-)metric space, or they might derive explicitly from numerical values assigned to each pair of objects. In the latter case, ob-

[1] [58, p. 55]

[2] [58, p. 1]

[3] This short quote from Carl von Linné's 1737 "Genera Planetarium" nicely illustrates the broad thrust. We have taken this quote from [40], where a longer version can also be found. In an earlier edition, Everitt also wrote, "A cluster is a set of entities that are alike, and entities from different clusters are not alike."

jects are not even required to have any content and are instead fully determined through their relations with other objects. Before going into any more detail, let us consider some particular examples.

1. *Sequence clustering*: Given a data set consisting of biological sequences, such as nucleic or, more generally, amino acid sequences, the goal of sequence clustering "is to predict homology and function, reduce redundancy, generate subsets that are tractable for more computationally expensive methods, compare data from different environments and quantify ecosystem diversity."[4] Consider, for example, the task of predicting the function of a protein, which to a large extent depends on its three-dimensional shape (its tertiary structure), solely from its amino acid sequence (its primary structure), which is much easier to analyze. The task is not necessarily to fit unfamiliar proteins into pre-established classes of known behavior (although this is a possible application — we will come back to this), but first and foremost to find commonalities in structure and thereby possibly function as well. As Sander and Schneider [69] note, structural homology can be inferred reasonably well from sequence homology when the alignment length (i.e., the length of the subsequences that could be matched) is large enough. On the one hand, this helps in predicting the function of certain proteins by relating them to other proteins with already-known behavior. On the other, one might even tentatively posit new classes of proteins, predicting that their behavior, despite currently unknown, will be similar. So this, then, is the goal: to cluster a given set of sequences into (a small number of) clusters, such that proteins in one cluster are structurally similar or homologous while proteins from different clusters are not. Such clusterings also allow one to simplify, or rather, summarize data sets (of which some can contain up to billions of protein sequences, see [74]) by storing from each cluster only a single sequence that nicely represents the whole cluster. Clustering algorithms such as CD-HIT [44], UCLUST [38], and Linclust [74] have been developed with this purpose in mind.

2. *Image segmentation:* In general, image segmentation aims to partition a given image into regions of interest, which might, for example, correspond to depicted objects. "It is a pre-processing phase of many image-based applications like biometric identification, medical imaging, object detection and classification, and pattern recognition."[5] Pixels can be clustered according to their location within the picture, their color, the intensity of those colors,

[4] [38, p. 1]
[5] [66, p. 1]

etc. In contrast to the preceding example, the measure is one of dissimilarity or distance within some metric space. A prominent application of image segmentation is the task of finding abnormalities in brain MRI images that could indicate tumors. Automated tumor detection can complement existing medical procedures and assist professionals in their work, especially where procedures can be very taxing.[6]

3. *Community detection in social networks:* This is a vast topic, and we are only interested in providing a short but hopefully sufficient example. At its simplest, a social network consists of an (un-)directed graph whose vertices represent people and whose edges represent acquaintance. Based on these relationships, one might try to identify or even partition the network into subgraphs with a high density of internal connections (meaning, for example, that their diameters are small). These could help track social influences, analyze current or emerging trends, and detect anomalies.

In each of these cases, despite the vast differences between the types of data considered, the goal is to classify or cluster objects (biological sequences, pixels, people, etc.) according to their measured similarity or dissimilarity. If these measurements are not entirely arbitrary but exhibit some structure or satisfy certain axioms (for example, when a dissimilarity measure forms a metric), then certain analyzable patterns start to emerge that could provide new insight into the data set or into the underlying problem that the data set traces. With cluster analysis, it all depends on how a given set of objects is distributed, whether relative to each other or positionally within some ambient space.

To further flesh out cluster analysis, it is helpful to contrast it with other approaches that resemble it, at least on a surface level. Consider the following statement by Jain and Dubes: "intrinsic classification is the essence of cluster analysis."[7] This quote places cluster analysis into the field of unsupervised learning and thereby opposes it to the extrinsic classification schemes of supervised learning.[8] The latter is less exploratory and presupposes that the classes into which objects are to be distributed are already known in advance to a certain degree, i.e. that they are already filled with some content, however minimal. Extrinsic classification schemes extract features and derive rules from an already correctly labeled set of example objects, the training set, that help assign newly encountered objects to the correct classes. In other words, the rules that measure the internal resemblance of objects to the pre-established classes should nicely extrapolate

[6]An image segmentation approach to this problem is outlined in [71].

[7] [58, p. 57]

[8] numerous books provide a quick introduction to these terms and machine learning in general, such as [70].

from the training set to other possible objects on a case-by-case basis. Consider, for example, the task of detecting skin cancer, or, more generally, of classifying skin lesions, such as moles, as to whether they might be malignant or not. This obviously important task — skin cancer is one of the most common forms of cancer — is very well suited to a supervised learning approach and has received much attention in the field (see [73, 72, 75, 67]). Instruments such as convolutional neural networks (CNNs) trained on large data sets of pictures of skin lesions that have already been correctly annotated by professionals achieve quite good results. For a comprehensive overview of this topic, see, for example, [79].

This is not true for cluster analysis or unsupervised learning in general. "The objective of cluster analysis is simply to find a convenient and valid organization of the data, not to establish rules for separating future data into categories."[9] None of the classes formed during cluster analysis are posited or filled with content prior to the task at hand; the internal constitution of objects is completely ignored in favor of their extrinsic relations. The goal is not to distribute unseen objects into already existing classes but rather to develop new classifications in the first place. However, while the clusters themselves are not specified beforehand, the overall clustering, or more generally, the method that produced it, still has to be evaluated in one way or another — different contexts or tasks just might favor different kinds of classifications, even for the same data set. Depending on the overall approach, it might be possible to compute clusterings using some heuristic and evaluate them purely empirically, but this is sometimes not possible and also not always desirable. From a theoretical perspective, we would instead like to ensure that clusterings or rather clustering algorithms satisfy certain a priori guarantees, which might concern:

1. Quantitative assessments which measure how well a given clustering attains a certain goal. For example, we could try to minimize the largest measured dissimilarity of objects placed into the same cluster — this would ensure that the clusters are not too spread out. Most clustering problems are formalized as optimization problems for certain objective functions and thus fall into this category.

2. Qualitative assessments which describe certain notions of structural or "functorial" stability of clustering algorithms. If clusterings highlight or illuminate certain fundamental structures of data sets, then similar data sets should yield similar clusterings. More precisely, if we change the data set in a controlled fashion, then the output of a good clustering algorithm should also change in a controlled fashion.

[9] [58, p. 15]

Whereas quantitative approaches consider solutions only in terms of the distances that separate them from optimal solutions in a linear order arranged by some objective function, and algorithms in terms of their worst-case approximation ratios, qualitative approaches directly consider the structural behavior of clustering algorithms themselves. One of the first papers to try to establish qualitative results was published in 2002 by Kleinberg ([61]). He lists three "natural properties", or axioms, that he thinks clustering algorithms should satisfy: scale-invariance, richness, and consistency. The first requires that the algorithm should be scale-invariant, i.e., the output should not change if all numerical measurements are uniformly scaled. The second requires that we can, in principle, always adjust the measurements in such a way that the algorithm yields any desired clustering specified beforehand. The last requires that the output of a clustering algorithm should not change if we decrease intra-cluster dissimilarity and increase inter-cluster dissimilarity. This third axiom is arguably too strong[10] and so, although Kleinberg proves that no such clustering algorithm can exist, the paper still allowed for subsequent research, such as the work of Carlsson and Mémoli (see for example [20, 21]), as well as Ackerman et al. (see for example [1, 2]). In the end, although being a very interesting field, research on qualitative assessments is much sparser than on quantitative assessments. One of the reasons might be that this line of research has put more emphasis on analyzing and understanding already existing algorithms and less on establishing new ones. Several axioms have been posited, but they also seem to have been somewhat exhausted. Another reason might be that such structural characteristics are not sufficient in themselves. Single-linkage, one of the structurally most stable hierarchical algorithms, is not very useful in practice since it quickly yields large clusters.

In this dissertation, we will instead focus on establishing results for quantitative assessments; every clustering problem will be framed as an optimization problem for some objective function. This treatment, of course, does not exhaust all of cluster analysis and warrants certain criticism, but from a theoretical perspective, it is highly significant. One issue is that the gap that it leaves between theoretical results and practical application can, at times, be quite large. Although each objective captures something desirable for any clustering whatsoever, their relevance might differ depending on the task at hand. For many concrete applications, it is unclear which objective function should best be optimized since they are often not tied directly to them. Here, some of the exploratory aspect of cluster analysis comes back, and, based on some prior intuition, different algorithms have to be weighed against each other. Although this is somewhat unsatisfying, it is also unavoidable. At some point, knowledge about the concrete application has to enter

[10]Changing the distances in this fashion might change the "correct" number of clusters that should comprise the clustering. See [27].

the picture. We agree that it would be nice to develop a better understanding of objective functions or better tie them to specific applications to make it easier for data analysts to choose between different algorithms, but this is not our goal. We just wish to extend research on quantitative assessments and develop new clustering algorithms or understand existing ones better.

Let us summarize the relevant points of the above discussion insofar as it pertains to this work: Ignoring some of the technicalities, we could describe the task of cluster analysis as partitioning a given set of objects into groups or clusters such that

(i) similar objects belong to the same cluster,

(ii) dissimilar objects belong to different clusters,

or, and this is just a rephrasing, such that

(i,) similar objects should **not** belong to different clusters,

(ii,) dissimilar objects should **not** belong to the same cluster.

Of course, we have yet to specify any kind of threshold that differentiates similar objects from dissimilar objects. Partially, this is because even when a context is specified, no global threshold has to exist. Locally, objects that are similar sometimes have to be put into different clusters or objects that are dissimilar into the same cluster. More importantly, though, it is also the case that satisfying both objectives simultaneously is a highly non-trivial, sometimes even impossible task. They pull in different, although not necessarily opposite, directions:

1. The first objective favors clusterings comprising just a few clusters. It ensures that objects are not wholly disconnected from each other, such as when all of them float completely detached within singleton clusters.

2. The latter favors clusterings comprising many clusters. It ensures that differences between objects do not fully dissolve, such as when every object is assigned to the same cluster.

There is an inherent tension here that cannot be resolved a priori. Both objectives are qualitatively different, and they should be understood in light of this difference. If we were to ignore one of them, we would miss out on possibly relevant structures. As we have already noted, what is at stake, among other things, is the number $k$ of clusters that should be formed. In most theoretical endeavors of partitional clustering, this number is bracketed or fixed beforehand and passed as a parameter to whatever clustering algorithm is used, even though determining such a number should be a part of the overall task of cluster analysis. Finding good values is, in fact, quite difficult, and so far, relatively little research regarding this problem has been published (see [16, 27]). For the most part, that is, whenever we deal with partitional clustering problems, we will assume that the number of desired clusters is known in advance.

Of course, whatever clustering problem is under consideration could also be solved for several values of $k$ and a seemingly good result selected. However, depending on the algorithm, the results, even when the number of clusters only differs by 1, might vary wildly and be especially difficult to compare. One way of making them more comparable would be to impose some form of hierarchical compatibility onto the computed clusterings, meaning that for any two clusterings $\mathscr{C}_1$ and $\mathscr{C}_2$ with $k_1 > k_2$ clusters, we require that $\mathscr{C}_1$ can be reached from $\mathscr{C}_2$ by clustering together some of the clusters already formed in $\mathscr{C}_2$. That is, in a sequence of hierarchically compatible clusterings, there must exist a way of nesting one clustering within the next, thereby "[permitting] the data to be understood simultaneously at different levels of granularity."[11] Figure 1.2 and Figure 1.3 provide visual representations of this constraint. Computing a sequence of hierarchically compatible and individually good clusterings is a problem that we will pick up in Chapter 2.

Beyond the individual clusterings that comprise such a sequence, the overarching hierarchical structure or taxonomy is also interesting in its own right and should be assessed as a whole. There are numerous applications in biology, knowledge management, and so on.[12] As a concrete example, consider the following hierarchical variant of sequence clustering. We have already mentioned a partitional variant above, but this time, the objective is quite different. Instead of clustering proteins in view of their tertiary structure, the goal now is to explore phylogenetic relationships of organisms and construct a "tree of life" (phylogenetic tree) that maps out and represents some possible evolutionary ancestry. We could visualize such an organization as a dendrogram or a vertical series of nested clusterings, with every point in a separate cluster at the bottom and a single cluster containing every point at the top. [13] Objectives (i) and (ii) change slightly for this example. At some point, every object will be contained in the same cluster, so the question is rather, how "early" this happens:

(i, , ) Similar objects should be clustered together early.

[11][32, p. 2]

[12]"Hierarchical techniques are popular in biological, social, and behavioral sciences because of the need to construct taxonomies. Partitional techniques are used frequently in engineering applications where single partitions are important. Partitional clustering methods are especially appropriate for the efficient representation and compression of large data bases." ([58, pp. 89–99])

[13] We do not necessarily claim the existence of a single universal ancestor. This is just how we will formalize hierarchical clusterings later on.

(ii, , ) Dissimilar objects should be clustered together late.

This concludes our informal introduction. Having established all relevant themes, we can now turn to mathematically precise formalizations.

## 1.2 Partitional Clustering Problems

The applications and especially the data sets we have encountered so far were pretty wide-ranging and diverse, including, for example,

1. nucleic acid sequences, where the degree of their similarity derives from their alignment,
2. pixels embedded into some Euclidean space, where the degree of their dissimilarity derives from their Euclidean distance.

To properly state cluster analysis problems in a mathematical fashion, it is first of all necessary to specify in detail the nature of the data sets we will consider. Since similarity and dissimilarity measures behave quite differently, at least from a theoretical perspective, we will restrict ourselves to one of them and only focus on the latter. A higher value will always signify a greater dissimilarity. Additionally, to impose a minimum of structure on the data set, we will require that such dissimilarity measures satisfy certain axioms. All dissimilarity measures considered in this dissertation turn the underlying set of objects into a metric space.

**Definition 1.** A *metric space* is a tuple $(X, d)$ consisting of a set $X$ and a mapping $d : X \times X \to \mathbb{R}_{\geq 0}$, such that

1. $d(x, y) = 0$ if and only if $x = y$,
2. $d(x, y) = d(y, x)$,
3. $d(x, z) \leq d(x, y) + d(y, z)$

for all $x, y, z \in X$. The elements of $X$ will be referred to as *points*, and $d$ will be referred to as *metric* or *distance function.*

The first axiom is a kind of identity of indiscernibles: if two objects are not dissimilar in any measurable way, then they are already the same. The second axiom tells us that the dissimilarity measure is symmetrical — it does not vary with “direction”. The third and last axiom, also known as triangle inequality, is the most interesting. It is this axiom that imparts $X$ with a minimal geometric structure and is the reason why the metric can be thought of as a distance function (and why we have decided to denote it by $d$) — taking a detour can never result in a shorter path. Most dissimilarity measures satisfy these properties and form metric spaces together with their objects. From now on, we will not talk about dissimilarity measures anymore, but only metrics and distance functions.

In the informal introduction, we have described cluster analysis as the task of partitioning data sets in such a way that the drawn distinctions are useful in a given context. Here, we had in mind the set-theoretical definition of a partitioning. A clustering for us thus amounts to the following.[14]

**Definition 2.** Let $(X, d)$ be a metric space. A (partitional[15]) *clustering* of $(X, d)$ is a set $\mathscr{C}$ of subsets of $X$, called clusters, such that

1. $C \cap C' = \varnothing$ for all $C, C' \in \mathscr{C}$ with $C \neq C'$,
2. $X = \bigcup_{C \in \mathscr{C}} C$.

If $\mathscr{C}$ consists of $k$ cluster and we want to highlight this, we will say that $\mathscr{C}$ is a $k$-clustering.

However, how are such clusterings evaluated? Quantitative assessments are done via objective functions, which introduce a gradient into the set of all clusterings and arrange them according to their purported usefulness. As we have discussed earlier, we are not interested in empirical evaluations, where a clustering is compared to an already known ground truth — this type of evaluation belongs more to supervised learning or extrinsic classification tasks. Instead, we would like to establish quantitative guarantees for algorithms that hold regardless of the data set under consideration. To make this more precise, recall our two goals from the previous section:

(i,) Similar points should **not** belong to different clusters.

(ii,) Dissimilar points should **not** belong to the same cluster.

Both goals can individually be turned into objective functions by assigning a numerical value to each cluster that derives from the metric and represents how well

[14]It should be noted that there are also applications, such as image recognition ([71]), where it is beneficial to assign objects to several different clusters at the same time and measure degrees of "belongingness". These clusterings are often termed *fuzzy* or *soft* in contrast to the *hard* or *exclusive* clusterings that we have defined just now. While there is research on fuzzy clustering, we are only concerned with strict partitionings in this dissertation: every object is assigned to exactly one cluster.

[15]If not otherwise specified, clusterings always refer to partitional clusterings. However, it should also be clear from the context whether we are dealing with partitional or hierarchical clusterings.

the goal has been attained. To formalize (i,), we could, for example, assign to a clustering the smallest distance induced by any pair of points contained in different clusters. The higher this value, the more separated the clusters, so our objective would be to maximize this function. Similarly, to formalize (ii,), we could assign to a clustering the largest distance induced by any pair of points that lie in the same cluster. The lower this value, the less spread out the clusters, so our objective would be to minimize this function. In each case, despite not being tied to a particular or concrete application, both assessments, in principle, capture certain desirable properties of any clustering whatsoever. Depending on the context, these properties might not always be of equal importance, but at least one of the two, if not both, are relevant to some extent. Of course, these particular assessments were just examples; other aspects of the same goal can be emphasized by formalizing it differently, as we will see. Nonetheless, this, in essence, is what each objective is: a cost function that derives from exactly one of the two goals and penalizes deviation from it.

Although we would like intra-cluster distances to be small and inter-cluster distances to be large simultaneously, as we have noted above, this is not necessarily possible. Both goals point in different directions, especially when considering the number of clusters that should comprise the clustering. The same holds for any derived objective function. Usually, only goal (ii,) is turned into a penalty function, whereas (i,) is completely ignored. To then prevent every point from being assigned to a separate cluster, one imposes an upper bound, given by some fixed parameter $k$, on the number of allowed clusters.[16] With quantitative assessments, clustering problems are stated as optimization problems for some objective function: find a $k$-clustering with optimal cost or try to approximate one as well as possible. Since it is, in almost all cases, NP-hard to compute optimal solutions, most research focuses on approximation algorithms. That is, the ratio of the computed solution's cost to the optimal solution's cost should be as close to 1 as possible.

### 1.2.1 The $k$-Diameter and $k$-Center Problems

One of the most straightforward formalizations of (ii,) is the one we have described above. If dissimilar points should not belong to the same cluster, then we should try to minimize the maximum distance between any two points contained in the same cluster. In other words, we should minimize the maximum diameter attained by any cluster in our clustering.

[16] There are also relaxations of this type of problem, known as *facility location* problems, where instead of fixing the number of clusters, one imposes additional costs when forming clusters. These costs are again part of the input and thus excluded from the actual clustering problem. We will not consider facility location problems much here because their range of application is usually quite different.

**Problem 3** (The $k$-Diameter Problem)**.** Let $(X, d)$ be a metric space and $k \in \mathbb{N}$. For any subset $C \subseteq X$, define its diameter to be the largest distance

$$\operatorname{diam}(C) = \max_{x,y \in Z} d(x, y)$$

between any of its points. The goal in the $k$-diameter problem is then to find a $k$-clustering $\mathscr{C}$ of $(X, d)$ that minimizes the maximal diameter

$$\operatorname{diam}(\mathscr{C}) = \max_{C \in \mathscr{C}} \operatorname{diam}(C)$$

of any cluster it contains.

This is one of the best-understood clustering problems from a purely theoretical perspective. It has long been known that this problem is NP-hard to approximate within a factor of $\alpha < 2$ for general metrics ([48]). This bound is also known to be tight: Gonzalez ([48]), as well as Hochbaum and Shmoys ([54]), have each provided elegant polynomial-time 2-approximation algorithms.

The $k$-center problem is closely related and similarly well understood, but we have to shift our focus a bit to state it properly. Recall that we are not always interested in partitionings directly but sometimes rather in summarizing a data set with the help of well-chosen representatives. For example, in the sequence clustering example, the goal was not only to cluster proteins or other molecules according to possible structural homology but to find for all clusters a particular protein that nicely represents it. The $k$-center problem is about finding such representatives.

**Problem 4** (The $k$-Center Problem)**.** Let $X \subseteq M$ be a subset of some ambient metric space $(M, d)$ and $k \in \mathbb{N}$. In the $k$-center problem, the goal is to find a subset $Z \subseteq M$, whose members are termed centers, of size $k$ that minimizes the largest distance

$$\max_{x \in X} \min_{z \in Z} d(x, z)$$

of any point in $X$ to its closest center in $Z$.

We allow centers to be drawn from a space larger than the actual data set because sometimes it is not necessary for centers to immediately correspond to objects directly contained in it. For example, when $X$ is a subset of some Euclidean space $\mathbb{R}^d$, each point $x \in \mathbb{R}^d$ could, in principle, represent an object despite not yet having been measured. These are continuous clustering problems, in contrast to the discrete variants in which $X$ coincides with $M$. Also note that, although the focus lies on representatives, their assigned value nonetheless relates them to a specific clustering that is implicitly attached to them. Given a set of $k$

centers $Z = \{z_1, \dots, z_k\} \subset M$ for $X$ we can construct from them the clustering $\mathscr{C} = \{C_1, \dots C_k\}$ that consists of the clusters

$$C_i = \{x \in X \mid \forall j \neq i \colon d(x, z_j) \geq d(x, z_i)\}.$$

That is, we group together all points that are closest to a given center. The cost that $Z$ induces for $X$ is then equal to

$$\max_i \max_{x \in C_i} d(x, z_i).^{17}$$

Note that an optimal $k$-center solution never costs more than an optimal $k$-diameter solution, and the latter is, in turn, at most twice as costly as the optimal $k$-solution.

Similar to the $k$-diameter problem, the discrete $k$-center problem is NP-hard to approximate within a factor of $\alpha < 2$ for general metrics ([52, 56]), and this bound was again shown to be tight by both Gonzalez [48], as well as Hochbaum and Shmoys [53]. Even when the points lie on the Euclidean plane, this lower bound is still as large as $\sqrt{2+\sqrt{3}} \approx 1.96$ ([65, 26]). However, if we consider the continuous case, where centers can be drawn from all of $\mathbb{R}^d$, then there exist $(1+\varepsilon)$-approximation schemes with a running time of $2^{(k/\varepsilon \log k)} dn$ and $O(n \log k) + (1/\varepsilon)^{O\left(2^d k^{1-1/d} \log k\right)}$.[18] The second algorithm is faster for constant dimension $d$, but for constant $k$ the former is guaranteed to run in polynomial time.

### 1.2.2 The $k$-Median and $k$-Means Problems

The most important and studied clustering problem likely is the $k$-means problem due to its relevance to several machine learning and engineering tasks. It is notable insofar as the corresponding objective function directly derives or is explicitly stated in them.

**Problem 5** (The $k$-Means Problem)**.** Let $X \subseteq M$ be a subset of some ambient metric space $(M, d)$. In the $k$-means problem the goal is to find a set of $k$ centers $Z \subseteq M$ that minimizes

$$\sum_{x \in X} \min_{z \in Z} d^2(x, z).$$

[17] A slight technical problem with this description is that some points might be equidistant from two different centers, meaning that their corresponding clusters would intersect. Of course, this can easily be solved by removing any such point from all but one of the clusters that contain it until all clusters are pairwise disjoint.

[18] The first algorithm, presented in [11], is due to Bădoiu and Clarkson. The second was established by Agarwal and Procopiuc in [4]. Although the latter state a better running time in their paper, Bandyapadhyay et al. ([14]) have noted that it is slightly incorrect and fixed it.

The set $M$ almost always coincides with some Euclidean space $\mathbb{R}^d$, although sometimes the discrete variant is also considered. A closely related problem, whose field of application, however, is quite different, is the $k$-median problem.

**Problem 6** (The $k$-Median Problem)**.** Let $X \subseteq M$ be a subset of some ambient metric space $(M, d)$. In the $k$-median problem the goal is to find a set of $k$ centers $Z \subseteq M$ that minimizes

$$\sum_{x \in X} \min_{z \in Z} d(x, z).$$

The only notable difference on the level of the problem statement is that the $k$-median cost function sums up the non-squared distances from each point to its closest center instead of the squared distances. Quite often, theoretical results are first established for the $k$-median problem and transported to the $k$-means problem only afterward at the cost of worse approximation ratios. As we have noted earlier, the number of theoretical guarantees and algorithms for both problems is quite high. However, since we only care about the discrete variants and most research, at least for the $k$-means problem, focuses instead on the continuous case, we can skip most of them. Cohen-Addad et al. ([28]) have shown that the $k$-median and $k$-means problems cannot be approximated with a factor better than $(1+2/e)$ and $(1+8/e)$, respectively, in FPT time, assuming Gap-ETH. On the positive side, Cohen-Addad et al. ([76]) have established a 2.67059-approximation algorithm for the discrete $k$-median problem, and Ahmadian et al. ([6]) a $9+\varepsilon$-approximation algorithm for the discrete $k$-means problem in general metric spaces.

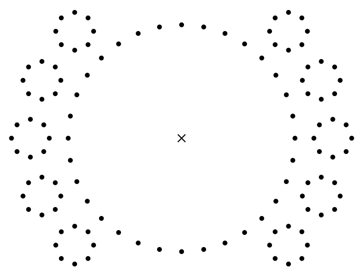
### 1.2.3 The $k$-MSR and $k$-MSD Problems

The $k$-min-sum-radii problem, $k$-MSR for short, resembles both the $k$-center and the $k$-median problem and takes an intermediate path: the objective is given by the sum of all radii. Contrary to the $k$-center objective, we do not completely ignore the fine-tuning of all $k-1$ smaller clusters since every cluster thus contributes to the objective cost. Contrary to the $k$-median objective, not every point contributes to it, so large individual costs cannot average out in the same manner.

**Problem 7** (The $k$-MSR Problem)**.** Let $X \subseteq M$ be a subset of some ambient metric space $(M, d)$. The goal in the $k$-MSR problem is to find a set of at most $k$ pairs $\{(c_1, r_1), \ldots, (c_\ell, r_\ell)\} \subset M \times \mathbb{R}_{\geq 0}$, each consisting of a center and a radius, that minimize $\sum_i r_i$ and guarantee that $X \subseteq \bigcup_i \mathrm{B}(c_i, r_i)$, where

$$\mathrm{B}(c_i, r_i) = \{x \in X \mid d(c_i, x) \leq r_i\}$$

denotes the set of all points of $X$ that are at a distance of at most $r_i$ from $c_i$.



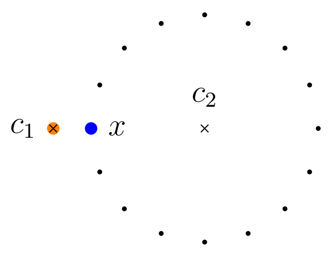


Figure 1.1: Left side: An example where $k$-min-sum-radii rather opens one cluster than eleven. Right side: An example where the cheapest $k = 2$-clustering keeps $c_1$ as a singleton rather than combining it with $x$, despite the fact that $c_1$ is closer to $x$ than the center $c_2$ of the big cluster. It is cheaper to *not* assign the blue point to the orange point even though that would be a closer center.[19]

Note that it is not sufficient just to provide a set of centers. This is because, contrary to the other center-based objectives, the $k$-MSR objective is not local in the sense that it is not always optimal to assign points to their closest center. It may be beneficial to assign them to centers further away (see Figure 1.1, right side). Even when $X$ is a subset of $\mathbb{R}$ and even when the centers are chosen optimally, assigning points to their closest centers can result in a clustering whose k-msr cost is three times the cost of an optimal solution (see [62]), and for more complicated metric spaces this factor grows quickly. As it turns out, the problem of finding the best assignment given a set of centers is NP-hard for the $k$-MSR objective, whereas computing such an assignment for the k-center objective is basically trivial. This is why we require a solution that does not solely consist of centers but also of radii (the costs that the centers induce). Note also that we also allow a solution to consist of less than $k$ pairs. Contrary to the other center-based objective, the cost of a $k$-MSR solution might actually *increase* if we open additional centers, i.e., increase the number of pairs with non-zero radii (see Figure 1.1, left side).

The $k$-MSR problem is closely connected to the base station placement problem arising in wireless network design [63], where the objective is to minimize the energy required for wireless transmission. Its mathematical model amounts to the *minimum-sum-radii-cover problem* where we have to cover a set of clients with a set of balls whose centers are located at a subset of server locations in such a way that the sum of the radii of the balls is minimized. Similar to before, we can also formulate a variation that focuses more on partitionings than on representatives. Instead of considering only the largest diameter, as in the $k$-diameter, we sum all of them up.[20]

[19]This figure was published in [35].

[20]In part, this summary of known results for the $k$-MSR problem was published in [35].

**Problem 8** (The $k$-MSD Problem)**.** Let $X \subseteq M$ be a subset of some ambient metric space $(M, d)$. The goal in the $k$-MSD for short, is to find a $k$-clustering $\mathscr{C} = \{C_1, \ldots, C_k\}$ of $(X, d)$ that minimizes the sum of diameters

$$\sum_i \operatorname{diam}(C_i)$$

over all clusters in $\mathscr{C}$.

One nice property of this objective we want to highlight is that the $k$-MSD objective is much more amenable than the $k$-diameter objective to allow the input to be adequately separated. The latter ensures that every cluster is small by minimizing the size of the largest cluster. However, this is not always desired. The data set might contain a relatively homogeneous and large cluster that should not be split apart since that would yield two clusters whose inter-cluster distance could be exceedingly small. In other words, no proper delineation within the data set would correspond to this separation. At the same time, since $k$-diameter only considers the largest cluster, it might also merge two small clusters that are actually quite well separated.

Although being quite natural clustering objectives, the $k$-MSR and $k$-MSD problems have received less attention than the other objectives in our list ($k$-center, $k$-diameter $k$-median, $k$-means). Or, rather, interest has picked up only relatively recently. In part, this might be because the objectives behave quite counter-intuitively at times and very unlike the other objectives (again, see Figure 1.1). Back in 2004, Charikar and Panigrahy [22] designed a 3.504-approximation algorithm for the metric $k$-MSR problem (and thereby also a 7.008 approximation algorithm for the $k$-MSD problem) based on the primal-dual framework by Jain and Vazirani for $k$-median problem. Recently, that is, in 2022, the approach has been refined by Friggstad and Jamshidian [42], who obtained a 3.389-approximation algorithm for $k$-MSR and a 6.546-approximation algorithm for the $k$-MSD problem (which is not just double the ratio of the $k$-MSR algorithm). The currently best-known algorithm, which is also a primal-dual approach, though in many respects simpler, is from Buchem et al. [17] and achieves a $(3 + \varepsilon)$-approximation ratio with a running time of $n^{O(1/\varepsilon)}$. This approximation ratio is guaranteed even when allowing outliers,[21] so their result also improves upon the previously best-known result of $(12.365 + O(\varepsilon))$. Although they will not concern us as much, constrained variants of the $k$-MSR problem have been studied as well. Ahmadian and Swamy [5] built upon [22] to obtain a 3.83-approximation for the non-

[21] An outlier is a point that is ignored and does not contribute to the objective cost. For variants of outlier problems, we usually specify a number $z$ as part of our input that denotes the number of outliers in our data set.

uniformly lower bounded[22] $k$-MSR problem. Inamdar and Varadarajan [57] derive a 28-approximation for the uniformly capacitated[23] $k$-MSR problem, but this algorithm is an FPT-approximation algorithm with running time $O(2^{O(k^2)} \cdot n^{O(1)})$. Bandyapadhyay, Lochet, and Saurabh [15] also give an FPT-approximation: They develop a $(4+\varepsilon)$-approximation algorithm with $2^{O(k \log(k/\varepsilon))} \cdot n^3$ running time for $k$-MSR with uniform capacities and a $(15+\varepsilon)$-approximation algorithm for $k$-MSR with non-uniform capacities that runs in time $2^{O(k^2 \log k)} \cdot n^3$.

If we are in the continuous case, where centers can be drawn from all of $\mathbb{R}^d$, it is possible to obtain better results. In the plane, an optimal set of centers partitions the plane into $k$ disjoint convex regions in such a way that each cluster from the corresponding clustering fully lies in one of them. The dual of that partition is an internally triangulated planar graph, a fact that Capolyleas et al. [19] use to solve the problem exactly by enumerating over all $O(n^{6k})$ possible constellations. Gibson et al. [47] also designed an exact algorithm for the Euclidean $k$-MSR problem in $\mathbb{R}^2$. Their algorithm is based on an involved dynamic programming approach and has a running time of $O(n^{881})$ for $d = 2$. Bandyapadhyay, Lochet and Saurabh [15] provide a $(1+\varepsilon)$-approximation algorithm for the capacitated Euclidean $k$-MSR problem that runs in $2^{O(kd \log(k/\varepsilon))} \cdot n^3$ time. If capacities are allowed to be violated by a $(1+\varepsilon)$ fraction, then the same approximation ratio can be achieved by a randomized algorithm whose running time only linearly depends on $d$.

### 1.2.4 The $k$-Separation Problem

We conclude this list with a clustering objective that draws much less attention: the $k$-separation objective. Contrary to the other objectives we have discussed, which are all formalizations of goal (ii,), this objective is a formalization of goal (i,). We have already outlined it at the beginning of this section: if similar points should not belong to different clusters, then we should try to maximize the smallest inter-cluster distance.

**Problem 9** (The $k$-Separation Problem)**.** Let $(X, d)$ be a metric sapce, and $k \in \mathbb{N}$. The goal in the *$k$-separation problem* is to find a $k$-clustering $\mathscr{C} = \{C_1, \ldots, C_k\}$ of $(X, d)$ that maximizes the smallest distance

$$\max_{i,j} d(C_i, C_j)$$

[22]In this variant we prescribe for each center a lower bound for the number of points that have to be assigned to this center. It is non-uniform if this number can vary from center to center.

[23]In this variant, which is in some sense the inverse of the lower bounded variant, we prescribe for each center an upper bound for the number of points that have to be assigned to this center. It is uniform if this number is the same for all centers.

between any two of its clusters, where

$$d(C_i, C_j) = \min_{(x_i, x_j) \in C_i \times C_j} d(x_i, x_j).$$

There are several things worth noting about this objective. First, this is not a clustering objective that's often (or at all) considered. Mostly, it crops up in the form of a clustering constraint where it requires that the smallest inter-cluster distance of any feasible clustering has to be at least as large as some threshold that was specified beforehand.[24] Second, this clustering objective is the only one on this list that does not necessarily favor isotropic clusterings. In Euclidean settings, optimal clusterings for all other objectives always consist of sets whose convex hulls are pairwise disjoint, meaning they have a very ball-like shape. Other constellations, such as the standard example of two encroaching crescents used to motivate spectral clustering, cannot be found or recognized by optimizing those objectives. Third, of the clustering objectives we have listed so far, this is the only one that should be maximized. Although we have said this objective does not draw much attention, this is only partially true. The corresponding problem is closely related to the classical minimum spanning tree problem. In the minimum spanning tree problem, one is given an undirected graph $G = (V, E)$ with weights $w : E \to \mathbb{R}_+$, and one tries to find a subtree containing all vertices whose overall weight is minimal. A standard algorithm to solve this problem is Kruskal's algorithm, where, starting with $(V, \varnothing)$, one successively adds a cheapest edge between different connected components. Consider now the complete graph on $X$, where the weight of an edge coincides with the distance of its points. Stopping Kruskal's algorithm right when it has formed $k$ connected components, the latter yield an optimal solution to the $k$-separation problem. This approach coincides with the Single-Linkage clustering algorithm we will discuss in the next section. Consequently, this makes the $k$-separation problem the only problem on this list that is easy to solve optimally. In that sense, the $k$-separation problem is less interesting than the others, but that changes once we combine it with some other objectives (see Chapter 3).

### 1.2.5 Constraints

Sometimes not all clusterings are valid in a given context. For example, it might be necessary that clusters have a certain size (lower-bounded and capacitated clustering variants), that certain attributes are well-represented within in cluster (fair variants), or that specific relations between certain objects are respected (must-link and cannot-link variants). In the corresponding variants of problems

[24] See, for example, the $\delta$-constraints outlined in [33].

for center-based objectives we can then not just compute representatives, since the optimal distribution of all other points relative to those centers do not necessarily assign every point to its closest center — the resulting clustering might not be allowed. Instead, we have to combine representatives and clusterings into the single notion of clustering assignments.

**Definition 10.** Let $X$ be a subset of some metric space $(M, d)$. A *$k$-clustering assignment* of $(X, d)$ is just a mapping $\sigma : X \to C$ of $X$ onto a subset of $C \subset M$. If $C$ consists of $k$ centers and we want to highlight this, we will say that $\sigma$ is a $k$-clustering assignment.

If $\mathsf{cost}$ denotes a general objective that evaluates such assignments, then the corresponding optimization problem for the constraint asks for a valid assignment that minimizes $\mathsf{cost}$. This come up in Chapter 3, for example.

## 1.3 Hierarchical Clustering Problems

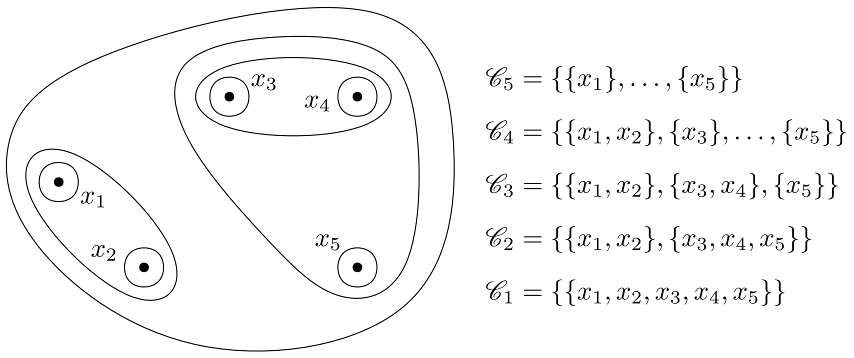
In the informal introduction, we also considered hierarchical clusterings. As we have discussed, they are relevant whenever one wants to understand the data set at different levels of granularity.

**Definition 11.** Let $(X, d)$ be a metric space. A clustering $\mathscr{C}$ of $(X, d)$ is *hierarchically compatible* with another clustering $\mathscr{C}'$ of $(X, d)$ if for every cluster $C' \in \mathscr{C}'$ there exists a cluster $C \in \mathscr{C}$, such that $C' \subseteq C$. In other words, $\mathscr{C}$ is hierarchically compatible with $\mathscr{C}'$, if $\mathscr{C}'$ is a set-theoretical refinement of $\mathscr{C}$.

**Definition 12.** Let $(X, d)$ be a metric space consisting of $n$ points. A collection $\mathscr{H} = \{\mathscr{C}_1, \ldots, \mathscr{C}_n\}$ is called a *hierarchical clustering* of $(X, d)$, if

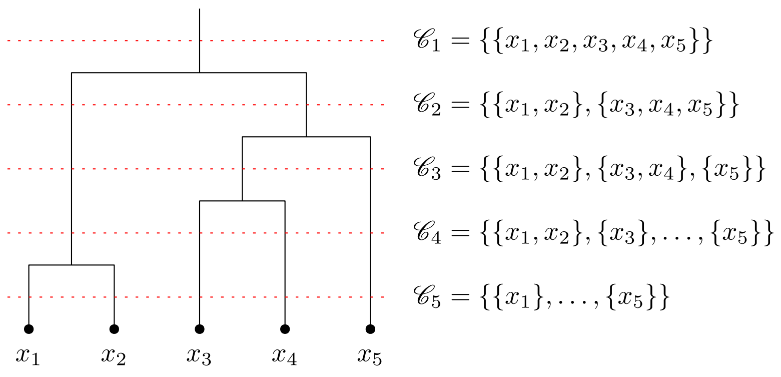
1. $\mathscr{C}_k$ is a $k$-clustering of $(X, d)$ for all $k \in \{1, \ldots, n\}$,
2. $\mathscr{C}_k$ is hierarchically compatible with $\mathscr{C}_{k+1}$ for all $k \in \{1, \ldots, n-1\}$.

There are two principal approaches to quantitatively assess such hierarchical clusterings: either on a level-by-level basis or as a whole. Let us consider the latter approach first, even if, in the end, we will only deal with the former. The first paper that introduced an objective function for hierarchical structures in their own right was published in 2015 by Dasgupta ([31]), and it is seminal in this regard. To focus directly on the hierarchical structure and not on derived partitional clusterings, he conceives of it not as a sequence of partitional clusterings but as a rooted tree whose leaves are the objects to be clustered. Each internal vertex then represents the cluster consisting of all leaves contained in the subtree starting at that vertex. Recall our general goal from the introduction: if two objects are similar, then they



$x_1$
$x_2$
$x_3$
$x_4$
$x_5$
$\mathscr{C}_5 = \{\{x_1\}, \ldots, \{x_5\}\}$
$\mathscr{C}_4 = \{\{x_1, x_2\}, \{x_3\}, \ldots, \{x_5\}\}$
$\mathscr{C}_3 = \{\{x_1, x_2\}, \{x_3, x_4\}, \{x_5\}\}$
$\mathscr{C}_2 = \{\{x_1, x_2\}, \{x_3, x_4, x_5\}\}$
$\mathscr{C}_1 = \{\{x_1, x_2, x_3, x_4, x_5\}\}$

Figure 1.2: A rendition of a hierachical clustering on 5 points. We start with a clustering $\mathscr{C}_5$ that consists only of singletons. In the next step we merge clusters $\{x_1\}$ and $\{x_2\}$ to produce $\mathscr{C}_4$. This is followed by merging $\{x_3\}$ with $\{x_4\}$ and then with $\{x_5\}$. In the end all points belong to the same cluster. Note that it is not clear from the picture at which time step each of the clusters exist. For example, without the sequence of clusterings provided on the right, one could also think that the first merge would be between $\{x_3\}$ and $\{x_4\}$. There are other representations, such as depicted in Figure 1.3, which this ambiguity is eliminated, should it matter.



$\mathscr{C}_1 = \{\{x_1, x_2, x_3, x_4, x_5\}\}$
$\mathscr{C}_2 = \{\{x_1, x_2\}, \{x_3, x_4, x_5\}\}$
$\mathscr{C}_3 = \{\{x_1, x_2\}, \{x_3, x_4\}, \{x_5\}\}$
$\mathscr{C}_4 = \{\{x_1, x_2\}, \{x_3\}, \ldots, \{x_5\}\}$
$\mathscr{C}_5 = \{\{x_1\}, \ldots, \{x_5\}\}$
$x_1$
$x_2$
$x_3$
$x_4$
$x_5$

Figure 1.3: A different rendition of the hierachical clustering given in Figure 1.2. This type of diagram is called a *dendrogram*. Cutting of the upper part of the dendrogram at each of the dotted red lines gives a forest, where the leaves of every tree within that forest represent a clustering: the lowest line yields $\mathscr{C}_5$, etc.

should be merged early, meaning that their smallest common ancestor in the tree should be closer to the leaves, whereas if they are dissimilar, then they should be merged late, meaning that their smallest common ancestor should be closer to the root. Dasgupta formalizes this intuition as follows. To each pair of objects, we assign the product of their similarity multiplied by the number of objects contained in the subtree rooted at the pair's smallest common ancestor. The goal is to minimize the sum of all these. If the objects are similar, then the smallest common ancestor should be close to the leaves, i.e., the subtree rooted at it should contain few leaves, and, inversely, if this subtree contains many leaves, then the similarity should be small. Note that Dasgupta has devised this objective with similarity measures in mind. While there are several interesting continuations (such as the paper [77] by Wang and Wang that also measure how much of a hierarchical structure the initial data set exhibits), for dissimilarity measures, the function is less interesting. Although it can readily be adjusted for them, and although Cohen-Addad et al. ([29]) were able to derive several relevant results for common algorithms (average-linkage is a $\frac{2}{3}$-approximation algorithm, whereas Single-Linkage and bisection 2-center can be arbitrarily bad), Chatziafratis et al. ([25]) showed that, in expectation, one already gets a $\frac{2}{3}$-approximation by recursively splitting clusters uniformly at random. Wang and Moseley ([78]) subsequently proposed an objective function for Euclidean data that does not have this problem. Despite still being somewhat intuitive, this objective feels decisively more artificial in some of its details. A problem that is already noticeable for partitional objectives, and which returns with more force here, is that it is pretty difficult to establish their relation to concrete problems. To intuit or gauge when such an objective is appropriate possibly becomes even more challenging. Still, they have introduced meaningful distinctions between already established clusters and gave rise to several new ones.

In any case, we will focus only on level-by-level assessments: each $k$-clustering in the hierarchical sequence is independently compared to an optimal $k$-clustering for this level through one of the partitional objective functions we have discussed in the previous section. All partitional clusterings comprising the sequence should have a good approximation ratio in their own right, i.e., the worst overall approximation ratio should be as close to 1 as possible. In other words, we would like to minimize

$$\max_k \frac{\mathsf{cost}(\mathscr{C}_k)}{\mathsf{cost}(\mathscr{O}_k)},$$

for any given clustering objective $\mathsf{cost}$, where $\mathscr{O}_k$ denotes an optimal $k$-clustering for $\mathsf{cost}$. This type of assessment is meaningful if, for example, one wants to extract from the hierarchical sequence one or several partitional clusterings, i.e., if one did not know which $k$ would be best beforehand and just wanted to compute several comparable clusterings. Noteworthy, because of the hierarchical constraint, there does not even have to exist a hierarchical sequence of independently optimal clusters (see [8]). The hierarchical clustering problems we are interested in most are the hierarchical $k$-center and the hierarchical $k$-diameter problem.

**Definition 13.** Let $(X, d)$ be a metric space consisting of $n$ points. Then the goal of the *hierarchical $k$-diameter problem* is to compute a hierarchical clustering $\mathscr{H} = \{\mathscr{C}_1, \dots, \mathscr{C}_n\}$, such that the maximal approximation factor

$$\max_k \frac{\operatorname{diam}(\mathscr{C}_k)}{\operatorname{diam}(\mathscr{O}_k)}$$

is minimal, where $(\mathscr{O}_k)_{k=1}^n$ are optimal $k$-diameter clusterings of $(X, d)$.

Although the partitional $k$-center problem concentrates on representatives, in the hierarchical $k$-center variant, we have to shift our focus to clusterings again. While it is possible to reconstruct an optimal clustering from a set of centers, getting a hierarchically consistent sequence of clusterings from a sequence of centers is not as straightforward.

**Definition 14.** Let $(X, d)$ be a metric space consisting of $n$ points. Then the goal of the *hierarchical $k$-center problem* is to compute a hierarchical clustering $\mathscr{H} = \{\mathscr{C}_1, \dots, \mathscr{C}_n\}$, such that the maximal approximation factor

$$\max_k \frac{\operatorname{rad}(\mathscr{C}_k)}{\operatorname{rad}(\mathscr{O}_k)}$$

is minimal, where $(\mathscr{O}_k)_{k=1}^n$ are optimal $k$-center clusterings of $(X, d)$. The radius of a cluster $C \in \mathscr{C}$ is given by

$$\operatorname{rad}(C) = \min_{z \in X} \max_{c \in C} d(z, c),$$

so that

$$\operatorname{rad}(\mathscr{C}) = \max_{C \in \mathscr{C}} \operatorname{rad}(C)$$

as usual.

There are two principal types of algorithms to compute hierarchical clusterings: agglomerative (or bottom-up) algorithms and divisive (or top-down) algorithms. If we consider the second axiom that hierarchical clusterings have to satisfy, then it becomes clear that all of them can be formed by applying an appropriate sequence of merges to $\mathscr{C}_n$. Indeed, if each clustering $\mathscr{C}_k$ is hierarchically compatible with the clustering $\mathscr{C}_{k+1}$ that immediately follows it and contains just one cluster less, then $\mathscr{C}_{k+1}$ has to contain two clusters $C$ and $C'$, such that $\mathscr{C}_k = (\mathscr{C}_{k+1} \setminus \{C, C'\}) \cup (C \cup C')$. That is, we can derive $\mathscr{C}_k$ from $\mathscr{C}_{k+1}$ by merging two of its clusters. The resulting approach is said to be agglomerative or bottom-up. If one instead descends from $\mathscr{C}_1$ to $\mathscr{C}_n$ by successively splitting up clusters, then this results in a divisive or top-down approach. The main algorithm established by Dasgupta and Long in [32] is an example of such a top-down algorithm, although a significant amount of pre-processing is involved. Another more direct example of a divisive algorithm is the bisecting $k$-means or $k$-center algorithm. However, linkage-based agglomerative algorithms are the most popular class of hierarchical clustering algorithms, be it due to their ease of use or widespread implementation. This additional description of being linkage-based indicates that each merge performed by the algorithm is determined by optimizing a particular linkage function, which assigns a cost to every pair of possible merges.

**Definition 15.** Let $(X, d)$ be a metric space. A *linkage function* for $(X, d)$ is a mapping $f : 2^X \times 2^X \to \mathbb{R}_{\geq 0}$ that assigns a non-negative real number to every pair of subsets.

At each step, an agglomerative algorithm merges a pair of clusters with minimal assigned value. Algorithm 1 provides the general outline of any agglomerative algorithm.

**Algorithm 1:** Linkage-Based Agglomerative Clustering Algorithm

```
Data: A metric space $(X, d)$ on $n$ points, a linkage function $f$ on $(X, d)$.
Result: A hierarchical clustering $\mathscr{H} = \{\mathscr{C}_1, \dots, \mathscr{C}_n\}$ of $(X, d)$.
1 $\mathscr{C}_n \leftarrow \{\{x\} \mid x \in X\}$;
2 for $i = n \dots 2$ do
3     $(C, C') \leftarrow \operatorname{argmin}\{f(D, D') \mid (D, D') \in \mathscr{C}_i \times \mathscr{C}_i$ with $D \neq D'\}$;
4     $\mathscr{C}_{i-1} = (\mathscr{C}_i \setminus \{C, C'\}) \cup (C \cup C')$;
5 end
6 return $\{\mathscr{C}_1, \dots, \mathscr{C}_n\}$;
```

All agglomerative algorithms are of this form and differ only by their linkage functions. A non-exhaustive list of linkage functions and their usages include:

1. $\mathsf{sep}(C, C') = d(C, C') = \min_{(c,c') \in C \times C'} d(c, c')$ can be used to maximize the minimal distance between two clusters. The corresponding algorithm is called *Single-Linkage* and is related to the *k-separation problem.*

2. $\mathsf{rad}(C, C') = \mathrm{rad}(C \cup C') = \min_{z \in X} \max_{c \in C \cup C'} d(c, z)$ can be used to minimize the largest radius in every step. The corresponding algorithm is called *Complete-Linkage* and is related to the *k-center problem.*

3. $\mathsf{diam}(C, C') = \operatorname{diam}(C \cup C') = \max_{x,x' \in C \cup C'} d(x, x')$ can similarly be used to minimize the largest diameter. Just like the previous algorithm, this one is also called *Complete-Linkage*, but this time it is related to the *$k$-diameter problem.* We denote the former by $\mathbf{CL}^{\text{rad}}$ and the latter by $\mathbf{CL}^{\text{diam}}$ to differentiate between both variants.

4. $\mathsf{avg}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{(c,c') \in C \times C'} d(c, c')$ can be used to minimize the average diameter of a cluster. It is one of several variants known as *Average-Linkage*.

5. $\mathsf{ward}(C, C') = \frac{|C| \cdot |C'|}{|C| + |C'|} \|\mu_C - \mu_{C'}\|^2$ can be used to minimize the increase in the sum of squares, where $\mu_A = \frac{1}{|A|} \sum_{a \in A} a$ denotes the *centroid* of a set $A \subset \mathbb{R}^d$. The corresponding agglomerative algorithm is called *Ward's Method* and is related to the *$k$-means problem.* In contrast to the other methods, it can only be applied to Euclidean clustering instances.

Although non-exhaustive, this list covers the most important linkage functions, such as Ward's Method. Each linkage function is related to a specific problem, which it optimizes in each step. However, greedily optimizing an objective function in such a way does not necessarily yield a series of independently optimal clusterings. In fact, this only holds for Single-Linkage. In all other cases, it might be possible that $\mathscr{C}_{n-1}$ is the only non-trivial optimal clustering. An initially worse merge might yield better clusterings in the long run. Since errors might accumulate over time, later clusterings might constitute quite bad solutions in their own right. As such, establishing level-by-level approximation guarantees for each of these algorithms is quite important. In the next chapter, we will contribute to these results by analyzing Single-Linkage and Complete-Linkage. They are nicely complementary in that the first wants to maximize inter-cluster distances in each step, whereas the second wants to minimize intra-cluster distances.

**Theorem 16.** *Let $(X, d)$ be a metric space consisting of $n$ points and $\mathscr{S}_n, \ldots, \mathscr{S}_1$ the hierarchical clustering computed by Single-Linkage on $(X, d)$. Then $\mathscr{S}_k$ is an optimal solution to the $k$-separation problem for all $k \in \{1, \ldots, n\}$.*

This result is anecdotally quite well known and follows, for example, relatively quickly from Lemma 69. In the next chapter, we will analyze how useful Single-Linkage is in optimizing the objective functions corresponding to Complete-Linkage.

For hierarchical $k$-center and $k$-diameter, the situation is quite different. Of course, if it is NP-hard to solve the corresponding partitional problems within a factor of $\alpha < 2$, as we have noted earlier, then the hierarchical variant is at least as hard. In fact, as it turns out, optimal $k$-clusterings are not necessarily hierarchically compatible, so even assuming unlimited computation power, 1-approximations generally do not exist. Das and Kenyon-Mathieu ([30]) provided an instance for the diameter and Großwendt ([50]) for the radius where the best hierarchical clustering is a 2-approximation. By building on the notion of nestings introduced by Lin et al. ([64]) Großwendt ([50]) proved an existential upper bound of 4 for hierarchical $k$-center. This line of research was then further improved by Arutyunova and Röglin ([8]). However, despite this, several constant-factor approximations are known. Dasgupta and Long ([32]), as well as Charikar et al. ([24]) both provide polynomial-time 8-approximations. The aforementioned notion of nestings ([64]) allows every approximation algorithm for a $k$-clustering objective that satisfies its nesting property to be converted into an algorithm for its hierarchical version. Especially $k$-median and $k$-means satisfy this property and thus (in combination with the currently best constant factor approximations for $k$-median ([18]) and $k$-means ([6])), polynomial time constant factor approximations do indeed exist for the hierarchical $k$-median/$k$-means problem. However, the resulting guarantees are relatively high ($\approx 56$ for $k$-median and $\approx 3662$ for $k$-means). Nesting can also be applied to $k$-center/$k$-diameter but does not improve upon the 8-approximation.

How well does Complete-Linkage fare? We will consider this question in the next chapter.

## Outlook

**In Chapter 2** we analyze Complete-Linkage on a level-by-level basis for general metric spaces. We improve the previously best-known lower bounds for the approximation ratio from $\Omega(\log k)$ to $\Omega(k)$ and provide the first non-trivial upper bounds. For $\mathbf{CL}^{\text{rad}}$ we are able to match the lower bound by showing that the algorithm always computes $O(k)$-approximations, while for $\mathbf{CL}^{\text{diam}}$ we are only able to prove an upper bound of $O(k^2)$ (which however can relatively easily be improved to $O(k^{\frac{\ln 3}{\ln 2}})$).

**In Chapter 3** we analyze how well various objectives ($k$-center, $k$-diameter, $k$-median, $k$-means, $k$-diameter) can be combined with the $k$-separation constraint. In each case we show that they cannot be approximated with constant factor simultaneously and thus turn to their Pareto sets.

**In Chapter 4** we study the Euclidean $k$-center PTAS established by Bădoiu et al. ([12]) for constant $k$ and modify it in such a way that it yields a PTAS for the Euclidean $k$-MSR problem, also for constant $k$. We also show that this algorithm is able to deal with outliers.

**In Chapter 5**, we showcase some preliminary (and somewhat outdated) results for the connected $k$-center problem.

# Chapter 2

# Lower and Upper Bounds for Complete-Linkage

## 2.1 Introduction

In this chapter we will analyze one of the main agglomerative clustering algorithms (besides Single-Linkage, Average-Linkage and Ward's method): Complete-Linkage. To recall, the Complete-Linkage algorithm computes a hierarchical clustering by successively merging pairs of clusters whose union will have minimal radius/diameter, starting with the trivial clustering in which every point is contained in a separate clustering. But how good is this approach, especially on a level-by-level basis? Clearly, the first merge is optimal, but this is not true for any of the subsequent merges. As is the case with many greedy algorithms, it might be beneficial to perform a bad merge first if that opens up better merges down the line. The further the algorithm progresses, the more such errors can accumulate. If we compare the clustering computed by Complete-Linkage on every level $k$ with the optimal $k$-center/$k$-diameter clustering for that level, then what is the worst attained approximation ratio?

Dasgupta and Long have already considered this question back in 2005 ([32]) and established a lower bound of $\Omega(\log k)$ on the approximation ratio of the $k$-clustering computed by Complete-Linkage. This is already quite a large lower bound and the factor seems even worse when we compare it with the new (divisive) algorithm designed by Dasgupta and Long that guarantees an approximation ratio of 8 over all levels. This was their main result. However, Complete-Linkage still finds use in practice, likely due to its simplicity and its widespread implementation, so it is still of great interest to establish a corresponding upper bound on it approximation ratio. (We should also keep in mind that the $k$-center and $k$-diameter objective functions are just used as guidance in several applications without directly corresponding to the problem at hand. A worse approximation factor could be compensated by some other positive aspect.) Ackermann et al. were the first to provide such upper bounds in 2014, although only for Euclidean spaces $\mathbb{R}^d$:

1. For the discrete $k$-center objective (where centers have to be chosen from the original set of points) they provide an upper bound of $O(d + \log k)$.
2. For the $k$-diameter objective they provide an upper bound of $f(d) \cdot O(\log k)$, where $f$ is an exponential function.
3. For the $k$-center objective they provide an upper bound of $f(d) \cdot O(\log k)$, where $f$ is a doubly exponential function.

They also established different lower bounds for $\ell_p$-norms and thus for more "natural" metrics than the one considered by Dasgupta and Long, although they did not improve the lower bound by Dasgupta and Long. Großwendt and Röglin subsequently improved the three upper bounds in 2017 by getting rid of the $\log k$ factors with the help of a better analysis of the last $k$ merges performed by Complete-Linkage ([49]). In the following sections, we will establish much better lower bounds of $\Omega(k)$ for both the $k$-center and $k$-diameter objectives on general metric spaces. This is a significant jump: in this regard Complete-Linkage is not even better than Single-Linkage! We will derive an upper bound of $O(k)$ for the $k$-center objective, showing that this bound is tight in this case. For the $k$-diameter objective we are only able to prove an upper bound of $O(k^2)$ (which can be improved to $O(k^{\ln 3/\ln 2})$ without too much extra work), still leaving a not-insignificant gap to the lower bound. These results have all been published in [9] and [10]. Recently, Dasgupta and Laber were able to, among other things, improve the still standing upper bound to $O(k^{1.30})$.

## 2.2 Approximation Guarantees for Single-Linkage

Before analyzing Complete-Linkage, we first spend some time with Single-Linkage to familiarize ourselves with some themes, notations, and methods relevant to general analyses of agglomerative clustering procedures. In some sense, the analysis below is not fair since Single-Linkage does optimize a very different objective from the one in regards to which we will assess it. Single-Linkage is excellent, in fact optimal, at separating clusters, meaning that it maximizes the smallest distance between any pair of clusters. This is very important when one wants to clearly delineate between different clusters and force similar objects to belong to the same cluster. However, this comes at the cost of potentially including very different objects in the same clusters. This is what Complete-Linkage does much better, at least to some extent.

### 2.2.1 Lower Bounds

Although it has been known for a long time that Single-Linkage is prone to chaining, we mainly restrict ourselves to analyzing the instance outlined by Dasgupta and Long [32]. Chaining refers to the process of merging clusters that are close to each other without regard to the resulting radius or diameter. So although Single-Linkage optimally solves the separation problem, its cluster can grow quite large and include very dissimilar points. In Dasgupta and Long's example, Single-Linkage builds chains of points in such a way that the resulting $k$-clustering has a radius (and diameter) that is worse by a factor of $\Omega(k)$ from that of an optimal $k$-clustering. We first discuss this instance and then show that this is, in fact, the worst scenario, meaning that fact Single-Linkage never computes something worse than an $O(k)$-approximative solution for the hierarchical $k$-center and $k$-diameter problem.

Single-Linkage is very good at finding linear structures — but only at the detriment of not being able to recognize more complex configurations. [1] Since it always merge clusters that are closest, this behavior can be exploited quite easily to build chain-clusters than span a large area. The example Dasgupta and Long give is a set of $n$ points $x_0, \ldots, x_{n-1}$ located on the real line with almost equal spacing between subsequent points. More precisely, the distance between all pairs of points $x_i$ and $x_{i+1}$ is equal to $1 - \varepsilon i$ for some fixed, but arbitrarily small $\varepsilon > 0$, as shown in Section 2.2.1. Although $\varepsilon$ might be arbitrarily small, we have thereby forced an ordering $d(x_{n-1}, x_{n-2}) < \ldots < d(x_1, x_2) < d(x_i, x_j)$ on the distances, where $i$ and $j$ are any indices with $|i - j| \geq 2$. As such, Single-Linkage will merge clusters from right to left until it reaches the clustering

$$\mathscr{C}_k = \{\{x_i\} \mid i \leq k-1\} \cup \{\{x_k, \ldots x_{n-1}\}\}.$$

All points to the left of $x_k$ (excluding $x_k$) remain singleton clusters, whereas all points to the right of $x_k$ (including $x_k$) have been merged into one cluster. The latter has a diameter of

$$d(x_k, x_{n-1}) = \sum_{i=k}^{n-2} (1 - \varepsilon i) = (n - k - 1) - \varepsilon \sum_{i=k}^{n-2} i,$$

which is much worse than just uniformly partitioning the point set. To see this, assume that $n = ks$ is a multiple of the cluster size. While not a strictly necessary

[1]This was noted as early as 1978 by Hansen and DeLattre ([51]): "clusters given by the single-link algorithm may have little homogeneity, "

assumption, this assumption allows us to simplify the analysis. Partitioning the point set along evenly spaced cuts, we get the clustering

$$\mathscr{O}_k = \{\{x_{is}, \ldots x_{(i+1)s-1}\} \mid i \in 0, \ldots, k-1\},$$

of which each cluster has a diameter of at most

$$d(x_{js}, x_{(j+1)s-1}) = \sum_{i=js}^{(j+1)s-2} (1 - \varepsilon i) = (s+1) - \varepsilon \sum_{i=k}^{n-2} i.$$

If $n \geq k^2$ we see that solution computed by Single-Linkage is only $\Omega(k)$-approximative:

$$\frac{\operatorname{diam}(\mathscr{C}_k)}{\operatorname{diam}(\mathscr{O}_k)} \approx \frac{n-k-1}{\frac{n}{k}+1} > k - \frac{k^2}{n} \geq k-1.$$

We just immediately discard the $\varepsilon$-terms since they are negligible if $\varepsilon$ is chosen small enough. Basically, the same analysis, although slightly more lengthy to account for optimal centers, also shows that the same result holds for the $k$-center objective without changing any of the arguments.

**Corollary 17.** *Single-Linkage computes $\Omega(k)$-approximative solutions for the hierarchical $k$-center and $k$-diameter problems in the worst case.*

Complete-Linkage avoids falling into this trap. If there are singletons (or other clusters with a relatively small cost) close to each other, Complete-Linkage will merge them first. However, as it turns out, if Complete-Linkage chooses bad merges for every cluster, one can still induce a particular type of chaining behavior, which we will analyze in the next section. This requires that all (or almost all) clusters are unfavorable for this to work, so it is still more robust than Single-Linkage; the average approximation ratio is much smaller, as we will see. It takes much longer for bad clusters to build in Complete-Linkage. Still, if one is interested in a specific $k$-clustering, where $k$ is small compared to $n$, then one might want to choose a different algorithm than Complete-Linkage. All of this will be shown explicitly in the next chapter.

### 2.2.2 Matching Upper Bounds

Having thus highlighted one of the fundamental problems of Single-Linkage — at least as it pertains to the size, or rather, to the diameter and radii of clusters — we still have to make sure that the above instance is, in fact, a worst-case instance and that Single-Linkage always computes solutions that are at least $O(k)$-approximative.

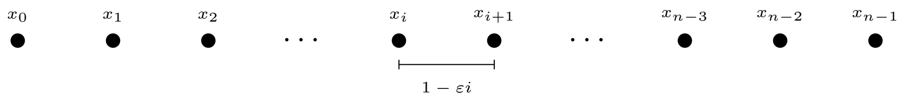


Figure 2.1: A rendition of Dasgupta and Long's worst-case example for Single-Linkage. The only difference here is that the indices start at 0 (and thereby slight shift the distances between subsequent points, but this is of no consequence). Although this figure is almost identical to Figure 9 (a) of [32] we have recreated it here to make this section self-contained.

We will consider the $k$-diameter and $k$-center objective function separately because even though the analyses are quite similar, the notational overlap is significant. This way, we can reuse the same names for different clusterings without further differentiating them.

**A Matching Upper Bound for the $k$-Diameter Objective** To provide an $O(k)$ upper bound for the approximation ratio of Single-Linkage with regards to the $k$-diameter objective function, consider any hierarchical sequence of clustering $(\mathscr{C}_k)_{k=1}^n$ computed by Single-Linkage for some instance $(X, d)$. What will allow us to estimate the cost of any clustering $\mathscr{C}_k$ nicely is the fact that — and this is the main characteristic of Single-Linkage — there always exists a relatively small sequence of relatively close points that spans the largest cluster contained in $\mathscr{C}_k$. Let $(\mathscr{O}_k)_{k=1}^n$ denote the optimal $k$-diameter solutions for $(X, d)$ that will serve as our points of reference.

To formalize our idea of estimating cluster costs by the triangle inequality over small sequences of points that do not include large gaps, we introduce an undirected graph $G_k$ on top of $\mathscr{O}_k$ and connect all pairs of vertices, depending on whether the corresponding optimal clusters are close to each other. More precisely, we add to $V(G_k) = \mathscr{O}_k$ the following edges:

$$E(G_k) = \{\{O, O'\} \subseteq \mathscr{O}_k \mid d(O, O') \leq \operatorname{diam}(\mathscr{O}_k)\}.$$

Whenever the distance between two optimal clusters is upper bounded by the value $\operatorname{diam}(\mathscr{O}_k)$ of that optimal solution, we connect those clusters. What is essential here — and what made us choose the above edge set — is that for any cluster $C \in \mathscr{C}_k$, the set of all optimal clusters intersected by $C$ induces a nice connected subgraph of $G_k$. Before proving this statement, let us properly introduce a notation for the set of all optimal clusters intersected by some set — we will use such sets a lot in the following analyses.

**Definition 18.** Let $X$ be a set. For any subset $Y \subset X$ and any set of subsets $\mathscr{X} \subset 2^X$, we set $\mathscr{X}(Y) = \{Z \in \mathscr{X} \mid Z \cap Y \neq \varnothing\}$.

To rephrase the above statement amounts to saying that $G_k[\mathscr{O}_k(C)]$ is connected for every cluster $C \in \mathscr{C}_k$ (see Lemma 19 below). Accordingly, there has to exist, for every pair of points $x, x' \in C$, an $O_x$-$O_{x'}$-path in $G_k$ that connects the optimal clusters $O_x, O_{x'}$ that respectively contain $x$ and $x'$. From this path, we can then pick a nice set of intermediary points between $x$ and $x'$, with the help of which we can then upper bound $d(x, x')$ using the triangle inequality. Since every optimal cluster is small and the distance between subsequent clusters in the $O_x$-$O_{x'}$-path is also small, we should have established enough intuition to be already able to conceive how an $O(k)$-approximation ratio might be possible. Figure 2.2 (a) shows the overall form of such paths. But first, let us prove that the induced subgraphs are connected.

**Lemma 19.** *Let $C \in \mathscr{C}_t$ be any cluster computed by Single-Linkage at some time step $t \geq k$. Then, the subgraph $G_k[\mathscr{O}_k(C)]$ is connected.*

*Proof.* We prove the lemma by induction. In the beginning (when $t = n$), the lemma holds since any cluster contained in $\mathscr{C}_n$ consists of a single point and thus intersects exactly one optimal cluster. Assume now that the claim holds for some $t > k$. By the pigeonhole principle there exist two clusters $D_1, D_2 \in \mathscr{C}_t$ that intersect a same cluster $O \in \mathscr{O}_k(D_1) \cap \mathscr{O}_k(D_2)$. We thus know that $d(D_1, D_2) \leq \operatorname{diam}(\mathscr{O}_k)$ and this is exactly the value that Single-Linkage minimizes. By extension, it follows that if Single-Linkage merges the cluster pair $C_1, C_2 \in \mathscr{C}_t$ in this time step, then these still have to satisfy the upper bound we just derived. In other words, we know that

$$d(\mathscr{O}_k(C_1), \mathscr{O}_k(C_2)) \leq d(C_1, C_2) \leq d(D_1, D_2) \leq \operatorname{diam}(\mathscr{O}_k),$$

which means that an edge between both sets of optimal clusters $\mathscr{O}_k(C_1)$ and $\mathscr{O}_k(C_2)$ exists. Combining this with the induction hypothesis that $G_k[\mathscr{O}_k(C_1)]$ and $G_k[\mathscr{O}_k(C_2)]$ are themselves connected, it follows that $G_k[\mathscr{O}_k(C_1 \cup C_2)]$ is also connected. $\square$

With the help of this lemma, we can now estimate the diameter of any Single-Linkage cluster $C$ by applying the triangle inequality onto (although not directly) an appropriate path in $\mathscr{O}_k(C)$ since both the inter-cluster and intra-cluster distances are upper bounded by $\operatorname{diam}(\mathscr{O}_k)$. Yields uncoiling this path within our original space $X$ yields our desired upper bound.

**Theorem 20.** *For all $1 \leq k \leq n$ it holds that $\operatorname{diam}(\mathscr{C}_k) \leq 2k \cdot \operatorname{diam}(\mathscr{O}_k)$.*

*Proof.* Consider an arbitrary cluster $C \in \mathscr{C}_k$ and two of its points $x, y \in C$. Both points are contained in some optimal clusters $O_x, O_y \in \mathscr{O}_k(C)$, which are in turn connected by some path $P$ in $G_k[\mathscr{O}_k(C)]$ since the latter is connected.

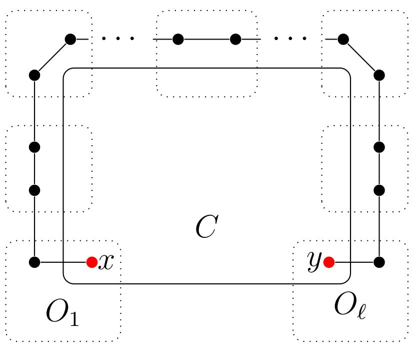


Figure 2.2: Cluster $C$ intersects the optimal clusters $O_1, \dots, O_\ell \in \mathscr{O}_k$ whose distance is at most $d(O_i, O_{i+1}) \leq \operatorname{diam}(\mathscr{O}_k)$. Since each optimal cluster has itself also a diameter of at most $\operatorname{diam}(\mathscr{O}_k)$, the diameter of $C$ is altogether at most $2\ell \operatorname{diam}(\mathscr{O}_k)$.

Without loss of generality, we can assume that $P$ passes through the optimal clusters $O_x = O_1, \dots, O_\ell = O_y$ in exactly this order by appropriately indexing the optimal clusters. Uncoiling this path in $X$ then gives us an upper bound of $2k \operatorname{diam}(\mathscr{O}_k)$ for $d(x,y)$ as follows. For each $i = 1, \dots, \ell - 1$ there exist points $x_i \in O_i$ and $y_{i+1} \in O_{i+1}$ such that both $d(y_i, x_i) \leq \operatorname{diam}(\mathscr{O}_k)$ and $d(x_i, y_{i+1}) \leq \operatorname{diam}(\mathscr{O}_k)$. Adding up these distances yields an upper bound of

$$\begin{aligned} d(x,y) &\leq d(x,x_1) + d(y_\ell, y) + \sum_{i=1}^{\ell-1} \left( d(x_i, y_{i+1}) + d(y_{i+1}, x_{i+1}) \right) \\ &\leq 2\ell \operatorname{diam}(\mathscr{O}_k) \leq 2k \operatorname{diam}(\mathscr{O}_k). \end{aligned}$$

Since $C$ and $x$ and $y$ were chosen arbitrarily, the claimed upper bound for Single-Linkage holds for the $k$-diameter objective function.

□

**A Matching Upper Bound for the $k$-Center Objective** We can prove in a similar fashion that the $k$-clustering computed Single-Linkage is an $O(k)$-approximative solution for the $k$-center problem. The only significant difference is that we must keep track of viable centers. As before, take any hierarchical clustering $(\mathscr{C}_k)_{k=1}^n$ computed by Single-Linkage for some instance $(X,d)$ and let $(\mathscr{O}_k)_{k=1}^n$ denote the optimal individual $k$-center clusterings. We will again formalize our argument with the help of some undirected graph $G_k$ on $\mathscr{O}_k$, which, however, has to be set up slightly differently this time. The vertices $V(G_k) = \mathscr{O}_k$ are connected by edges whenever their distance is upper bounded by $2 \operatorname{rad}(\mathscr{O}_k)$, meaning that

$$E(G_k) = \{\{O, O'\} \subset \mathscr{O}_k \mid d(O, O') \leq 2 \operatorname{rad}(\mathscr{O}_k)\}.$$

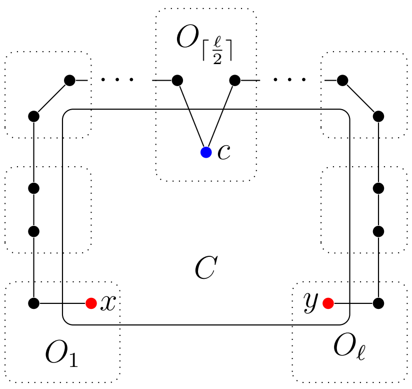


Figure 2.3: Cluster $C$ intersects the optimal clusters $O_1, \dots, O_\ell$ whose consecutive distances are upper bounded by $d(O_i, O_{i+1}) \leq 2\,\mathrm{rad}(\mathscr{O}_k)$. We see that choosing a center $c$ from $O_{\lceil\frac{\ell}{2}\rceil}$ induces a radius of at most $2(\ell+1)\,\mathrm{rad}(\mathscr{O}_k)$.

**Lemma 21.** *Let $C \in \mathscr{C}_t$ be any cluster computed by Single-Linkage at some time step $t \geq k$. Then, the subgraph $G_k[\mathscr{O}_k(C)]$ is connected.*

*Proof.* The proof is very similar to that of Lemma 19 and also proceeds by induction. Initially (when $t = n$), the lemma holds since any cluster contained in $\mathscr{C}_n$ consists of a single point and thus intersects exactly one optimal cluster. Assume now that the claim holds for some $t > k$. By the pigeonhole principle there exist two clusters $D_1, D_2 \in \mathscr{C}_t$ whose centers $c_1 \in D_1$, $c_2 \in D_2$ lie in a same optimal cluster $O \in \mathscr{O}_k(D_1) \cap \mathscr{O}_k(D_2)$. We thus know that

$$d(D_1, D_2) \leq d(c_1, c_2) \leq 2\,\mathrm{rad}(O) \leq 2\,\mathrm{rad}(\mathscr{O}_k)$$

and this is exactly the value that Single-Linkage minimizes. By extension, it follows that if Single-Linkage merges the cluster pair $C_1, C_2 \in \mathscr{C}_t$ in this time step, then these still have to satisfy the upper bound we just derived. In other words, we know that

$$d(\mathscr{O}_k(C_1), \mathscr{O}_k(C_2)) \leq d(C_1, C_2) \leq d(D_1, D_2) \leq 2\,\mathrm{rad}(\mathscr{O}_k),$$

which means that an edge between both sets of optimal clusters $\mathscr{O}_k(C_1)$ and $\mathscr{O}_k(C_2)$ exists. Combining this with the induction hypothesis that $G_k[\mathscr{O}_k(C_1)]$ and $G_k[\mathscr{O}_k(C_2)]$ are themselves connected, it follows that $G_k[\mathscr{O}_k(C_1 \cup C_2)]$ is also connected. □

**Theorem 22.** *For all $1 \leq k \leq n$ it holds that $\mathrm{rad}(\mathscr{C}_k) \leq 2(k+1) \cdot \mathrm{rad}(\mathscr{O}_k)$.*

*Proof.* Consider an arbitrary cluster $C \in \mathscr{C}_k$ and denote by $P$ a longest simple path in $G_k[\mathscr{O}_k(C)] \subseteq G_k$. Without loss of generality, we can assume that $P$ passes through the optimal clusters $O_1, \dots, O_\ell$ in exactly this order. Then, if we choose an arbitrary point $c \in C \cap O_{\lceil \frac{\ell}{2} \rceil}$ from an optimal cluster $O_{\lceil \frac{\ell}{2} \rceil}$ lying in the middle of $P$, every other point will be close enough to it to derive the desired bound for $\operatorname{rad}(C)$. Since $P$ is a longest path, every optimal cluster in $G_k[\mathscr{O}_k(C)]$ is reachable from $O_{\lceil \frac{\ell}{2} \rceil}$ by a path of length at most $\lceil \frac{\ell}{2} \rceil$. (Recall that $G_k[\mathscr{O}_k(C)]$ is indeed connected by Lemma 21.) Uncoiling these paths in $X$ now gives us an upper bound of $2(k+1)\operatorname{rad}(\mathscr{O}_k)$ for the distance between $c$ and any other point $z \in C$ as follows: If $O_z \in \mathscr{O}_k$ is the optimal cluster containing $z$, then by our choice of $O_{\lceil \frac{\ell}{2} \rceil}$, there exists a path $O_{\lceil \frac{\ell}{2} \rceil} = O_{i_1}, \dots, O_{i_{\lambda+1}} = O_z$ in $G_k[\mathscr{O}_k(C)]$ of length $\lambda + 1 \leq \lceil \frac{\ell}{2} \rceil \leq \lceil \frac{k}{2} \rceil$ connecting them. That means, for each $j = 1, \dots, \lambda$ there exist points $x_i \in O_{i_j}, y_{i+1} \in O_{i_{j+1}}$ such that $d(x_i, y_{i+1}) \leq 2\operatorname{rad}(\mathscr{O}_k)$. Hence

$$\begin{aligned} d(c,z) &\leq d(c,x_1) + \sum_{i=1}^{\lambda-1} \left(d(x_i,y_{i+1}) + d(y_{i+1},x_{i+1})\right) \\ &\quad + d(x_\lambda, y_{\lambda+1}) + d(y_{\lambda+1}, z) \\ &\leq 2(2\lambda+1)\operatorname{rad}(\mathscr{O}_k) \leq 2(k+1)\operatorname{rad}(\mathscr{O}_k). \end{aligned}$$

We are done since $C$ was an arbitrary cluster and $z \in C$ an arbitrary point.

□

## 2.3 Lower Bounds for Complete-Linkage

Theoretical analyses of Complete-Linkage are sparse, and of those, only three are relevant to this chapter. The first is by Dasgupta and Long [32], who were one of the first to analyze Complete-Linkage from a purely theoretical perspective and provide a preliminary assessment of the performance guarantees satisfied by the Complete-Linkage algorithm. However, this assessment was purely negative. In the worst case, it only provided a lower bound of $\Omega(\log k)$ for the approximation ratio attainable by Complete-Linkage, leaving several questions unanswered. [2] Most importantly, as Dasgupta and Long themselves ask at the end of their paper, is this lower bound already the worst possible? It took almost a decade until Ackermann et al. [3], focusing on $\ell_p$-metrics, would establish further lower bounds and derive the first non-trivial upper bounds. Regarding the lower bounds, Ackermann et al. showed that even for $\ell_p$-metrics a lower bound of $\Omega(\sqrt[p]{\log_2 k})$ can be established for the discrete $k$-center and $k$-diameter variants, providing an alternative to Dasgupta and Long's lower bound which was itself only confirmed by a somewhat

[2] This is understandable since they also designed — and this was their primary focus — an algorithm with a much better (i.e., constant) approximation ratio. Nonetheless, Complete-Linkage is still used in practice, so establishing positive bounds is very much desirable.

"artificial" metric. Regarding the upper bounds, Ackermann et al. were able to show that the approximation ratio of Complete-Linkage is in $f(d) \cdot \log k$, where $f(d)$ is linear in the dimension $d$ in the case of the discrete $k$-center variant, exponential in $d$ in the case of the continuous $k$-center variant and doubly exponential in $d$ in the case of the $k$-diameter variant. These upper bounds were subsequently improved by Großwendt and Röglin [49] in 2017 by refining the analysis of the last $k$ merges, thereby getting rid of the $\log k$ factor in the upper bound. In other words, the approximation ratio of Complete-Linkage in all its three variants now only depends on $d$. Note that this does not contradict the lower bounds previously established by Ackermann et al. since their worst-case instances have a dimension that depends linearly on $k$. Also note, that Großwendt and Röglin's analysis does not make much use of the geometric properties of $\mathbb{R}^d$, if at all, but cannot by itself yield any bounds for Complete-Linkage on general (that is, abstract) metric spaces since it builds upon the preliminary cost analyses by Ackermann et al. and improves the assessment of subsequent merges. What is missing is an initial estimation that does not utilize any geometric structure inherent to $\ell_p$-metrics. Our contribution, and the goal of this chapter, is to provide lower and upper bounds for Complete-Linkage in its processing of abstract metrics. More precisely, we will show that the previously best-known lower bounds of $\Omega(\log k)$ for both $\mathbf{CL}^{\text{diam}}$ and $\mathbf{CL}^{\text{rad}}$ can be raised to $\Omega(k)$, tying them with Single-Linkage in the worst case.

The structure of this section is relatively straightforward. We first review and discuss both previously established lower bounds in detail (once by Dasgupta and Long and once by Ackermann et al.) and try to work out some of the fundamental themes undergirding their analyses. Each discussion has, in turn, been divided into two parts, the first being an introduction to and an outline of the respective worst-case instance, the second being an in-depth breakdown of the resulting behavior of Complete-Linkage. Both lower bounds share significant conceptual similarities, which have, however, already reached their limit in some sense. We will have to considerably change our approach to constructing such instances to push the lower bound beyond just a logarithmic dependency on $k$. We will do this in the subsequent section by drawing on the intuition we have built up thus far. Again, this section is split into a first part that introduces the worst-case instance and a second part that analyses this instance. This time, the second part is much longer and is itself subdivided into two parts, accounting separately for $\mathbf{CL}^{\text{diam}}$ and $\mathbf{CL}^{\text{rad}}$. The latter is slightly more complicated to analyze since we additionally have to keep track of optimal centers. In the end, we close this chapter with a primarily technical addendum that fills in what could be considered a hole in our discussion. While we provide an instance for which $\mathbf{CL}^{\text{diam}}$ and $\mathbf{CL}^{\text{rad}}$ might both yield only $\Omega(k)$-approximative solutions, these solutions are not enforced. By this, we mean that they depend on how Complete-Linkage breaks upcoming ties during its runtime. Although this is in some respects not that significant, since no evaluation could distinguish good from bad tiebreaks, we describe a modification of our instance that forces Complete-Linkage at each step to choose the worst merge. So even with “luck”, Complete-Linkage cannot compute good solutions for this instance.

### 2.3.1 Dasgupta and Long's Lower Bound

In 2005, Dasgupta and Long [32] have constructively shown that for each $k \in \mathbb{N}$, there exists an instance $(X_k, d)$ on which $\mathbf{CL}^{\text{diam}}$ produces a $k$-clustering whose *diameter* is worse by a factor of $\Omega(\log k)$ than that of an optimal $k$-clustering. We go through this construction and the corresponding analysis in full since, already here, we encounter the most important ideas that guide all later worst-case constructions.

#### The Instance

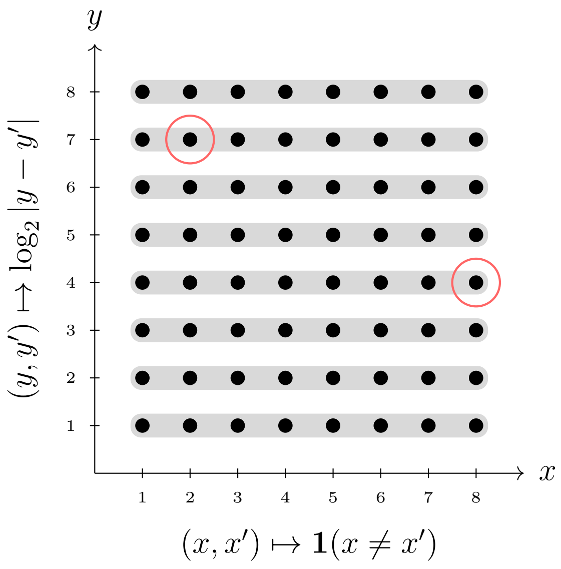
The instance that Dasgupta and Long designed consists of $k^2$ points placed on a $k \times k$ grid together with a metric that clearly distinguishes between vertical and horizontal distances, measuring them differently and only later adding them up. Merging points along the $x$-axis will be optimal, while merging points along the $y$-axis will accrue a much larger cost. The trick is that, in the way those distances are set up, Complete-Linkage will not be able to distinguish between those two choices. Since this description was quite vague, let us formally define the instance first before trying to motivate all choices involved in this definition. Dasgupta and Long's worst-case instance $(X_k, d)$ consists of

1. the underlying set $X_k = \{1, \dots, k\}^2 = \{(i, j) \mid i, j \in \{1, \dots, k\}\}$
2. the metric $d((x, y), (x', y')) = \mathbb{1}[x \neq x'] + \log_2(1 + |y - y'|)$,

where $\mathbb{1}[x \neq x']$ denotes the indicator-function, which is 1, when $x \neq x'$ and 0 otherwise. A rendition is given in Section 2.3.1 for $k = 8$. To get a feeling for this metric, consider the points $(2, 7)$ and $(8, 4)$, highlighted in the figure. Then, their distance is computed by

$$d((2, 7), (8, 4)) = \mathbb{1}[2 \neq 8] + \log_2(1 + |7 - 4|) = 1 + \log_2 4 = 3.$$

The horizontal component of both points is different, so they incur a cost of 1 from the indicator function $\mathbb{1}[x \neq x']$. Here, it does not matter how large the horizontal distances are! (This will become very important later on.) At the same time, their vertical distance is 3, which, when plugged into the logarithmic component of the metric, evaluates to 2. Hence, the distance is 3 altogether.



$y$
$(y, y') \mapsto \log_2 |y - y'|$
8
7
6
5
4
3
2
1
$x$
1 2 3 4 5 6 7 8
$(x, x') \mapsto \mathbf{1}(x \neq x')$

Figure 2.4: A rendition of the Dasgupta and Long's Complete-Linkage instance for $k = 8$. The gray lines indicate an optimal $k$-clustering.

### The Metric

Now that we have established Dasgupta and Long's worst-case instance and done some first computations let us take a closer look at the metric. Because this metric seems somewhat unintuitive at first sight, we now spend some time just examining and dissecting it, envisioning $(X_k, d)$ as the $\ell_1$-product of two relatively simple metric spaces $(H_k, d_h)$ and $(V_k, d_v)$. By this, we mean that the underlying set will be defined as the cartesian product $X_k = H_k \times V_k$ and the metric as the sum $d(x, x') = d_h(\pi_1(x), \pi_1(x')) + d_v(\pi_2(x), \pi_2(x'))$, where $\pi_1 : X_k \to H_k$ is the usual projection onto the first coordinate and $\pi_2$ the projection onto the second. (This is why we have split the metric into two constituent parts, each placed along the axis, in Section 2.3.1.) According to a more general but well-known and easily verified notion of product metrics, this construction indeed forms a metric space.

**Proposition 23** (Product Metric)**.** *Let $(Y_1, d_1), \ldots, (Y_m, d_m)$ be metric spaces and $p \geq 1$. Then $(Y, d)$ with $Y = \prod_i Y_i$ and*

$$d((y_1, \ldots, y_m), (y_1', \ldots, y_m')) = \left( \sum_{i=1}^{m} d_i(y_i, y_i') \right)^{\frac{1}{p}}$$

*also forms a metric space.*

As we have already seen, visually, this $\ell_1$-product should consist of $k^2$ points placed on a $k \times k$ grid with a spacing of 1 and a different distance function for each axis. Horizontally, the distance is equal to the discrete metric; vertically, it is measured by a logarithmic function. The reason for this decoupling is that, given such a setup, $\mathbf{CL}^{\text{diam}}$ might unite vertically neighboring points and so construct the columns $\{\pi_1^{-1}(1), \ldots, \pi_1^{-1}(k)\}$ as its $k$-clustering, instead of uniting horizontally neighboring points to get the rows $\{\pi_2^{-1}(1), \ldots, \pi_2^{-1}(k)\}$. Whereas the former $k$-clustering has a diameter of 1, the latter has a diameter of $\log_2 k$. This is the core insight. $\mathbf{CL}^{\text{diam}}$ is unable to distinguish merges, where a cost is incurred only once (the horizontal merges), and merges, where a cost is incurred every time (the vertical merges).

To properly formalize this sketch, set the underlying sets of both auxiliary spaces to $H_k = V_k = \{1, \ldots, k\}$. The first metric $d_h : H_k \times H_k \to \mathbb{R}$ is then nothing but the discrete metric, meaning that $d_h(x, x') = \mathbb{1}[x \neq x']$, but the second is slightly more intricate. We start with the standard metric $(a, b) \mapsto |a - b|$ for $\mathbb{R}$ and deform it with the help of a metric-preserving function, which, by definition, always yields a metric when composed with some other metric.[3] To arrive at our vertical distance measure $d_v(x, x') = \log_2(1 + |x - x'|)$, all we have to do is to show that $x \mapsto \log_2(1 + x)$ belongs to this class of functions. But this follows from the well-known fact that all monotonically increasing and concave functions are metric-preserving, so long as they vanish precisely and only at $x = 0$ (Theorem 3 in [34]).

Before analyzing this instance in detail, it should be noted that we have slightly modified Dasgupta and Long's instance, specifically $V_k$, to simplify later descriptions. The trade-off is that this simplification introduces some ambiguity in the form of additional but significant tiebreaks. Whereas the tiebreaks in the original instance do not influence the overall clustering process of $\mathbf{CL}^{\text{diam}}$, always steering it toward a bad clustering, in this new instance, it could, "by chance", compute an optimal clustering. Depending on how ties are broken, the rows could be reconstructed. However, no general rule can differentiate between good and bad tiebreaks, so this is not a big deal. So, while we could minimally reduce the vertical space, as Dasgupta and Long have done, to force vertical merges, we leave that aside for now to better focus on the overall picture.

## The Analysis

As mentioned above, the main problem that this instance poses for $\mathbf{CL}^{\text{diam}}$ is that the latter might merge points along the columns instead of the rows. In the latter case, a single merge already pays for all upcoming merges within that

[3]For more information on metric-preserving functions, we refer the reader to [34].

row. That is, the diameter of the resulting cluster does not increase with any subsequent horizontal merges. The same, however, does not hold in the other case. Whenever two vertically neighboring clusters are merged, the diameter increases by a strictly positive amount. In the worst case, this amount equals 1, the same as any horizontal merge induces. Each such merge doubles the number of intersected optimal clusters, so they can be stacked up to $\log_2 k$ times, yielding a $k$-clustering worse by a factor of $\Omega(\log_2 k)$.

Before analyzing how these cases unfold, we must introduce a few definitions to handle any ensuing tiebreaks properly.

**Definition 24.** Given a clustering $\mathscr{C}$ and a linkage function $f$, we say that a merge $C = C_1 \cup C_2$ with $C_1 \neq C_2 \in \mathscr{C}$ is *$f$-viable*, if it is cheapest among all merges available with regards to $f$, i.e., if it minimizes

$$f(C) = \min\{f(C_1' \cup C_2') \mid C_1' \neq C_2' \in \mathscr{C}\}.$$

The clustering $\mathscr{C}$ itself is said to be *$f$-viable*, if all merges leading up to it, starting from $\{\{x\} \mid x \in X_k\}$, are $f$-viable. That is, $\mathscr{C}$ is $f$-viable if a sequence of tiebreaks exists such that Algorithm 1 produces it.

Since it is usually apparent from the context which linkage function we currently consider, we will just write *viable* instead of the sometimes more clumsy $f$-viable. For example, in this section, we only work with the diameter linkage function. Having established these definitions, we can now formulate the main claim.

**Theorem 25** (Cf. Section 4.3 of [32])**.** *Consider $(X_k, d)$ as constructed above for an arbitrary $k \in N$. Then*

1. *The rows $\{\pi_2^{-1}(1), \dots, \pi_2^{-1}(k)\}$ form an optimal $k$-clustering of $X_k$ with a diameter of 1.*
2. *The columns $\{\pi_1^{-1}(1), \dots, \pi_1^{-1}(k)\}$ form a viable $k$-clustering of $X_k$ with a diameter of $\log_2 k$.*

The first of these two points can be proven relatively quickly. It follows almost directly from the fact that the distance between any two non-equal points in $X_k$ is at least 1.

**Observation 26.** Let $x, x' \in X_k$ be two different points. Then $d(x, x') \geq 1$.

*Proof.* Since they are non-equal, $x$ and $x'$ must differ in at least one coordinate. If they differ in the first coordinate, then

$$d(x, x') \geq d_h(\pi_1(x), \pi_1(x')) = \mathbb{1}[\pi_1(x) \neq \pi_1(x')] = 1.$$

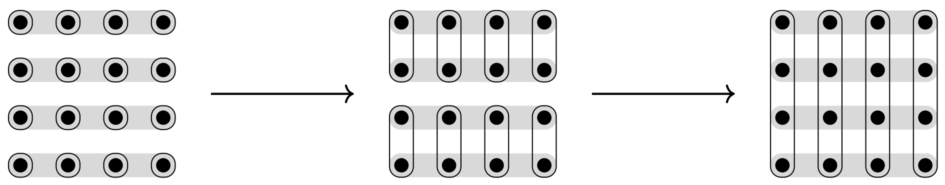


Figure 2.5: An outline of the first two phases (represented by the arrows) when $k = 4$. They gray rows again indicate the optimal clustering.

If they differ in the second, then

$$d(x, x') \geq d_v(\pi_2(x), \pi_2(x')) \geq \log_2(1 + |\pi_2(x) - \pi_2(x')|) \geq \log_2 2 = 1,$$

since the vertical spacing between points is at least 1. □

**Corollary 27.** *The rows* $\mathscr{C}^* = \{\pi_2^{-1}(1), \ldots, \pi_2^{-1}(k)\}$ *of* $X_k$ *form an optimal* $k$*-clustering of diameter* $\operatorname{diam}(\mathscr{C}^*) = 1$*.*

*Proof.* Since $X_k$ consists of $k^2$ many points, any $k$-clustering $\mathscr{C}$ of $X_k$ has to contain a cluster $C$ of size at least 2. From the previous observation, it thus follows that $\operatorname{diam}(\mathscr{C}) \geq \operatorname{diam}(C) \geq 1$ and hence that an optimal $k$-clustering has to have a cost of at least 1. But then the rows, whose diameter is purely measured by the discrete metric and thus equal to 1, necessarily are optimal. □

Having established that the rows form an optimal $k$-clustering, we can now turn toward the more interesting analysis of the columns.

**Corollary 28.** *The columns* $\mathscr{C} = \{\pi_1^{-1}(1), \ldots, \pi_1^{-1}(k)\}$ *of* $X_k$ *form a viable* $k$*-clustering with* $\operatorname{diam}(\mathscr{C}) = \log_2(k)$*.*

*Proof.* We will show that the columns form a viable clustering by providing an appropriate merge sequence, which is itself split into different phases. To simplify, we will presume, like Dasgupta and Long, that $k = 2^\lambda$ is some power of 2. The general case unfolds no differently but requires a more elaborate description. Intuitively, the following happens in each phase: divide each column into sets of 2 neighboring clusters and merge those. Since the vertical measure is logarithmic, the diameter of the resulting clusters increases by an additional factor of 1, which is at least as good as any horizontal or diagonal merge. A rendition of the phases can be found in Section 2.3.1 when $k = 4$.

To properly outline the process, let $C_{i,j}^{(0)} = \{(i, j)\}$, where $i$ and $j$ range over the columns and rows respectively, denote the clusters with which $\mathbf{CL}^{\text{diam}}$ initially starts. Formalizing the above description, we inductively assemble new clusters

$C_{i,j}^{(\ell)} = C_{i,2j-1}^{(\ell-1)} \cup C_{i,2j}^{(\ell-1)}$ with each phase. What we want to show now is that the corresponding clusterings

$$\mathscr{C}^{(\ell)} = \left\{ C_{i,j}^{(\ell)} \mid i \in \{1, \ldots, 2^\lambda\}, j \in \{1, \ldots, 2^{\lambda-\ell}\} \right\}$$

after each phase are, in fact, viable clusterings of diameter $\ell$.

Visually, the part about the diameter is almost self-evident. By construction, each cluster in $\mathscr{C}^{(\ell)}$ consists of a sequence of $2^\ell$ vertically neighboring points. More precisely, each such cluster is of the form

$$C_{i,j}^{(\ell)} = \{(i, j') \mid j' \in \{(j-1)2^\ell + 1, \ldots, j\, 2^\ell\}\}$$

for appropriate $i$ and $j$ and thus must indeed have a diameter of $\mathrm{diam}(C) = \log_2(1 + (2^\ell - 1)) = \ell$.

To show that this clustering is also viable, consider the two clusters $C_{i,2j-1}^{(\ell-1)}$, $C_{i,2j}^{(\ell-1)} \in \mathscr{C}^{(\ell-1)}$ from which $C_{i,j}^{(\ell)}$ derives. Merging either of these with any other cluster $C_{i,j'}^{(\ell-1)}$ that is also contained in the $i$th column always necessarily induces an additional cost of at least 1 (strictly more than one, if the clusters are not neighboring). Then, what about horizontal merges? Consider merging $C_{i,2j-1}^{(\ell-1)}$ with $C_{i',2j-1}^{(\ell-1)}$ for some $i' \neq i$. This also increases the diameter by 1, since from the topmost point in the first to the bottommost point in the second cluster, we not only pay $\ell - 1$ for the change in the second coordinate but also an additional fee of 1, orthogonal to the first, for the change in the first coordinate. All remaining merges induce a higher cost since not just one but both the horizontal and the vertical costs separately increase by at least 1. Hence $C_{i,j}^{(\ell)}$ and thus $\mathscr{C}^{(\ell)}$ as a whole is viable. □

This wraps up our in-depth review of Dasgupta and Long's lower bound for $\mathbf{CL}^{\mathrm{diam}}$. Broadly speaking, their main idea is to exploit Complete-Linkage's greedy behavior as follows. In their instance, $\mathbf{CL}^{\mathrm{diam}}$ can always grow clusters in two different directions — horizontally and vertically — between which it cannot differentiate. Both horizontal and vertical merges increase the cost by the same amount; however, in the first direction, the diameter increases only once, while it increases with each subsequent merge in the other direction. With every bad merge, the number of intersected optimal clusters is doubled so that the worst-case diameter equals $\log_2 k$. At this point, one might ask at least three further questions:

1. The instance was designed with the diameter in mind. What happens when we consider the radius? Does there also exist a lower bound for $\mathbf{CL}^{\mathrm{rad}}$?

2. The instance, or rather the metric, seems somewhat artificial (at least, this is something that Ackermann et al. remark [3]). Does the lower bound also hold for more "natural" metrics?

3. Is the lower bound of $\Omega(\log_2 k)$ tight? That is, does a matching upper bound exist?

We will not analyze how $\mathbf{CL}^{\mathrm{rad}}$ would process the above instance. Instead, we will look at a different instance in the next section that not only provides a lower bound for radii but is also based on a more natural metric, the $\ell_1$-metric. Although the third question can be answered affirmatively for this metric, the lower bound can be raised to $\Omega(k)$ for more general metrics. Establishing this lower bound is the main goal of this chapter.

## 2.3.2 Ackermann et al.'s Lower Bound

In 2014, Ackermann et al. ([3]) further expanded theoretical knowledge for several Complete-Linkage variants, establishing lower and upper bounds for different norm-induced metrics on $\mathbb{R}^d$. We will look at some upper bounds in the next chapter and only consider the lower bound for the $\ell_1$-metric here.

### The Instance

Ackermann et al. outline a different $\Omega(\log_2 k)$ lower bound instance $(X_k, d)$ for Complete-Linkage, only this time based on the Manhattan metric $d = \ell_1$. Underlying this space is a $(k + \log_2 k)$ dimensional set $X$ that consists of the following $k^2$ points:

$$X = \left\{ \begin{bmatrix} e_i \\ b \end{bmatrix} \middle| \, i \in \{1, \dots, k\},\, b \in \{0,1\}^{\log_2 k} \right\}.$$

Although the instance looks quite different, the main idea is mostly the same. At each step, there are two "directions" — this time to be taken more loosely — along which Complete-Linkage can merge clusters. Let us call the vector comprising the first $k$ coordinates *indicator* and the vector comprising the remaining $\log_2 k$ coordinates *signature.* Merging clusters with the same signature yields the $k$-clustering

$$\left\{ \left\{ \begin{bmatrix} e_i \\ b \end{bmatrix} \middle| \, i \in \{1, \dots, k\} \right\} \middle| \, b \in \{0,1\}^{\log_2 k} \right\},$$

which turns out to be optimal. Two points within such a cluster have a distance of exactly 2 since they only differ in their indicators (recall that those are just canonical unit vectors). However, $\mathbf{CL}^{\mathrm{diam}}$ will instead always merge clusters whose indicator is the same, ultimately yielding a $k$-clustering

$$\left\{ \left\{ \begin{bmatrix} e_i \\ b \end{bmatrix} \middle| \, b \in \{0,1\}^{\log_2 k} \right\} \middle| \, i \in \{1, \dots, k\} \right\},$$

of diameter $\log_2 k$. (Note how the inner and outer set specifications have switched places.) Just as in Section 2.3.1, this latter clustering has to be built step by step, the diameter increasing by 1 with each phase. We will now outline this process without going as in-depth as in the previous section.

### The Analysis

Again, since all coordinates are from $\{0,1\}$ and $d$ is the Manhattan metric, the distance between any pair of points is at least 1. In the first phase, $\mathbf{CL}^{\mathrm{diam}}$ might thus construct clusters of the form

$$\left\{ \begin{bmatrix} e_i \\ 0 \\ b' \end{bmatrix}, \begin{bmatrix} e_i \\ 1 \\ b' \end{bmatrix} \;\middle|\; b' \in \{0,1\}^{(\log_2 k)-1} \right\},$$

whose points only differ in the first coordinate of the signature. Since their distance is 1, all of the merges are viable. Extending the argument, as in the Dasgupta and Long lower bound, it is not difficult to see that $\mathbf{CL}^{\mathrm{diam}}$ might continue with such merges in the future, meaning that

$$\left\{ \begin{bmatrix} e_i \\ 0 \\ 0 \\ b' \end{bmatrix}, \begin{bmatrix} e_i \\ 0 \\ 1 \\ b' \end{bmatrix}, \begin{bmatrix} e_i \\ 1 \\ 0 \\ b' \end{bmatrix}, \begin{bmatrix} e_i \\ 1 \\ 1 \\ b' \end{bmatrix} \;\middle|\; b' \in \{0,1\}^{(\log_2 k)-2} \right\},$$

etc., are all viable clusterings. The resulting $k$-clustering then has a diameter of $\log_2 k$ since each cluster contains a point with all coordinates of the signature equal to 0 and one with all coordinates of the signature equal to 1. Thus, the lower bound established in the previous chapter is not just bound to a very specific and maybe artificial metric — it also holds for the much more common Manhattan distance. Notably, these clusters' diameter is already equal to their radius, so $\mathbf{CL}^{\mathrm{rad}}$ might run into the same problem. The only difference is that the optimal $k$-clustering, in that case, has a cost of 1 instead of 2.

In the next section, we raise the lower bound and show that there even exist instances in which both $\mathbf{CL}^{\mathrm{diam}}$ and $\mathbf{CL}^{\mathrm{rad}}$ yield $k$-clusterings that are worse by a factor of $\Omega(k)$ — and not just $\Omega(\log_2 k)$. However, this surprising leap is only possible by considering more general metrics. Indeed, Ackermann et al. had shown that the lower bound is tight for the $\ell_1$-metric. Nonetheless, there are enough cases in which the distance between two points corresponds to the length of a shortest path connecting them in a weighted graph. We show that in this case, Complete-Linkage produces $k$-clusterings that are as bad as those computed by Single-Linkage, even though Single-Linkage optimizes an entirely different objective function and is known for its chaining behavior.

### 2.3.3 New and Improved Lower Bounds

In this section, we will now show that both Complete-Linkage variants $\mathbf{CL}^{\mathrm{diam}}$ and $\mathbf{CL}^{\mathrm{rad}}$ surprisingly do, in the worst case, not perform asymptotically better than Single-Linkage. That is, for every $k \in \mathbb{N}$, we can provide an instance $X_k$ for which $\mathbf{CL}^{\mathrm{diam}}$ as well as $\mathbf{CL}^{\mathrm{rad}}$ compute a $k$-clustering that is worse than an optimal solution for the respective objective by a factor of $\Omega(k)$. This result improves upon both known lower bounds of $\Omega(\log_2(k))$, established first by Dasgupta and Long and subsequently extended by Ackermann et al. to new classes of metric spaces.

Neither of the two instances discussed so far can readily be adjusted to yield a lower bound of $\Omega(k)$. The problem is that any horizontal merge already pays in full for all the involved rows, meaning all clusters parallel to the merge can be incorporated without increasing the cost. To reach the worst case, one is thus only allowed to merge vertically, but this can be done at most $\log_2(k)$ times. Every vertical merge doubles the number of intersected optimal clusters. The idea is then to set up an instance in such a way that bad merges increase the number of intersected optimal clusters only by 1 while ensuring at the same time that they do not pay for any possible future parallel merges. We construct such an instance inductively by combining smaller components, always shifting them diagonally by 1 in a precise fashion.

#### The Instance

The basic building blocks of our instance will be so-called *$k$-components*.

**Definition 29.** A *$k$-component* $K_k = (G_k, \phi_k)$ is a certain combination of an undirected, weighted graph $G_k$ and a mapping $\phi_k : V(G_k) \to \{1, \ldots, k\}$. The latter, also called $k$-mapping, designates to each point a *level* that will be used to specify inter-component distances and later on even determine the shape of an optimal clustering. The former, referred to as a *$k$-graph*, is a graph consisting of $2^{k-1}$ vertices with edge weights $w_k : E(G_k) \to \mathbb{N}$ that define the distances between the levels. Our worst-case instance, that is, the metric space that we will analyze in this section, is then nothing other than the metric closure of this graph. Both graph and mapping are inductively defined as follows.

1. The 1-graph consists of a single point $x$ without any edges. Setting the level of this singular to $\phi_1(x) = 1$ gives us our 1-component.

2. Suppose we have already specified the $(k-1)$-component. Take two such components $(G_k^{(0)}, \phi_k^{(0)})$, $(G_k^{(1)}, \phi_k^{(1)})$ and combine them as follows. First, start with a preliminary definition of the $k$-graph $G_k = G_{k-1}^{(0)} \coprod G_{k-1}^{(1)}$ as the disjoint

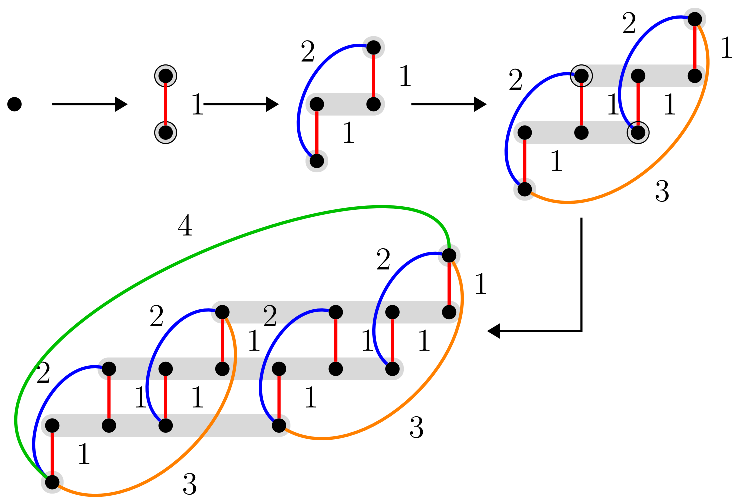


Figure 2.6: The progression of the first 5 components $K_1, \ldots, K_5$. The gray sets indicate points on the same level and form the optimal clusters. When analyzing the instance for the radius, the encircled points in the $K_2$ and $K_4$ component indicate their optimal centers.

union of both $(k-1)$-graphs. All we have to do now is to specify the levels and add some inter-component edges. The former is simply taken care of by setting $\phi_k(x) = \phi_{k-1}^{(i)}(x) + i$ for each $x \in V(G_{k-1}^{(i)}) \subset V(G_k)$. The levels stay the same in the first $k$-component, whereas all vertices move up exactly 1 level in the second. To complete $G_k$, we add one final edge of weight $k-1$ connected the unique point $s \in V(G_k)$ with $\phi_k(s) = 1$ and the unique point $t \in V(G_k)$ with $\phi_k(t) = k$.

The progression of the first five components is shown in Figure 2.6.

We can now construct our worst-case instance $(X_k, d)$. Let $K_k^{(1)}, \ldots, K_k^{(k+1)}$ be $k+1$ different $k$-components. Take the disjoint union $H_k = \coprod_i G_k^{(i)}$ of the corresponding $k$-graphs and add edges $\{x, y\}$ of weight 1 between all pairs of points $x \in V(G_k^{(i)})$ and $y \in V(G_k^{(j)})$ whose levels are the same, i.e., $\phi_k^{(i)}(x) = \phi_k^{(j)}(y)$. To simplify notation, we will usually omit the superscripts and write $\phi_k(x)$ to denote the level of a point $x \in V(G_k^{(j)})$, irrespective of its component. $(X_k, d)$ is now defined as follows:

1. The underlying set is just $X_k = V(H_k)$.

2. The distance $d(x, x')$ for all pairs of points $x, x' \in X_k$ is equal to the length of a shortest $x$-$x'$-path in $H_k$. In other words, $d$ is the metric closure over the weights of the graph.

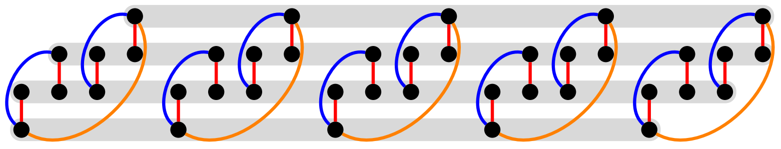


Figure 2.7: The instance $X_k$ for $k = 4$.

A rendition of the final worst-case instance is given in Figure 2.7 in the case of $k = 4$. Note that the edges connecting the individual $k$-components have turned each level into a clique of diameter and radius equal to 1 so that they collectively, in fact, constitute an optimal $k$-clustering for both objective functions. (Obviously, any $k$-clustering must have a diameter or radius of at least 1.)

### The Analysis for $\mathbf{CL}^{\text{diam}}$

Let $(\mathscr{C}_\ell)_{\ell=1}^n$ denote the clustering sequence produced by $\mathbf{CL}^{\text{diam}}$ on $(X_k, d)$. Recall that $\mathscr{C}_{\ell-1}$ arises from $\mathscr{C}_\ell$ by merging that pair of clusters $A, B \in \mathscr{C}_\ell$, whose union has the smallest diameter. (Of course, this does not have to specify the clusters to be merged uniquely — several merges can induce the same cost.) Instead of working with this sequence of clusterings, it will be more helpful to consider just a certain subsequence that is established relative to the clusterings cost. The following definition will be used throughout the analysis.

**Definition 30.** Denote by $t_{\leq x} = \min\{\ell \mid \operatorname{diam}(\mathscr{C}_\ell) \leq x\}$ that point in time, immediately after which all $\mathbf{CL}^{\text{diam}}$-clusterings costs strictly more than $x$. In other words, since the cost is increasing monotonously, it is the last point in time during which a $\mathbf{CL}^{\text{diam}}$-cluster still has cost at most $x$. The corresponding $\mathbf{CL}^{\text{diam}}$-clustering is denoted by $\mathscr{H}_x = \mathscr{C}_{t_{\leq x}}$.

Let us now properly establish what we aim to prove in this section. We wish to show that each cluster contained in $\mathscr{H}_{k-1}$ coincides with the vertex set of one of the $k+1$ different $k$-graphs that make up the instance. Since the last merge will push the cost to $k$, this yields our main result.

**Theorem 31.** *For every $k \in \mathbb{N}$, the $k$-clustering computed by $\mathbf{CL}^{diam}$ on $(X_k, d)$ has a diameter of $k$, even though there exists an optimal $k$-clustering of diameter 1.*

In the next section, we show that the same instance also yields a lower bound of $\Omega(k)$ for $\mathbf{CL}^{\text{rad}}$. For now, however, we focus on the diameter case. As in the two preceding sections, we first establish a lower bound on the distance between any two points.

**Lemma 32.** *The distance between two points $x, x' \in X_k$ is at least as big as the difference in levels. In other words, $d(x, x') \geq |\phi_k(x) - \phi_k(x')|$.*

*Proof.* Due to the specific construction of the components, an edge of weight $w$ can cross at most $w$ levels. Hence the distance between $x$ and $x'$ is at least $|\phi_k(x) - \phi_k(x')|$. $\square$

Consider now any $\ell$-graph $G_\ell$. With the help of the above lemma, we can show that the diameter of the underlying set of this component is upper bounded by $\operatorname{diam}(V(G_\ell)) \leq k - 1$. At the same time, we can inductively also show that the diameter is not strictly less than $k-1$. In other words, the underlying set of every $\ell$-component within $X_k$ has a diameter of exactly $k - 1$, as the following lemma proves.

**Lemma 33.** *For all $\ell \in \{1, \ldots, k\}$ the underlying set of any $\ell$-graph $G_\ell$ contained in $X_k$ has a diameter of* $\operatorname{diam}(V(G_\ell)) = \ell - 1$.

*Proof.* We first prove the upper bound $\operatorname{diam}(V(G_\ell)) \leq \ell - 1$ by induction. The 1-graphs are points, so the claim follows trivially for $\ell = 1$. Assume now that we have shown the claim for $\ell - 1$. Let $s, t \in V(G_\ell)$ be points such that $d(s, t) = \operatorname{diam}(V(G_\ell))$. If these points lie in the same graph, say $G^{(0)}_{\ell-1}$, of the two $(\ell - 1)$-graphs $G^{(0)}_{\ell-1}$ and $G^{(1)}_{\ell-1}$ that make up $G_\ell$, then

$$\operatorname{diam}(V(G_\ell)) = d(s, t) \leq \operatorname{diam}(V(G^{(0)}_{\ell-1})) \leq \ell - 2 < \ell - 1$$

by induction, and we are done. Otherwise we may assume that $s \in V(G^{(0)}_{\ell-1})$ and $t \in V(G^{(1)}_{\ell-1})$. This leaves us with another case analysis. If $s$ is the unique point with level 1 and $t$ is the unique point in level $\ell$ in $G_\ell$ then we are again done, since by construction, there exists an edge between $s$ and $t$ of weight $\ell - 1$. Otherwise, one of $s$ or $t$ must share a level with a point not in the same $(\ell - 1)$-graph as themselves. Without loss of generality we may assume that $s$ lies in the same level as some $u \in V(G^{(1)}_{\ell-1})$. By induction $d(u, t) \leq \ell - 2$ and so

$$\operatorname{diam}(V(G_\ell)) = d(s, t) \leq d(s, u) + d(u, t) \leq 1 + \ell - 2 = \ell - 1.$$

To derive the lower bound $\operatorname{diam}(V(G_\ell)) \geq \ell - 1$, we simply apply Lemma 32 to the unique point $s$ with level 1 and the unique point $t$ with level $\ell$ in $G_\ell$. This shows that $\operatorname{diam}(V(G_\ell)) \geq d(s, t) \geq \ell - 1$ as claimed. $\square$

The goal now is to show that $\mathbf{CL}^{\text{diam}}$ indeed reconstructs these graphs as clusters. Given the above calculation of the cost of an $\ell$-graph, we can show that all these eventual merges are viable for $\mathbf{CL}^{\text{diam}}$.

**Lemma 34.** $\mathbf{CL}^{diam}$ *might merge clusters on* $(X_k, d)$ *in such a way that for all* $\ell \leq k$*, the clustering* $\mathscr{H}_{\ell-1}$ *consists exactly of the* $\ell$*-graphs that make up* $X_k$*. In other words, we want to show that the latter are all viable clusterings.*

*Proof.* We again prove the claim by induction. As always, $\mathbf{CL}^{\text{diam}}$ starts with every point in a separate cluster. Since those are exactly the 1-graphs and any merge costs at least 1, the claim holds for $\ell = 1$. Suppose now that $\mathscr{H}_{\ell-1}$ consists exactly of the $\ell$-graphs that make up the instance. Since we are dealing with integer weights, any subsequent merge increases the cost by at least 1. In other words, it is viable to merge all $\ell$-graphs into their corresponding $(\ell+1)$-graphs. All of them are cheapest merges since they only increase the cost from $\ell - 1$ to $\ell$ (see Lemma 33). To finish the proof, all that's left to do is to show that there are no more free merges left. Take any two $(\ell+1)$-graphs $G_{\ell+1} \neq G'_{\ell+1}$ contained in the current clustering. If they do not exactly cover the same levels, then the distance between the point in the lowest level to the point in the highest level is strictly more than $\ell$ by Lemma 32. Hence, we can assume that they share the same levels, say level $\lambda$ up to level $\ell + \lambda$. Denote by $s$ the unique point in $V(G_{\ell+1})$ with $\phi_k(s) = \lambda$ and by $t$ the unique point in $V(G'_{\ell+1})$ with $\phi_k(t) = \ell + \lambda$. A shortest path connecting $s$ and $t$ must contain an edge $\{u, w\}$ with $u \in V(G_{\ell+1})$ and $w \in X_k \setminus V(G_{\ell+1})$. Such an edge either weights at least $\ell+1$ or weights exactly 1 while connecting points in the same level, i.e., $\phi_k(u) = \phi_k(w)$. In the first case, we immediately obtain $d(s,t) \geq \ell + 1$; in the second case, we use Lemma 32 and obtain

$$\begin{aligned} d(s,t) &= d(s,u) + d(u,w) + d(w,t) \\ &\geq |\phi_k(s) - \phi_k(u)| + 1 + |\phi_k(w) - \phi_k(t)| \\ &= |\phi_k(s) - \phi_k(t)| + 1 \\ &= \ell + 1. \end{aligned}$$

It follows that $\mathscr{H}_\ell$ consists exactly of the $(\ell+1)$-graphs that make up $X_k$. □

*Proof of Theorem 31.* Lemma 34 shows that $\mathscr{H}_{k-1}$ can consist of all the $k$-graphs that make up $X_k$. Since there are exactly $k+1$ of them, one merge remains for us to end up with a $k$-clustering. By definition of $\mathscr{H}_{k-1}$, this last merge then increases the cost at least by 1, so that the $k$-clustering produced by $\mathbf{CL}^{\text{diam}}$ costs at least $k$. On the other hand, the levels collectively make up a $k$-clustering of diameter exactly 1. □

## The Analysis for $\mathbf{CL}^{\text{rad}}$

In this section, we prove that $(X_k, d)$ also already provides a lower bound of $\Omega(k)$ for $\mathbf{CL}^{\text{rad}}$, which, as we will see in the next chapter is, contrary to the $\mathbf{CL}^{\text{diam}}$

analysis, actually asymptotically tight. For now, we want to prove the following theorem.

**Theorem 35.** *For every $k \in \mathbb{N}$, the $k$-clustering computed by $\mathbf{CL}^{rad}$ on $(X_k, d)$ as constructed above has a diameter of $\frac{k}{2}$, even though the latter has an optimal $k$-clustering of radius 1.*

Proving this theorem is more involved than the previous section and requires additional work since we now have to keep track of optimal centers for each cluster. To ascertain the radii of the underlying sets of each component (see Lemma 36 below), we inductively show that for any $2\ell$-graph $G_{2\ell}$ in $X_k$, there always exists a point $z$ with the following properties:

1. For all but one of the $\ell$-graphs that constitute $G_{2\ell}$ there exists a point that is at distance 1 from $z$. Since the diameter of the underlying sets of these graphs is $\ell - 1$ (by induction), it follows that all points within those $\ell$-graphs are within distance $\ell$ from $z$.

2. The remaining $\ell$-graph lies in the same $(\ell + 1)$-graph as $z$. Since the underlying sets of all $\ell$-graphs have a diameter of $\ell - 1$ (see Lemma 33), we thus know that $V(G_{2\ell})$ has to have a radius of at most $\ell$.

3. At the same time, there cannot exist any point that induces a smaller radius. This is again because all weights are integers, and the $V(G_{2\ell})$ has a diameter of $2\ell - 1$ by Lemma 33.

Additionally, to keep track of all optimal centers, we prove that all points outside of $G_{2\ell}$ induce a higher radius for $V(G_{2\ell})$ and are thus not optimal. We need this observation to show that, once all $2\ell$-graphs have been established, all subsequent merges necessarily increase the radius. This follows with Lemma 37.

**Lemma 36.** *Let $G_{2\ell}$ be any one of the $2\ell$-graphs that constitute $X_k$ for any arbitrary $1 \leq \ell \leq \frac{k}{2}$. Then it holds that (i) $\operatorname{rad}(V(G_{2\ell})) = \ell$ and (ii) all optimal centers that induce this cost are themselves contained only within $G_{2\ell}$ (and not in any other $2\ell$-graph).*

*Proof.* We prove the properties listed directly above the lemma in reverse order. As mentioned in the third item, we already know that $V(G_{2\ell})$ has to have a radius of at least $\ell$, since its diameter is exactly $2\ell - 1$ by Lemma 33. To show that the radius of $V(G_{2\ell})$ at the same time does not exceed $\ell$ suppose that $G_{2\ell}$ intersects levels $\lambda$ up to $\lambda + 2\ell - 1$ of $X_k$.

Let $G_{\ell+1}$ be the unique $(\ell + 1)$-graph contained in $G_{2\ell}$ that intersects levels $\lambda + \ell - 1$ up to $\lambda + 2\ell - 1$ and let $z \in G_{\ell+1}$ be that unique point with level $\lambda + \ell - 1$.

This will be our optimal center. From Lemma 33 we know that the diameter of $V(G_{\ell+1})$ is $\ell$, so any point also contained in $G_{\ell+1}$ is at distance at most $\ell$ to $z$.

Consider any of the remaining points $x \in V(G_{2\ell}) \setminus V(G^z_{\ell+1})$ together with the $\ell$-graph $G'_\ell$ containing it. We claim that $G'_\ell$ necessarily intersects level $\lambda + \ell - 1$, i.e., that it contains a point $y$ with $\phi_k(y) = \lambda + \ell - 1$. If this were not the case, then $G'_\ell$ would have to intersect all levels starting from $\lambda + \ell$ all the way up to $\lambda + 2\ell - 1$ and therefore contain the unique point in $G_{2\ell}$ with level $\lambda + 2\ell - 1$. But this is not possible, since that point already belongs to in $G_{\ell+1}$. So combining the fact that the diameter of $V(G'_\ell)$ is $\ell - 1$ and $\phi_k(z) = \phi_k(y)$ we obtain

$$d(z, x) \leq d(z, y) + d(y, x) \leq 1 + (\ell - 1) = \ell,$$

which proves that $\mathrm{rad}(V(G_{2\ell})) \leq \ell$, as claimed.

Let us move on to the second part of the lemma and show that all optimal centers for $V(G_{2\ell})$ are already part of it. Quickly rephrasing this statement, this amounts to showing that $\max_{x' \in V(G_{2\ell})} d(z', x') \geq \ell + 1$ holds for all possible centers $z' \in X_k \setminus V(G_{2\ell})$. The reasoning is quite simple: we can enter $V(G_{2\ell})$ from the outside only via one of the levels, in which case we cannot, by the same token, cover any vertical ground. (Actually, there are other edges entering the component, but their cost is much too high.) To properly prove this claim, we make a case distinction over $\phi_k(z')$.

Suppose that $\phi_k(z') \leq \lambda + \ell - 1$. Then we claim that $d(z', x) \geq \ell + 1$ for the unique point $x \in V(G_{2\ell})$ with level $\lambda + 2\ell - 1$. Indeed, consider the first edge $\{u, w\}$ lying on some shortest $z'$-$x$-path with $u \in X_k \setminus V(G_{2\ell})$ and $w \in V(G_{2\ell})$. By construction $\{u, w\}$ either weights at least $2\ell$ (those are the only edges leaving a $2\ell$-graph) in which case

$$d(z', x) \geq 2\ell \geq \ell + 1$$

or it weights 1, in which case $\phi_k(u) = \phi_k(w)$, so that

$$\begin{aligned} d(z', x) &= d(z', u) + d(u, v) + d(v, x) \\ &\geq |\phi_k(z') - \phi_k(u)| + 1 + |\phi_k(w) - \phi_k(x)| \\ &= |\phi_k(z') - \phi_k(x)| + 1 \\ &\geq \ell + 1. \end{aligned}$$

If instead $\phi_k(z') \geq \lambda + \ell$ holds we argue in exactly that $d(z', y) \geq \ell + 1$ for the unique point $y$ in $G_{2\ell}$ with level $\lambda$. □

In order to now demonstrate that $\mathbf{CL}^{\mathrm{rad}}$ reconstructs these components, or rather, that all components are viable clusters, we first have to show, as hinted at above, that merging parallel $2\ell$-graphs increases the cost of our solution by a positive amount. Here, we make use of the fact that sets of optimal centers for any pair of $2\ell$-graphs cannot intersect.

**Lemma 37.** *Let $C, D$ be two subsets of $X_k$ with* $\operatorname{rad}(C) = \operatorname{rad}(D)$*. Let $Z(C)$ and $Z(D)$ denote the set of all optimal centers for $C$ and $D$, respectively. If $Z(C) \cap Z(D) = \varnothing$, then* $\operatorname{rad}(C \cup D) > \operatorname{rad}(C)$*.*

*Proof.* Consider an arbitrary point $x \in X_k$. Since $Z(C) \cap Z(D) = \varnothing$, this point cannot be optimal for both sets simultaneously. Assume without loss of generality that $x \notin Z(D)$. Then

$$\max_{y \in C \cup D} d(y, x) \geq \max_{y \in D} d(y, x) > \operatorname{rad}(D) = \operatorname{rad}(C)$$

Since $x$ was chosen arbitrarily, this proves the claim. $\square$

Given all these auxiliary results, we can finally analyze the merge behavior of $\mathbf{CL}^{\text{rad}}$ and prove that it reconstructs our components in the worst case. Theorem 35 is then an immediate consequence of Corollary 38.

**Corollary 38.** $\mathbf{CL}^{rad}$ *might merge clusters for $(X_k, d)$ in such a way, that for all $1 \leq \ell \leq \frac{k}{2}$, the clustering $\mathscr{H}_\ell$ consists exactly of the $2\ell$-graphs that make up $X_k$.*

*Proof.* The proof is an induction analogous to Lemma 34. First, consider the base case $\ell = 1$. Since all weights are integers, the first merge increases the cost to 1, which already equals the cost of any 2-graph (by Lemma 36). In other words, all of these clusters are viable. Combining Lemma 36 and Lemma 37 shows that merging any resulting 2-graphs increases the cost by some additional amount. Hence, $\mathscr{H}_1$ consists precisely of the 2-graphs.

Assume now that the claim holds for $\mathscr{H}_\ell$. The induction step works essentially the same as the base case. Any merge will increase the cost of the solution by at least 1 by definition of $\mathscr{H}_\ell$ and so we might as well merge all $2\ell$-graphs that together compose a $(2\ell + 2)$-graph as this is a cheapest choice (Lemma 36). Furthermore, any additional merge would increase the cost to at least $\ell + 2$ (again by Lemma 37) and so $\mathscr{H}_{\ell+1}$ consist of the $(2\ell + 2)$-graphs. $\square$

This concludes our analysis of $\mathbf{CL}^{\text{rad}}$. In the next section, which will serve mainly as an addendum, we tackle one last, maybe somewhat unsatisfying detail pertaining to the lower bound presented here and in the previous section. For either Complete-Linkage variant, we have only shown that worst-case merges that lead to an $\Omega(k)$-approximate solution are viable. In the section on Dasgupta and Long's lower bound, we have argued that this is not an important issue because no general procedure (applicable to Complete-Linkage) could distinguish good from problematic merges. However, to absolutely rule out any good result, we will show in the next section that, with appropriate technical adjustments, we can force worst-case merges for both Complete-Linkage variants. In other words, Complete-Linkage cannot compute a good $k$-clustering, even "by chance". This will form the final section of this chapter.

### 2.3.4 Addendum: Removing Tiebreaks

In this section, we modify the instance $(X_k, d)$ from the previous sections in such a way that the only viable merges are those that combine two $\ell$-graphs $G_\ell$, $G'_\ell$ which are part of the same $(\ell + 1)$-graph. All other merges will be slightly more expensive, so Complete-Linkage will necessarily compute a bad approximation. This strengthens the previous results and lays some remaining reservations to rest, even if at the cost of a more technical analysis. Since the original instance was already quite complex, the underlying modification is not as straightforward as the one that yields Dasgupta and Long's actual instance. Although these analyses overlap significantly with the previous two sections in their overall structure, we will thoroughly work through them to ensure that no problems crop up — something that could quickly happen with instances of this complexity.

#### Adjusting the Instance for $\mathbf{CL}^{\text{diam}}$

Let us first focus on the modified construction of the $k$-components for the diameter variant. We introduce a new factor $\varepsilon \in (0, \frac{1}{2})$, arbitrarily small if one wishes, by which we will skew some of the inter-level distances.

Now, concerning the components, the definition of $K_1$ stays the same, and as before a $k$-component is constructed from two copies $K_{k-1}^{(0)}, K_{k-1}^{(1)}$ of the $(k-1)$-component by taking the disjoint union of the corresponding graphs and increasing the level of each point in $K_{k-1}^{(1)}$ by one. The new construction differs from the old only through its inter-level edges and their weights. Contrary to before, we do not just add an edge of weight $k-1$ between the unique point $s \in V(G_{k-1}^{(0)})$ with level 1 and $t \in V(G_{k-1}^{(1)})$ with level $k$. Instead, we complete $G_k$ by adding edges of weight $(k-1)(1-\varepsilon)$ between all pairs $x \in V(G_{k-1}^{(0)})$ and $y \in V(G_{k-1}^{(1)})$, whenever they do not lie on the same level, i.e., whenever $\phi_k(x) \neq \phi_k(y)$. Note that most of these edges do not actually lie on any shortest path and are, in fact, superfluous. We only added them to simplify the description of the construction. Given these components, the subsequent arrangement of $X_k$ is almost identical. Only the number of $k$-graphs changes. This time, we take the disjoint union of only $k$ copies $G_k^{(1)}, \dots, G_k^{(k)}$ of $k$-graphs and connect them by adding edges $\{x, y\}$ of weight 1 for every two points $x \in V(G_k^{(i)})$ and $y \in V(G_k^{(j)})$ with $\phi_k^{(i)}(x) = \phi_k^{(j)}(y)$. We could have also reduced the number of copies in the original instance. The only difference would be that the lower bound of $k$ is reduced to a lower bound of $k-1$ since the last merge does not occur.

Overall, our goal is to show that the clustering computed by $\mathbf{CL}^{\text{diam}}$ on $(X_k, d)$ at time $t_{\leq \ell(1-\varepsilon)}$ consists precisely of the $(\ell+1)$-graphs that make up the instance. We start by establishing a new lower bound on the distance between any pair of points.

**Lemma 39.** *The distance between two points $x, y \in X_k$ is at least $|\phi_k(x) - \phi_k(y)|(1-\varepsilon)$.*

*Proof.* From the construction of the components, it is clear that an edge that crosses $w$ levels costs at least $w(1-\varepsilon)$. Hence the distance between any pair of points $x$ and $y$ is at least $|\phi_k(x) - \phi_k(y)|(1-\varepsilon)$. □

As before, we use this lemma to show that the diameter of any $\ell$-graph in $X_k$ is exactly equal to $(\ell-1)(1-\varepsilon)$.

**Lemma 40.** *Let $G_\ell$ be any $\ell$-graph contained in $X_k$. Then* $\operatorname{diam}(V(G_\ell)) = (\ell - 1)(1-\varepsilon)$.

*Proof.* We prove the upper bound, that $\operatorname{diam}(V(G_\ell)) \leq (\ell-1)(1-\varepsilon)$, by induction. The 1-graphs are points, so the claim trivially follows for $\ell = 1$. Assume now that we have proven the claim for all $\ell - 1$-graphs. Take any $\ell$-graph $G_\ell$ and consider an arbitrary pair $s, t \in V(G_\ell)$ contained within it. If these points lie in the same $(\ell-1)$-graph, say $G_{\ell-1}$, then

$$d(s,t) \leq \operatorname{diam}(V(G_{\ell-1})) \leq (\ell-2)(1-\varepsilon) < (\ell-1)(1-\varepsilon)$$

by induction, and we are done. Otherwise, if $s$ and $t$ belong to different $(\ell-1)$-graphs, we make the following case distinction:

**Case 1:** If $\phi_k(s) = \phi_k(t)$, these points are connected by an edge of weight one by construction. Note that if $\ell \leq 2$, no $\ell$-graph contains points from the same level. Using $\varepsilon \leq \frac{1}{2}$ and $\ell \geq 3$ we obtain

$$d(s,t) = 1 \leq (\ell-1)(1-\varepsilon).$$

**Case 2:** If $s$ and $t$ are on different levels there is an edge of weight $(\ell-1)(1-\varepsilon)$ between $s$ and $t$ by construction.

Since $s$ and $t$ were chosen arbitrarily, we indeed obtain that

$$\operatorname{diam}(V(G_\ell)) = d(s,t) \leq (\ell-1)(1-\varepsilon).$$

To see that the lower bound $\operatorname{diam}(V(G_\ell)) \geq (\ell-1)(1-\varepsilon)$ holds, we apply Lemma 39 to the unique point $s \in V(G_\ell)$ with $\phi_\ell(s) = 1$ and the unique point $t \in V(G_\ell)$ with $\phi_\ell(t) = \ell$. Altogether this yields $\operatorname{diam}(V(G_\ell)) = (\ell-1)(1-\varepsilon)$ as claimed. □

Finally, we show that $\mathbf{CL}^{\text{diam}}$ must, during its execution, in fact reconstruct each component.

**Lemma 41.** $\mathbf{CL}^{diam}$ *must merge clusters on* $(X_k, d)$ *in such a way that for all* $\ell < k$*, the clustering* $\mathscr{H}_{\ell(1-\varepsilon)}$ *consists exactly of the* $(\ell+1)$*-graphs that make up* $X_k$*.*

*Proof.* As usual, we prove the claim by induction. Since the singleton clusters, with which $\mathbf{CL}^{\text{diam}}$ starts, are exactly the 1-graphs, and since any merge of two points costs at least $(1-\varepsilon)$, the claim follows for $\ell = 1$. Suppose now that $\mathscr{H}_{(\ell-1)(1-\varepsilon)}$ consists exactly of the $\ell$-graphs of the instance. Consider two $\ell$-graphs $G_\ell \neq G'_\ell$ contained in the current clustering. We make the following case distinction to compute the cost of merging them.

**Case 1:** If they are contained in the same $(\ell+1)$-graph $G_{\ell+1}$, then merging $G_\ell$ with $G'_\ell$ results in $G_{\ell+1}$ which itself costs $\operatorname{diam}(V(G_{\ell+1})) = \ell(1-\varepsilon)$ (Lemma 40).

**Case 2:** If they are not contained in the same $(\ell+1)$-graph, we show that merging $G_\ell$ with $G'_\ell$ costs strictly more than $\ell(1-\varepsilon)$. Note that the following holds.

1. The edges connecting $x \in V(G_\ell)$ and $y \in V(G'_\ell)$ with $\phi_k(x) \neq \phi_k(y)$ are of weight $\geq (\ell+1)(1-\varepsilon)$.
2. There exist $s \in V(G_\ell)$ and $t \in V(G'_\ell)$ with $|\phi_k(s) - \phi_k(t)| \geq \ell - 1$.

The last observation is because both graphs contain two points whose difference in level is exactly $\ell - 1$. We now prove that $d(s,t) > \ell(1-\varepsilon)$, which shows that merging $G_\ell$ with $G'_\ell$ in fact costs more than $\ell(1-\varepsilon)$.

Any shortest path connecting $s$ and $t$ in $X_k$ must contain an edge $\{u, w\}$ between a point $u \in V(G_\ell)$ and a point $w \in V(G'_\ell)$. By the above observation, this edge is either of weight $\geq (\ell+1)(1-\varepsilon)$ or $u$ and $w$ are on the same level, and the edge is of weight 1. In the first case, we conclude that

$$d(s,t) \geq (\ell+1)(1-\varepsilon) > \ell(1-\varepsilon).$$

And similarly, in the second case, that

$$\begin{aligned} d(s,t) &= d(s,u) + 1 + d(w,t) \\ &\geq |\phi_k(s) - \phi_k(u)|(1-\varepsilon) + 1 + |\phi_k(w) - \phi_k(t)|(1-\varepsilon) \\ &= |\phi_k(s) - \phi_k(t)|(1-\varepsilon) + 1 \\ &\geq (\ell-1)(1-\varepsilon) + 1 \\ &> \ell(1-\varepsilon). \end{aligned}$$

Ultimately, this proves that $\mathscr{H}_{\ell(1-\varepsilon)}$ consists exactly of all $(\ell+1)$-graphs of $X_k$. $\square$

In particular, we get that $\mathscr{H}_{(k-1)(1-\varepsilon)}$ consists exactly of all $k$-graphs that make up $X_k$. There are exactly $k+1$ of them, thus the $k$-clustering produced by $\mathbf{CL}^{\mathrm{diam}}$ costs $(k-1)(1-\varepsilon)$.

**Corollary 42.** *However the tie-breaks are resolved,* $\mathbf{CL}^{diam}$ *computes a $k$-clustering on $(X_k, d)$ with diameter $(k-1)(1-\varepsilon)$ while the optimal $k$-clustering has diameter* 1.

## Adjusting the Instance for $\mathbf{CL}^{\mathrm{rad}}$

Lastly, we explain how to adjust the construction of the $k$-components for $\mathbf{CL}^{\mathrm{rad}}$. Again, we introduce an auxiliary magnitude $\varepsilon \in (0, \frac{1}{2})$, which we will use to alter some of the inter-level distances. The definition of $K_1$ again does not change and as before a $k$-component is assembled from two copies $K_{k-1}^{(0)}, K_{k-1}^{(1)}$ of the $(k-1)$-component by taking the disjoint union of the corresponding graphs $V(G_{k-1}^{(0)})$, $V(G_{k-1}^{(1)})$ and increasing the level of each point in $K_{k-1}^{(1)}$ by one. We then complete $G_k$ by adding edges between $x \in V(G_{k-1}^{(0)})$ and $y \in V(G_{k-1}^{(1)})$ if $\phi_k(x) \neq \phi_k(y)$ and we assign this edge a weight of $\lceil\frac{k}{2}\rceil(1-\varepsilon)$ if $|\phi_k(x)-\phi_k(y)| \leq \lceil\frac{k}{2}\rceil - 1$ and otherwise a weight of $|\phi_k(x) - \phi_k(y)|(1-\varepsilon)$. The final set $X_k$ is defined as usual.

Note that Lemma 39 still holds and that the diameter of an $\ell$-graph is still upper bounded by $(\ell-1)(1-\varepsilon)$.

**Lemma 43.** *Let $G_{2\ell}$ be any of the $2\ell$-graphs that constitute $X_k$ for $1 \leq \ell \leq \frac{k}{2}$. It holds that* $\mathrm{rad}(G_{2\ell}) = \ell(1-\varepsilon)$. *Furthermore, if $G'_{2\ell}$ denotes a second $2\ell$-graph which is not contained in the same $2(\ell+1)$-graph as $G_{2\ell}$, then any cluster containing both $G_{2\ell}$ and $G'_{2\ell}$ costs at least $\ell(1-\varepsilon)+1$.*

*Proof.* Proving that $\mathrm{rad}\, G_{2\ell} \geq \ell(1-\varepsilon)$ again is the easier of the two bounds. We know that $G_{2\ell}$ contains points $s$ and $t$ with $|\phi_k(s) - \phi_k(t)| = 2\ell - 1$. It follows that $\max\{|\phi_k(s)-\phi_k(x)|, |\phi_k(t)-\phi_k(x)|\} \geq \ell$ for all $x \in X_k$. But then, Lemma 39 proves that $\max\{d(s,x), d(t,x)\} \geq \ell(1-\varepsilon)$, so that indeed $\mathrm{rad}(G_{2\ell}) \geq \ell(1-\varepsilon)$.

To prove the upper bound suppose that $G_{2\ell}$ covers levels $\lambda$ up to $\lambda + 2\ell - 1$ of $X_k$. Consider the unique $(\ell+1)$-graph $H_{\ell+1}$ contained in $G_{2\ell}$ covering levels $\lambda + \ell - 1$ to $\lambda + 2\ell - 1$. Let $c$ be the unique point in $H_{\ell+1}$ with level $\lambda + \ell - 1$. Remember that the diameter of $H_{\ell+1}$ is at most $\ell(1-\varepsilon)$, so any point in $H_{\ell+1}$ is at distance $\leq \ell(1-\varepsilon)$ to $c$. Consider now a point $x \in V(G_{2\ell}) \setminus V(H_{\ell+1})$. We know that $\phi_k(x) < \lambda + 2\ell - 1$. Thus $|\phi_k(x) - \phi_k(c)| \leq \ell - 1$. By construction, there exists an edge of weight at most $\ell(1-\varepsilon)$ between $x$ and $c$ and thus $d(x,c) \leq \ell(1-\varepsilon)$.

It is left to show that any cluster containing $G_{2\ell}$ and $G'_{2\ell}$ costs at least $\ell(1-\varepsilon)+1$. Let $y \in X_k$ and let $H_{2(\ell+1)}$ be the $2(\ell+1)$-graph containing $y$. Assume without loss of generality that $G_{2\ell}$ is not contained in $H_{2(\ell+1)}$. Let $x \in V(G_{2\ell})$ be a point with $|\phi_k(x) - \phi_k(y)| \geq \ell$. We claim that $d(x,y) \geq (\ell-1)(1-\varepsilon)+1$. A shortest path connecting $x$ and $y$ must contain an edge $\{u, w\}$ with $u \in X_k \setminus V(H_{2(\ell+1)})$ and $w \in V(H_{2(\ell+1)})$. We know by construction that either $\phi_k(u) = \phi_k(w)$, or the edge weights at least $(\ell + 2)(1 - \varepsilon)$. In the first case we use Lemma 39 and obtain

$$\begin{aligned} d(x, y) &= d(x, u) + d(u, w) + d(w, y) \\ &\geq |\phi_k(x) - \phi_k(u)|(1 - \varepsilon) + 1 + |\phi_k(w) - \phi_k(y)|(1 - \varepsilon) \\ &= |\phi_k(x) - \phi_k(y)|(1 - \varepsilon) + 1 \\ &\geq \ell(1 - \varepsilon) + 1 \end{aligned}$$

and in the second case, we obtain

$$d(x, y) \geq (\ell + 2)(1 - \varepsilon) \geq \ell(1 - \varepsilon) + 1.$$

□

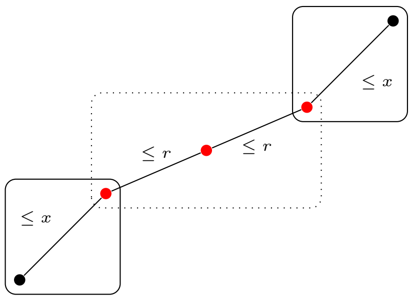
This immediately leads to the following results.

**Corollary 44.** $\mathbf{CL}^{rad}$ *must merge clusters on* $(X_k, d)$ *in such a way that for all* $1 \leq \ell \leq \frac{k}{2}$*, the clustering* $\mathscr{H}_{\ell(1-\varepsilon)}$ *consists exactly of the* $2\ell$*-graphs that make up* $X_k$*.*

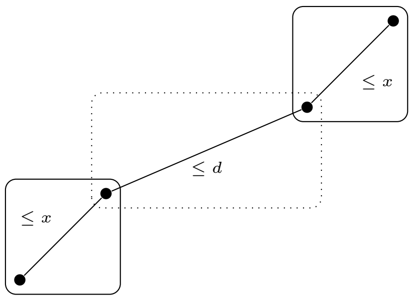
**Corollary 45.** *However the tie-breaks are resolved,* $\mathbf{CL}^{rad}$ *computes a* $k$*-clustering on* $(X_k, d)$ *with radius* $\frac{k}{2}(1 - \varepsilon)$*, while the optimal* $k$*-clustering has radius* 1.

## 2.4 Upper Bounds for Complete-Linkage

We have just seen that $\mathbf{CL}^{\text{diam}}$ and $\mathbf{CL}^{\text{rad}}$ are at most $\Omega(k)$-competitive relative to their respective objective functions. In the worst case, Complete-Linkage is thus not better than Single-Linkage, even though the latter optimizes an entirely different objective function. Of course, this statement is a bit deceptive or misleading, and we could further flesh out differences in their behavior. Clearly, Single-Linkage's chaining characteristic can yield $\Omega(k)$-approximative solutions much more quickly. In Figure 2.1, we have seen that such solutions can arise with only $k - 1$ merging steps. Thus, whereas our worst-case instance for Complete-Linkage consists of $\Omega(2^k)$ points, we only require $2k$ points to construct a worst-case instance for Single-Linkage. For now, we leave these considerations to the side because we first of all still have to establish that the lower bounds derived in the previous section are, in fact, worst cases. We still have to establish that Complete-Linkage is an $O(k)$-approximation for the $k$-center and $k$-diameter objective functions. We attempt to show this in this section, although we will only partially succeed for $\mathbf{CL}^{\text{diam}}$. Our results will be the first non-trivial upper bounds for Complete-



(a) As long as $|\mathscr{H}_x| > k$ there have to exist two clusters $C, C' \in \mathscr{H}_x$ whose centers lie in the same optimal cluster $O$. Merging them yields a new cluster of radius at most $2r + x$, where $r = \mathrm{rad}(O)$ is the radius of the optimal cluster.



(b) As long as $|\mathscr{H}_x| > k$ there have to exist two clusters $C, C' \in \mathscr{H}_x$ that intersect a same optimal cluster $O$. Merging them yields a new cluster of diameter at most $2x + d$, where $d = \mathrm{diam}(O)$ is the diameter of the optimal cluster.

Figure 2.8: The cost increase can be upper bounded much better for $\mathbf{CL}^{\mathrm{rad}}$ than for $\mathbf{CL}^{\mathrm{diam}}$. In the former case (a), the radius will increase by an additive amount of at most twice the optimum cost, which is fixed and doesn't change during the execution of $\mathbf{CL}^{\mathrm{rad}}$. In the latter case (b), however, the diameter might increase to such a degree that it more than doubles the previous diameter. As such, we cannot derive any logarithmic bound as in Proposition 49.

Linkage on general metrics spaces — that is, metric spaces without any additional geometric structure, such as is attached, for example, to Euclidean spaces. In the case of $\mathbf{CL}^{\mathrm{rad}}$, the upper bound turns out to be $O(k)$, matching the lower bound at least asymptotically. However, in the case of $\mathbf{CL}^{\mathrm{diam}}$, we only manage to prove an upper bound of $O(k^2)$. Although this bound can be improved to $O(k^{\ln 3/\ln 2})$ by tightening an important estimation, the gap between this ratio and the lower bound remains too large. One of the difficulties of analyzing $\mathbf{CL}^{\mathrm{diam}}$ relative to the $k$-diameter objective function, as opposed to $\mathbf{CL}^{\mathrm{rad}}$, is that the cost of clusters can increase significantly with a single merge. We will provide further insight later on, but the main point is already captured in Figure 2.8. For now, we will focus on $\mathbf{CL}^{\mathrm{rad}}$.

### 2.4.1 Upper Bounds for $\mathbf{CL}^{\mathbf{rad}}$

In this section, we want to show that the radius of the $k$-clustering $\mathscr{C}_k$ produced by $\mathbf{CL}^{\mathrm{rad}}$ for any instance $(X, d)$ and any $k$ is always an $O(k)$-approximation with regards to the $k$-center objective function.

**Theorem 46.** *Let $(\mathscr{C}_k)_{k=1}^n$ be the hierarchical clustering computed by Complete-Linkage on $(X, d)$ optimizing the radius. For all $1 \leq k \leq n$ the radius $\mathrm{rad}(\mathscr{C}_k)$ is* *upper bounded by $O(k)\,\mathrm{rad}(\mathscr{O}_k)$, where $\mathscr{O}_k$ is an optimal $k$-center clustering.*

Together with the lower bound derived in the previous section, this shows that our analysis is asymptotically tight relative to the $k$-center objective function.

## Part 0: Preliminaries and an Outline of the Proof

Throughout this section, every result will pertain to a fixed but arbitrary general metric space $(X, d)$ and a similarly fixed but arbitrary clustering number $k$. Since the behavior of Complete-Linkage is invariant with regard to scaling, [4] we can assume that the optimal $k$-clustering, from now on denoted $\mathscr{O} = \mathscr{O}_k$, has a cost of exactly $\mathrm{rad}(\mathscr{O}) = \frac{1}{2}$.

The proof of Theorem 46 is split into two parts. In the first, we crudely upper bound the increase in cost during the execution of Complete-Linkage in a similar vein to Ackermann et al. [3], who use the same bound to estimate the cost of later merge steps. Proposition 49 shows that the difference in cost between $\mathscr{C}_k$ and $\mathscr{C}_t$ for $t > k$ is at most $\lceil\log(t-k)\rceil + 1$. That is, $\mathrm{rad}(\mathscr{C}_k) \leq \lceil\log(t-k)\rceil + 1 + \mathrm{rad}(\mathscr{C}_t)$ holds for all $1 \leq k < t \leq n$, establishing a logarithmic increase in cost relative to the difference in time steps. A clustering $\mathscr{C}_t$ whose cost we can estimate directly (i.e., without referring to any other clustering) can then serve as a starting point to derive a proper upper bound for $\mathrm{rad}(\mathscr{C}_k)$. Ideally, this clustering should consist of relatively few clusters (so that $\lceil\log(t-k)\rceil$ is small) while at the same time not being too expensive. However, these criteria clearly oppose each other. Naively choosing the initial clustering $\mathscr{C}_t = \mathscr{C}_n$ is not good enough. Although its cost is minimal, the number of clusters is too high, only yielding an upper bound of $\mathrm{rad}(\mathscr{C}_k) \leq \lceil\log(n-k)\rceil + 1$. In the second part of the proof, we thus set out to find a different clustering to start from — one that nicely balances a cost-to-size ratio.

## Part 1: Estimating the relative difference in cost

When dealing with radii, any merge performed by Complete-Linkage prior to reaching $k$ clusters increases the cost by at most $2\,\mathrm{rad}(\mathscr{O}) = 1$ (Figure 2.8). This is because the centers of two clusters are necessarily contained in the same optimal cluster.

We show that Complete-Linkage clusterings at times $t_{\leq x}$ and $t_{\leq x+1}$ can have at most $k$ clusters in common. All other clusters from $\mathscr{H}_x$ are merged in $\mathscr{H}_{x+1}$.

**Lemma 47.** *For all $x \geq 0$, the clustering $\mathscr{H}_{x+1}$ contains at most $k$ clusters of cost at most $x$. In particular, it holds that $|\mathscr{H}_{x+1} \cap \mathscr{H}_x| \leq k$.*

[4] This is easy to see. Scaling preserves the ordering of merges according to their cost.

*Proof.* Assume on the contrary that there exist $k+1$ pairwise different Complete-Linkage clusters $D_1, \ldots, D_{k+1}$ at time $t_{\leq x+1}$ of cost at most $x$. Denote by $d_i \in D_i$ a point that induces the smallest radius, i.e. $\operatorname{rad}(D_i) = \max_{d \in D_i} d(d, d_i)$ for all $i$. Then, two of these points, say $d_1$ and $d_2$, must be contained in the same optimal cluster $O \in \mathscr{O}$. Hence, we know that

$$\operatorname{rad}(D_1 \cup D_2) \leq 1 + \max_{i \in \{1,2\}} \operatorname{rad}(D_i) \leq 1 + x$$

because $d(d_1, d_2) \leq 2\operatorname{rad}(O) \leq 2\operatorname{rad}(\mathscr{O}) = 1$ and $\operatorname{rad}(D_i) \leq x$ for $i = 1, 2$. This contradicts the definition of $\mathscr{H}_{x+1}$, as $D_1$ and $D_2$ can still be merged without pushing the cost beyond $x+1$. $\square$

In other words, most clusters will get merged between time steps $t_{\leq x}$ and $t_{\leq x+1}$, at least halving their number. By extending the argument over several time steps, we can establish the following relation between the size of a clustering and the relative time at which it appears.

**Corollary 48.** *For all $i \in \mathbb{N}_+$ and $x \geq 0$ it holds that $|\mathscr{H}_{x+i}| \leq k + \frac{1}{2^i}(|\mathscr{H}_x| - k)$.*

*Proof.* First, we consider what happens when we increase the cost by 1. We fix an arbitrary $x' \geq 0$. Lemma 47 shows that at most $k$ clusters from $\mathscr{H}_{x'}$ are left untouched, while the remaining $|\mathscr{H}_{x'}| - k$ clusters have to be merged with at least one other cluster (thus at least halving the number of those clusters) to get to $\mathscr{H}_{x'+1}$. This yields a bound of

$$|\mathscr{H}_{x'+1}| \leq k + \frac{1}{2}(|\mathscr{H}_{x'}| - k).$$

The case for general $i \in \mathbb{N}$ follows from a straightforward induction. We have just shown that the claim is true for $i = 1$, where we set $x' = x$. For the induction step, suppose that

$$|\mathscr{H}_{x+i-1}| \leq k + \frac{1}{2^{i-1}}(|\mathscr{H}_x| - k).$$

Substituting this into the inequality

$$|\mathscr{H}_{x+i}| \leq k + \frac{1}{2}(|\mathscr{H}_{x+i-1}| - k),$$

derived from the first part of our proof with $x' = x + i - 1$, yields

$$|\mathscr{H}_{x+i}| \leq k + \frac{k + \frac{1}{2^{i-1}}(|\mathscr{H}_x| - k) - k}{2} = k + \frac{1}{2^i}(|\mathscr{H}_x| - k)$$

as claimed. $\square$

Since the particular time steps appearing above already refer to the cost of the corresponding clusterings ($\mathscr{H}_x$ refers to the clustering at time $t_{\leq x}$), we can thereby estimate the increase in cost from one time step to another. This increase turns out to be logarithmic in the distance between the respective time steps.

**Proposition 49.** *For all $k < t \leq n$ it holds that* $\operatorname{rad}(\mathscr{C}_k) \leq \lceil \log(t-k) \rceil + 1 + \operatorname{rad}(\mathscr{C}_t)$.

*Proof.* Let $x = \operatorname{rad}(\mathscr{C}_t)$, so that $\mathscr{H}_x$ consists of at most $t$ clusters. Applying Corollary 48 with $i = \lceil \log(t-k) \rceil + 1$ then shows that

$$|\mathscr{H}_{x+i}| < k + \frac{1}{t-k}(|\mathscr{H}_x| - k) \leq k + 1.$$

That is, $\mathscr{H}_{x+i}$ emerges from $\mathscr{C}_k$ by merging some (or none) of its clusters, and we can conclude that $\operatorname{rad}(\mathscr{C}_k) \leq \operatorname{rad}(\mathscr{H}_{x+i}) \leq x + i = \operatorname{rad}(\mathscr{C}_t) + \lceil \log(t-k) \rceil + 1$. $\square$

This particular part of the analysis is already present in Ackermann et al. [3]. We have mostly adjusted the proof to fit our overall notation.

### Part 2: A cheap clustering with few clusters

Suppose we know of the existence of a Complete-Linkage clustering $\mathscr{C}_t$ with $t \in O(2^k)$ clusters and $\operatorname{rad}(\mathscr{C}_t) \in O(k)$. Applying Proposition 49 the $\mathscr{C}_t$ then yields

$$\operatorname{rad}(\mathscr{C}_k) \in \log(O(2^k)) + 1 + O(k) = O(k) = O(k) \operatorname{rad}(\mathscr{O})$$

and we have proven Theorem 46 (recall that $\operatorname{rad}(\mathscr{O}) = \frac{1}{2}$). We show that $\mathscr{C}_t = \mathscr{H}_{4k+2}$ is a sufficiently good choice. Here, we diverge entirely from Ackermann et al. Whereas they were able to make use of certain properties of Euclidean space to estimate the cost of an initial $2k$-clustering, we have to approach the problem by other means.

To estimate the size of $\mathscr{H}_{4k+2}$, we distinguish between regular and irregular clusters. (Remember that $\mathscr{O}(C) = \{O \in \mathscr{O} \mid O \cap C \neq \varnothing\}$ is the set of optimal clusters intersected by $C$.) Contrary to Single-Linkage, Complete-Linkage sometimes merges clusters that are quite far apart. That is, contrary to Single-Linkage, Complete-Linkage can produce clusters that are very expensive relative to the number of optimal clusters intersected by them. Luckily for us, Complete-Linkage cannot construct many such *irregular* clusters within a short time frame. Since the circumstances under which they can come about are very restrictive, we can establish a relatively small lower bound for the number of irregular clusters at any given time step. On the other hand, the number of the remaining *regular* clusters can be potentially large. However, since they are regular, we can easily upper-bound them once the cost of our clustering reaches $4k + 1$.

**Definition 50.** We call a cluster $C$ *regular*, if $\operatorname{rad}(\mathscr{O}(C)) \leq 4|\mathscr{O}(C)|$ and *irregular* otherwise.

Lemma 60 establishes an upper bound for irregular and Lemma 62 for regular clusters. The former is proved by induction: irregular clusters can descend from cheap or other irregular clusters. On the one hand, since we know from Lemma 47 that at most $k$ cheap clusters can exist at any given time, we get that the number of irregular clusters at time step $t_{\leq 4k+1}$ is at most $4k^2$. On the other hand, the number of regular clusters can be upper-bounded quite easily just by showing that within the same time step, no optimal cluster can contain centers of two different regular clusters.

**Lemma 51.** *$\mathscr{H}_{4k+1}$ contains at most $4k^2$ irregular clusters.*

*Proof.* We show that the number of irregular clusters created between $t_{\leq x-1}$ and $t_{\leq x}$ is at most $k$ for all $x \in \mathbb{N}$. In other words, if $m_x$ denotes the number of irregular clusters in $\mathscr{H}_x$, then $m_x \leq m_{x-1} + k$ holds for all $x \in \mathbb{N}$. Since all clusters are regular at time step $t = 1$, we can inductively infer that $m_{4k+1} \leq 4k^2$, as claimed.

Consider an irregular cluster $D \in \mathscr{H}_x$ for a fixed but arbitrary $x \in \mathbb{N}$. Let $D_1, \ldots, D_\ell \in \mathscr{H}_{x-1}$ be an enumeration of all ancestors of $D$ at time step $t_{\leq x-1}$. We show that none of them can cost at least $x - 2$ and be regular simultaneously. If that is the case, each irregular cluster in $\mathscr{H}_x$ has to descend from

$$\{C \in \mathscr{H}_{x-1} \,|\, C \text{ is irregular}\} \cup \{C \in \mathscr{H}_{x-1} \mid \operatorname{rad}(C) < x - 2\}. \tag{2.1}$$

The set on the left has cardinality $m_{x-1}$, and the set on the right has a cardinality of at most $k$ (by Lemma 47), i.e., $m_x \leq m_{x-1} + k$.

What is left to show is that the ancestors of $D$ are indeed contained in (2.1). Suppose that this is not the case, i.e., that $4|\mathscr{O}(D_i)| \geq \operatorname{rad}(\mathscr{O}(D_i)) \geq \operatorname{rad}(D_i) \geq x - 2$ for some $i$. Since regularity only concerns the distribution of intersected optimal clusters, we know that $\mathscr{O}(D)$ has to be a proper superset of $\mathscr{O}(D_i)$, i.e., $|\mathscr{O}(D_i)| < |\mathscr{O}(D)|$. But then

$$\begin{aligned}\operatorname{rad}(\mathscr{O}(D)) &\leq \operatorname{rad}(D) + 2 \leq x + 2 \leq \operatorname{rad}(D_i) + 4 \\ &\leq \operatorname{rad}(\mathscr{O}(D_i)) + 4 \leq 4|\mathscr{O}(D_i)| + 4 \leq 4(|\mathscr{O}(D_i)| + 1) \leq 4|\mathscr{O}(D)|\end{aligned}$$

still contradicts the assumption that $D$ is irregular. It follows that all ancestors $D_1, \ldots, D_\ell$ must be contained in (2.1) and the proof is complete. □

**Lemma 52.** *There are at most $2^k$ regular clusters in $\mathscr{H}_{4k+1}$.*

*Proof.* At time $t_{\leq 4k+1}$ there cannot exist two regular clusters $C_1$ and $C_2$ with $\mathscr{O}(C_1) \subseteq \mathscr{O}(C_2)$. Indeed, since $C_2$ intersects all the optimal clusters also intersected by $C_1$, we get that

$$\operatorname{rad}(C_1 \cup C_2) \leq \operatorname{rad}(C_2) + 1 \leq 4|\mathscr{O}(C_2)| + 1 \leq 4k + 1$$

and so $C_1$ or $C_2$ would have already gotten merged in $\mathscr{H}_{4k+1}$. Now, if there are more than $2^k$ regular clusters in $\mathscr{H}_{4k+1}$, then at least two must intersect the same set of optimal clusters. Since we have just ruled this out, the lemma follows. $\square$

Since all clusters are either regular or irregular, we can upper bound the number of clusters present at time step $t_{\leq 4k+1}$.

**Corollary 53.** *$\mathscr{H}_{4k+1}$ consists of at most $2^k + 4k^2 \in O(2^k)$ clusters.*

Theorem 46 is now an immediate consequence of combining Corollary 53 with Proposition 49.

## 2.4.2 An Upper Bound for $\mathbf{CL^{diam}}$

The main challenge in proving an upper bound on the approximation guarantee of Complete-Linkage when replacing the $k$-center objective by the $k$-diameter objective is to deal with possibly large increases of cost for particular merge steps, as shown in Figure 2.8. Whereas it was possible to upper bound this increase by an additive factor of 1 relative to the $k$-center objective function (see Lemma 47), the same cannot be done with regard to the $k$-diameter objective function. We cannot a priori rule out an increase by a *multiplicative* factor of 2. This possible doubling of the diameter prevents us from establishing a result similar to Proposition 49 and requires us to approach the problem in a completely different fashion. Not only will the analysis be much more complex, but it will also only guarantee an $O(k^2)$ approximation guarantee. [5]

#### Part 0: Preliminaries

Recall our proof of the fact that Single-Linkage is an $O(k)$-approximation for the $k$-center and $k$-diameter objective functions. At its core, this proof was concerned with Single-Linkage clusters only indirectly — first and foremost, it was about the relative placement of optimal clusters and how this placement influences the behavior of Single-Linkage. Whenever two optimal clusters are too far apart, they will never direct a merge, in the sense that there would be a point in one optimal cluster and a point in the other and that that pair would minimize the sep linkage function. This behavior guarantees that no Single-Linkage cluster crosses a large gap and thus stays relatively small altogether.

We will also approach the upper bound of Complete-Linkage for the $k$-diameter problem by considering what Complete-Linkage clusterings are possible for a given

[5] At least, according to our computations. As it turns out, one estimation — we will note which one — turns out not to be tight. So, although we will not do so, this factor could be improved to $O(k^{\ln 3/\ln 2})$

arrangement of optimal clusters. There are two cases, situated at either end of all possible arrangements, that are easy to analyse and that we will consider soon. First, we have to establish some notation. As before, fix some arbitrary $k$ as a parameter for the number of clusters and let $\mathscr{O}$ denote an optimal $k$-diameter solution for $(X, d)$, which, after scaling the instance appropriately, has diameter $\mathrm{diam}(\mathscr{O}) = 1$. We again introduce a *cluster graph* $G = (\mathscr{O}, E)$ on the set of optimal clusters. In the Single-Linkage case, we added edges between optimal clusters whenever their distance is at most 1. We could do the same to analyze the extreme cases here, but later on, a different setup will be more helpful. So instead, we will add edges between two optimal clusters $O$ and $O'$ if there exists some cluster $C \in \mathscr{H}_1$ that intersects both of them, i.e., $O, O' \in \mathscr{O}(C)$. With this setup, we do not add edges between any optimal clusters whose distance exceeds 1, so it is not too different from the cluster graph we constructed for Single-Linkage. The two extreme cases are now the following.

1. *$G$ is completly disconnected, meaning that $E = \varnothing$.* In this case, $\mathscr{H}_1 = \mathscr{O}$ and $\mathbf{CL}^{\mathrm{diam}}$ has successfully recovered the optimal clustering. No Complete-Linkage cluster can intersect two different optimal clusters — otherwise, there would be an edge — and no two Complete-Linkage clusters can be contained in the same optimal cluster — otherwise, we could merge them without increasing the cost.

2. *$G$ is fully connected, meaning that it consists of a single connected component.* In this case, the analysis is the same as for Single-Linkage. The instance has to be pretty small overall since merging all of its points into just one cluster costs at most $2k-1$. In particular, any algorithm will necessarily yield an $O(k)$-approximative solution.

While these edge cases are easy to handle, everything in between is much more complicated. Contrary to Single-Linkage, $\mathbf{CL}^{\mathrm{diam}}$ can form clusters that intersect several different connected components, making it difficult to estimate their cost properly. Complete-Linkage can build up large clusters within the connected components to the point that one can connect with a cluster contained in a different connected component. If the number of such "irregular" clusters grows much slower than the number of "regular" clusters decreases, we can still deal with them. Although the setup is quite different, the counting part is very similar to the one we employed for $\mathbf{CL}^{\mathrm{rad}}$. However, this time, the number of irregular clusters has to be much smaller since we cannot use any result similar to Proposition 49. Our goal is to balance out the following trade-off nicely:

1. Connected components in the cluster graph should be small enough so that we can use them to upper bound the size of Complete-Linkage clusters completely contained within them.

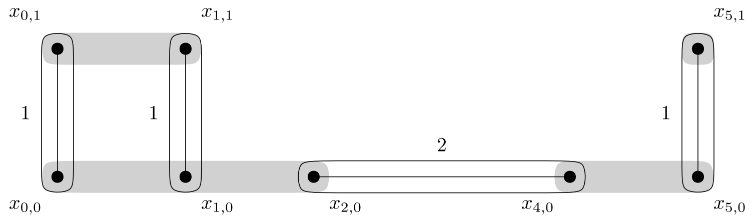


Figure 2.9: An example for instance where Complete-Linkage crosses gap of $2 = 2\,\mathrm{diam}(\mathscr{O})$. Distances between points are equal to the length of a shortest path connecting them in the graph. Note that the gray sets again indicate cliques of diameter 1. These sets also form the only 4-diameter solution of cost 1. All other clusterings contain some cluster that joins two points $x_{i,j}$ and $x_{i',j'}$ with $i \neq i'$ and $j \neq j'$ whose distance necessarily is at least 2. The clusters computed by Complete-Linkage are indicated by black outlines. First it forms the clusters $\{x_{0,0}, x_{0,1}\}$, $\{x_{0,1}, x_{1,1}\}$ and $\{x_{5,0}, x_{5,1}\}$. The precise order in which these merges happen is not significant, what is relevant is that every following merge has to induce a cost of at least 2. Complete-Linkage might thus merge $\{x_{2,0}\}$ and $\{x_{4,0}\}$ in the following step, whose distance is exactly 2. Of course there are other viable clusters, but we could also deduct some small $\varepsilon$ from the length of the edge connecting $x_{2,0}$ and $x_{4,0}$ and thus force this merge.

2. Connected components in the cluster graph should be large enough so that the number of Complete-Linkage clusters intersecting different connected components should be small. If the number of regular clusters decreases fast enough, hopefully, we will be able to push the overall number of clusters below $k$ at some favorable time step $t_{\leq x}$.

It all depends on how we set up our cluster graph or our series of cluster graphs. What we do is successively add edges between optimal clusters in such a way that $\mathrm{diam}(Z) = \mathrm{diam}(\cup_{A \in V(Z)} A) \leq |V(Z)|^2$ for all connected components $Z$ at any given point in time. The diameter of clusters contained within such connected components can thus never exceed $k^2$. This is the first side of the trade-off. Although this upper bound of $k^2$ is larger than we would like, it is what ultimately allows to upper bound the number of irregular clusters at time $t_{\leq k^2}$ in such a way that $\mathscr{H}_{k^2}$ is guaranteed to consist of at most $k$ clusters. This yields our main theorem.

**Theorem 54.** *Let $(\mathscr{C}_k)_{k=1}^n$ be the hierarchical clustering computed by $\mathbf{CL}^{diam}$ on $(X, d)$ optimizing the diameter. For all $1 \leq k \leq n$ the diameter $\mathrm{rad}(\mathscr{C}_k)$ is upper bounded by $k^2\,\mathrm{rad}(\mathscr{O}_k)$, where $\mathscr{O}_k$ is an optimal $k$-diameter clustering.*

Essential for our analysis is the following sequence of *cluster graphs* $G_t = (V_t, E_t)$ for $t = 1, \ldots, k^2$ constructed directly on the set $V_t = \mathscr{O}$ of optimal $k$-

clusters. We start with the cluster graph $G_1$ that contains edges $\{A, B\}$ for every two vertices $A, B \in V_1 = \mathscr{O}$ that are intersected by a common cluster from $\mathscr{H}_1$. We successively add edges based on some vertex labeling to create the remaining cluster graphs $G_2, \ldots, G_{k^2}$. The labeling distinguishes vertices as either *active* or *inactive*. We denote the set of active vertices in $V_t$ by $V_t^a$ and the set of inactive ones by $V_t^i$. In the beginning ($t = 1$), the inactive vertices are set to precisely those that are isolated: $V_1^i = \{O \in V_1 \mid \delta_{G_1}(O) = \varnothing\}$. For $t \geq 2$, the labeling is outlined in Definition 55. Over time, active vertices may become inactive, but inactive vertices never become active again.

Given some labeling for $V_{t+1}$, we construct $G_{t+1}$ from $G_t$ by adding additional edges: If there are two active vertices $A, B \in V_{t+1}^a$ that are both intersected by a common cluster from $\mathscr{H}_{t+1}$, we add an edge $\{A, B\}$ to $E_{t+1}$.

**Definition 55.** Let $A \in V_{t+1}$ be an arbitrary optimal cluster and $Z_A$ the connected component in $G_t$ that contains $A$. We call $A$ *inactive* (i.e., $A \in V_{t+1}^i$ ) if $\lceil \operatorname{diam}(Z_A) \rceil \leq t$, and *active* otherwise. Here, and in the following $\operatorname{diam}(Z_A) = \operatorname{diam}\left(\bigcup_{B \in V(Z_A)} B\right)$ denotes the cost of merging all optimal clusters contained in $V(Z_A)$.

Thus, if a connected component in $G_t$ has a small cost, then all vertices in this component become inactive by definition in $G_{t+1}$.

**Part 1: Inactive Clusters**

We state the following useful properties of inactive vertices in $(G_t)_{t=1}^{k^2}$.

**Lemma 56.** *If $Z$ is a connected component in $G_{t+1}$ with $V(Z) \cap V_{t+1}^i \neq \varnothing$, then*

1. *$Z$ is also a connected component in $G_t$ and $\lceil \operatorname{diam}(Z) \rceil \leq t$,*
2. *we have $V(Z) \subseteq V_{t+1}^i$, i.e., all vertices in $Z$ become inactive at the same time.*

*Moreover, we have $V_t^i \subseteq V_{t+1}^i$, so once vertices become inactive, they stay inactive. Equivalently, $V_{t+1}^a \subseteq V_t^a$.*

*Proof.* Take any inactive vertex $A \in V_{t+1}^i \cap V(Z)$ and consider the connected component $Z_A$ in $G_t$ containing $A$. By Definition 55, we have that $\lceil \operatorname{diam}(Z_A) \rceil \leq t$ and so all other vertices in $Z_A$ have to be in $V_{t+1}^i$ as well. We observe that $E_{t+1} \setminus E_t$ only contains edges between vertices from $V_{t+1}^a$ by construction. This shows that $Z = Z_A$.

It is left to show that inactive vertices stay inactive. For $t = 1$, the inactive vertices $V_1^i$ are already connected components with cost at most 1. As such, they remain inactive at step $t = 2$. For $t \geq 2$, consider an inactive vertex $A \in V_t^i$ and the connected component $Z \subseteq G_t$ containing it. We showed previously that $V(Z) \subset V_t^i$ and so $Z$ is also a connected component in $G_{t+1}$ with $\lceil \operatorname{diam}(Z) \rceil \leq t - 1 < t$ and thus $A \in V(Z) \subset V_{t+1}^i$. $\square$

**Definition 57.** Let $C \in \mathscr{H}_t$ for some fixed $t \in \mathbb{N}$. We define $\mathscr{I}_t = \{C \in \mathscr{H}_t \mid \mathscr{O}(C) \cap V_t^i \neq \varnothing\}$ as the set of all clusters in $\mathscr{H}_t$ which intersect at least one inactive vertex of $G_t$. We call these clusters inactive and those from $\mathscr{H}_t \backslash \mathscr{I}_t$ active.

We prove the following easy property about active clusters.

**Lemma 58.** *If $C \in \mathscr{H}_t \setminus \mathscr{I}_t$, then $G_t[\mathscr{O}(C)]$ forms a clique. In particular, there exists a connected component in $G_t$ that fully contains $\mathscr{O}(C)$.*

*Proof.* By definition of $\mathscr{I}_t$, $\mathscr{O}(C)$ must consist exclusively of active vertices. Since all of them are intersected by $C \in \mathscr{H}_t$, there exists an edge $\{A, B\} \in E_t$ for every pair $A, B \in \mathscr{O}(C)$. In other words, $G_t[\mathscr{O}(C)]$ forms a clique and the claim follows. $\square$

This does not necessarily hold for an inactive cluster $C \in \mathscr{I}_t$. As $C$ contains at least one inactive vertex, the connected component $Z$, which contains this vertex, does not grow. If $\mathbf{CL}^{\text{diam}}$ merges $C$ with another cluster later on, the result is an inactive cluster that may intersect vertices outside of $Z$. So $G_{t'}$ does not reflect the progression of $C$ for $t' \geq t$. However, as we will show in Lemma 60, the number of such clusters is at each time step upper bounded by the number of inactive vertices. Before proving, we first have to establish the following auxiliary result.

**Lemma 59.** *Let $C, D \in \mathscr{I}_t$ be two clusters, all of whose ancestors at time step $t-1$ are contained in $\mathscr{H}_{t-1} \setminus \mathscr{I}_{t-1}$. Then $\mathscr{O}(C) \cap \mathscr{O}(D)$ cannot contain any vertex $I \in V_t^i$. In other words, no two clusters can become inactive during the same time step because of the same shared vertex.*

*Proof.* Suppose, on the contrary, that such a vertex $I$ exists. Let $C', D' \in \mathscr{H}_{t-1} \setminus I_{t-1}$ denote those ancestors of $C$ and $D$ that intersect $I$ at time step $t-1$. If $Z_O$ denotes the connected component of $G_{t-1}$ that contains $O$, then

1. $\operatorname{diam}(Z_O) \leq t - 1$: This follows directly from the definition of inactive vertices.

2. $\mathscr{O}(C') \cup \mathscr{O}(D') \subset Z_O$: This follows from Lemma 58. Both induced subgraphs $G_{t-1}[\mathscr{O}(C')]$ and $G_{t-1}[\mathscr{O}(D')]$ constitute cliques that contain $O$.

However, then $\operatorname{diam}(C' \cup D') \leq t-1$, so they could not exist together at time step $t - 1$. $\square$

We now use this result to build our assignment.

**Lemma 60.** *The number of inactive clusters in $\mathscr{H}_t$ is, at most, the number of inactive vertices at time $t$. That is, $|\mathscr{I}_t| \leq |V_t^i|$ holds for all $t \in \mathbb{N}$.*

*Proof.* We prove the claim by showing that the following inductive construction defines a family of injective mappings $\phi_t : \mathscr{I}_t \to V_t^i$:

- Let $C \in \mathscr{I}_1$ be an inactive cluster. By definition, $C$ thus has to intersect an optimal cluster $I \in V_1^i$. As it turns out, $C$ already has to coincide with $I$: On the one hand, $C$ cannot intersect any other optimal cluster $I'$ since that would induce an edge $\{I, I'\}$ in $G_1$, turning $I$ active. This shows that $C$ is at least a subset of $I$. On the other hand, the same holds for any other cluster $D \in \mathscr{H}_1$ intersecting $I$. Thus, $\mathrm{diam}(C \cup D) \leq \mathrm{diam}(I) \leq 1$, so we could have merged them before time step $t = 1$. In effect, setting $\phi_1(C) = I$ and doing the same for all other inactive clusters yields an injective mapping.
- For $t > 1$ and $C \in \mathscr{I}_t$ we distinguish two cases. On the one hand, if $C$ descends from some cluster $C' \in \mathscr{I}_{t-1}$, then we can just set $\phi_t(C) = \phi_{t-1}(C')$. This is not a problem in itself since $\phi_{t-1}$ is already assumed to be injective. On the other hand, if $C$ only descends from clusters contained in $\mathscr{H}_{t-1} \setminus \mathscr{I}_{t-1}$, then $\mathscr{O}(C)$ has to contain a vertex $I \in V_t^i$ that was not yet inactive at time step $t - 1$, and we can set $\phi_t(C) = I$. Two observations guarantee that this yields a unique assignment. First, all clusters $D \in \mathscr{I}_t$ that descend from some cluster $D' \in \mathscr{I}_{t-1}$ have already been assigned to a different cluster $\phi_t(D) \in V_{t-1}^i$. Second, due to Lemma 59, no other cluster that becomes inactive at this time step can intersect $I$. □

**Part 2: Active Clusters**

Active clusters from $\mathscr{H}_t$ are nicely represented by the graph $G_t$ as shown in Lemma 58. We can indirectly upper-bound the cost of active clusters by upper-bounding the cost of the connected components in which they are contained.

**Lemma 61.** *Let $Z$ be a connected component in $G_t$. If $V(Z) \subset V_t^a$, we have* $\mathrm{diam}(Z) \leq |V(Z)|^2$.

*Proof.* Again, we prove this via an induction over $t$. The base case $t = 1$ is very similar to the Single-Linkage analysis of Theorem 20. For every pair of clusters $A, B \in V(Z)$ there exists a simple path $A = Q_1, \ldots, Q_s = B \in V(Z)$ of at most $|Z|$ optimal clusters with $d(Q_i, Q_{i+1}) \leq 1$. Using the triangle inequality we can thus upper bound $\mathrm{diam}(Z) \leq 2|V(Z)| - 1 \leq |V(Z)|^2$.

For $t > 1$ let $Z_1, \ldots, Z_u$ denote the connected components in $G_{t-1}$ with $V(Z) = \bigcup_{j=1}^{u} V(Z_j)$. Let $j, j' \in \{1, \ldots, u\}$. We observe that $V(Z_j) \subset V(Z) \subset V_t^a \subset V_{t-1}^a$. Thus, we obtain by induction that

$$\operatorname{diam}(Z_j) \leq |V(Z_j)|^2. \tag{2.2}$$

Suppose that $\lceil \operatorname{diam}(Z_j) \rceil \leq t - 1$. Then $V(Z_j) \subset V_t^i$ by definition, which is a contradiction to $V(Z) \cap V_t^i = \varnothing$. So we must have

$$t \leq \lceil \operatorname{diam}(Z_j) \rceil. \tag{2.3}$$

Combining (2.2) and (2.3) we obtain

$$t \leq \sqrt{\lceil \operatorname{diam}(Z_j) \rceil \lceil \operatorname{diam}(Z_{j'}) \rceil} \leq \sqrt{|V(Z_j)|^2 |V(Z_{j'})|^2} = |V(Z_j)||V(Z_{j'})|. \tag{2.4}$$

For $A, B \in V(Z)$, we want to upper bound the distance between $p \in A$ and $q \in B$. Let $A = Q_1, \ldots, Q_s = B$ be a simple path connecting $A$ and $B$ in $Z$ which enters and leaves every connected component $Z_j$ for $j \in \{1, \ldots, u\}$ at most once. We divide the path into several parts such that every part lies in one connected component from $\{Z_1, \ldots, Z_u\}$. Let $1 = m_1 < m_2 < \ldots < m_l = s$ such that $Q_{m_j} \ldots, Q_{m_{j+1}-1}$ lie in one connected component $Z^{(j)} \in \{Z_1, \ldots, Z_u\}$ and $Z^{(j)} \neq Z^{(j+1)}$ for all $j \in \{1, \ldots, l\}$. Since $(Q_{m_j-1}, Q_{m_j}) \in E_t$ we know that there exists a cluster in $\mathscr{H}_t$ that intersects $Q_{m_j-1}$ and $Q_{m_j}$, thus there is a pair of points $p_j \in Q_{m_j-1}$ and $q_j \in Q_{m_j}$ such that $d(p_j, q_j) \leq t$. We obtain

$$\begin{aligned} d(p,q) &\leq \sum_{j=1}^{l-1} \big( \operatorname{diam}(Z^{(j)}) + d(p_j, q_j) \big) + \operatorname{diam}(Z^{(l)}) \leq \sum_{j=1}^{l} \big( |V(Z^{(j)})|^2 + t \big) \\ &\leq \Big( \sum_{j=1}^{l} |V(Z^{(j)})| \Big)^2 = |V(Z)|^2. \end{aligned}$$

For the second inequality we use (2.2) and $d(p_j, q_j) \leq t$. For the third inequality, we use (2.4). So, we obtain the claimed upper bound on the cost of $Z$. □

A connected component in $G_{k^2}$ cannot contain two active clusters, yielding the following upper bound.

**Lemma 62.** *At time $t_{\leq k^2}$, the number of active clusters is less than or equal to the number of active vertices. In other words, $|\mathscr{H}_{k^2} \setminus \mathscr{I}_{k^2}| \leq |V_{k^2}^a|$.*

*Proof.* By Lemma 58 we know that every cluster $C \in \mathscr{H}_{k^2} \setminus \mathscr{I}_{k^2}$ is fully contained in a connected component $Z_C$ from $G_{k^2}$. We show that mapping any such $C$ to an arbitrary vertex in $Z_C$ yields an injective map $\varphi : \mathscr{H}_{k^2} \setminus \mathscr{I}_{k^2} \hookrightarrow V^a_{k^2}$. First, notice that $\varphi$ is well-defined: If $Z_C$ contains an inactive vertex, then all its vertices are inactive (Lemma 56), contradicting the choice of $C$ as active.

Suppose now that there are two different clusters $C, C' \in \mathscr{H}_{k^2} \setminus \mathscr{I}_{k^2}$ that are mapped to the same vertex $\varphi(C) = \varphi(C')$. Then the connected components $Z_C$ and $Z_{C'}$, in which they are embedded, already have to coincide ($Z_C = Z_{C'}$). But we have just shown (Lemma 61), that $\operatorname{diam}(Z_C) \leq |V(Z_C)|^2 \leq k^2$ and so $C$ and $C'$ would have already been merged in $\mathscr{H}_{k^2}$. As such, the images of both cannot coincide, and the map is injective. □

Together with the bound for the number of inactive clusters, we can now prove the theorem.

*Proof of Theorem 54.* Using Lemma 60 and Lemma 62 we obtain

$$|\mathscr{H}_{k^2}| = |\mathscr{H}_{k^2} \setminus \mathscr{I}_{k^2}| + |\mathscr{I}_{k^2}| \leq |V^a_{k^2}| + |V^i_{k^2}| = k$$

and thus $\operatorname{diam}(\mathscr{C}_k) \leq \operatorname{diam}(\mathscr{H}_{k^2}) \leq k^2 \operatorname{diam}(\mathscr{O})$. □

## 2.5 The Average Approximation Factor

Although we have just increased the lower bound on the approximation guarantees of Complete-Linkage for both the $k$-diameter and $k$-center objective functions with regard to $k$, if we instead consider the approximation guarantees with regard to the number of points of the instance, then the lower bound has not been improved. The lower bound established by Dasgupta and Long was only $\Omega(\log k)$, but their instance was exponentially smaller than ours, containing just $n = k^2$ points. So in terms of $n$ they have proven a lower bound of $\Omega(\log n)$. Complete-Linkage only yields an $\Omega(k)$-approximate solution for our instance, but this instance contains $n = k2^k$ points and so we still get a lower bound of $\Omega(\log n)$. From this perspective nothing much has changed. In fact, Proposition 49 proves that Complete-Linkage always returns an $O(\log n)$-approximative solution, so we cannot increase the lower bound. However, if we also analyze Single-Linkage from this perspective then we see that it is much worse than Complete-Linkage. As we have established, with an instance consisting of only $n = 2k$ points we can force Single-Linkage to compute a solution that is worse by a factor of $\Omega(k) = \Omega(n)$. We will conclude this chapter by showing these lower bounds even hold when we average over all computed clusterings. This substantiates our earlier claim that Complete-Linkage builds up worst-case solutions much slower than Single-Linkage. Single-Linkage computes an asymptotically much worse clustering than Complete-Linkage in average.

**Definition 63.** Let $(\mathscr{C}_k)_{k=1}^n$ be an arbitrary hierarchical clustering on $(X, d)$ and let $(\mathscr{O}_k)_{k=1}^n$ be optimal solutions for the $k$-center or $k$-diameter objective functions. We denote by

$$\operatorname{avg}((\mathscr{C}_k)_{k=1}^n) = \frac{1}{n} \sum_{k=1}^{n} \frac{\mathsf{cost}(\mathscr{C}_k)}{\mathsf{cost}(\mathscr{O}_k)}$$

the *average approximation factor* of $(\mathscr{C}_k)_{k=1}^n$.

The following corollary is an immediate consequence of Proposition 49.

**Corollary 64.** *Let $(\mathscr{C}_k)_{k=1}^n$ be the hierarchical clustering computed by Complete-Linkage for the radius. We have*

$$\operatorname{avg}((\mathscr{C}_k)_{k=1}^n) \leq \lceil \log(n) \rceil.$$

However, this upper bound of $\lceil \log n \rceil$ seems too pessimistic. It would be interesting to know whether Complete-Linkage does not in fact compute a constant factor approximation on average and whether similar results hold for the diameter. Still let us compare it with the average approximation factor of Single-Linkage.

**Proposition 65.** *Let $X = \{1, \ldots, 2^s\} \subset \mathbb{R}$ for some $s$. Then the average approximation factor achieved by Single-Linkage on $(X, \|\cdot\|_1)$ for both, radius and diameter, is at least $\frac{n}{24} - 1$.*

*Proof.* Let $n = 2^s$. We can assume that in the $k$-th step Single-Linkage merges the two clusters containing $x_{n-k}$ and $x_{n-k+1}$ as the distance between these clusters is 1. The $k$-clustering computed by Single-Linkage on $(X, \|\cdot\|_1)$ then equals

$$\mathscr{C}_k = \{\{x_1\}, \ldots, \{x_{k-1}\}, \{x_k, \ldots, x_n\}\}$$

and has diameter $n - k$.

On the other hand for $0 \leq t \leq s$ the optimal $2^t$-clustering has diameter $2^{s-t} - 1$ and consists of clusters with $2^{s-t}$ consecutive points in $X$

$$\mathscr{O}_{2^t} = \{\{x_1, \ldots, x_{2^{s-t}}\}, \ldots, \{x_{2^{s-t}(2^t-1)+1}, \ldots, x_{2^s}\}\}.$$

Thus we obtain for the diameter

$$\begin{aligned}
\operatorname{avg}((\mathscr{C}_k)_{k=1}^n) &= \frac{1}{n}\sum_{t=0}^{s-1}\sum_{k=2^t+1}^{2^{t+1}} \frac{\mathsf{cost}(\mathscr{C}_k)}{\mathsf{cost}(\mathscr{O}_k)} \geq \frac{1}{n}\sum_{t=0}^{s-1} 2^t \frac{\mathsf{cost}(\mathscr{C}_{2^{t+1}})}{\mathsf{cost}(\mathscr{O}_{2^t})} \\
&\geq \frac{1}{n}\sum_{t=0}^{s-1} 2^t \frac{2^s - 2^{t+1}}{2^{s-t}} = \frac{1}{n}\sum_{t=0}^{s-1} 2^{2t}\frac{2^{s-t}-2}{2^{s-t}} \\
&\geq \frac{1}{2n}\sum_{t=0}^{s-2} 4^t = \frac{1}{2^{s+1}}\frac{4^{s-1}-1}{3} = \frac{2^{s-1}}{12} - \frac{1}{3\cdot 2^{s+1}} \\
&\geq \frac{n}{24} - 1
\end{aligned}$$

The same computation can be done for the radius, as the radius of $\mathscr{C}_{2^{t+1}}$ equals $\frac{2^s-2^{t+1}}{2}$ and the radius of $\mathscr{O}_{2^t}$ equals $\frac{2^{s-t}}{2}$. $\square$

# Chapter 3

# Separated $k$-Clustering

In the introduction, we have spoken of a certain tension between the two main goals of cluster analysis as we have established them. We would like to partition a data set in such a way that, on the one hand, dissimilar points lie in different clusters, and on the other, similar points lie in the same cluster. However, this tension, which cannot be resolved a priori and can only be experimented with on a case-by-case basis, is usually pushed aside. The first goal is turned into a penalizing objective ($k$-center, $k$-diameter, $k$-MSR, $k$-MSD, $k$-median, $k$-means, etc.), which measures the spread of a clustering, while the second is omitted altogether. As a result, clusters do not have to be separated well, and no clear lines of demarcation have to exist. The second goal can only be traced implicitly through the imposition of a pre-determined number $k$ of allowed clusters.[1] This is true also for this dissertation. In almost all chapters, we have treated clustering problems as optimization problems where dissimilarity is penalized, but the similarity is ignored. In this chapter, we would like to prove a different outlook by trying to combine each of the objectives deriving from the first goal with the $k$-separation objective, which, to recall, measures the separation of a clustering, i.e., the minimal distance between any of its clusters. These results form part of a yet unpublished project on bi-objective clustering. We have decided not to deviate from the larger context and allow each objective to refer to a separate metric. For the most part, we will thus work with two metrics $d_1, d_2$ defined on a single set $X$.

## 3.1 The $k$-Separation Objective

Although we have already introduced the $k$-separation objective in the introduction, let us quickly recap the main points. The separation $\operatorname{sep}(\mathscr{C}, d)$ of a clustering

[1] This also holds for hierarchical clusterings that are evaluated based on their individual levels.

$\mathscr{C}$ of some finite metric space $(X, d)$ is defined to be minimal distance

$$\operatorname{sep}(\mathscr{C}, d) = \max_{C,C' \in \mathscr{C}} \min_{(c,c') \in C \times C'} d(c, c')$$

between any two of its clusters. Note that we also pass the metric as an argument to the objective. This is because we will work with two different metrics simultaneously and have to explicitly state in relation to which the objective is evaluated. Contrary to all the other objectives discussed in this work, the goal is to *maximize* this function instead of minimizing it. A larger objective value implies that distinctions between different clusters are more pronounced. If we consider the set of all clusterings that satisfy a given separation value, we will find that it can be characterized quite nicely.

**Definition 66.** Let $(X, d)$ be a finite metric space and $\tau \geq 0$. A clustering $\mathscr{C}$ of $(X, d)$ is *$\tau$-separated*, if $\tau \leq \operatorname{sep}(\mathscr{C})$.

As it turns out, imposing $\tau$-separability amounts to imposing a specific set of must-link constraints. The latter consists of a set $L_\tau \subset \binom{X}{2}$ of links, which are satisfied by a given clustering $\mathscr{C} = \{C_1, \ldots, C_k\}$, if and only if

$$x \in C_i \wedge \{x, y\} \in L_\tau \Rightarrow y \in C_i$$

for all $x, y \in X$ and all $i \in \{1, \ldots, k\}$. If we conceive of these links as edges on the set of points (as we will do), then a clustering has to partition this graph in such a way that no edge is cut. To see the connection between $\tau$-separability and must-link constraints, consider the following graph.

**Definition 67.** Let $(X, d)$ be a finite metric space and $\tau \geq 0$. The *$\tau$-separation graph* $G_{(X,d)}(\tau)$ of $X$ consists of

- vertices $V_{(X,d)}(\tau) = X$ and
- edges $E_{(X,d)}(\tau) = \{\{x, y\} \mid x, y \in X \colon 0 < d(x, y) < \tau\}$.

In other words, if the distance between two different points is smaller than $\tau$, then we impose a must-link constraint for that pair: $L_\tau = E_{(X,d)}(\tau)$. Intuitively, it should already be clear how these must-link constraints amount to $\tau$-separability, but we will also prove it in just a bit.

**Definition 68.** Let $(X, d)$ be a finite metric space and $\tau \geq 0$. We denote by $\mathscr{S}_{(X,d)}(\tau)$ the clustering consisting of the connected components of $G_{(X,d)}(\tau)$.

What is noteworthy about must-link constraints (and therefore also $\tau$-separability), as opposed to other constraints, is that there exists something like a minimal clustering for that constraint, in that it is a refinement of every clustering satisfying the constraints. For $\tau$-separability this is $\mathscr{S}_{(X,d)}(\tau)$.

**Lemma 69.** *Let $(X, d)$ be a metric space and $\tau \geq 0$. Then*

1. *$\mathscr{S}_{(X,d)}(\tau)$ is $\tau$-separated.*
2. *If $\mathscr{C}$ is clustering of $(X, d)$ then it is $\tau$-separated if and only if $\mathscr{S}_{(X,d)}(\tau)$ is a refinement clustering $\mathscr{C}$. This means that for every $S \in \mathscr{S}_{(X,d)}(\tau)$ there has to exist some $C \in \mathscr{C}$ such that $S \subseteq C$.*

*In this sense, $\mathscr{S}_{(X,d)}(\tau)$ can be called the* minimally $\tau$-separated clustering *of $(X, d)$.*

*Proof.* This lemma immediately follows from two easy observations:

1. The minimal distance between two different clusters $S, S' \in \mathscr{S}_{(X,d)}(\tau)$ is strictly more than $\tau$. Otherwise, if there exist points $x \in S$ and $x' \in S'$ whose distance is smaller than $\tau$, we would have added an edge between them, meaning that neither $S$ nor $S'$ would be connected components.
2. We cannot split any cluster $S \in \mathscr{S}_{(X,d)}(\tau)$ without lowering its separation to something strictly smaller than $\tau$: since $S$ is connected in $G_{(X,d)}(\tau)$ at least one edge would have to be cut. □

In particular, if we want to compute a $\tau$-separable clustering, then we only have to merge the clusters contained in $S \in \mathscr{S}_{(X,d)}(\tau)$, which itself can be computed relatively easily for any specific $\tau$.

**Lemma 70.** *Let $(X, d)$ be a metric space consisting of $n$ elements and $\tau \geq 0$. We can compute $\mathscr{S}_{(X,d)}(\tau)$ in $O(n^2)$ time.*

*Proof.* The $\tau$-separation graph can be computed in $O(n^2)$ time by passing over all pairs of points. In turn, connected components can be computed in linear time. □

**Lemma 71.** *Let $(X, d)$ be a metric space consisting of $n$ elements. We can compute the minimally $\tau$-separated clusterings $\mathscr{S}_{(X,d)}(\tau)$ for all $\tau \in d(X \times X)$ in $O(n^2 \log n)$ time.*

*Proof.* We can just run Kruskal's algorithm on the complete graph on $X$, where the edge weights are given by the respective distances. The set of connected components at any given point in time equals $\mathscr{S}_{(X,d)}(\tau)$, where $\tau$ is the weight of the last introduced edge. This is the same as running Single-Linkage on $(X, d)$. □

With this established, let us continue our task and combine this objective with other objectives.

## 3.2 Combining $k$-Diameter and $k$-Separation

What do we have in mind when discussing the combination of two different objectives? The best-case scenario would be that of finding a clustering that approximates both objectives well, independently of each other. However, this is not feasible, at least for the $k$-diameter and $k$-separation objectives, as Observation 72 will show. Both objectives are too opposed to each other to properly approximate them simultaneously. For any fixed number of clusters, there is no guarantee that a solution exists that yields constant-factor approximations for both objectives, not even for subspaces of the Euclidean line $\mathbb{R}$.

**Observation 72.** For any fixed $k, m \in \mathbb{N}_{\geq 1}$ consider the set

$$X_{k,m} = \{1, \dots, k-1\} \cup \underbrace{\{-x/m \mid x \in \{0, \dots, m(k-1)^2\}\}}_{A_{k,m}}$$

together with the usual metric $d_1(x, y) = d_2(x, y) = |x - y|$ from $\mathbb{R}$. We now show that all clusterings can have a good (i.e., constant-factor) approximation ratio with regard to at most one of the two objectives; when measured against the other, its approximation ratio is necessarily quite bad.

1. The only clustering that achieves a constant-factor approximation ratio with regard to sep is $\mathscr{C} = \{\{1\}, \dots, \{k-1\}, A_{k,m}\}$ with $\operatorname{sep}(\mathscr{C}, d_1) = 1$. Every other clustering $\mathscr{C}'$ necessarily breaks up $A_{k,m}$ into at least two sets and so has to have a separation value of $\operatorname{sep}(\mathscr{C}', d_1) = 1/m$, which is just the distance between two consecutive points in $A_{k,m}$.

   However, precisely by not splitting up $A_{k,m}$ does $\mathscr{C}$ have a bad approximation ratio with regard to the $k$-diameter objective. Its maximal diameter is equal to $\operatorname{diam}(A_{k,m}, d_2)$, and thus at least $(k-1)^2$ while partitioning $[-(k-1)^2, k-1] \supset X_m$ into $k$ intervals of equal length yields a clustering of diameter at most
$$\frac{(k-1)^2 + (k-1)}{k} = \frac{k(k-1)}{k} = k - 1.$$
   It is thus, at most, a $k$-approximation with regard to diam.

2. The same argument also shows that constant-factor approximations for diam perform poorly with regards to sep. A solution with a constant-factor approximation ratio relative to the former must break up $A_{k,m}$ and thus has a separation value at least $m$ times worse than the optimal.

Either case is untenable in its own right; the approximation ratios are just too large. Instead of analyzing the cost of a solution independently for each objective, we turn to the computation of *Pareto sets*, which encode trade-offs between the two objectives and which are a standard object of study in multi-objective optimization.[2] In our case, it amounts to the following: a clustering $\mathscr{C}$ belongs to the Pareto set for this combination, meaning that it is *Pareto optimal*, if and only if no clustering $\mathscr{C}'$ with $\operatorname{diam}(\mathscr{C}', d_2) \leq \operatorname{diam}(\mathscr{C}, d_2)$ and $\operatorname{sep}(\mathscr{C}', d_1) \geq \operatorname{sep}(\mathscr{C}, d_1)$ exists, where at least one of both inequalities is strict. (In the following sections, we will replace the diameter objective with various other objectives.) In other words, when given a Pareto optimal clustering, we cannot improve its value regarding either objective without worsening the other. Despite being quite an interesting problem, such a bi-objective clustering scenario was published in 2017 by Alamdari and Shmoys ([7]). However, they are interested in a very different type of combination: that of the $k$-center and the $k$-median objective.

Finding Pareto sets is NP-hard because we would still have to solve the vanilla $k$-diameter problem (this is the case when the separation value equals the minimal distance between any two points). Instead, we will try to find clusterings that closely approximate Pareto-optimal solutions. As it turns out, due to the simple structure that must-link constraints impose, we can relatively easily compute a set that contains for every Pareto optimal solution $\mathscr{C}^*$ a clustering $\mathscr{C}$, such that $\operatorname{diam}(\mathscr{C}, d_2) \leq 2 \operatorname{diam}(\mathscr{C}^*, d_2)$ and $\operatorname{sep}(\mathscr{C}, d_1) \geq \operatorname{sep}(\mathscr{C}^*, d_1)$. We call such a set a $(1, 2)$-approximative Pareto set. (The first factor always refers to the separation objective, and the second always to whatever other objective is considered.) To this end, we slightly modify the 2-approximative $k$-center/$k$-diameter algorithm of Hochbaum and Shmoys (given in [54]) in such a way that it does not just cluster points but rather merges specific pre-established clusters. For every possible separation value $\tau$, we can then use this algorithm as a subroutine (applied to the clusters of the minimally $\tau$-separated clustering) to compute a 2-approximate $\tau$-separated $k$-diameter clustering. In other words, we just approximate the $\tau$-separated $k$-diameter problem defined below for every $\tau$.

**Definition 73** (The $\tau$-separated $k$-diameter problem)**.** Let $d_1$, $d_2$ be two metrics over a finite set $X$, $k \in \mathbb{N}$, and $\tau \geq 0$. The goal in the *$\tau$-separated $k$-diameter problem* is to find a clustering $\mathscr{C} = \{C_1, \ldots, C_k\}$ that minimizes

$$\operatorname{diam}(\mathscr{C}, d_2) = \max_i \max_{x,y \in C_i} d_2(x, y)$$

and that guarantees that for all $S \in \mathscr{S}_{(X,d_1)}(\tau)$ there exists some $i \in \{1, \ldots, k\}$ with $S \subseteq C_i$.

Algorithm 2 shows a modification of the algorithm of Hochbaum and Shmoys that yields 2-approximations for the $\tau$-separated $k$-diameter problem. In a first

[2]For a general book on multi-objective optimization see, for example, [39].

step, the algorithm computes the minimally $\tau$-separated clustering $\mathscr{S}_{(X,d)}(\tau)$ for $(X, d)$, whose clusters it then merges in a second step in such a way that the number of clusters falls below $k+1$ and the diameter stays small. The underlying idea in the second step is exactly the same as in the algorithm of Hochbaum and Shmoys. We sort of guess the maximal diameter of an optimal $\tau$-separated $k$-diameter clustering and use it to further merge the clusters of our minimally $\tau$-separated clustering. Since there are at most $O(|X|^2)$ values for the maximal diameter, this can be done in polynomial time.

Two quick notes on the running time of Algorithm 2, without going into too much depth: Despite our phrasing, it is not necessary to test every possible cost that the solution could take:

1. Since the cost of an optimal $\tau$-separated $k$-diameter clustering has to be at least as large as the cost of the minimally $\tau$-separated clustering, we know that we only have check $\delta$'s, which are at least as large as
$$\max_{S \in \mathscr{S}_{(X,d_1)}(\tau)} \max_{x,y \in S} d_2(x, y).$$
2. We only have to find the smallest $\delta$ for which the maximally independent set consists of at most $k$ clusters, meaning that the loop can be implemented as a binary search, though we will not get into the details.

**Theorem 74.** *Algorithm 2 is a 2-approximation algorithm for the $\tau$-separated $k$-diameter problem that runs in polynomial time.*

*Proof of Theorem 74.* Let $\mathscr{C}^* = \{C_1^*, \ldots, C_k^*\}$ be an optimal $\tau$-separated $k$-diameter clustering, and $\{S_1^*, \ldots, S_\ell^*\}$ the independent set computed by Algorithm 2 during iteration $\delta^* = \operatorname{diam}(\mathscr{C}^*, d_2)$. From Lemma 69, we know that $\mathscr{S}$ is necessarily a refinement of $\mathscr{C}^*$, so each cluster contained in the independent set has to be a subset of some cluster from $\mathscr{C}^*$. This implies that $\ell \leq k$ by the pigeonhole principle. (Otherwise, some optimal cluster, of which there are only $k$, would have to contain two clusters from $\mathscr{S}$, of which there would be at least $k+1$, whose union would have a diameter strictly greater than $\delta^*$.) By merging the clusters from $\mathscr{S}$ via star graphs, we ensure that the diameter of all resulting clusters $X_1, \ldots, X_\ell$ is at $2\delta^*$. Indeed, if for any two points $x, y$ contained in some $X_i$, we know that
$$d_2(x, y) \leq d_2(x, z) + d_2(z, y) \leq \delta^* + \delta^* = 2\delta^*$$
for all $z \in S_i$. Finally, since we only merged clusters from $\mathscr{S}$ and did not split up any of them, the separation of the final clustering is at least $\tau$. The separation can only increase. □

**Algorithm 2:** $k$-Diameter Pareto Approximation

```
Input  : Two metrics $d_1, d_2$ over a finite set $X$, a number $k \in \mathbb{N}$, a
         threshold $\tau \in \mathbb{R}$.
Output: A 2-approximation $\mathscr{C} = \{C_1, \ldots C_k\}$ for the $\tau$-separeted
         $k$-diameter problem.
1  $\mathscr{C} \leftarrow \{X\}$
2  $\mathscr{S} \leftarrow \mathscr{S}_{(X,d_1)}(\tau)$
3  for $\delta \in d_2(X \times X)$ do
4      $G_{\leq\delta} \leftarrow (\mathscr{S}, \{\{S, S'\} \mid \operatorname{diam}(S \cup S', d_1) \leq \delta\})$
5      $\{S_1, \ldots, S_\ell\} \leftarrow$ maximal independent set in $G_{\leq\delta}$
6      if $\ell \leq k$ then
7          partition $G_{\leq\delta}$ into star-graphs $X_1, \ldots, X_\ell$ centered at $S_1, \ldots, S_\ell$
8          if $\operatorname{diam}(\{X_1, \ldots, X_\ell\}, d_2) < \operatorname{diam}(\mathscr{C}, d_2)$ then
9              $\mathscr{C} \leftarrow \{X_1, \ldots, X_\ell\}$
10         end
11     end
12 end
13 return $\mathscr{C}$
```

Finally, we can compute our approximate Pareto set by just running the algorithm for every $\tau \in d_1(X \times X)$. We could also accomplish this slightly more efficiently by pre-computing all minimally separated clusterings using Single-Linkage, respectively, Kruskal's algorithm.

**Corollary 75.** *Given two metrics $d_1$ and $d_2$ over a finite set $X$, we can compute in polynomial time a $(1, 2)$-approximate Pareto with respect to the objectives $f_1 = \operatorname{sep}$ and $f_2 = \operatorname{diam}$.*

Running Algorithm 2 with a threshold of $\tau = \min_{x \neq y} d_1(x, y)$ amounts to finding a 2-approximative solution for the vanilla $k$-diameter problem. Since no polynomial-time $(2 - \varepsilon)$-approximation algorithm for this problem can exist for any $\varepsilon > 0$ unless P = NP ([41]), we also know that the $(1, 2)$-approximation ratio we have established in Theorem 74 cannot really be improved.

## 3.3 Combining $k$-Center and $k$-Separation

We can also combine the $k$-separation objective with the $k$-center objective. As we have seen in the introduction, the latter is closely related to the $k$-diameter objective; both draw on the same intention, and there is merely a shift of focus from partitionings to representatives. Consequently, the analyses conducted here will be very similar to those of the previous section. For instance, Observation 72 unfolds almost the same for the $k$-center objective and can be reused as is. As such, we will not repeat it and instead focus primarily on the algorithm(s), the only part where qualitative (though not stark) changes are required.

**Definition 76** (The $\tau$-separated $k$-center problem)**.** Let $d_1$, $d_2$ be two metrics over a finite set $X$, $k \in \mathbb{N}$, and $\tau \geq 0$. The goal in the *$\tau$-separated $k$-center problem* is to find an assignment $\sigma : X \to C$ from $X$ to a set of $k$ centers $C = \{c_1, \dots, c_k\}$ that minimizes

$$\operatorname{rad}(\sigma, d_2) = \max_{x \in X} d_2(x, \sigma(x))$$

and that guarantees that for all $S \in \mathscr{S}_{(X,d_1)}(\tau)$ there exists some $i \in \{1, \dots, k\}$ with $S \subseteq \sigma^{-1}(c_i)$.

Algorithm 3 again builds on the algorithm by Hochbaum and Shmoys and is just a slight variation of Algorithm 2. Interestingly, we can choose any arbitrary point from the clusters of the minimally separated clustering as a center for our assignment.

**Algorithm 3:** $k$-Center Pareto Approximation

```
Input  : Two metrics $d_1, d_2$ over a finite set $X$, a number $k \in \mathbb{N}$, a
         threshold $\tau \in \mathbb{R}$.
Output: A 2-approximation $\sigma : X \to C$ for the $\tau$-separeted $k$-center
         problem.
1  $\sigma \equiv x$ for some $x \in X$;
2  for $\rho \in d_2(X \times X)$ do
3      $G_{\leq 2\rho} \leftarrow \big(\mathscr{S}_{(X,d_1)}(\tau), \{\{S, S'\} \mid \operatorname{diam}(S \cup S', d_1) \leq 2\rho\}\big)$;
4      $\{S_1, \dots, S_\ell\} \leftarrow$ maximal independent set in $G_{\leq 2\rho}$;
5      if $\ell \leq k$ then
6          partition $G_{\leq 2\rho}$ into star-graphs $X_1, \dots, X_\ell$ centered at $S_1, \dots, S_\ell$;
7          $c_i \leftarrow$ arbitrary point from $S_i$ for all $i$;
8          define $\sigma'$, such that $\sigma'(x) = c_i$, iff $x \in X_i$;
9          if $\operatorname{rad}(\sigma', d_2) < \operatorname{rad}(\sigma, d_2)$ then
10             $\sigma \leftarrow \sigma'$;
11         end
12     end
13 end
14 return $\sigma$;
```

**Theorem 77.** *Algorithm 3 is a 2-approximation algorithm for the $\tau$-separated $k$-center problem that runs in polynomial time.*

*Proof of Theorem 77.* Let $\sigma^*$ be an optimal $\tau$-separated $k$-center assignment, and $\{S_1^*, \dots, S_\ell^*\}$ the independent set computed by the algorithm during iteration $\rho^* = \operatorname{rad}(\sigma^*, d_2)$. From Lemma 69, we know that $\mathscr{S}_{(X,d_1)}(\tau)$ necessarily is a refinement of the clustering induced by $\sigma^*$, so each cluster contained in the independent set has to be a subset of some cluster from $\sigma^*$. This implies that $\ell \leq k$ by the pigeonhole principle. Otherwise, some optimal cluster would have to contain two clusters from $\mathscr{S}_{(X,d_1)}(\tau)$ whose union would have diameter strictly greater than $2\rho^*$ and thus radius strictly greater than $\rho^*$. By merging the clusters from $\mathscr{S}_{(X,d_1)}(\tau)$ via star graphs, we ensure that the diameter of all resulting clusters $C_1, \dots, C_\ell$ is at most $2\rho^*$, so for any $x \in C_i$ it holds that

$$d_2(x, \sigma(x)) = d_2(x, c_i) \leq \operatorname{diam}(C_i, d_2) \leq 2\rho^*.$$

Finally, since we have only merged clusters from $\mathscr{S}_{(X,d_1)}(\tau)$ and did not split up any of them, the separation can only increase, and has to be at least $\tau$. $\square$

We get the following result by running this algorithm for every $\tau$, or rather by applying it to the hierarchical clustering computed by Single-Linkage.

**Corollary 78.** *Given two metrics $d_1$ and $d_2$ over a finite set $X$, we can compute in polynomial time a $(1,2)$-approximate Pareto set $\mathcal{P}$ with respect to the objectives $f_1 = \operatorname{sep}$ and $f_2 = \operatorname{rad}$. Since the Pareto set has size at most $\binom{n}{2}$, the same holds for $\mathcal{P}$.*

Again, the $(1,2)$-approximation factor cannot be improved. For $\tau = 0$, the Algorithm 3 computes 2-approximative solutions for $k$-center, which is the lowest approximation ratio achievable in polynomial time unless P = NP ([54]).

As a quick side note, Line 3 can be pretty expensive. Instead of building the graph on top of the clusters of the minimally separated clustering, we can instead build the graph on top of good representatives for each cluster. Algorithm 4 shows the corresponding modification, which is still closer to the algorithm by Hochbaum and Shmoys.

**Theorem 79.** *Algorithm 3 is a 2-approximation algorithm for the $\tau$-separated $k$-center problem that runs in polynomial time.*

*Proof.* Let $\sigma^{\text{opt}} : X \to C^{\text{opt}}$ be an optimal $\tau$-separated $k$-center assignment, and $\{c_1^*, \dots, c_\ell^*\}$ the independent set computed by the algorithm during iteration $\rho^{\text{opt}} = \operatorname{rad}(\sigma^{\text{opt}}, d_2)$. From Lemma 69, we know that $\mathscr{S}$ is necessarily hierarchically compatible with the clustering induced by $\sigma^{\text{opt}}$, so each cluster corresponding

**Algorithm 4:** Pareto Approximation

```
Input  : Two metrics $d_1, d_2$ over a finite set $X$, a number $k \in \mathbb{N}$, a
         threshold $\tau \in \mathbb{R}$.
Output: A 2-approximation $\sigma : X \to C$ for the $\tau$-separated $k$-center
         problem.
1  $\sigma \equiv x$ for some $x \in X$;
2  $\mathscr{S} \leftarrow\in \mathscr{S}_{(X,d_1)}(\tau)$;
3  $c_S \leftarrow$ arbitrary point from $\mathrm{argmin}_{c\in X} \max_{x\in S} d_1(x,c)$ for all $S \in \mathscr{S}$;
4  $C_{\mathscr{S}} \leftarrow \{c_S \mid S \in \mathscr{S}\}$;
5  for $\rho \in d_2(X \times X)$ do
6      $G^2_\rho \leftarrow (C_{\mathscr{S}}, \{\{c,c'\} \mid \exists x \in X : d_2(c,x) \le \rho,\ d_2(c',x) \le \rho\})$;
7      $\{c_1, \dots, c_\ell\} \leftarrow$ maximal independent set in $G^2_\rho$;
8      if $\ell \le k$ then
9          $\sigma' \leftarrow$ for all $S \in \mathscr{S}$ set $\sigma'(S) = \{c_i\}$ if $d_2(c_S, c_i) \le \rho$ and resolve
            tiebreaks in such a way to stay hierarchically compatible with $\mathscr{S}$;
10         if $\mathrm{rad}(\sigma', d_2) < \mathrm{rad}(\sigma, d_2)$ then
11             $\sigma \leftarrow \sigma'$;
12         end
13     end
14 end
15 return $\sigma$;
```

to a center contained in the independent set has to be a subset of some cluster from $\sigma^{\mathrm{opt}}$. This implies that $\ell \le k$ by the pigeonhole principle. Otherwise, some optimal cluster would have to contain two clusters from $\mathscr{S}$ whose centers cannot be covered by a ball of radius $\rho^{\mathrm{opt}}$ centered at any point in $X$. By merging the clusters from $\mathscr{S}$ via $\sigma$ we ensure that the radius of all resulting clusters $C_1, \dots, C_\ell$ is at most $2\rho^{\mathrm{opt}}$: for any $x \in S$ it holds that

$$d_2(x, \sigma(x)) = d_2(x, c_S) + d_2(c_S, \sigma(x)) \le \rho^{\mathrm{opt}} + \rho^{\mathrm{opt}}.$$

Here we have used the fact that $\rho^{\mathrm{opt}}$ is to be at least

$$\max_{S\in\mathscr{S}} \max_{x\in S} d_2(x, c_S)$$

by choice of each center $c_S$. Finally, since we have only merged clusters from $\mathscr{S}$ and did not split up any of them, the separation can only increase and has to be at least $\tau$. $\square$

## 3.4 Combining $k$-Median/$k$-Means with $k$-Separation

As in the previous two sections, Observation 72 can be used to show that there is no hope of finding a clustering that performs well with regards to both the $k$-median (or $k$-means) and the $k$-separation objective, meaning that we again have to focus on Pareto approximations.

**Observation 80.** Consider a set of points $\{x/m \mid x \in \{0, \dots, m\ell\}\}$ for any $\ell$ and even $m$. The 1-median cost of this set is given by

$$\begin{aligned}\sum_{x=0}^{m\ell} \left| \frac{x}{m} - \frac{\ell}{2} \right| &= 2 \sum_{x=0}^{\frac{m\ell}{2}-1} \left( \frac{\ell}{2} - \frac{x}{m} \right) \\ &= \frac{2}{m} \sum_{x=0}^{\frac{m\ell}{2}-1} \left( \frac{m\ell}{2} - x \right) \\ &= \frac{2}{m} \sum_{x=1}^{\frac{m\ell}{2}} x \\ &= \frac{1}{4}\ell(m\ell + 2)\end{aligned}$$

since the points are evenly spaced, and thus, the optimal center coincides with $\ell/2$. Now, if we consider the same set

$$X_{k,m} = \underbrace{\{-x/m \mid x \in \{0, \dots, m(k-1)^2\}\}}_{A_{k,m}} \cup \{1, \dots, k-1\}$$

as in Observation 72, then we run into the same issues. By the above computation, $A_{k,m}$ has a 1-median cost that is in $\Omega(k^4)$, whereas the subsets that we get from dividing $[-(k-1)^2, k-1]$ into $k$ intervals of equal length each have a 1-median cost that is in $O(k^2)$. The $k$-median cost of the only clustering $\mathscr{C} = \{\{1\}, \dots, \{k-1\}, A_{k,m}\}$ that achieves a constant-factor approximation ratio with regard to sep is worse by factor of $\Omega(k)$ from that of the optimum. Conversely, if we split up $A_{k,m}$ to achieve a good $k$-median cost, then the separation value will necessarily be bad.

Thus, as before, we have shown that all clusterings can have a good (i.e., constant-factor) approximation ratio either with regard to the $k$-median objective or with regard to the $k$-separation objective, but not both simultaneously. The approximation ratio must be quite bad for at least one of them.

The issue we are facing now is that competitive algorithms for the $k$-median objective are much more involved than for $k$-center of $k$-diameter objectives, and it is not obvious whether they could be adjusted in the same manner as in the previous two sections to incorporate $\tau$-separation. While this might be possible (algorithms with the best approximation ratios are mostly Primal-Dual algorithms), we will instead make use of the several results of Lin et al. ([64]) that provide us with a certain black box approach. This independence of any specific algorithm comes at the cost of increasing the approximation by an additive factor of 2. The idea was in a certain way already present in the previous section: we first compute a minimally separated clustering and then merge its clusters in such a way that the cost stays small. Whereas we could easily guess the value (radius/diameter) of an optimal solution — there are at most $O(n^2)$ different possibilities — in the $k$-median case, this is not possible in polynomial time, so instead, we merge those clusters in relation to an approximate solution for the vanilla $k$-median problem.

**Definition 81** (The $\tau$-separated $k$-median problem)**.** Let $d_1$, $d_2$ be two metrics over a finite set $X$, $k \in \mathbb{N}$, and $\tau \geq 0$. The goal in the *$\tau$-separated $k$-median problem* is to find an assignment $\sigma : X \to C$ from $X$ to a set of $k$ centers $C = \{c_1, \ldots, c_k\}$ that minimizes

$$\mathrm{med}(\sigma, d_2) = \sum_{x \in X} d_2(x, \sigma(x))$$

and that guarantees that for all $S \in \mathscr{S}_{(X,d_1)}(\tau)$ there exists some $i \in \{1, \ldots, k\}$ with $S \subseteq \sigma^{-1}(c_i)$.

**Theorem 82.** *Given an $\alpha$-approximation algorithm for the vanilla $k$-median problem, we can find a $(2+\alpha)$-approximate solution for the $\tau$-separated $k$-median problem in polynomial time.*

This is possible because, as Lin et al. ([64]) show, the $k$-median objective admits of the $(2,1)$-nesting property.

*Proof.* Let $\sigma^* : X \to C^*$ be an optimal $\tau$-separated $k$-median assignment. From Lemma 69, we know that it is hierarchically compatible with $\mathscr{S} = \mathscr{S}_{(X,d_1)}(\tau)$, so each cluster contained in $\mathscr{S}$ has to be a subset of some cluster from $\sigma^*$. We turn the partitioning $\mathscr{S}$ into an assignment $\sigma_{\mathscr{S}} : X \to C_{\mathscr{S}}$ by picking, for each $S \in \mathscr{S}$, an arbitrary point

$$c_S \in \operatorname*{argmin}_{c \in X} \sum_{x \in S} d_2(x, c_S)$$

and defining $\sigma_{\mathscr{S}}(x) = c_S$, if $x \in S$. We now merge clusters from $\sigma_{\mathscr{S}}$ according to the assignments of their centers within an $\alpha$-approximative solution $\sigma_\alpha : X \to C_\alpha$ for the vanilla $k$-median problem on $(X, d_2)$. What we wish to show is that the cost of $\sigma = \sigma_\alpha \circ \sigma_{\mathscr{S}}$ is at most $(2 + \alpha)$ times larger than $\mathrm{med}(\sigma^*, d_2)$. To bound this cost, we first bound the distance of each point $x \in X$ to its center $\sigma(x)$ in terms of its distance to $\sigma_{\mathscr{S}}$ and $\sigma_\alpha$:

$$\begin{aligned} d_2(x, \sigma(x)) &= d_2(x, \sigma_\alpha(\sigma_{\mathscr{S}}(x))) \\ &\leq d_2(x, \sigma_{\mathscr{S}}(x)) + d_2(\sigma_{\mathscr{S}}(x), \sigma_\alpha(\sigma_{\mathscr{S}}(x))) \\ &\leq d_2(x, \sigma_{\mathscr{S}}(x)) + d_2(\sigma_{\mathscr{S}}(x), \sigma_\alpha(x)) \\ &\leq 2 \cdot d_2(x, \sigma_{\mathscr{S}}(x)) + d_2(x, \sigma_\alpha(x)) \end{aligned}$$

The first and third inequality hold due to the triangle inequality, the second, because $\sigma_\alpha$ assigns each point to the center closest to it: $\sigma_\alpha(x)$ cannot be closer to $\sigma_{\mathscr{S}}(x)$ than $\sigma_\alpha(\sigma_{\mathscr{S}})(x)$. Taking the sum over all points then gives us the following:

$$\begin{aligned} \sum_{x \in X} d_2(x, \sigma(x)) &\leq \sum_{x \in X} (2 \cdot d_2(x, \sigma_{\mathscr{S}}(x)) + d_2(x, \sigma_\alpha(x))) \\ &= 2 \cdot \mathrm{med}(\sigma_{\mathscr{S}}, d_2) + \mathrm{med}(\sigma_\alpha, d_2) \\ &\leq 2 \cdot \mathrm{med}(\sigma^*, d_2) + \delta \cdot \mathrm{med}(\sigma^*, d_2) \\ &= (2 + \delta) \cdot \mathrm{med}(\sigma^*, d_2), \end{aligned}$$

where the last inequality holds because:

1. The induced clustering of $\sigma_{\mathscr{S}}$ is a refinement of the induced clustering of $\sigma^*$ and thus cheaper than it,
2. The assignment $\sigma_\alpha$ costs at most $\alpha$ times more than the optimal assignment for the vanilla $k$-median problem, which cannot cost more than the optimal solution for the $\tau$-separated $k$-median problem. □

Similar to the case of the $k$-center objective where the value of the $k$-separation function is determined by only two of the points in $P$, we know that for the same reasons that the Pareto set for this combination consists of at most $\binom{n}{2}$ different solutions.

**Corollary 83.** *Given two metrics $d_1$ and $d_2$ over a finite set $X$, we can compute in polynomial time a $(1, 2+\alpha)$-approximate Pareto set $\mathcal{P}$ with respect to the objectives $f_1 = \mathrm{sep}$ and $f_2 = \mathrm{med}$, where $\alpha$ is the best approximation ratio of any polynomial-time algorithm for the vanilla $k$-median problem. Since the Pareto set has size at most $\binom{n}{2}$, the same holds for $\mathcal{P}$.*

Plugging in the approximation ratio $\alpha = 2.67059$ established by Cohen-Addad et al. ([76]) then directly yields a $(1, 4.67059)$-approximate Pareto set.

**Definition 84** (The $\tau$-separated $k$-means problem)**.** Let $d_1$, $d_2$ be two metrics over a finite set $X$, $k \in \mathbb{N}$, and $\tau \geq 0$. The goal in the *$\tau$-separated $k$-means problem* is to find an assignment $\sigma : X \to C$ from $X$ to a set of $k$ centers $C = \{c_1, \ldots, c_k\}$ that minimizes

$$\operatorname{med}(\sigma, d_2) = \sum_{x \in X} d_2^2(x, \sigma(x))$$

and that guarantees that for all $S \in \mathscr{S}_{(X,d_1)}(\tau)$ there exists some $i \in \{1, \ldots, k\}$ with $S \subseteq \sigma^{-1}(c_i)$.

We can combine the $k$-separation and the $k$-means objective in a similar fashion with the help of the $(8, 2)$-nesting property that Lin et al. establish for the $k$-means objective. Note that Observation 80 also holds for this combination.

**Theorem 85.** *Given an $\alpha$-approximation algorithm for the vanilla $k$-means problem, we can find a $(8 + 2\alpha)$-approximate solution for the $\tau$-separated $k$-means problem in polynomial time.*

Similarly to the result for the $k$-median combination, this one follows from the fact that, as Lin et al. show, the $k$-means objective function admits of an $(8, 2)$-nesting.

**Corollary 86.** *Given two metrics $d_1$ and $d_2$ over a finite set $X$, we can compute in polynomial time a $(1, 8 + 2\alpha)$-approximate Pareto set $\mathcal{P}$ with respect to the objectives $f_1 = \text{sep}$ and $f_2 = \text{mean}$, where $\alpha$ is the best approximation ratio of any polynomial-time algorithm for the vanilla $k$-means problem. Since the Pareto set has size at most $\binom{n}{2}$, the same holds for $\mathcal{P}$.*

Plugging in the approximation ratio $\alpha = 9 + \varepsilon$ established by Ahmadian et al. ([6]) then directly yields a $(1, 26 + 2\varepsilon)$-approximate Pareto set.

## 3.5 Combining $k$-MSR and $k$-Separation

In this section, we would like to combine the $k$-separation and the $k$-MSR objectives. In the introduction, we have seen that the latter differs quite drastically from other center-based objectives on a fundamental level. Whereas it is always optimal regarding the $k$-center, the $k$-median, or the $k$-means objectives to assign points to their closest center, the same does not hold for the $k$-MSR objective. Even on a line graph and even when the centers are chosen optimally, assigning points to their closest centers can result in a clustering whose $k$-MSR cost is thrice the cost of an optimal solution ([62]), and for more complicated metric spaces this factor grows quickly. In particular, the black box approach we used in the previous two sections is not applicable here since it relies heavily on this property. Instead, we will work through the explicit primal-dual algorithm of Buchem et al. ([17]) and show that it still works if we replace single points with preformed clusters. Before that, however, let us again quickly ascertain that these two objectives cannot be approximated well simultaneously.

**Observation 87.** The example we now provide does not turn out to be as bad as the ones given in Observation 72 or Observation 80. This is because we can often merge clusters that are close to each other without increasing the $k$-MSR cost by that much. If we were to consider the other examples, then we would quickly find that the same arguments do not hold. For example, in the $k$-diameter case, we only had to compare the clustering

$$\mathscr{C} = \{\{1\}, \ldots, \{k-1\}, A_{k,m}\}$$

with one that evenly partitions $X_{k,m}$ as a subset of $[-(k-1)^2, k-1]$ into $k$ sets of equal length $k-1$, to deduce that $\mathscr{C}$, which is the only constant-factor approximation with regard to the $k$-separation objective, is only $\Omega(k)$-approximative with regard to the $k$-diameter objective. However, regarding the $k$-MSR objective, we would have to sum up all the radii of the second partitioning and end up with clustering whose cost is also in $\Omega(k^2)$. In other words, this example does not show that $\mathscr{C}$ is a bad clustering with regard to the $k$-MSR objective. Instead, we consider a graph metric consisting of two parts:

1. a line graph consisting of $k-1$ points, where all edges have weight 1;
2. a clique consisting of $k-1$ points, where all edges have weight $\sqrt{k}$.

Each distance $d(x, y)$ on the union of both graphs is given as the weight of a shortest path between connecting $x$ and $y$. For such a path to always exist, we can add a single edge with an arbitrarily large weight somewhere between the first and second parts. Now, let us consider the only constant-factor approximation with regard to the $k$-separation objective. As before, this clustering results from merging all points in the line graph and leaving all points in the clique separate. Its separation value is $\sqrt{k}$, whereas every other clustering has to have a separation value of 1 since it has to split up the line graph. However, the $k$-MSR cost of this solution is $\lceil \frac{k-1}{2} \rceil$, which is worse by a factor of $\sqrt{k}$ than merging the clique and leaving each point in the line graph separate, as the latter has a $k$-MSR cost of exactly $\sqrt{k}$.

As in the previous chapter, instead of working with explicit assignments, we rather conceive $k$-MSR clusters as sets of points that are all at a certain distance from a given center. This is possible since the cost of a $k$-MSR cluster derives from only two of its points, as in the case of the $k$-center objective, and by resolving tiebreaks arbitrarily, we can always extract an explicit assignment from them. Let us quickly revisit the definition of a ball, which we have to adjust slightly to denote the metric with regard to which we consider it. This is necessary since we now work with two different metrics at once.

**Definition 88.** Let $(X, d)$ be a finite metric space. For a pair $(c, r) \in X \times \mathbb{R}$ let $\mathrm{B}_{(X,d)}(c,\, r) = \{x \in X \mid d(x, c) \leq r\}$ denote the set of points in $X$ that are at distance at most $r$ from $c$.

The general approach will be the same as in the previous sections: compute minimally separated clusterings and merge their clusters in a second step. As such, it makes sense to again consider something like a $\tau$-separated $k$-MSR problem.

**Definition 89** (The $\tau$-separated $k$-MSR problem)**.** Let $d_1$, $d_2$ be two metrics over a finite set $X$, $k \in \mathbb{N}$, and $\tau \geq 0$. The goal in the *$\tau$-separated $k$-MSR problem* is to find a set of at most $k$ pairs[3] $P = \{(c_1, r_1), \ldots, (c_k, r_k)\}$ that minimizes

$$\operatorname{msr}(P) = \sum_{i=1}^{k} r_i$$

and that guarantees that for all $S \in \mathscr{S}_{(X,d_1)}(\tau)$ there exists some $i \in \{1, \ldots, k\}$ with $S \subseteq \mathrm{B}_{(X,d_2)}(c_i,\, r_i)$.

By slightly adjusting the algorithm by Buchem et al. ([17]), we will be able to prove the following result, which can then, in turn, be used to construct a $(1, 3 + \varepsilon)$-Pareto approximation.

**Theorem 90.** *There exists a $(3 + \varepsilon)$-approximation algorithm for the $\tau$-separated $k$-MSR problem that runs in $n^{O(1/\varepsilon)}$ time.*

The approximation factor is exactly the same as in the original paper and thus, in this sense, tight. Before getting into the details of Buchem et al.'s primal-dual algorithm, we have to highlight one of the authors' principal observations (Lemma 92) — which is of general interest for the $k$-MSR problem, not just for their algorithm in specific.

**Definition 91.** Let $(X, d)$ be a finite metric space and $P \subset X \times \mathbb{R}$ a finite set of pairs. The *intersection graph* $G_{(X,d)}(P)$ of $P$ consists of

- vertices $V_{(X,d)}(P) = P$ and
- edges $E_{(X,d)}(P) = \{\{(c, r), (c', r')\} \mid \mathrm{B}_{(X,d)}(c,\, r) \cap \mathrm{B}_{(X,d)}(c',\, r') \neq \emptyset\}$.

[3]To exclude the possibility of the trivial solution dominating every other solution, we just assign it a separation value of $\infty$.

(This graph should not be confused with the separation graph that we introduced earlier.)

One of the most interesting properties of the $k$-MSR objective is that the cost of a solution can sometimes be reduced by merging clusters — this cannot happen with other center-based objectives. We have already seen this in the introduction. For example, whenever we have a solution where one center (say $c_i$) is contained in the ball of a different pair (i.e., $c_i \in \mathrm{B}_{(X,d)}(c_j,\, r_j)$ for some $j$), then we can close the first pair (i.e., set $r_i = 0$) and increase the radius of the other pair (i.e., set $r_j \leftarrow r_j + r_i$) without increasing the overall cost. More generally, if the centers of two clusters are at a distance of $\delta$ from each other, then we can merge them in such a way that the overall increase in cost is upper bounded by $\delta$. This analysis can be pushed much further: Buchem et al. were able to show that sets of intersecting clusters can always be covered by a single ball whose radius is at most three times the sum of the radii of the original set.

**Lemma 92** (Cf. Appendix C of [17]). *Let $(X, d)$ be a finite metric space, $P \subseteq X \times \mathbb{R}$ a finite set of pairs whose intersection graph $G_{(X,d)}(P)$ is connected, and $I \subset V_{(X,d)}(P)$ an independent set that maximizes* $\operatorname{msr}(I)$. *Then there exists a pair $(c^*, r^*) \in X \times \mathbb{R}$ such that*

1. $\bigcup_{(c,r)\in P} B_{(X,d)}(c,\, r) \subset B_{(X,d)}(c^*,\, r^*)$ *and*
2. $r^* \le 3 \sum_{(c,r)\in I} r$.

The proof of this result is dispersed over several technical and intermediary results contained in Appendix C of [17]. We will not repeat it here since it does not concern our new constraints. Instead, we will just work through their primal-dual algorithm in the remainder of this chapter and ensure that we can prepend the Single-Linkage algorithm without introducing too many problems. From now on, fix an arbitrary $\tau$-separated $k$-MSR instance $(X, d_1, d_2)$ and let $(c_1^*, r_1^*), \ldots, (c_k^*, r_k^*)$ denote the pairs of one of its optimal solutions $P^*$ in non-increasing order of their radii.

The first step is common to all primal-dual (or Langragian relaxation) approaches for the $k$-MSR problem ([23, 43, 17]) and consists of guessing the $1/\varepsilon$ largest pairs $(c_1^*, r_1^*)$, …, $(c_{1/\varepsilon}^*, r_{1/\varepsilon}^*)$ of the optimal solution, where we assume wlog. that $1/\varepsilon \in \mathbb{N}$. This can be done in $n^{O(1/\varepsilon)}$ time and leaves us in the position where we only have to approximate the remaining $k' = k - 1/\varepsilon$ pairs $(c_{1/\varepsilon}^*, r_{1/\varepsilon}^*), \ldots, (c_k^*, r_k^*)$. Of course, we can assume here that $k > 1/\varepsilon$ since we would be done otherwise. The pairs comprising the optimal solution can be of drastically different sizes, but this segmentation ensures that all leftover clusters

$$\mathscr{S}' = \{S \in \mathscr{S}_{(X,d_1)}(\tau) \mid \forall i \in \{1, \ldots, 1/\varepsilon\} \colon S \not\subseteq \mathrm{B}_{(X,d_2)}(c_i,\, r_i)\}$$

can be covered by balls of radius at most $r_{1/\varepsilon} \leq \varepsilon \cdot \mathrm{msr}(P)$. This will give us some breathing room later on. From now on, we can focus on the $\tau$-separated $k'$-msr problem for $X' = \bigcup_{S \in \mathscr{S}'} S$, which consists of all yet uncovered points.

The next step requires us to set up the corresponding primal and dual LPs. While they look quite similar to those for the vanilla $k$-MSR problem (cf. [17]), conceptually they are slightly different. Whereas Buchem et al. only have to cover one individual point at a time, we have to ensure that all points from a given cluster $S \in \mathscr{S}$ are simultaneously covered by the same ball. To formulate the LPs enumerate $\mathscr{S}' = \{S_1, \ldots, S_\ell\}$.

**Primal LP:**

$$\begin{array}{lrcll} \text{minimize} & \sum_{(c,r)} r \cdot x_{(c,r)} & & & \\ & & & & \\ \text{subject to} & \sum_{(c,r): S_j \subset \mathrm{B}_{(X,d_2)}(c,r)} x_{(c,r)} & \geq & 1 & \forall j \in \{1, \ldots, \ell\} \\ & \sum_{(c,r)} x_{(c,r)} & \leq & k' & \\ & x_{(c,r)} & \geq & 0 & \end{array}$$

Here the variables $x_{(c,r)}$ range over all $(c,r) \in X \times R$, where

$$R = \{d(x,y) \mid x, y \in X \colon d(x,y) \leq r^*_{1/\varepsilon}\}.$$

The first inequality ensures that every cluster $S_i$ is fully covered, and the second ensures that at most $k$ clusters are formed (at least fractionally). The corresponding dual LP is then as follows.

**Dual LP:**

$$\begin{array}{lrcll} \text{maximize} & \sum_j \alpha_j - \lambda k' & & & \\ & & & & \\ \text{subject to} & \sum_{j\colon S_j \subset \mathrm{B}_{(X,d_2)}(c,r)} \alpha_j & \leq & r + \lambda & \forall (c,r) \\ & \alpha_j & \geq & 0 & \forall j \in \{1, \ldots, \ell\} \\ & \lambda & \geq & 0 & \end{array}$$

As expected, the second step now consists of computing a solution for the dual LP that can almost fully pay for a solution of the original $\tau$-separated $k$-MSR problem. While this is the standard for primal-dual algorithms or other approaches via Lagrangian relaxations, Buchem et al.'s approach differs significantly from those established at an earlier point by Charikar and Panigrahy ([23]), as well as Friggstad and Jamshidian ([43]). The latter, in the spirit of Jain and Vazirani ([59]), first compute a suitable bi-point solution (which consists of a solution with at least $k$ centers and a solution with at most $k$ centers) via a binary search over the Lagrangian multiplier, and then combine them to get a good feasible solution for the $k$-MSR problem. Buchem et al., on the other hand, do not have to guess a suitable Lagrangian multiplier, as they consider $\lambda$ a variable that can be adjusted during the execution of the algorithm, which arguably simplifies the analysis of their approach, especially after proving Lemma 92.

**Definition 93.** Let $(\alpha, \lambda)$ be a solution for the dual LP and $\mu = \frac{r^*_{1/\varepsilon}}{|X|^2}$. A pair $(c, r) \in X \times R$ is *almost tight*, if the corresponding dual constraint is tight up to an additive factor of $\mu$, i.e., if

$$\sum_{j \colon S_j \subset \mathrm{B}_{(X,d_2)}(c,\, r)} \alpha_j \geq r + \lambda - \mu.$$

Starting with the trivial solution $\alpha_1 = \ldots = \alpha_\ell = 0$ and $\lambda = 0$, Buchem et al. successively increase the values in such a way that the number of connected components of the intersection graph of almost tight pairs decreases while still guaranteeing that the solution stays valid and that every point is contained in the ball of at least one almost tight pair. Roughly speaking, once they reach $k'$ connected components, they are able to derive a good solution for $k'$-msr problem on $X'$ with the help of Lemma 92.

The overall approach is the same for us, but we have to modify the invariant to ensure that the clusters $S \in \mathscr{S}'$ are *fully* contained within the ball of at least one tight pair. Now, contrary to the vanilla case, in which the trivial solution satisfies the invariant, because every point $x \in X'$ is contained within the respective ball $\mathrm{B}_{(X,d_2)}(x,\, 0)$, the trivial solution for our LP does not necessarily satisfy our modified invariant. This is because the balls of radius 0 do not necessarily cover all clusters $S \in \mathscr{S}'$. But this can easily be rectified by successively increasing the $\alpha_j$'s for clusters $S_j$ that are not yet covered at a uniform speed and end when no such cluster remains. At the end of this phase, the invariant will be satisfied, and we can continue in the same manner as Buchem et al. Let $\mathcal{T}$ denote the set of currently tight pairs. Since no $\alpha_j$ can be increased on its own, from now on, we also have to increase $\lambda$ to offset this. Even then, if there is an (almost) tight pair whose ball covers two different clusters from $\mathscr{S}'$, then we cannot increase both of them, as this would (quickly) lead to a violation of the respective constraint. Like Buchem et al. we greedily choose a maximal set of $\alpha_j$'s in such a way that no two corresponding clusters are fully contained within the ball of some pair from $\mathcal{T}$. These variables, as well as $\lambda$, are increased uniformly by some value until one additional pair that until then was not almost tight becomes tight. Due to the choice of the $\alpha_j$'s, the validity of the solution is guaranteed and the invariant satisfied. As long as the intersection graph of the updated set $\mathcal{T}$ contains strictly more than $k'$ connected components, we will increase the $\alpha_j$ values of at least $k' + 1$ clusters, which means that the objective value of the solution increases as well. By relating this increase in cost during each iteration to the upper bound $k'r^*_{1/\varepsilon}$ of the primal LP, Buchem et al. prove that the number of possible iterations is upper bounded by $O(|X|^4)$ (or rather $O(k|X|^3)$). In particular, this shows that the process can be continued until, at most, $k'$ connected components are left.

From this point onward, the remaining analysis is independent of the newly imposed constraints, so we will not cover them in detail. Just note that the resulting set of almost tight clusters $\mathcal{T}$ satisfies all the requirements that are necessary for us to conclude the proof in the same manner as Buchem et al. have done (see the proof of Lemma 2.3 in [17]). Finally, by considering all thresholds $\tau \in d_1(X \times X)$, we get the following result on the Pareto set for $f_1 = \mathrm{sep}$ and $f_2 = \mathrm{msr}$.

**Corollary 94.** *Given two metrics $d_1$ and $d_2$ over a finite set $X$, we can compute in polynomial time a $(1, 3+\varepsilon)$-approximate Pareto set $\mathcal{P}$ with respect to the objectives $f_1 = \mathrm{sep}$ and $f_2 = \mathrm{msr}$. Since the Pareto set has size at most $\binom{n}{2}$, the same holds for $\mathcal{P}$.*

# Chapter 4

# Euclidean $k$-MSR with Outliers

## 4.1 Introduction

In 2002, Bădoiu, Har-Peled and Indyk established a $(1+\varepsilon)$-approximation algorithm for the Euclidean $k$-center problem that runs in $2^{O\left(\varepsilon^{-2}k\log k\right)}$ time [12]. In particular, it yields a PTAS for constant $k$ but arbitrary dimension $d$. Our goal in this chapter is to try and use the same underlying idea to approach the much more complicated $k$-MSR problem (with outliers). We published a more general algorithm in 2023 ([35]), but in preparation for this dissertation, we discovered a significant error in the paper that was not easily resolvable. As a result, we decided to weaken the claims, and instead of providing an algorithm for general mergeable constraints, we focused only on outliers. Also noteworthy is that roughly during the same time period, Bandyapadhyay et al. ([15]) established several great results on the capacitated $k$-MSR problem (as mentioned in the introduction) that partially overlap with our own. However, since they did not deal with outliers and our approach is nonetheless different, we feel this chapter is still worth presenting. Our main result will be the following.

**Theorem 95.** *There exists a $(1+\varepsilon)$-approximation algorithm for the $k$-MSR problem with $z$ outliers that runs in $O\left(nd \cdot k^z \cdot f(k, \varepsilon^{-1})\right)$ time. Assuming that $k$ is constant and $z \in O(\log n)$, such an algorithm yields a PTAS for the $k$-MSR problem with outliers.*

Roughly speaking, the idea of Bădoiu et al. is to guess small portions — which they call coresets[1] — of an optimal solution that are nonetheless significant enough

[1]Since coresets play a much more significant role for Bădoiu et al. than becomes apparent in our presentation, even appearing in the title of their paper (“Approximate Clustering via Core-Sets”), we should spend just a little time on explaining the concept. Generally, coresets are small representations (weighted subsets) of a clustering instance that retain almost all of the structure

for the whole clustering to be reconstructed almost exactly. A coreset of some cluster $C \subset \mathbb{R}^d$ is small if its size is independent of $d$ and $|C|$; it is significant enough if its MEB is almost as large as the MEB of $C$. In other words, a coreset

1. has a large enough spread that its MEB yields an almost optimal 1-center approximation for $C$ and

2. is small enough that, once computed, we can easily guess its assignment within an optimal clustering.

However, it is not the case that coresets are computed first and that approximate solutions are extracted only subsequently. Instead, the computation of good centers and radii is intertwined with the very construction of these coresets. They are built by the successive incorporation of new points, each chosen relative to a previously established set of centers, where this set of centers is modified in turn by this incorporation. Bădoiu et al. were able to show that such coresets can be computed for all optimal clusters simultaneously while appropriately guessing their optimal assignments. This then yields their overall algorithm.

To prove Theorem 95 we will proceed as follows:

1. We start with a quick overview of a few basic notions and results from affine geometry that will help us formalize the ideas and geometrical intuitions central to the remaining chapter.

2. Next, we will consider the simple 1-center problem and show how the theory we have just introduced is utilized. The algorithms developed here form the basis for all subsequent approaches.

3. The following section then describes how the algorithms developed for the 1-center case can be combined to solve the general $k$-center problem for $k > 1$. Most of our presentation until there is a considerable expansion of papers [13, 12], where detailed descriptions are relatively sparse.

relevant for whichever center-based clustering objective is considered. Barely any significant information is lost in their construction in the sense that any solution for the coreset yields a solution with almost the same cost for the original instance and vice versa. More specifically, a coreset — let us consider $k$-center coresets as an example — is supposed to guarantee the following: for every set of $k$ centers, the maximum radius induced by those centers for the coreset differs from the maximum radius induced for the original input by at most an $\varepsilon$-fraction. Solving the $k$-center problem on the coreset then yields a good $k$-center solution for the original input. This requires that the cost really is approximated for all possible sets of $k$ centers and not just for the optimal center set since this set is not known in advance. While one could establish some continuity between this more widely employed notion of coresets and the one appearing in [12], there are also significant differences. Regardless, we will not outline them, partially because Bădoiu et al. do not provide a rigorous definition in their paper (this is remedied in later research), but also because it is not that important for us.

4. Finally, we will move on to the $k$-MSR objective and discuss the problems that the approach developed so far faces and then provide modifications to circumvent them. The resulting algorithm still works, even when we introduce outliers.

## 4.2 Basic Affine Geometry

**Definition 96.** For a finite set $X \subset \mathbb{R}^d$ let

$$\zeta(X) = \underset{z \in \mathbb{R}^d}{\operatorname{argmin}} \max_{x \in X} \|z - x\|$$

denote the optimal Euclidean 1-center solution and

$$\rho(X) = \max_{x \in X} \|x - \zeta(X)\|$$

the induced radius.

In contrast to the previous chapters, which were concerned with abstract metric spaces only, $\mathbb{R}^d$ carries additional geometric information: there are not only distances but also angles.[2] Moving forward, we will thus make use of two different interpretations of elements of $\mathbb{R}^d$: as points and as vectors. The former interpretation encapsulates position; the latter encapsulates orientation. Combining both interpretations yields the notion of affine spaces.

**Definition 97.** An *affine space* is a tuple $(A, V, +)$ consisting of a set $A$, a vector space $V$ and an action $+ : A \times V \to A$, such that the following axioms are satisfied:

1. $a + 0 = a$ for all $a \in A$,
2. $(a + v) + w = a + (v + w)$ for all $a \in A$ and $v, w \in V$,
3. $V \to A, v \mapsto a + v$ is a bijection for every $a \in A$. Equivalently, we could require that for every pair $a, b \in A$, a unique vector, denoted $b - a \in V$, exists, such that $a + (b - a) = b$.

Principally, affine spaces allow us to forget or rather displace the origin of vectors. Indeed, any point $a_0 \in A$ can serve as the origin by considering the set $\{a - a_0 \mid a \in A\}$ which can be identified with $V$ via the bijection $A \to V, a \mapsto a - a_0$ from Axiom 3. Note that, in our case, $A = V = \mathbb{R}^d$ just acts on itself via translations. Since the underlying sets of $A$ and $V$ are thus the same, we cannot do much to distinguish them notationally; sometimes, the same element has to be

[2]For a more comprehensive introduction see [45], for example.

interpreted both as a vector and as a point. However, it should always be apparent from the context with which interpretation we are working.

The geometric properties of Euclidean space relevant to us are all encapsulated in the inner product

$$\langle \cdot, \cdot \rangle : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}, (x, y) \mapsto x^\top y.$$

It relates to distances and angles in the following way:

1. The squared distance between two points $x, y \in \mathbb{R}^d$ is given by

$$\|x - y\|^2 = \langle x - y, \, x - y \rangle.$$

2. If $v, w \in \mathbb{R}^d$ are non-zero vectors and $\theta$ the angle between them, then

$$\|v\| \, \|w\| \cos\theta = \langle v, \, w \rangle.$$

The central geometric insight we want to establish in the following paragraphs concerns the distribution of points located on the boundary of the MEB of a finite set $X \subset \mathbb{R}^d$. No matter how the MEB is split into two equal (closed) halves, both sides have to contain a point from $X$ that is positioned exactly on the boundary of the MEB. If such a point lacks on one side, then we could move the center of the MEB further toward the other side and reduce the induced radius. However, that would contradict the assumption that the original ball was an MEB. This geometrical insight is crucial for all algorithms developed in this chapter. To properly formalize this gesture of splitting the MEB (or rather, all of $\mathbb{R}^d$) into two, we have to introduce a few basic notions from affine geometry.

**Definition 98.** Let $(A, V, +)$ be an affine space. A subset $B \subseteq A$ is an *affine subspace* of $(A, V, +)$, if the set of induced translations $\{b - x \mid b \in B\}$ forms a linear subspace of $V$ for every $x \in B$.

In other words, for $B$ to be an affine subspace, it must be parallel to a linear subspace $W$ of $V$ in the sense that $B = x + W$ for some $x \in B$. (The subspace $W$ would be equal to the set of translations of $B$ relative to $x$.) Since we are dealing with Euclidean space, another way of defining affine subspaces would be to require that $B$ be closed under affine combinations, i.e., that $b + \lambda(b' - b) \in B$ for all $b, b' \in B$ and $\lambda \in \mathbb{R}$.

**Definition 99.** A subset $A \subset \mathbb{R}^d$ is an affine subspace *of dimension $\ell$*, if $A$ is parallel to an $\ell$-dimensional subspace of $\mathbb{R}^d$. In the special case that $\ell = d - 1$, we call $A$ a *hyperplane*.

Any hyperplane $H$ cuts $\mathbb{R}^d$ into two and thus plays a role analogous to planes in $\mathbb{R}^3$. To see this, fix any point $z \in H$ and consider the parallel linear subspace $V \subset \mathbb{R}^d$ with $H = z + V$. Since $H$ is a hyperplane, $V$ has to have dimension $d - 1$ and its orthogonal complement

$$V^\perp = \{w \in \mathbb{R}^d \mid \forall v \in V \colon \langle v,\, w\rangle = 0\}$$

has to have dimension 1. If we pick any $x \in \mathbb{R}^d$ different from $z$, but such that $x - z \in V^\perp$, then

$$\begin{aligned} H &= z + V \\ &= \{z + v \mid v \in \mathbb{R}^d \colon \langle v,\, x - z\rangle = 0\} \\ &= \{p \in \mathbb{R}^d \mid \langle p - z,\, x - z\rangle = 0\} \end{aligned}$$

yields another equivalent definition of hyperplanes.

**Corollary 100.** *A subset $H \subset \mathbb{R}^d$ is a hyperplane if and only if there exist two different points $z, x \in \mathbb{R}^d$, such that*

$$H = H_{z,x} = \{p \in \mathbb{R}^d \mid \langle p - z,\, x - z\rangle = 0\}.$$

The displacement of the origin is quite obvious here: if we interpret elements as vectors starting from $z$, then $H_{z,x}$ contains all those that are orthogonal to $x$. In other words, it comprises all those points $p$ for which the triangle spanned by $x$, $z$, and $p$ has a right angle at $z$. From this point of view it becomes also clear that $H_{z,x}$ necessarily splits up $\mathbb{R}^d$ into two disjoint (convex) sets:

$$\mathbb{R}^d \setminus H_{z,x} = \{p \in \mathbb{R}^d \mid \langle p - z,\, x - z\rangle > 0\} \cup \{p \in \mathbb{R}^d \mid \langle p - z,\, x - z\rangle < 0\}.$$

The left set contains all points $p$ whose angle with $x$ relative to $z$ is strictly acute, while the one on the right continues all those whose angle is strictly obtuse.

**Definition 101.** Let $x, z \in \mathbb{R}^d$ be two different points and $H_{z,x}$ the associated hyperplane centered at $z$ and orthogonal to $x - z$. Then

$$H^+_{z,x} = \{p \in \mathbb{R}^d \mid \langle p - z,\, x - z\rangle \geq 0\}$$

and

$$H^-_{z,x} = \{p \in \mathbb{R}^d \mid \langle p - z,\, x - z\rangle \leq 0\}$$

are the (closed) *half-spaces* induced by $H_{z,x}$.

With these definitions in place, we can now analyze the distribution of points on the boundary of MEBs.

**Lemma 102** (Cf. Lemma 2.2 of [12]). *Let $X \subset \mathbb{R}^d$ be a finite set and $p \in \mathbb{R}^d$ an arbitrary point that is different from $\zeta(X)$. Then there exists a point $x \in H^-_{\zeta(X),p} \cap X$ with $\|x - \zeta(X)\| = \rho(X)$.*

*Proof.* Notice that $p \in H^+_{\zeta(X),p}$, since $\langle p - z, p - z\rangle = \|p - z\|^2 > 0$. So, visually, this statement is quite intuitive: if $H^-_{\zeta(X),p}$ would not contain a point from $X$ that is exactly on the boundary of the MEB, then we could shift $\zeta(X)$ slightly towards $p$ and cover $X$ by a ball with a radius strictly smaller than $\rho(X)$.

1. Since $H^-_{\zeta(X),p}$ is closed, its complement $\overline{H^-_{\zeta(X),p}}$ must be open. In particular, there has to exist some positive error factor $\delta > 0$ that is nonetheless small enough to guarantee that $\delta < \|x - y\|$ for all $x \in \overline{H^-_{\zeta(X),p}} \cap X$ and $y \in H^-_{\zeta(X),p}$.
2. At the same time, since $H^-_{\zeta(X),p} \cap X$ does not contain any point that is at a distance of $\rho(X)$ from $\zeta(X)$, there must exist some other positive error factor $\delta' > 0$ that is small enough to guarantee that $\rho(X) - \delta' > \|x - \zeta(X)\|$ for all $x \in X \cap H^-_{\zeta(X),p}$.

Both observations hold at once for $\varepsilon = \min\{\delta, \delta'\}$. Now, if we consider the distances of points to the shifted center $\zeta(X) + \frac{\varepsilon}{2}(p - \zeta(X))$, then we can see that for all points in $\overline{H^-_{\zeta(X),p}} \cap X$ they are still at most $\rho(X) - \frac{\varepsilon}{2}$, while for all points in $H^-_{\zeta(X),p} \cap X$ they have been reduced by some positive amount. However, this cannot happen since $\rho(X)$ is the smallest radius of any ball covering $X$. □

The following two results will help us derive a useful lower bound on the maximal distance from any given point $x$ to a point inside an MEB.

**Lemma 103.** *Let $x, z \in \mathbb{R}^d$ be two different points and $H_{x,z}$ the associated hyperplane. Then*

$$\|p - z\|^2 \geq \|p - x\|^2 + \|z - x\|^2$$

*for all $p \in H^-_{x,z}$ and*

$$\|p - z\|^2 \leq \|p - x\|^2 + \|z - x\|^2$$

*for all $p \in H^+_{x,z}$.*

*Proof.* This lemma is just a rephrasing of the Law of Cosines. The triangle spanned by $x$, $z$ and $p$ has an acute angle at $z$ if $p \in H^-_{x,z}$ and an obtuse angle for $p \in H^+_{x,z}$. However, we can also prove it directly with the theory we have set up. Since $p - z = (p - x) - (z - x)$, we can apply the bilinearity of the inner product to get that

$$\|p - z\|^2 = \|p - x\|^2 + \|z - x\|^2 - 2\langle p - x, z - x\rangle.$$

The last term is negative for all $p \in \mathbb{R}^d \setminus H^+_{x,z}$ (because $\langle p,\, z - x\rangle < \langle x,\, z - x\rangle$ and thus $\langle p - x,\, z - x\rangle < 0$) and positive for all all $p \in \mathbb{R}^d \setminus H^- x, z$. This yields the claimed inequalities. $\square$

Now follows the geometrical insight that will be exploited throughout this chapter. It has been established by Bădoiu et al. in [13] but was basically already in use in [12].

**Corollary 104** (Cf. Lemma 2.1 of [13])**.** *Let $X \subset \mathbb{R}^d$ be a finite set. Then for any point $p \in \mathbb{R}^d$ there exists a point $x \in X$ with*

$$\|p - x\| \geq \sqrt{\|p - \zeta\,(X)\|^2 + \rho\,(X)^2}.$$

*Proof.* This corollary is a straightforward combination of previously established lemmata. First, we apply Lemma 102 to $H^-_{\zeta(X),p}$ to infer the existence of some point $x \in X \cap H^-_{\zeta(X),p}$ at distance exactly $\rho\,(X)$ from $\zeta\,(X)$. Since $p \in H^-_{\zeta(X),x}$ (the angle at $\zeta\,(X)$ is obtuse), we apply Lemma 103 to deduce that

$$\|x - p\| \geq \sqrt{\|x - \zeta\,(X)\|^2 + \|x - \zeta\,(X)\|^2} = \sqrt{\|z - \zeta\,(X)\|^2 + \rho\,(X)^2}. \qquad \square$$

With this, we have established all the affine geometry we need moving forward.

## 4.3 Approximating the 1-Center problem

Let us now consider the 1-center problem. There are two ways in which we could approximate an optimal solution: we could approximate the optimal center directly (which means that we would find a center close to it), or we could approximate the optimal radius (which means that we would find a center whose induced radius is close to it).

**Definition 105.** Let $X \subset \mathbb{R}^d$ be a finite set, and $0 < \varepsilon < 1$. A point $c \in \mathbb{R}^d$ is

- an *$\varepsilon$-approximate* center for $X$, if $\|c - x\| \leq (1 + \varepsilon)\rho\,(X)$ for all $x \in X$
- an *$\varepsilon$-close* center for $X$, if $\|c - \zeta\,(X)\| \leq \varepsilon\rho\,(X)$.

The latter is a (strictly[3]) stronger requirement, as the following lemma shows. Although $\varepsilon$-approximate centers already yield $(1 + \varepsilon)$-approximate solutions by definition, it is not much more difficult to compute $\varepsilon$-close centers (Algorithm 5), and this sometimes comes in handy.

[3] We will not prove this claim. However, it should be intuitive that, depending on the distribution of points on the boundary of the MEB, an $\varepsilon$-approximate center can be placed further from the optimal 1-center.

**Lemma 106.** *Let $X \subset \mathbb{R}^d$ be a finite set and $c \in \mathbb{R}^d$ an $\varepsilon$-close center for $X$. Then $c$ is also an $\varepsilon$-approximate center for $X$.*

*Proof.* This lemma is an immediate consequence of applying the triangle inequality. For any $x \in X$ we have

$$\|x - c\| \leq \|x - \zeta(X)\| + \|\zeta(X) - c\| \leq \rho(X) + \varepsilon\rho(X) = (1+\varepsilon)\rho(X). \qquad \square$$

**Algorithm 5:** $\varepsilon$-Close Center

```
    Input  : A finite set $X \subset \mathbb{R}^d$, an error-parameter $0 < \varepsilon < 1$.
    Output: An $\varepsilon$-close center for $X$.
1   $c \leftarrow$ an arbitrary point from $X$;
2   for $i = 1, \ldots, \lceil \varepsilon^{-2} \rceil$ do
3       $p \leftarrow \mathrm{argmax}_{x \in X} \|x - c\|$;
4       $c \leftarrow c + \frac{1}{i+1}(p - c)$;
5   end
6   return $c$;
```

Bădoiu and Clarkson [13] have devised a very simple algorithm (see Algorithm 5), that computes an $\varepsilon$-close center for any finite set $X \subset \mathbb{R}^d$ in $O\left(|X| d\varepsilon^{-2}\right)$ time. And, although this algorithm cannot be used directly to find $\varepsilon$-close centers for several clusters at once (that is, it does not generalize to the $k$-center problem for $k > 1$), we will nonetheless discuss it in detail for the following reasons:

1. working out why the algorithm cannot be used to compute $\varepsilon$-close centers for several clusters simultaneously is quite insightful;
2. we will use it as a subroutine later on (it can easily replaced, though);
3. it provides an opportunity to show how the preliminary geometric insights from the previous section are utilized.

**Lemma 107** (Cf. Claim 3.1 in [13])**.** *Let $X \subset \mathbb{R}^d$ be a finite set consisting of $n$ points and $0 < \varepsilon < 1$ an arbitrary error-parameter. Then Algorithm 5 returns an $\varepsilon$-close center in $O\left(nd\varepsilon^{-2}\right)$ time.*

*Proof.* The running time is obvious. To prove the correctness, let $c^{(i)}$ refer to variable $c$ right before the $i$-th iteration of the for-loop, and $p^{(i)}$ to the next chosen point, meaning that $c^{(i+1)} = c^{(i)} + \frac{1}{i+1}\left(p^{(i)} - c^{(i)}\right)$. Bădoiu and Clarkson prove that $\left\|\zeta(X) - c^{(i)}\right\| \leq \frac{\rho(X)}{\sqrt{i}}$ for all $i$ by induction. If that's the case, then after iteration

$i = \lceil \varepsilon^{-2} \rceil$ we have that $\left\| \zeta(X) - c^{\left(\varepsilon^{-2}\right)} \right\| \leq \frac{\rho(X)}{\varepsilon}$, proving that the algorithm indeed returns an $\varepsilon$-close center.

For the base case $i = 1$ the hypothesis is obviously true: $\left\| \zeta(X) - c^{(i)} \right\| \leq \rho(X) = \frac{\rho(X)}{1}$, just because $c^{(1)}$ was chosen from $X$. For the induction step, assume that the hypothesis holds for some arbitrary $i$. If $c^{(i)}$ already coincides with the optimal center, that is, if $c^{(i)} = \zeta(X)$, then

$$\begin{aligned} \left\| \zeta(X) - c^{(i+1)} \right\| &= \left\| \zeta(X) - \left( \zeta(X) + \frac{1}{i+1} \left( p^{(i)} - \zeta(X) \right) \right) \right\| \\ &= \frac{1}{i+1} \left\| p^{(i)} - \zeta(X) \right\| \\ &\leq \frac{\rho(X)}{i+1} \leq \frac{\rho(X)}{\sqrt{i+1}} \end{aligned}$$

as claimed. Otherwise, we can consider the hyperplace $H_{\zeta(X),c^{(i)}}$ and Lemma 102 yields the existence of a point $x \in X$ with $\left\| x - c^{(i)} \right\|^2 \geq \left\| c^{(i)} - \zeta(X) \right\|^2 + \rho(X)^2$. Since $p^{(i)}$ was chosen such that it maximizes the distance to $c^{(i)}$ and since

$$\left\| x' - c^{(i)} \right\|^2 \leq \left\| c^{(i)} - \zeta(X) \right\|^2 + \rho(X)^2$$

for all points $x' \in X \cap H^{+}_{\zeta(X),c^{(i)}}$, we know that $p^{(i)} \in X \cap H^{-}_{\zeta(X),c^{(i)}}$.

There are now two possible cases that deserve separate attention: whether $c^{(i+1)} \in H^{-}_{\zeta(X),c^{(i)}}$ or whether $c^{(i+1)} \in H^{+}_{\zeta(X),c^{(i)}}$. We will only consider the first case, as the other one is more difficult to prove rigorously. (The broad idea can be found in Bădoiu and Clarkson's paper [13].) Recall that $c^{(i+1)}$ is a convex combination of $c^{(i)}$ and $p^{(i)}$: the algorithm places $c^{(i+1)}$ exactly $\frac{1}{i+1}$ of the way along the line going from $c^{(i)}$ to $p^{(i)}$. If we displace $c^{(i)}$ by some vector $v$, then $c^{(i+1)}$ is displaced by $\left(1 - \frac{1}{i+1}\right) v$. In particular, if $c^{(i+1)} \in H^{-}_{\zeta(X),c^{(i)}}$, then by moving $c^{(i)}$ closer to $\zeta(X)$, we necessarily push $c^{(i+1)}$ away from $\zeta(X)$. That means that the distance between $c^{(i+1)}$ and $\zeta(X)$ is maximized for $c^{(i)} = \zeta(X)$ and we can conclude, as we have done earlier, that $\left\| c^{(i+1)} - \zeta(X) \right\| \leq \frac{\rho(X)}{\sqrt{i+1}}$. □

Now is an excellent moment to discuss how Bădoiu et al. move from the 1-center problem to the $k$-center problem. Roughly speaking, the idea is to guess an optimal $k$-center solution and run a 1-center algorithm for each of its clusters. Of course, guessing a whole solution cannot be done in a reasonable amount of time, so we must be more precise in our description. The trick is that Algorithm 5 only considers $\varepsilon^{-2}$ many points from the input, so if we were only to guess their assignment within the optimal solution, we would incur a running time, whose only exponential dependencies are $\varepsilon^{-1}$ and $k$. However, we cannot do this with

Algorithm 5. The reason is simply that we do not know in advance which points from the respective clusters maximize the distance to the already established centers, and, as we have seen, this choice is crucial for the proper functioning of the algorithm. This would re-introduce the original problem with the running and again force us to guess the proper assignments of basically all points. The next algorithm we will discuss does not have that same weakness, so we can use it to solve the $k$-center problem for $k > 1$.

**Algorithm 6:** $\varepsilon$-Approximate Center (1st Variant)

```
Input  : A point set $X$, an error-parameter $0 < \varepsilon < 1$.
Output: A pair $(c, r)$ consisting of an $\varepsilon$-approximate center $c$ of $X$ and
        the radius $r$ it induces.
1 $x \leftarrow$ some arbitrary point from $X$;
2 $y \leftarrow \operatorname{argmax}_{p \in X} \|x - p\|$;
3 $S \leftarrow \{x, y\}$;
4 while there exists a point $p \in X \setminus B(\zeta(S), (1+\varepsilon)\rho(S))$ do
5     $S \leftarrow S \cup \{p\}$;
6 end
7 return $(\zeta(S), \rho(S))$;
```

Although the algorithm also tries to maximize the distance to a previously selected point, it only happens once, and we have a lot more leeway doing so; the point does not actually have to maximize the distance to the previously selected center. The only penalty is that the computed coreset $S$ increases in size, which in turn also increases the running time.

**Lemma 108** (Cf. Lemma 2.3 of [12]). *For every finite set $X \subset \mathbb{R}^d$ Algorithm 6 returns an $\varepsilon$-approximate center after at most $\frac{48}{\varepsilon^2}$ iterations.*

Let us first prove that Algorithm 6 terminates after $O(\varepsilon^{-2})$ many iterations. This is because the MEB of the points collected in $S$ grows sufficiently with every newly added point.

**Lemma 109** (Cf. Lemma 2.3 in [12]). *Let $X \subset \mathbb{R}^d$ be a finite set, $0 < \varepsilon \leq 1$, and $p \in \mathbb{R}^d$ a point with $\|p - \zeta(X)\| \geq (1+\varepsilon)\rho(X)$. Then*

$$\rho(X \cup \{p\}) \geq \left(1 + \frac{\varepsilon^2}{4(1+\sqrt{2})}\right)\rho(X).$$

*Proof.* The proof proceeds in the form of a case distinction over the distance between the old center $\zeta(X)$ and the new center $\zeta(X \cup \{p\})$. If this distance is small, then $p$, which is far from $\zeta(X)$, is also far from $\zeta(X \cup \{p\})$, and the new radius $\rho(X \cup \{p\})$ would be sufficiently large. On the other hand, if the distance $\|\zeta(X \cup \{p\}) - \zeta(X)\|$ is already large, then we can apply Corollary 104 to find a point $x \in X$ lying at the other end of the MEB. This point is far from $\zeta(X \cup \{p\})$, and the new radius would be sufficient again.

1. Suppose that $\|\zeta(X) - \zeta(X \cup \{p\})\| < \frac{\varepsilon}{2}\rho(X)$. In this case, the triangle inequality implies that

$$\begin{aligned} \rho(X \cup \{p\}) &\geq \|\zeta(X \cup \{p\}) - p\| \\ &\geq \|\zeta(X) - p\| - \|\zeta(X \cup \{p\}) - \zeta(X)\| \\ &\geq (1+\varepsilon)\rho(X) - \frac{\varepsilon}{2}\rho(X) \\ &\geq \left(1 + \frac{\varepsilon}{2}\right)\rho(X). \end{aligned}$$

   Since $\varepsilon \in [0,1]$ this last term is lower bounded by $\left(1 + \frac{\varepsilon^2}{4(1+\sqrt{2})}\right)\rho(X)$.

2. Suppose now that $\|\zeta(X) - \zeta(X \cup \{p\})\| \geq \frac{\varepsilon}{2}$. Then Corollary 104 implies that there exists a point $x \in X$ with

$$\begin{aligned} \|\zeta(X \cup \{p\}) - x\| &\geq \sqrt{\|\zeta(X \cup \{p\}) - \zeta(X)\|^2 + \rho(X)^2} \\ &\geq \sqrt{\left(1 + \frac{\varepsilon^2}{4}\right)\rho(X)^2} \\ &\geq \sqrt{\left(1 + \frac{\varepsilon^2}{4}\right)}\rho(X) \\ &\geq \left(1 + \frac{\varepsilon^2}{4(1+\sqrt{2})}\right)\rho(X), \end{aligned}$$

   where the last inequality follows from the straightforward observation that $\sqrt{1+r} \geq 1 + \frac{r}{1+\sqrt{2}}$ for all $r \in [0,1]$.

□

By showing that the distance between the first two selected points $x$ and $y$ in Algorithm 6 is sufficiently large, we can derive the desired upper bound on the number of iterations. However, before we get to that, we will prove a simple (and quite obvious) result that concerns MEBs of subsets.

**Lemma 110.** *Let $X \subset \mathbb{R}^d$ be a finite set and $Y \subset X$ an arbitrary subset. Then $\rho(Y) \leq \rho(X)$.*

*Proof.* Apply Corollary 104 to $\zeta(X)$ in relation to the MEB of $Y$, which yields the existence of a point $y \in Y \subset X$ with

$$\rho(X) \geq \|\zeta(X) - y\| \geq \sqrt{\|\zeta(X) - \zeta(Y)\|^2 + \rho(Y)^2} \geq \rho(Y).$$

□

Unsurprisingly, the radius of an MEB of a subset is upper bound by the radius of the MEB of the overall set. We are now able to prove Lemma 108.

*Proof of Lemma 108.* Denote the state of variable $S$ right before the $i$-th iteration by $S^{(i)}$ (in particular, this means that $S^{(1)} = \{x, y\}$ consists only of the points chosen in Line 1 and 2). The fact that $y$ was chosen such that it maximizes the distance to $x$ immediately implies that

$$2\rho(S^{(1)}) = \|x - y\| \geq \frac{\operatorname{diam}(X)}{2}.$$

Indeed, the contrary assumption yields a contradiction: if every point $z \in X$ would be at a distance of $\|x - z\| \leq \|x - y\| < \frac{\operatorname{diam}(X)}{2}$ from $x$, then we could cover $X$ by a ball $\mathrm{B}(x, \|x - y\|)$ whose diameter is strictly smaller than $\operatorname{diam}(X)$. This gives us a lower bound on the initial radius of the MEB of $S$. Combining this with the lower bound on the increase in radius with every iteration we have established in Lemma 109 then yields the claim.

Together, we thus know that the number of iterations is upper bounded by the smallest $t \in \mathbb{N}$ for which

$$\left(1 + \frac{\varepsilon^2}{4(1+\sqrt{2})}\right)^t \frac{\operatorname{diam} X}{4} \geq \operatorname{diam} X.$$

We could solve this exactly, but a weaker bound is sufficient here. From the observations we have established so far, we know that the radius always increases by an additive factor of

$$\frac{\varepsilon^2}{4(1+\sqrt{2})} \frac{\operatorname{diam}(X)}{4} = \frac{\varepsilon^2}{16(1+\sqrt{2})} \operatorname{diam}(X)$$

with every iteration. Clearly, this can happen at most $\frac{16(1+\sqrt{2})}{\varepsilon^2} \leq \frac{48}{\varepsilon^2}$ many times.

To prove correctness note that $\rho\left(S^{(i)}\right) \leq \rho(X)$ for all $i$ by Lemma 110. Once the algorithm terminates, every point $x \in X$ is thus within a distance of $(1 + \varepsilon)\rho\left(S^{(i)}\right) \leq (1 + \varepsilon)\rho(X)$ from $\zeta\left(S^{(i)}\right)$. In other words, it is an $\varepsilon$-approximate center for $X$. □

We have only provided an upper bound on the number of iterations and not an actual running time because computing MEBs is quite costly. In the following, we will adjust the algorithm to work with $\varepsilon$-close centers instead of optimal ones to circumvent an exponential dependency on $d$ in the running time. Bădoiu et al. have only mentioned this in passing, without providing much or any detail. Sadly, our modification increases the size of the coreset from $O\left(\varepsilon^{-2}\right)$ to $O\left(\varepsilon^{-4}\right)$ and impacts the running accordingly.

**Algorithm 7:** Approximate Center (2nd Variant)

```
   Input  : A point set $X$, an error-parameter $0 < \varepsilon < 1$.
   Output: A pair $(c, r)$ consisting of an $\varepsilon$-approximate center $c$ of $X$ and
           the radius $r$ it induces.
 1 $\delta \leftarrow \frac{\varepsilon^2}{8}$;
 2 $\gamma \leftarrow 1 + \delta + 2\sqrt{\delta}$;
 3 $x \leftarrow$ some arbitrary point from $X$;
 4 $y \leftarrow \operatorname{argmax}_{p \in X} \|x - p\|$;
 5 $S \leftarrow \{x, y\}$;
 6 $(c, r) \leftarrow \left(\frac{1}{2}(x + y), \frac{1}{2}\|x - y\|\right)$;
 7 while there exists a point $p \in X \setminus B(c, (1+\gamma)r)$ do
 8     $S \leftarrow S \cup \{p\}$;
 9     $(c, r) \leftarrow \delta$-close center of $S$ with induced radius $r$;
10 end
11 return $(c, r)$;
```

As a quick side note: the variable $S$ in Algorithm 7 is only necessary for the analysis of the algorithm. It can be removed without introducing any problems.

**Lemma 111.** *For every finite set $X \subset \mathbb{R}^d$ Algorithm 7 returns an $\varepsilon$-approximate center after $O(\varepsilon^{-4})$ iterations and thus has a runtime of $O\left(ndf\left(\varepsilon^{-1}\right)\right)$ for some polynomial $f$.*

*Proof.* Again, let $S^{(i)}$ denote the state of variable $S$ right before the $i$th iteration and $p^{(i)}$ the point selected during iteration $i$. Similarly, let $c^{(i)}$ and $r^{(i)}$ denote the approximate centers and radii computed during this iteration. We will again establish that $\rho\left(S\right)$ grows significantly with each iteration. Since nothing has changed in the construction of $S^{(1)}$, we can conclude the proof in the same manner as in the proof of Lemma 108. Lower bounding the increase in radius is slightly more complicated because points are now selected relative to an approximate and not optimal center. Points that are at first sight sufficiently far away from this approximate center might still be too close to the optimal center. To offset this, we have to change the increase in radius in Line 7 by the multiplicative factor $1+\gamma$. As we will see, this ensures that

$$\mathrm{B}\big(\zeta\left(S^{(i)}\right), (1+\delta)\rho\left(S^{(i)}\right)\big) \subset \mathrm{B}\big(c^{(i)}, (1+\gamma)r^{(i)}\big),$$

allowing us to apply Lemma 109.

Since $p^{(i)}$ was chosen relative to $c^{(i-1)}$ and not $\zeta\left(S^{(i-1)}\right)$, we have to pass through the former to establish any relation between $\zeta\left(S^{(i-1)}\right)$ and $\zeta\left(S^{(i)}\right)$. First, we show that $c^{(i-1)}$ and $\zeta\left(S^{(i-1)}\right)$ cannot be too far apart from each other. From Corollary 104 we know that there exists a point $z \in X$, such that

$$\left\|c^{(i-1)} - z\right\|^2 \geq \left\|c^{(i-1)} - \zeta\left(S^{(i-1)}\right)\right\|^2 + \rho\left(S^{(i-1)}\right)^2.$$

Since $c^{(i-1)}$ is an $\delta$-approximate center for $S^{(i-1)}$ we get that

$$\begin{aligned}(1+\delta)^2 \cdot \rho\left(S^{(i-1)}\right)^2 &\geq \left(r^{(i-1)}\right)^2 \\ &\geq \left\|z - c^{(i-1)}\right\|^2 \\ &\geq \rho\left(S^{(i-1)}\right)^2 + \left\|c^{(i-1)} - \zeta\left(S^{(i-1)}\right)\right\|^2,\end{aligned}$$

which in turn implies that

$$\begin{aligned}\left\|c^{(i-1)} - \zeta\left(S^{(i-1)}\right)\right\| &\leq \sqrt{\delta(2+\delta)}\rho\left(S^{(i-1)}\right) \\ &\leq \sqrt{3\delta}\rho\left(S^{(i-1)}\right) \\ &\leq 2\sqrt{\delta}\rho\left(S^{(i-1)}\right).\end{aligned}$$

We are now able to carry out the same case distinction as before. First, if

$$\left\|\zeta\left(S^{(i-1)}\right) - \zeta\left(S^{(i)}\right)\right\| < \frac{\delta}{2}\rho\left(S^{(i-1)}\right),$$

then using the triangle inequality twice yields

$$\begin{aligned}\rho\left(S^{(i)}\right) &\geq \left\|p^{(i)} - \zeta\left(S^{(i)}\right)\right\| \\ &\geq \left\|p^{(i)} - \zeta\left(S^{(i-1)}\right)\right\| - \left\|\zeta\left(S^{(i-1)}\right) - \zeta\left(S^{(i)}\right)\right\| \\ &\geq \left\|p^{(i)} - c^{(i-1)}\right\| - \left\|c^{(i-1)} - \zeta\left(S^{(i-1)}\right)\right\| - \left\|\zeta\left(S^{(i-1)}\right) - \zeta\left(S^{(i)}\right)\right\| \\ &\geq \gamma r^{(i-1)} - 2\sqrt{\delta}\rho\left(S^{(i-1)}\right) - \frac{\delta}{2}\rho\left(S^{(i-1)}\right) \\ &\geq \left(\gamma - 2\sqrt{\delta} - \frac{\delta}{2}\right)\rho\left(S^{(i-1)}\right) \\ &= \left(1 + \frac{\delta}{2}\right)\rho\left(S^{(i-1)}\right) \\ &\geq \left(1 + \frac{\delta^2}{4(1+\sqrt{2})}\right)\rho\left(S^{(i-1)}\right)\end{aligned}$$

and we are done. (Recall that we precisely set $\gamma = 1 + \delta + 2\sqrt{\delta}$.) Next, consider the case where $\left\|\zeta\left(S^{(i-1)}\right) - \zeta\left(S^{(i)}\right)\right\| \geq \frac{\delta}{2}\rho\left(S^{(i-1)}\right)$. Then Corollary 104 yields the existence of a point $z \in S^{(i-1)}$, such that

$$\begin{aligned}
\rho\left(S^{(i)}\right) &\geq \left\|\zeta\left(S^{(i)}\right) - z\right\| \\
&\geq \sqrt{\rho\left(S^{(i-1)}\right)^2 + \left\|\zeta\left(S^{(i-1)}\right) - \zeta\left(S^{(i)}\right)\right\|^2} \\
&\geq \sqrt{\rho\left(S^{(i-1)}\right)^2 + \frac{\delta^2}{4}\left(\rho\left(S^{(i-1)}\right)\right)^2} \\
&\geq \sqrt{\left(1 + \frac{\delta^2}{4}\right)}\,\rho\left(S^{(i-1)}\right) \\
&\geq \left(1 + \frac{\delta^2}{4(1+\sqrt{2})}\right)\rho\left(S^{(i-1)}\right)
\end{aligned}$$

for $0 < \delta < 1$. As before, the number of iterations is thus upper bounded by $\frac{48}{\delta^2} = O\left(\varepsilon^{-4}\right)$. Since we used Algorithm 5 to compute $\delta$-close centers for the coresets, the running time follows from Lemma 107.

We still have to make sure that the resulting point is an $\varepsilon$-approximate center for $X$. Once the algorithm terminates $\|x - c\| \leq (1+\gamma)r$ for all $x \in X$. Since $c$ is an $\varepsilon$-close center for $S$ we know that $r \leq (1+\delta)\rho(X)$, so every point is at a distance from $c$ by at most $(1 + \gamma(1+\delta) + \delta)\rho(X)$. All that's left now, is to upper bound $\gamma(1+\delta)+\delta$: since $\delta = \frac{\varepsilon^2}{8}$ we get that

$$\begin{aligned}
\gamma(1+\delta)+\delta &= (1+\delta+2\sqrt{\delta})(1+\delta)+\delta \\
&= 1 + 3\delta + 2\sqrt{\delta} + \delta^2 + 2\delta\sqrt{\delta} \\
&\leq 1 + 4\delta + 4\sqrt{\delta} \\
&\leq 1 + 8\sqrt{\delta} \\
&= 1 + \varepsilon
\end{aligned}$$

as claimed. □

As a quick side note, it would have been sufficient to work with $\varepsilon$-approximate centers instead of $\varepsilon$-close centers. Algorithm 5 could be replaced with any other coreset algorithm, such as the one presented by Yildirin in [80]. In any case, we can now approach the general $k$-center problem. Instead of maximizing the distance of the second selected point to the first one, we just make sure that it is sufficiently far away to provide a useful lower bound on the initial radius of the coreset.

## 4.4 Approximating the $k$-Center Problem

For the sake of clarity, we will ignore the reasons that led us to posit Algorithm 7 and only show how to solve the $k$-center problem using Algorithm 6. We are not concerned with the running time per se but only with the underlying idea, which is the same, whether we work with MEBs directly or instead with $\varepsilon$-approximations. So, suppose we can run above Algorithm 6 for all clusters of a given $k$-center clustering $\mathscr{C} = \{C_1, \ldots, C_k\}$ simultaneously. After $O\left(k\varepsilon^{-2}\right)$ steps, we will have found $k$ sets $S_1 \subset C_1$, ..., $S_k \subset C_k$ of size at most $48\varepsilon^{-2}$ each, such that

$$X \subset \bigcup_i \mathrm{B}((1+\varepsilon)\zeta\left(S_i\right), \rho\left(S_i\right)).$$

In such a case, the number of points is small enough to guess these coresets directly. Although trying to find an optimal clustering purely by brute force usually takes $O\left(k^n\right)$ time, since for each of the $n$ points contained in $X$, we have $k$ possible clusters into which to put it, once we pass to coresets we only have to decide this distribution for $O\left(k\varepsilon^{-2}\right)$ points, which is possible if $k$ is small. In other words, with a multiplicative overhead of $k^{O\left(k\varepsilon^{-2}\right)}$, we can guess for each point, newly selected during the coreset construction, where to put it. This is then the algorithm: to brute-force the coresets of an optimal solution on a much more restricted "solution space". We could also think of this in terms of oracles. Whenever we select a new point, the oracle will tell us exactly which coreset $S_i$ to put it into, multiplying the running time by $k$ each time. But how do we select new points to incorporate into our coresets? There are two things to take care of right now:

1. We have to outline how several $\varepsilon$-coreset constructions can be executed "simultaneously". In particular, we must consider how points will be selected without knowing to which coreset they should belong.

2. We must show that the approach yields a PTAS for constant $k$.

Algorithm 8 shows an outline of the approach. We have decided to use the oracle formulation first since the resulting algorithm is more concise and intuitively clearer. We will discuss a different representation where the running time is easier to establish.

**Theorem 112.** *Let $X \subset \mathbb{R}^d$ a finite set, $\mathscr{C}^* = \{C_1^*, \ldots, C_n^*\}$ an optimal $k$-center clustering of $X$, and $u^* : X \to \{1, \ldots, k\}$ the corresponding oracle (meaning that $u^*(x) = i$ for $x \in C_i$). Then Algorithm 8 terminates after $O\left(k\varepsilon^{-2}\right)$ iterations and returns a $(1+\varepsilon)$-approximate $k$-center solution.*

*Proof.* First, we will show that the resulting solution is a $(1+\varepsilon)$-approximation. This is the case because:

**Algorithm 8:** *k*-Center via Coresets

```
Input  : A finite set $X \subset \mathbb{R}^d$, a number $k \in \mathbb{N}$, an error-parameter
         $0 < \varepsilon < 1$, an oracle $u : X \to \{1, \ldots, k\}$.
Output: A $(1+\varepsilon)$-approximate solution $(c_1, r_1), \ldots (c_k, r_k)$.
1 $S_j \leftarrow \varnothing$ for all $j \in \{1, \ldots, k\}$;
2 $r \leftarrow 0$;
3 $S_{u(x)} \leftarrow \{x\}$ for some arbitrary $x \in X$;
4 while $\exists x \in \operatorname{argmax}_{p \in X \setminus \bigcup_j B(\zeta(S_j),\,(1+\varepsilon)r)} \min_j \|p - \zeta(S_j)\|$ do
5     $S_{u(x)} \leftarrow S_{u(x)} \cup \{x\}$;
6     $r \leftarrow \rho\left(S_{u(x)}\right)$;
7 end
8 return $\{\zeta(S_1), \ldots, \zeta(S_k)\}$
```

1. The algorithm terminates once all points are within a distance of $(1+\varepsilon)r$ from some center. In other words,

$$X \subset \bigcup_j \mathrm{B}(\zeta(S_j),\, (1+\varepsilon)r).$$

2. The variable $r$ always equals the radius of some subset $S_i \subset C_i^* \in \mathscr{C}^*$. Lemma 110 thus implies that

$$r \leq \max_j \rho\left(C_j^*\right) = \mathrm{rad}(\mathscr{C}^*).$$

Combining both observations shows that the solution computed by the algorithm is upper-bounded by

$$(1+\varepsilon)r \leq (1+\varepsilon)\max_j \rho\left(C_j^*\right) = (1+\varepsilon)\,\mathrm{rad}(\mathscr{C}^*).$$

Note that although the radius by which we open a center $\zeta(S_i)$ might significantly exceed $\rho(C_i^*)$, this is not a problem since only the largest radius is relevant to the $k$-center objective. Strictly speaking, the $S_i$ do not necessarily form coresets, but we will still refer to them as such.

Next, we will upper bound the number of iterations and show that the simultaneous construction of coresets does not incur additional overhead. Whereas $r$ still grows by a multiplicative factor of at least $1 + \frac{\varepsilon^2}{4(1+\sqrt{2})}$ with each new addition, we have to establish the initial lower bound differently. This is because new points are chosen not by maximizing the distance to the center of the coreset, which represents their cluster in the optimal solution, but by maximizing the smallest distance to all already computed centers. However, as it turns out, this is not a big problem. Because we consider all radii to be the same, saved in variable $r$, we only have to consider the first time $r$ is set to a non-zero value. Indeed, this non-zero value has to be at least $\mathrm{rad}(\mathscr{C}^*)$: otherwise, the process by which we select new points would guarantee that every point would be at a distance from some center that is strictly smaller than $\mathrm{rad}(\mathscr{C}^*)$. The centers collected so far would thus constitute a cheaper-than-optimal solution, which is impossible. In other words, the initial lower bound for each coreset $S_i$ is at least

$$\mathrm{rad}(\mathscr{C}^*) \geq \rho\,(C_i^*) \geq \frac{1}{2}\,\mathrm{diam}(C_i^*)$$

and we can conclude as before that each $S_i$ can reach a size of at most $48\varepsilon^{-2}$. This upper bounds the number of iterations by $48k\varepsilon^{-2}$, at least, when $u^*$ is an optimal oracle. □

The above proof only works if we have an oracle for the optimal solutions. Three questions remain:

1. How can the oracle be modeled so that we can properly use it?
2. How do we find the oracle?
3. What is the actual running time of the algorithm?

As we said earlier, we are leaving out the running time for now and will only return to it in the next section for the $k$-MSR objective. Switching from MEBs to approximations introduces several technical complications that are not particularly interesting at the moment. Nothing much is lost by skipping this aspect for now. However, we have also partially skipped it because we have yet to formalize oracles properly. In the remaining two paragraphs, we will sketch our approach and answer the first two questions. We can envision an oracle $u$ as a sequence of numbers drawn from $\{1, \ldots, k\}$, where the $i$-th entry of the sequence tells us to which coreset the point selected in iteration $i$ should be added. As we have seen, for every optimal clustering, there exists a sequence of length at most $48k\varepsilon^{-2}$ that allows us to reconstruct it approximately, and those are the only ones we care about. Computation can always be terminated after $48k\varepsilon^{-2}$ iterations; if the resulting solution does not cover everything, then it can be thrown away. This ensures that the running time does not explode even when the sequence representing an oracle is unfavorable. We can then return the one with the smallest objective value from those that yield viable clusterings.

One last problem remains before we can actually model oracles as such sequences. While the sequence fixes a specific order in which the points are to be distributed, the points chosen by the algorithm are not yet fully determined: at several moments (for example, when we select the very first point), there might be several viable choices for newly selected points. Without further control, even if we brute-force all possible oracles, we might still miss the distribution of an optimal solution simply due to bad tiebreaks. To fix this problem, all we have to do is to provide a fixed rule by which tiebreaks are resolved. This can be achieved quite simply by enumerating the points in $X$ and, at any tiebreak, selecting the one whose index is smallest, meaning that in the same scenario, tiebreaks are always resolved the same way and, in particular, that the first selected point is always the same. This representation will be used in the next section to tackle the $k$-MSR problem.

## 4.5 Approximating the $k$-MSR Problem

By now, the approach should be obvious. We would like to successively select points and guess their assignments within an optimal solution in such a way that the coresets remain small. However, this is much more complex in the $k$-MSR setting. The problem lies in establishing proper lower bounds for the initial sizes of the coresets due to the possibly wildly different radii sizes in an optimal solution (all of which enter into the objective). Assume, for example, that the next point selected by the algorithm is always chosen from

$$\underset{p \in X - \bigcup_j \mathrm{B}(\zeta(S_j),\, (1+\varepsilon)\rho(S_j))}{\operatorname{argmax}} \; \min_j \|p - \zeta\left(S_j\right)\| - \rho\left(S_j\right),$$

where $j$ ranges over all coresets containing at least one element in both cases. We have slightly adjusted the function from Algorithm 8 on the right side by factoring in the radii of the coresets. This ensures that $p$ is always chosen furthest from all constructed MEBs, which is somewhat necessary if we want to argue in a similar fashion to the $k$-center case. To establish a lower bound by contradiction, we could try the following: once we add a second point $x$ to some coreset $S_{u(x)}$ that consisted of a single point $y$ until then, then $\|x - y\| \geq \frac{1}{k}\rho\left(C^*_{u(x)}\right)$. This is already quite a bad lower bound and would increase the running time exponentially by a multiplicative factor of $k$ (one multiplicative factor of $k$ for each additional point for which we would have to guess the assignment). Even worse, this need not necessarily hold. The argument would be as follows: if $\|x - y\| < \frac{1}{k}\rho\left(C^*_{u^*(x)}\right) =: r$, then we could try to increase the radii of all other centers by $r - \delta$ and open $y$ by $r - \delta$ for some $\delta > 0$. The hope would be that this would yield a $k$-MSR solution of cost strictly less than $\mathrm{rad}(\mathscr{C}^*)$. However, the resulting balls do not necessarily cover all points despite our choice of $x$, as we have not considered the points within the $\varepsilon$-enlarged balls of any of the other centers. This was necessary to ensure that the MEBs grew sufficiently quickly. However, now, we might be in a position where there is a point within the $\varepsilon$-enlarged ball $\mathrm{B}(\zeta(S_i), (1+\varepsilon)\rho(S_i))$ of its center, but that is still further away from that ball than $\|x-y\|$. It would not be covered. What would need to hold is that

$$\varepsilon\rho(S_i) \leq \|x-y\| < \frac{1}{k}\rho\left(C^*_{u^*(x)}\right)$$

where the first term could be equal to $\varepsilon \max_j \rho\left(C^*_j\right)$ in the worst-case. To ensure this, we would have to shrink $\varepsilon$ to $\frac{\varepsilon}{k^2}$ and lead the proof to a contradiction, not relative to an optimal solution, but relative to an optimal $\varepsilon$-balanced solution (we will define this type of solution later on). However, this change would exponentially impact the running time, which is even worse than the increase we mentioned just earlier. Instead, we will guess a good lower bound for each coreset beforehand and pass it to the algorithm.

### 4.5.1 Guessing Good Lower Bounds

We will essentially guess the radii of an (almost) optimal solution up to a multiplicative factor of 2. First, we will guess the radius of the largest ball and then use that value in the next step to guess the remaining radii. To do this, we will exploit the following relation between the largest radius in an approximately optimal $k$-MSR solution and the value of an optimal $k$-center solution.

**Lemma 113.** *Let $r_\alpha^{center}$ denote the value of an $\alpha$-approximate $k$-center solution and $r_\beta^{msr}$ the largest radius of a $\beta$-approximative $k$-MSR solution for the same instance. Then, it holds that*

$$r_\beta^{msr} \in \left[\frac{r_\alpha^{center}}{\alpha}, \beta \cdot k^2 \cdot r_\alpha^{center}\right].$$

*Proof.* Let $r_{\mathsf{opt}}^{\text{center}}$ denote the value of an optimal $k$-center solution. Since the centers of any a $k$-MSR solution yield a solution for the $k$-center problem, we know that $r_\beta^{\text{msr}} \geq r_{\mathsf{opt}}^{\text{center}}$. Combining this with the fact that $r_\alpha^{\text{center}} \leq \alpha \cdot r_{\mathsf{opt}}^{\text{center}}$ yields the lower bound $r_\beta^{\text{msr}} \geq r_{\mathsf{opt}}^{\text{center}} \geq r_\alpha^{\text{center}}/\alpha$.

On the other hand, since every $k$-center solution also yields a solution to the $k$-MSR problem by assigning points to their closest center, we know that $r_{\mathsf{opt}}^{\text{center}} \leq k \cdot r_\alpha^{\text{center}}$. This forces $r_\beta^{\text{msr}} \leq \beta \cdot k^2 \cdot r_\alpha^{\text{center}}$, because otherwise $r_\beta^{\text{msr}} > \beta \cdot k^2 \cdot r_\alpha^{\text{center}} \geq \beta \cdot k \cdot r_{\mathsf{opt}}^{\text{center}}$ would contradict the assumption that $r_\beta^{\text{msr}}$ is the largest radius in a $\beta$-approximation for the $k$-MSR problem. □

This lemma already yields a candidate set (which we will discretize in a second) for the largest radius. A few quick remarks:

1. For our purposes, it is enough that $\alpha = 2$, so we can use either the algorithm by Gonzalez or the one by Hochbaum and Shmoys.

2. The reason that we are considering $\beta$-approximations for the $k$-MSR problem instead of optimal solutions, as we have done so far, will become clear in just a second or once we build a candidate set for the other radii.

By utilizing standard discretization techniques on this interval, we will be able to obtain a finite candidate set such that

1. its size only depends on $\varepsilon$, $k$, $\alpha$ and $\beta$, and

2. it contains a 2-approximation for each value in the interval.

Once we have guessed the largest radius, we can apply a similar technique to obtain a candidate set for the remaining radii. However, this requires that the other radii are not too small compared to the largest. More precisely, we will assume that the solution we are interested in is $\varepsilon$-balanced.

**Definition 114.** Let $\varepsilon > 0$. A $k$-MSR solution $\{(c_1, r_1), \dots, (c_k, r_k)\}$ is *$\varepsilon$-balanced,* if $r_i \geq \frac{\varepsilon}{k} \max_j r_j$ for all $i \in \{1, \dots, k\}$.

Since we will guess the solution of an $\varepsilon$-balanced solution later on, let us quickly show that such a solution always exists that is worth guessing.

**Lemma 115.** *Let $(X, d)$ be a finite metric space, $k \in \mathbb{N}$ a number, and $\varepsilon > 0$ an error parameter. Then there exists an $\varepsilon$-balanced solution that is also a $(1 + \varepsilon)$-approximation.*

*Proof.* Let $(c_1^*, r_1^*), \dots, (c_k^*, r_k^*)$ denote an optimal $k$-MSR solution for $(X, d)$. Define a new $k$-MSR solution on the same centers with the following radii:

$$r_i = \begin{cases} r_i^*, & \text{if } r_i^* \geq \frac{\varepsilon}{k} \max_j r_j^* \\ \frac{\varepsilon}{k} & \text{otherwise.} \end{cases}$$

This new solution, which covers at least as much as the optimal solution (we have only increased the radii of some balls), has a cost that is larger than the optimum by an additive factor of at most

$$k \cdot \frac{\varepsilon}{k} \max_j r_j^* \leq \varepsilon \sum_i r_i^*.$$

$\square$

Given such an $\varepsilon$-balanced solution, we can conclude this part with the following statement.

**Lemma 116.** *Let $\varepsilon > 0$ and $(c_1, r_1), \dots, (c_k, r_k)$ an $\varepsilon$-balanced solution with $r_1 \geq \dots \geq r_k$. Then, we can compute*

1. *a set of size $O(\log k)$ that contains a number $\widetilde{r_1}$ with $\frac{1}{2}r_1 \leq \widetilde{r_1} \leq r_1$.*
2. *a set of size $O(\log(k\varepsilon^{-1}))$ that contains, for each $r_i$, a number $\widetilde{r_i}$ with $\frac{1}{2}r_i \leq \widetilde{r_i} \leq r_i$.*

We start with the following helpful lemma to later prove Lemma 116. It encapsulates the common technique of "covering" an interval $[a, b]$ by $O(\log(b/a))$ smaller intervals to obtain a (somewhat reasonably sized) discrete set of values that contains for each value in $[a, b]$ a 2-approximation.

**Lemma 117.** *Let $I = [a, b] \subset \mathbb{R}$ be an interval consisting of non-negative numbers. Then for every $r \in I$, there exists a number $\widetilde{r}$ within*

$$R = \{2^{i-1}a \mid i \in \{0, \dots, \lceil \log(b/a) \rceil\}\},$$

*such that $\frac{1}{2}r \leq \widetilde{r} \leq r$.*

*Proof.* Consider the exponential growing set of intervals $I_j = [2^{j-1}a,\ 2^j a]$ for $j \in \{0, \dots, \lceil \log(b/a) \rceil\}$. Then

1. the union of these intervals covers $I$, and
2. $R$ consists exactly of the left endpoints of each of them.

Both of these observations combined prove the claim. $\square$

This lemma can directly be applied to the interval given by Lemma 113 to obtain a candidate set for the largest radius in any feasible solution whose size does not depend on $n$.

**Lemma 118.** *Let $(X, d)$ be a finite metric space and $(c_1, r_1), \dots, (c_k, r_k)$ a $\beta$-approximative $k$-MSR solution with $r_1 \geq \dots \geq r_k$. If $r_\alpha^{center}$ denotes the value of an $\alpha$-approximate $k$-center solution, then the set*

$$R = \left\{ 2^{i-1} \frac{r_\alpha^{center}}{\alpha} \mid i \in \{0, \dots, 2\lceil \log(\alpha\beta k) \rceil\} \right\}$$

*contains a number $\widetilde{r_1}$ with $\frac{1}{2}r_1 \leq \widetilde{r_1} \leq r_1$.*

Once the largest radius is fixed, we can obtain a candidate set for all other radii in a similar manner. The difference here is that we need a suitable lower bound for the radii so that the candidate set does not get too large. We achieve this by assuming that the solution we are trying to approximate is $\varepsilon$-balanced, as introduced in Definition 114.

**Lemma 119.** *Let $(X,d)$ be a finite metric space and $(c_1,r_1),\ldots,(c_k,r_k)$ an $\varepsilon$-balanced solution with $r_1 \geq \ldots \geq r_k$. Then the set*

$$R = \left\{ 2^{i-1}\frac{\varepsilon}{k} r_1 \mid i \in \left\{0,\ldots,\frac{k}{\varepsilon}\right\}\right\}$$

*contains for each $r_i$ a number $\widetilde{r_i}$ with $\frac{1}{2}r_i \leq \widetilde{r_i} \leq r_i$.*

*Proof.* Since the solution is $\varepsilon$-balanced, we necessarily have that $r_i \in \left[\frac{\varepsilon r_1^*}{k}, r_1^*\right]$ for all $i$. The claim then immediately follows from Lemma 117. □

We are now able to prove Lemma 116.

*Proof of Lemma 116.* Lemma 118 shows how to obtain the candidate set for the largest radius of the desired size, Lemma 119 shows it for the remaining radii. □

### 4.5.2 The Algorithm

Instead of establishing a lower bound via the workings of the algorithm, we will now supply it directly. Algorithm 9 shows the final algorithm (at least for a given oracle and set of lower bounds). We have finally fixed an order on the points in $X$ and replaced the oracle with a sequence that represents the assignments of newly selected points. As usual, the union in Line 5 runs only over those $j$ for which $c_j$ has been set.

Suppose we correctly guessed the oracle and the lower bounds for some $k$-MSR solution. Then the following lemma shows that Algorithm 9 yields a solution whose cost is at most $(1+\varepsilon)$ times higher.

**Lemma 120.** *Let $X \subset \mathbb{R}^d$ be a finite ordered set, $\varepsilon > 0$ an error parameter, and $\widetilde{r_1},\ldots,\widetilde{r_k}$ radii, such that $\frac{1}{2}r_i^* \leq \widetilde{r_i} \leq r_i^*$ for some k-MSR solution $(c_1^*,r_1^*),\ldots,(c_k^*,r_k^*)$. Then there exists a sequence $u^* \in \{1,\ldots,k\}^{768k\varepsilon^{-4}}$, such that running Algorithm 9 on $X$, $u^*$, and $(\widetilde{r_i})_i$ returns a k-MSR solution $(c_1,r_k)$, …, $(c_k,r_k)$ with*

$$\sum_i r_i \leq (1+\varepsilon)\sum_i r_i^*.$$

*Proof.* We construct $u^*$ by recording the proper assignments during the selection process. This is possible because

1. after fixing an ordering of the points, the algorithm is deterministic,
2. assignments do not have to be specified before points are selected.

**Algorithm 9:** $k$-MSR Algorithm

```
Input  : A finite ordered set $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$, a number $k \in \mathbb{N}$, an
         error-parameter $0 < \varepsilon < 1$, a sequence $u \in \{1, \dots, k\}^*$, a set of
         radii $r_1, \dots, r_k$.
Output: A set of $k$ pairs $\{(c_1, r_1), \dots (c_k, r_k)\}$.
 1 $\delta \leftarrow \frac{\varepsilon^2}{8}$;
 2 $\gamma \leftarrow 1 + \delta + 2\sqrt{\delta}$;
 3 $S_j \leftarrow \varnothing$ for all $j \in \{1, \dots, k\}$;
 4 for $i = 1, \dots, |u|$ do
 5     $R \leftarrow X - \bigcup_j \mathrm{B}(c_j,\ (1+\gamma)r_j)$;
 6     if $R = \varnothing$ then
 7         exit loop;
 8     end
 9     $x \leftarrow$ point in $R$ with smallest index;
10     $S_{u_i} \leftarrow S_{u_i} \cup \{x\}$;
11     $(c_{u_i}, r_{u_i}) \leftarrow \delta$-close center of $S_{u_i}$ with induced radius;
12 end
13 return $\{(c_1, r_1), \dots, (c_k, r_k)\}$
```

During the first iteration, if the first selected point $x^{(1)}$ lies within $\mathrm{B}\big(c^*_{i_1},\ r^*_{i_1}\big)$, where $i_1$ is the smallest such index, we set $u_1 = i_1$. Continuing in this fashion, we ensure that $S_i \subseteq \mathrm{B}(c^*_i,\ r^*_i)$ for all $i$. Furthermore, since we also pass a value $\widetilde{r}_i$ with $\frac{1}{2}r^*_i \leq \widetilde{r}_i \leq r^*_i$ as a lower bound to the algorithm, we know that

$$\rho\left(C^*_i\right) \geq \rho\left(S_i\right) \geq \frac{1}{2}\rho\left(C^*_i\right) \geq \frac{1}{4}\,\mathrm{diam}\left(C^*_i\right)$$

holds for the initial radius of $S_i$ (i.e., when $S_i$ contains exactly two points). Finally, we can apply the same estimation from the proof of Lemma 111 to show that $\rho\left(C^*_i\right)$ always grows by a factor of $1+\frac{\delta^2}{4(1+\sqrt{2})}$ with every new addition and that the size of $S_i$ is thus upper bounded by $2 \cdot 48\delta^{-2} = 96\delta^{-2} = 768\varepsilon^{-4}$. This proves that $R = \varnothing$ after processing $768k\varepsilon^{-4}$ many entries of $u^*$, so we can upper bound the length of $u^*$ accordingly. $\square$

Finally, we get to the main theorem.

**Theorem 121.** *Let $X \subset \mathbb{R}^d$ be a finite ordered set, $\varepsilon > 0$ an error parameter, and $k \in \mathbb{N}$. Then there exists an algorithm that computes a $(1+\varepsilon)$-approximate $k$-MSR solution for $X$ in $O\left(nd \cdot f(k, \varepsilon^{-1})\right)$ time. Thus, assuming that $k$ is constant, this algorithm yields a PTAS for the $k$-MSR problem.*

*Proof.* Let $(c_1^*, r_1^*), \ldots, (c_k^*, r_k^*)$ be an $\varepsilon$-balanced and $(1+\varepsilon)$-approximative $k$-MSR solution for $X$. If we have guessed the lower bounds and the oracle correctly, then Lemma 120 shows that we can compute a $(1+\varepsilon)^2$-approximative solution in $O(ndk\varepsilon^{-4})$ time. The oracle can be guessed in $k^{O\left(k\varepsilon^{-4}\right)}$ time and the lower bounds in $O\left(\log \frac{k}{\varepsilon}\right)^k$ time by going over all possible combinations. This yields a running time of

$$O\left(nd\varepsilon^{-4} \cdot k^{O\left(k\varepsilon^{-4}\right)} \cdot O\left(\log \frac{k}{\varepsilon}\right)^k\right). \qquad \square$$

### 4.5.3 Outliers

In this short final section, we would like to show that outliers can be incorporated into the above algorithm. That is, we can even allow the user to specify a number $z$ of how many points can be ignored when assessing the cost of a solution. All we have to do is to add a $(k+1)$-th set $S_{k+1}$ for the outliers, whose radius remains 0 throughout. If there are $z$ outliers, then the oracle is drawn from $\{1, \ldots, k+1\}^{768k\varepsilon^{-4}+z}$, where at most $z$ entries are equal to $k+1$. This then yields Theorem 95.

# Chapter 5

# Connected Clustering

## 5.1 Introduction

In this chapter, we provide a simple assignment algorithm for the connected $k$-center problem.

**Definition 122** (Connected $k$-Center Problem)**.** Let $(X, d)$ be a finite metric space, $G = (X, E)$ a graph on $X$, and $k \in \mathbb{N}$. The goal in the *connected $k$-center problem* is to find an assignment $\sigma : X \to C$ to a set of $k$ centers $C \subset X$ that minimizes $\max_{x \in X} d(x, \sigma(x))$ and guarantees that $G[\sigma^{-1}(c)]$ is connected for all $c \in C$.

In other words, this $k$-center variant allows us to consider object relations that go beyond the metric that measures their dissimilarity. Ge et al. ([46]) first came up with this restriction in order to separate attribute data (a set of points $X$ in Euclidean space) and relationship data (a graph $G$ on top of $X$), which is of interest in market segmentation (which is "the process of dividing a market into distinct customer groups with homogeneous needs"[1]) and community detection within social networks, for example. In the first case, clusterings are derived primarily from attributes, such as demographic data; in the second, they are primarily derived from social relations. However, in both cases, the other data is also of importance. Social relations along which word of mouth can spread are important to markets and commonalities of interest to social formations. We can also draw examples from geodesy. Consider the task of reconstructing regional of global mean sea levels from recordings of gauge stations situated all across the world, such as the PSMSL data set ([68], [55]). These stations, however, are distributed very unevenly around the globe, skewing the result. One way of handling this is to cluster the stations and consider good representatives to thin

[1] [46], p. 3

out the data. This entails clustering stations according to their gauge time series while ensuring that stations from different geographic regions are not clustered together. In other words, stations must be viewed through two different lenses: (1) as points on the globe and (2) as sources of sea level data. The first yields the metric (via the Fréchet distance, for example), and the second the relations (via coastal neighborhood, for example).

As a generalization of the $k$-center problem, it is known to be NP-hard in general, but we can solve it optimally in polynomial time for simpler graph structures, such as trees. This was already proven by Ge et al., who provided a DP that solves the problem in $O(n^2 \log n)$ time, where $n$ is the number of points. In this short chapter, we present a simple algorithm that solves the corresponding assignment problem. This algorithm was established as a preliminary result for [37] (and can be found in the full version [36]) during a time when we were unaware of the existence of the paper [46] by Ge at al. Even without considering these results, the assignment algorithm was superseded by a different DP (also outlined in [36]) that optimally solves the whole connected $k$-center problem, not just the assignment problem. In any case, we hope this algorithm might still be of some interest, even if just as a preparation for the DP.

## 5.2 Hardness of the Assignment Problem

In the assignment version of the connected $k$-center version, we are also given, apart from the finite metric space $(X, d)$ and the graph $G = (X, E)$, a predetermined set of $k$ centers $C$. The task is only to assign the points from $X$ to $C$ in the best possible way, meaning that the resulting assignment $\sigma : X \to C$ should minimize $\mathrm{rad}(\sigma, d)$. Finding such an assignment is trivial for the vanilla $k$-center problem: just assign every point to its closest center. However, this might not be possible for the connected $k$-center problem since the resulting clusters might not induce connected subgraphs. In fact, there is no $(3 - \varepsilon)$-approximation algorithm for the general assignment problem for any $\varepsilon > 0$, unless P = NP. We will have to be more careful later on.

**Theorem 123.** *It is NP-hard to approximate the assignment variant of the connected $k$-center problem with an approximation factor smaller* 3*, even if $k = 2$.*

*Proof.* We prove this claim via a reduction from 3-SAT (which is known to be NP-hard [60]). Given a set of variables $X = \{x_1, ..., x_n\}$ and a set of clauses $\mathcal{Z} = \{z_1, z_2, ..., z_m\}$ consisting of 3 literals each, the 3-SAT problem consists in determining, whether there exists a truth assignment $f : X \to \{T, F\}$, such that each clause contains at least one true literal. Here, the set of literals $L = \{x, \overline{x} \mid$

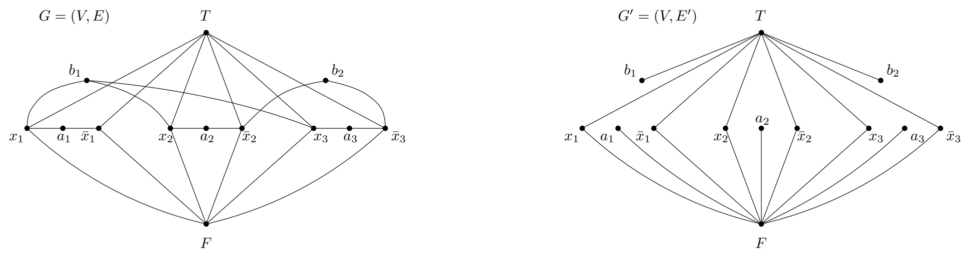


Figure 5.1: The connectivity graph and metric corresponding to 3SAT instance $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_2 \vee \bar{x}_3)$

$x \in X\}$ just consists of variables and their negations. We will turn any such 3-SAT instance into an instance for the connected $k$-center problem in such a way that an assignment of cost 1 exists if and only if the 3-SAT instance is solvable. Otherwise, there only exists an assignment of cost 3. The connectivity graph $G = (V, E)$ consists of vertices

$$V = \{T, F\} \cup \{a_i \mid x_i \in X\} \cup L \cup \mathcal{Z}$$

and edges

$$\begin{aligned} E = \quad & \{\{\lambda, T\}, \{\lambda, F\} \mid \lambda \in L\} \\ & \cup \{\{x_i, a_i\}, \{\overline{x_i}, a_i\} \mid x_i \in X\} \\ & \cup \{\{\lambda, z\} \mid \lambda \in z\}. \end{aligned}$$

The distance between two vertices is given by the length of a shortest path in the graph $G' = (V, E')$, where

$$E' = \{\{\lambda, T\}, \{\lambda, F\} \mid \lambda \in L\} \cup \{\{a_i, F\} \mid x_i \in X\} \cup \{\{z, T\} \mid z \in \mathcal{Z}\},$$

and the pre-established set of centers is given by $C = \{T, F\}$. For a picture of the final instance, see Figure 5.1.

If $f$ is a truth assignment satisfying the above 3-SAT instance, then the following cluster assignment $\sigma$ provides a solution with radius 1 for the connected $k$-center problem. First, set $\sigma(a_i) = F$ for all $x_i \in X$ and $\sigma(z) = T$ for all $z \in \mathcal{Z}$. This part of the assignment does not depend on $f$. Then, if $f(x_i) = T$, set $\sigma(x_i) = T$ and $\sigma(\overline{x_i}) = F$. Otherwise, switch both of these assignments. It is easy to verify that this assignment induces a maximal radius of 1, and it is only slightly harder to show that the subgraphs induced by the clusters are connected. The only vertices at issue are the $a_i$s since all other vertices are directly connected to their assigned centers. However, since each $a_i$ is adjacent to $x_i$ and $\overline{x_i}$, and since one of these vertices gets assigned to $F$, the claim follows.

For the other direction, suppose that we are given a feasible assignment $\sigma : V \to \{T, F\}$ whose maximal radius is less than 2. By definition of $E'$ this necessitates that $\sigma(a_i) = F$ for all $x_i \in X$ and $\sigma(z) = T$ for all $z \in \mathcal{Z}$. We now wish to show that the truth assignment $f(x_i) = \sigma(x_i)$ satisfies all clauses in $\mathcal{Z}$. However, this is not difficult to see. Since no clause is connected directly to $T$, at least one of its neighbors must be adjacent to $T$. However, the only neighbors of clauses in $(V, E)$ are the literals of which they consist. Hence, all clauses must be satisfied, and $f$ is a feasible solution.

After combining both directions, we see that if there were to exist a $(3-\varepsilon)$-approximation algorithm for the assignment variant of the connected $k$-center problem, then we could also solve 3-SAT. □

## 5.3 An Assignment Algorithm for Trees

Although the assignment variant of the connected $k$-center problem is NP-hard to approximate to within a factor of 3 for general connectivity graphs, the same is not true for simpler connectivity graphs, such as trees. This is what we wish to show in the remainder of this chapter. From now on, assume that the underlying connectivity graph is a tree $T = (X, E)$. We will consider a simplification in which every pre-established center is a leaf in $T$, as this allows us to more readily exploit the tree structure of the connectivity graph to build an assignment from the ground up.

Given a guess for the optimal radius $r$, we process the vertices from leaves to root. For each vertex $v \in V$, we maintain two sets: $n(v)$ and $z(v)$. The former set consists of all previously processed vertices that must be served together with $v$, meaning they must be assigned to the same center as $v$. The set $z(v)$ consists of all centers that are reachable from $v$, where a center $c$ is said to be reachable from $v$ if $d(x, c) \leq r$ all $x \in n(v)$. In the end, if $z(t)$ is empty, then no center is reachable from vertex $t$, and we can conclude that $r$ is too small. Otherwise, process the points from top to bottom and assign them to reachable centers. A more precise description is given in Algorithm 10. For any $v$, let $\pi(v)$ denote its parent and $P_{v,c}$ the unique $v$-$c$-path connecting it to center $c$. From line 1 to line 3, we initialize the set, which will hold all processed vertices, as well as the lists $n(\cdot)$ and $z(\cdot)$. From line 4 to line 11 and line 12 to 21, we update both lists for all points bottom-up, from leaves to root. After processing a point $v$, if $z(v)$ is empty, then $v$ has to be assigned to a center the same as its parent $\pi(v)$, and we add all of $n(v)$ to $n(\pi(v))$. During lines 22-24 and 25-32, the algorithm either fails, implying that $r$ is too small, or outputs a feasible assignment.

**Lemma 124.** *For any $v \in V$ and $c \in z(v)$, it must hold that $d(y, c) \leq r$ for all points $y \in \bigcup_{x \in P_{v,c}} n(x)$.*

**Algorithm 10:** Assignment Algorithm

**Input:** A tree $T = (V, E)$ with root $t$, centers $C$ that are also leaves, a metric $d$, a radius $r$.

**Result:** If an assignment with radius $r$ exists, then the algorithm will find such an assignment. Otherwise, it will fail.

```
$M \leftarrow \emptyset$;
$z(v) \leftarrow \emptyset$ for all $v \in V$;
$n(v) \leftarrow \{v\}$ for all $v \in V$;
forall leaves $\ell \in V$ do
    $M \leftarrow M \cup \{\ell\}$;
    if $\ell \in C$ then
        $z(\ell) \leftarrow z(\ell) \cup \{\ell\}$;
    else
        $n(\pi(\ell)) \leftarrow n(\pi(\ell)) \cup n(\ell)$;
    end
end
while $M \neq V$ do
    pick $v \in V$ such that $u \in M$ for each child $u$ of $v$;
    forall child $u$ of $v$ do
        $z(v) \leftarrow z(v) \cup \{c \in z(u) \mid \forall x \in n(v) : d(x, c) \leq r\}$;
    end
    if $z(v) = \emptyset$ then
        $n(\pi(v)) \leftarrow n(\pi(v)) \cup n(v)$;
    end
    $M \leftarrow M \cup \{v\}$;
end
if $z(t) = \emptyset$ then
    fail;
end
while $M \neq \emptyset$ do
    let $v$ be the point in $M$ that was added last;
    pick $c \in z(v)$ arbitrarily and let $P_{v,c}$ denote the unique $v$-$c$-path;
    forall $x \in P_{v,c}$ do
        assign all points in $n(x)$ to $c$;
        $M \leftarrow M \setminus n(x)$;
    end
end
return assignment
```

*Proof.* For all $x \in P_{v,c}$, we know that $c \in z(x)$ since $c$ can only be passed upward along path $P_{v,c}$ (see lines 12 and 13) in a tree. If some $z(x)$ does not contain $c$, then $z(v)$ can neither. Line 13 then ensures that the distance between each point in $n(x)$ and center $c$ should be at most $r$, which completes the proof. □

We are now ready to analyze the correctness of the algorithm using the following case distinction.

1. The algorithm succeeds. Then, applying Lemma 124 to the assignments computed in lines 27-29 shows that the maximal radius of the resulting solution is at most $r$.
2. The algorithm fails. Then $z(v) = \emptyset$ for some point $v \in V$. Since the subgraph induced by the clusters of any feasible assignment cluster has to be connected and the connectivity graph is a tree, $v$ must be assigned to the same center as point $\pi(v)$. Since we process the points in a bottom-up fashion, from leaves to root, root $t$ is processed at the end, and thus $z(t) = \emptyset$ implies that there is no feasible assignment with radius $r$.

Altogether, Algorithm 10 needs at most $O(nk)$ time since we check the distance from every point to every center at most once. While this would suggest an overall running time of $O(nk \log n)$, which includes guessing $r$, we initially assumed that the centers are all leaves and have yet to deal with the general case. If some center $c$ is not a leaf, we remove it from $T$, add $|\delta(c)|$ new vertices, and pair them with the old neighbors. That is, each new copy is connected to a different neighbor from $c$ in $T$. After doing this for each non-leaf center, we get a connectivity forest, in which each center is a leaf. The trees making up this forest can then be individually passed to Algorithm 10 and the resulting assignments combined into a single feasible solution, increasing the overall running time since the number of centers can be in $\Omega(n)$.

**Theorem 125.** *There exists an optimal assignment algorithm on trees that runs in $O(n^2 \log n)$ time.*

# Acknowledgments

**Chapter 2** is based on [10, 9] which I've co-authored with Anna Arutyunova, Anna Großwendt, Heiko Röglin and Melanie Schmidt. My main contribution concerned the lower bounds, while Anna Arutyunova was the driving force behind the upper bounds. After I constructed several $\mathbf{CL}^{\mathrm{diam}}$ worst-case examples for particular $k$ by hand, Anna Großwendt realized that they exhibited a pattern that could be continued inductively to yield worst-case examples for all $k$. I then formalized and fleshed out this approach and also used it to construct lower bounds for $\mathbf{CL}^{\mathrm{rad}}$. (It should be noted that Anna Arutyunova found another relatively different worst-case example for $\mathbf{CL}^{\mathrm{rad}}$. However, that example was not included in the publications.)

**Chapter 3** is based on a not yet published paper, which I have worked on with Anna Arutyunova, Jan Eube, Heiko Röglin, Melanie Schmidt, and Sarah Sturm. When I heard that the other authors researched Pareto sets for combinations of different clustering objectives, I approached them with the idea of also considering the $k$-separation objective. Anna Arutyunova and Sarah Sturm quickly devised a concrete algorithm to combine this objective with the $k$-center and $k$-diameter objective, which I then fleshed out. While I was responsible for the combination of the $k$-separation objective with the $k$-MSR objective, the remaining combinations with the $k$-median and $k$-means objectives are due to Sarah Sturm. I have merely rewritten those parts for this thesis. The initial example that shows that $k$-separation and $k$-diameter are not simultaneously constant-factor approximable is from Anna Arutyunova and myself. I have later used this example to establish similar results for other combinations with the $k$-separation objective.

**Chapter 4** is based on [35], which I have co-authored with Lukas Drexler, Annika Hennes, Abhiruk Lahiri, and Melanie Schmidt. I joined the group late and did not participate in initial discussions and preparations. To a large extent, these consisted of fleshing out Bădiou et al.'s description for the $k$-center algorithm. However, I have contributed considerably to the paper. The main lemma, the lemma that shows that there always exist good

clusterings that are both balanced and separated, the method of guessing radii for general mergeable constraints, as well as the proof of the correctness of the main theorem, are in their final form all written by me. However, while preparing the paper for this thesis, I sadly discovered a significant error in one of the proofs. Although the problem might be fixable, I have instead weakened the result. Chapter 4 is quite different from the published paper and was, apart from Section 4.5.1, which goes back to Lukas Drexler, entirely written by me.

**Chapter 5** is based on [37, 36], which I have co-authored with Lukas Drexler, Jan Eube, Kelin Luo, Heiko Röglin and Melanie Schmidt. Beyond some initial discussions, I have not contributed much to the paper and, hence, have only prepared a small part of it for this dissertation. Kelin Luo had the initial idea for the assignment algorithm, which I then fleshed out. The NP-hardness result is due to Jan Eube.

# Bibliography

[1] Margareta Ackerman, Shai Ben-David, and David Loker. "Characterization of Linkage-based Clustering". In: *Annual Conference Computational Learning Theory.* 2010. URL: https://api.semanticscholar.org/CorpusID:58691790.

[2] Margareta Ackerman, Shai Ben-David, and David Loker. "Towards Property-Based Classification of Clustering Paradigms". In: *Advances in Neural Information Processing Systems.* Ed. by J. Lafferty et al. Vol. 23. Curran Associates, Inc., 2010. URL: https://proceedings.neurips.cc/paper_files/paper/2010/file/f93882cbd8fc7fb794c1011d63be6fb6-Paper.pdf.

[3] Marcel R. Ackermann et al. "Analysis of Agglomerative Clustering". In: *Algorithmica* 69.1 (2014), pp. 184–215. DOI: https://doi.org/10.1007/s00453-012-9717-4.

[4] P. K. Agarwal and C. M. Procopiuc. "Exact and Approximation Algorithms for Clustering". In: *Algorithmica* 33.2 (June 2002), pp. 201–226. ISSN: 1432-0541. DOI: 10.1007/s00453-001-0110-y. URL: https://doi.org/10.1007/s00453-001-0110-y.

[5] Sara Ahmadian and Chaitanya Swamy. "Approximation Algorithms for Clustering Problems with Lower Bounds and Outliers". In: *Proc. of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP).* Vol. 55. 2016, 69:1–69:15. URL: https://doi.org/10.4230/LIPIcs.ICALP.2016.69.

[6] Sara Ahmadian et al. "Better Guarantees for k-Means and Euclidean k-Median by Primal-Dual Algorithms". In: *SIAM J. Comput.* 49.4 (2020). DOI: https://doi.org/10.1137/18M1171321.

[7] Soroush Alamdari and David Shmoys. "A Bicriteria Approximation Algorithm for the k-Center and k-Median Problems". In: *Approximation and Online Algorithms.* Ed. by Roberto Solis-Oba and Rudolf Fleischer. Cham: Springer International Publishing, 2018, pp. 66–75. ISBN: 978-3-319-89441-6.

[8] Anna Arutyunova and Heiko Röglin. "The Price of Hierarchical Clustering". In: *30th Annual European Symposium on Algorithms, ESA 2022, September 5-9, 2022, Berlin/Potsdam, Germany.* Ed. by Shiri Chechik et al. Vol. 244. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 10:1–10:14. DOI: 10.4230/LIPICS.ESA.2022.10. URL: https://doi.org/10.4230/LIPIcs.ESA.2022.10.

[9] Anna Arutyunova et al. "Upper and Lower Bounds for Complete Linkage in General Metric Spaces". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021).* Ed. by Mary Wootters and Laura Sanità. Vol. 207. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, 18:1–18:22. ISBN: 978-3-95977-207-5. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2021.18. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.APPROX/RANDOM.2021.18.

[10] Anna Arutyunova et al. "Upper and lower bounds for complete linkage in general metric spaces". In: *Machine Learning* 113.1 (Jan. 2024), pp. 489–518. ISSN: 1573-0565. DOI: 10.1007/s10994-023-06486-8. URL: https://doi.org/10.1007/s10994-023-06486-8.

[11] Mihai Badoiu and Kenneth L. Clarkson. "Smaller core-sets for balls". In: *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA).* ACM/SIAM, 2003, pp. 801–802. URL: http://dl.acm.org/citation.cfm?id=644108.644240.

[12] Mihai Badoiu, Sariel Har-Peled, and Piotr Indyk. "Approximate clustering via core-sets". In: *Proceedings on 34th Annual ACM Symposium on Theory of Computing (STOC).* ACM, 2002, pp. 250–257. URL: https://doi.org/10.1145/509907.509947.

[13] Mihai Bădoiu and Kenneth L. Clarkson. "Optimal core-sets for balls". In: *Computational Geometry* 40.1 (2008), pp. 14–22. ISSN: 0925-7721.

[14] Sayan Bandyapadhyay, Zachary Friggstad, and Ramin Mousavi. "Parameterized Approximation Algorithms and Lower Bounds for k-Center Clustering and Variants". In: *Algorithmica* (May 2024). ISSN: 1432-0541. DOI: 10.1007/s00453-024-01236-1. URL: https://doi.org/10.1007/s00453-024-01236-1.

[15] Sayan Bandyapadhyay, William Lochet, and Saket Saurabh. "FPT Constant-Approximations for Capacitated Clustering to Minimize the Sum of Cluster Radii". In: *39th International Symposium on Computational Geometry (SoCG).* Vol. to appear. 2023. URL: https://doi.org/10.48550/arXiv.2303.07923.

[16] Chiranjib Bhattacharyya, Ravindran Kannan, and Amit Kumar. “How many Clusters? - An algorithmic answer”. In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 2607–2640. DOI: 10.1137/1.9781611977073.102. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611977073.102.

[17] Moritz Buchem et al. “A $(3 + \varepsilon)$-approximation algorithm for the minimum sum of radii problem with outliers and extensions for generalized lower bounds”. In: *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1738–1765. DOI: 10.1137/1.9781611977912.69. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611977912.69.

[18] Jaroslaw Byrka et al. “An Improved Approximation for $k$-Median and Positive Correlation in Budgeted Optimization”. In: *ACM Trans. Algorithms* 13.2 (2017), 23:1–23:31. DOI: https://doi.org/10.1145/2981561.

[19] Vasilis Capoyleas, Günter Rote, and Gerhard Woeginger. “Geometric clusterings”. In: *Journal of Algorithms* 12.2 (1991), pp. 341–356. ISSN: 0196-6774. URL: https://www.sciencedirect.com/science/article/pii/019667749190007L.

[20] Gunnar Carlsson and Facundo Mémoli. “Characterization, Stability and Convergence of Hierarchical Clustering Methods”. In: *Journal of Machine Learning Research* 11.47 (2010), pp. 1425–1470. URL: http://jmlr.org/papers/v11/carlsson10a.html.

[21] Gunnar E. Carlsson and Facundo Mémoli. “Classifying Clustering Schemes”. In: *Foundations of Computational Mathematics* 13 (2010), pp. 221–252. URL: https://api.semanticscholar.org/CorpusID:9097762.

[22] Moses Charikar and Rina Panigrahy. “Clustering to minimize the sum of cluster diameters”. In: *J. Comput. Syst. Sci.* 68.2 (2004), pp. 417–441. URL: https://doi.org/10.1016/j.jcss.2003.07.014.

[23] Moses Charikar and Rina Panigrahy. “Clustering to minimize the sum of cluster diameters”. In: *J. Comput. Syst. Sci.* 68.2 (Mar. 2004), pp. 417–441. ISSN: 0022-0000.

[24] Moses Charikar et al. “Incremental Clustering and Dynamic Information Retrieval”. In: *SIAM J. Comput.* 33.6 (2004), pp. 1417–1440. DOI: https://doi.org/10.1137/S0097539702418498.

[25] Vaggos Chatziafratis, Rad Niazadeh, and Moses Charikar. “Hierarchical Clustering with Structural Constraints”. In: *Proceedings of the 35th International Conference on Machine Learning.* Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, Oct. 2018, pp. 774–783. URL: `https://proceedings.mlr.press/v80/chatziafratis18a.html`.

[26] Raymond Chen. “On Mentzer’s Hardness of the k-Center Problem on the Euclidean Plane”. In: (2021). URL: `https://digitalcommons.dartmouth.edu/cs_tr/383`.

[27] Vincent Cohen-Addad, Varun Kanade, and Frederik Mallmann-Trenn. “Clustering redemption–beyond the impossibility of kleinberg’s axioms”. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems.* NIPS’18. Montréal, Canada: Curran Associates Inc., 2018, pp. 8526–8535.

[28] Vincent Cohen-Addad et al. “Tight FPT Approximations for k-Median and k-Means”. In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).* Ed. by Christel Baier et al. Vol. 132. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019, 42:1–42:14. ISBN: 978-3-95977-109-2. DOI: `10.4230/LIPIcs.ICALP.2019.42`. URL: `https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ICALP.2019.42`.

[29] Vincent Cohen-addad et al. “Hierarchical Clustering: Objective Functions and Algorithms”. In: *J. ACM* 66.4 (June 2019). ISSN: 0004-5411. DOI: `10.1145/3321386`. URL: `https://doi.org/10.1145/3321386`.

[30] Aparna Das and Claire Kenyon-Mathieu. “On Hierarchical Diameter-Clustering and the Supplier Problem”. In: *Theory Comput. Syst.* 45.3 (2009), pp. 497–511. DOI: `https://doi.org/10.1007/s00224-009-9186-6`.

[31] Sanjoy Dasgupta. “A cost function for similarity-based hierarchical clustering”. In: *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing* (2015). URL: `https://api.semanticscholar.org/CorpusID:2262168`.

[32] Sanjoy Dasgupta and Philip M. Long. “Performance guarantees for hierarchical clustering”. In: *J. Comput. Syst. Sci.* 70.4 (2005), pp. 555–569. DOI: `https://doi.org/10.1016/j.jcss.2004.10.006`.

[33] Ian Davidson and S. S. Ravi. "Clustering With Constraints: Feasibility Issues and the $k$-Means Algorithm". In: *Proceedings of the 2005 SIAM International Conference on Data Mining (SDM)*, pp. 138–149. DOI: 10.1137/1.9781611972757.13. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611972757.13.

[34] Jozef Doboš. *Metric preserving functions.* Štroffek Košice, 1998.

[35] Lukas Drexler et al. "Approximating Fair k-Min-Sum-Radii in Euclidean Space". In: *Approximation and Online Algorithms - 21st International Workshop, WAOA 2023, Amsterdam, The Netherlands, September 7-8, 2023, Proceedings.* Ed. by Jaroslaw Byrka and Andreas Wiese. Vol. 14297. Lecture Notes in Computer Science. Springer, 2023, pp. 119–133. DOI: 10.1007/978-3-031-49815-2_9. URL: https://doi.org/10.1007/978-3-031-49815-2%5C_9.

[36] Lukas Drexler et al. *Connected k-Center and k-Diameter Clustering.* 2023. arXiv: 2211.02176 [cs.DS].

[37] Lukas Drexler et al. "Connected k-Center and k-Diameter Clustering". In: *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023).* Ed. by Kousha Etessami, Uriel Feige, and Gabriele Puppis. Vol. 261. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, 50:1–50:20. ISBN: 978-3-95977-278-5. DOI: 10.4230/LIPIcs.ICALP.2023.50. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ICALP.2023.50.

[38] Robert C. Edgar. "Search and clustering orders of magnitude faster than BLAST". In: *Bioinformatics* 26.19 (Aug. 2010), pp. 2460–2461. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btq461. eprint: https://academic.oup.com/bioinformatics/article-pdf/26/19/2460/48857155/bioinformatics_26_19_2460.pdf. URL: https://doi.org/10.1093/bioinformatics/btq461.

[39] Matthias Ehrgott. *Multicriteria Optimization (2. ed.)* Springer, 2005. ISBN: 978-3-540-21398-7. DOI: 10.1007/3-540-27659-9. URL: https://doi.org/10.1007/3-540-27659-9.

[40] Brian S. Everitt et al. *Cluster Analysis.* Wiley Series in Probability and Statistics. 2011.

[41] Tomás Feder and Daniel Greene. "Optimal Algorithms for Approximate Clustering". In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing.* STOC '88. Chicago, Illinois, USA: Association for

Computing Machinery, 1988, pp. 434–444. ISBN: 0897912640. DOI: 10.1145/62212.62255. URL: https://doi.org/10.1145/62212.62255.

[42] Zachary Friggstad and Mahya Jamshidian. “Improved Polynomial-Time Approximations for Clustering with Minimum Sum of Radii or Diameters”. In: *30th Annual European Symposium on Algorithms (ESA)*. Vol. 244. 2022, 56:1–56:14. URL: https://doi.org/10.4230/LIPIcs.ESA.2022.56.

[43] Zachary Friggstad and Mahya Jamshidian. “Improved Polynomial-Time Approximations for Clustering with Minimum Sum of Radii or Diameters”. In: *30th Annual European Symposium on Algorithms (ESA 2022)*. Ed. by Shiri Chechik et al. Vol. 244. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 56:1–56:14. ISBN: 978-3-95977-247-1. DOI: 10.4230/LIPIcs.ESA.2022.56. URL: https://drops-dev.dagstuhl.de/entities/document/10.4230/LIPIcs.ESA.2022.56.

[44] Limin Fu et al. “CD-HIT”. In: *Bioinformatics* 28.23 (Dec. 2012), pp. 3150–3152. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bts565. URL: https://doi.org/10.1093/bioinformatics/bts565.

[45] Jean Gallier. *Geometric methods and applications: for computer science and engineering*. Berlin, Heidelberg: Springer-Verlag, 2000. ISBN: 0387950443.

[46] Rong Ge et al. “Joint cluster analysis of attribute data and relationship data: The connected k-center problem, algorithms and applications”. In: *ACM Trans. Knowl. Discov. Data* 2.2 (July 2008). ISSN: 1556-4681. DOI: 10.1145/1376815.1376816. URL: https://doi.org/10.1145/1376815.1376816.

[47] Matt Gibson et al. “On Clustering to Minimize the Sum of Radii”. In: *SIAM Journal of Computing* 41.1 (2012), pp. 47–60. URL: https://doi.org/10.1137/100798144.

[48] Teofilo F. Gonzalez. “Clustering to minimize the maximum intercluster distance”. In: *Theoretical Computer Science* 38 (1985), pp. 293–306. ISSN: 0304-3975. DOI: https://doi.org/10.1016/0304-3975(85)90224-5. URL: https://www.sciencedirect.com/science/article/pii/0304397585902245.

[49] Anna Großwendt and Heiko Röglin. “Improved Analysis of Complete-Linkage Clustering”. In: *Algorithmica* 78.4 (2017), pp. 1131–1150. DOI: https://doi.org/10.1007/s00453-017-0284-6.

[50] Anna-Klara Großwendt. “Theoretical Analysis of Hierarchical Clustering and the Shadow Vertex Algorithm”. PhD thesis. University of Bonn, 2020. URL: http://hdl.handle.net/20.500.11811/8348.

[51] Pierre Hansen and Michel Delattre. "Complete-Link Cluster Analysis by Graph Coloring". In: *Journal of the American Statistical Association* 73.362 (1978), pp. 397–403. ISSN: 01621459. URL: http://www.jstor.org/stable/2286672 (visited on 06/07/2024).

[52] Dorit S. Hochbaum. "When are NP-hard location problems easy?" In: *Ann. Oper. Res.* 1.3 (1984), pp. 201–214. DOI: https://doi.org/10.1007/BF01874389.

[53] Dorit S. Hochbaum and David B. Shmoys. "A Best Possible Heuristic for the $k$-Center Problem". In: *Math. Oper. Res.* 10.2 (1985), pp. 180–184. DOI: https://doi.org/10.1287/moor.10.2.180.

[54] Dorit S. Hochbaum and David B. Shmoys. "A Unified Approach to Approximation Algorithms for Bottleneck Problems". In: *J. ACM* 33.3 (May 1986), pp. 533–550. ISSN: 0004-5411. DOI: 10.1145/5925.5933. URL: https://doi.org/10.1145/5925.5933.

[55] Simon J. Holgate et al. "New Data Systems and Products at the Permanent Service for Mean Sea Level". In: *Journal of Coastal Research* 29.3 (2013), pp. 493–504. DOI: 10.2112/JCOASTRES-D-12-00175.1. URL: https://doi.org/10.2112/JCOASTRES-D-12-00175.1.

[56] Wen-Lian Hsu and George L. Nemhauser. "Easy and hard bottleneck location problems". In: *Discret. Appl. Math.* 1.3 (1979), pp. 209–215. DOI: https://doi.org/10.1016/0166-218X(79)90044-1.

[57] Tanmay Inamdar and Kasturi R. Varadarajan. "Capacitated Sum-Of-Radii Clustering: An FPT Approximation". In: *Proc. of the 28th Annual European Symposium on Algorithms (ESA)*. Vol. 173. 2020, 62:1–62:17. URL: https://doi.org/10.4230/LIPIcs.ESA.2020.62.

[58] Anil K. Jain and Richard C. Dubes. "Algorithms for Clustering Data". In: 1988.

[59] Kamal Jain and Vijay V. Vazirani. "Approximation algorithms for metric facility location and $k$-Median problems using the primal-dual schema and Lagrangian relaxation". In: *Journal of the ACM* 48.2 (2001), pp. 274–296. URL: https://doi.org/10.1145/375827.375845.

[60] Richard M Karp. "Reducibility among combinatorial problems". In: *Complexity of computer computations*. Springer, 1972, pp. 85–103.

[61] Jon Kleinberg. "An Impossibility Theorem for Clustering". In: *Proceedings of the 15th International Conference on Neural Information Processing Systems*. NIPS'02. Cambridge, MA, USA: MIT Press, 2002, pp. 463–470.

[62] Nissan Lev-Tov and David Peleg. "Polynomial time approximation schemes for base station coverage with minimum total radii". In: *Computer Networks* 47.4 (2005), pp. 489–501. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2004.08.012. URL: https://www.sciencedirect.com/science/article/pii/S1389128604002427.

[63] Nissan Lev-Tov and David Peleg. "Polynomial time approximation schemes for base station coverage with minimum total radii". In: *Comput. Networks* 47.4 (2005), pp. 489–501. URL: https://doi.org/10.1016/j.comnet.2004.08.012.

[64] Guolong Lin et al. "A General Approach for Incremental Approximation and Hierarchical Clustering". In: *SIAM J. Comput.* 39.8 (2010), pp. 3633–3669. DOI: https://doi.org/10.1137/070698257.

[65] Stuart Mentzer. "Approximability of Metric Clustering Problems". In: (Mar. 1988).

[66] Himanshu Mittal et al. "A comprehensive survey of image segmentation: clustering methods, performance parameters, and benchmark datasets". In: *Multimedia Tools and Applications* 81.24 (Oct. 2022), pp. 35001–35026. ISSN: 1573-7721. DOI: 10.1007/s11042-021-10594-9. URL: https://doi.org/10.1007/s11042-021-10594-9.

[67] Simona Moldovanu et al. "Refining skin lesions classification performance using geometric features of superpixels". In: *Scientific Reports* 13.1 (July 2023), p. 11463. ISSN: 2045-2322. DOI: 10.1038/s41598-023-38706-5. URL: https://doi.org/10.1038/s41598-023-38706-5.

[68] Permanent Service for Mean Sea Level (PSMSL), 2024, "Tide Gauge Data", Retrieved 03 February 2022 from http://www.psmsl.org/data/obtaining/.

[69] Chris Sander and Reinhard Schneider. "Database of homology-derived protein structures and the structural meaning of sequence alignment". In: *Proteins: Structure, Function, and Bioinformatics* 9.1 (1991), pp. 56–68. DOI: https://doi.org/10.1002/prot.340090107. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.340090107.

[70] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. DOI: 10.1017/CBO9781107298019.

[71] Motahare Shekari and Milad Rostamian. "Brain tumor segmentation from MRI using FCM clustering, morphological reconstruction, and active contour". In: *Multimedia Tools and Applications* (Oct. 2023). ISSN: 1573-7721. DOI: 10.1007/s11042-023-17233-5. URL: https://doi.org/10.1007/s11042-023-17233-5.

[72] Bhuvaneshwari Shetty et al. “Skin lesion classification of dermoscopic images using machine learning and convolutional neural network”. In: *Scientific Reports* 12.1 (Oct. 2022), p. 18134. ISSN: 2045-2322. DOI: 10.1038/s41598-022-22644-9. URL: https://doi.org/10.1038/s41598-022-22644-9.

[73] Luis R. Soenksen et al. “Using deep learning for dermatologist-level detection of suspicious pigmented skin lesions from wide-field images”. In: *Science Translational Medicine* 13.581 (2021), eabb3652. DOI: 10.1126/scitranslmed.abb3652. URL: https://www.science.org/doi/abs/10.1126/scitranslmed.abb3652.

[74] Martin Steinegger and Johannes Söding. “Clustering huge protein sequence sets in linear time”. In: *Nature Communications* 9.1 (June 2018), p. 2542. ISSN: 2041-1723. DOI: 10.1038/s41467-018-04964-5. URL: https://doi.org/10.1038/s41467-018-04964-5.

[75] Jitendra V. Tembhurne et al. “Skin cancer detection using ensemble of machine learning and deep learning techniques”. In: *Multimedia Tools and Applications* 82.18 (July 2023), pp. 27501–27524. ISSN: 1573-7721. DOI: 10.1007/s11042-023-14697-3. URL: https://doi.org/10.1007/s11042-023-14697-3.

[76] Vincent Cohen-Addad Viallat et al. “Breaching the 2 LMP Approximation Barrier for Facility Location with Applications to $k$-Median”. In: *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 940–986. DOI: 10.1137/1.9781611977554.ch37. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611977554.ch37.

[77] Dingkang Wang and Yusu Wang. “An Improved Cost Function for Hierarchical Cluster Trees”. In: *J. Comput. Geom.* 11 (2018), pp. 283–331. URL: https://api.semanticscholar.org/CorpusID:54446891.

[78] Yuyan Wang and Benjamin Moseley. “An Objective for Hierarchical Clustering in Euclidean Space and Its Connection to Bisecting K-means”. In: *AAAI Conference on Artificial Intelligence*. 2020. URL: https://api.semanticscholar.org/CorpusID:213720113.

[79] Yinhao Wu et al. “Skin Cancer Classification With Deep Learning: A Systematic Review”. In: *Frontiers in Oncology* 12 (2022). ISSN: 2234-943X. DOI: 10.3389/fonc.2022.893972. URL: https://www.frontiersin.org/articles/10.3389/fonc.2022.893972.

[80] E. Alper Yildirim. “Two Algorithms for the Minimum Enclosing Ball Problem”. In: *SIAM Journal on Optimization* 19.3 (2008), pp. 1368–1391. URL: https://doi.org/10.1137/070690419.