Approximate Clustering via Metric Partitioning*

Sayan Bandyapadhyay[1] and Kasturi Varadarajan[2]

1 **Department of Computer Science, University of Iowa, Iowa City, USA**
sayan-bandyapadhyay@uiowa.edu
2 **Department of Computer Science, University of Iowa, Iowa City, USA**
kasturi-varadarajan@uiowa.edu

Abstract

In this paper we consider two metric covering/clustering problems – *Minimum Cost Covering Problem* (MCC) and k-clustering. In the MCC problem, we are given two point sets X (clients) and Y (servers), and a metric on $X \cup Y$. We would like to cover the clients by balls centered at the servers. The objective function to minimize is the sum of the α-th power of the radii of the balls. Here $\alpha \geq 1$ is a parameter of the problem (but not of a problem instance). MCC is closely related to the k-clustering problem. The main difference between k-clustering and MCC is that in k-clustering one needs to select k balls to cover the clients.

For any $\varepsilon > 0$, we describe quasi-polynomial time $(1+\varepsilon)$ approximation algorithms for both of the problems. However, in case of k-clustering the algorithm uses $(1+\varepsilon)k$ balls. Prior to our work, a 3^α and a c^α approximation were achieved by polynomial-time algorithms for MCC and k-clustering, respectively, where $c > 1$ is an absolute constant. These two problems are thus interesting examples of metric covering/clustering problems that admit $(1+\varepsilon)$-approximation (using $(1+\varepsilon)k$ balls in case of k-clustering), if one is willing to settle for quasi-polynomial time. In contrast, for the variant of MCC where α is part of the input, we show under standard assumptions that no polynomial time algorithm can achieve an approximation factor better than $O(\log |X|)$ for $\alpha \geq \log |X|$.

1998 ACM Subject Classification I.3.5 Computational Geometry and Object Modeling

Keywords and phrases Approximation Algorithms, Clustering, Covering, Probabilistic Partitions

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2016.15

1 Introduction

We consider two metric covering/clustering problems. In the first problem, we are given two point sets X (clients) and Y (servers), and a metric d on $X \cup Y$. For $z \in X \cup Y$ and $r \geq 0$, the ball $B(z,r)$ centered at z and having radius $r \geq 0$ is the set $\{y \in X \cup Y | d(z,y) \leq r\}$. A *cover* for a subset $P \subseteq X$ is a set of balls, each centered at a point of Y, whose union contains P. The *cost* of a set $\mathcal{B} = \{B_1, \ldots, B_k\}$ of balls, denoted by $\text{cost}(\mathcal{B})$, is $\sum_{i=1}^{k} r(B_i)^\alpha$, where $r(B_i)$ is the radius of B_i, and $\alpha \geq 1$ is a parameter of the problem (but not of a problem instance). The goal is to compute a minimum cost *cover* for the clients X. We refer to this problem as the *Minimum Cost Covering Problem* (MCC).

In the second problem, we are given a set X of n points, a metric d on X, and a positive

* This material is based upon work supported by the National Science Foundation under Grant CCF-1318996.

integer k. Unlike in the case of MCC, here each ball is centered at a point in X.[1] The *cost* $\text{cost}(\mathcal{B})$ of a set $\mathcal{B}$ of balls is defined exactly in the same way as in the case of MCC. The goal is to find a set $\mathcal{B}$ of k balls whose union contains all the points in X and $\text{cost}(\mathcal{B})$ is minimized. We refer to this problem as k-clustering.

Inspired by applications in wireless networks, MCC has been well studied [22]. One can consider the points in Y as the potential locations of mobile towers and the points in X as the locations of customers. A tower can be configured in a way so that it can serve the customers lying within a certain distance. But the service cost increases with the distance served. The goal is to serve all the customers minimizing the total cost. For modelling the energy needed for wireless transmission, it is common to consider the value of α to be at least 1.

For the MCC problem with $\alpha = 1$, a primal-dual algorithm of Charikar and Panigrahy [10] leads to an approximation guarantee of 3; their result generalizes to $\alpha \geq 1$, with an approximation guarantee of 3^α. The problem is known to be NP-hard for $\alpha > 1$, even when X and Y are points in the Euclidean plane [2]. The case $\alpha = 1$ has received particular attention. The first PTAS for the Euclidean plane was designed by Lev-Tov and Peleg [22]. Later, Gibson *et. al* [17] have designed a polynomial time exact algorithm for this problem when X and Y are points in the plane, and the underlying distance function d is either the l_1 or l_∞ metric. For the l_2 metric they also get an exact algorithm if one assumes two candidate solutions can be compared efficiently; without this assumption, they get a $(1+\varepsilon)$ approximation. Their algorithm is based on a separator theorem that, for any optimal solution, proves the existence of a balanced separator that intersects with at most 12 balls in the solution. In a different work they have also extended the exact algorithm to arbitrary metric spaces [16]. The running time is quasi-polynomial if the aspect ratio of the metric (ratio of maximum to minimum interpoint distance) is bounded by a polynomial in the number of points. When the aspect ratio is not polynomially bounded, they obtain a $(1+\varepsilon)$ approximation in quasi-polynomial time. Their algorithms are based on a partitioning of the metric space that intersects a small number of balls in the optimal cover.

When $\alpha > 1$, the structure that holds for $\alpha = 1$ breaks down. It is no longer the case, even in the Euclidean plane, that there is a good separator (or partition) that intersects a small number of balls in an optimal solution. In the case $\alpha = 2$ and the Euclidean plane, the objective function models the total area of the served region, which arises in many practical applications. Hence this particular version has been studied in a series of works. Chuzhoy developed an unpublished 9-factor approximation algorithm for this version. Freund and Rawitz [15] present this algorithm and give a primal fitting interpretation of the approximation factor. Bilo *et. al* [9] have extended the techniques of Lev-Tov and Peleg [22] to get a PTAS that works for any $\alpha \geq 1$ and for any fixed dimensional Euclidean space. The PTAS is based on a sophisticated use of the *shifting strategy* which is a popular technique in computational geometry for solving problems in $\mathbb{R}^d$ [13, 19]. For general metrics, however, the best known approximation guarantee for $\alpha > 1$ remains the already mentioned 3^α [10].

The k-clustering problem has applications in many fields including Data Mining, Machine Learning and Image Processing. Over the years it has been studied extensively from both theoretical and practical perspectives [9, 10, 12, 16, 17, 23]. The problem can be seen as a variant of MCC where $Y = X$ and at most k balls can be chosen to cover the points in X. As one might think, the constraint on the number of balls that can be used in k-clustering

[1] Our results do generalize to the problem where we distinguish between clients and servers as in the MCC.

makes it relatively harder than MCC. Thus all the hardness results for MCC also hold for k-clustering. For $\alpha = 1$, Charikar and Panigrahy [10] present a polynomial time algorithm with an approximation guarantee of about 3.504. Gibson *et. al* [16, 17] obtain the same results for k-clustering with $\alpha = 1$ as the ones described for MCC, both in $\mathbb{R}^d$ and arbitrary metrics. Recently, Salavatipour and Behsaz [8] have obtained a polynomial time exact algorithm for $\alpha = 1$ and metrics of unweighted graphs, if we assume that no singleton clusters are allowed. However, in case of $\alpha > 1$ the best known approximation factor (in polynomial time) for general metrics is c^α, for some absolute constant $c > 1$; this follows from the analysis of Charikar and Panigrahy [10], who explicitly study only the case $\alpha = 1$. In fact, no better polynomial time approximation is known even for the Euclidean plane. We note that though the polynomial time algorithm in [9] yields a $(1+\varepsilon)$ approximation for k-clustering in any fixed dimensional Euclidean space and for $\alpha \geq 1$, it can use $(1+\varepsilon)k$ balls.

In addition to k-clustering many other clustering problems (k-means, k-center, k-median etc.) have been well studied [5, 11, 24, 18].

In this paper we address the following interesting question. Can the techniques employed by [9] for fixed dimensional Euclidean spaces be generalized to give $(1+\varepsilon)$ approximation for MCC and k-clustering in any metric space? Our motivation for studying the problems in a metric context is partly that it includes two geometric contexts: (a) high dimensional Euclidean spaces; and (b) shortest path distance metric in the presence of polyhedral obstacles in $\mathbb{R}^2$ or $\mathbb{R}^3$.

1.1 Our Results and Techniques

In this paper we consider the metric MCC and k-clustering with $\alpha \geq 1$. For any $\epsilon > 0$, we design a $(1+\epsilon)$-factor approximation algorithm for MCC that runs in quasi-polynomial time, that is, in $2^{(\log mn/\varepsilon)^c}$ time, where $c > 0$ is a constant, $m = |Y|$, and $n = |X|$. We also have designed a similar algorithm for k-clustering that uses at most $(1+\varepsilon)k$ balls and yields a solution whose cost is at most $(1+\varepsilon)$ times the cost of an optimal k-clustering solution. The time complexity of the latter algorithm is also quasi-polynomial. As already noted, somewhat stronger guarantees are already known for the case $\alpha = 1$ of these problems [16], but the structural properties that hold for $\alpha = 1$ make it rather special.

The results in this paper should be compared with the polynomial time algorithms [10] that guarantee 3^α approximation for MCC and c^α approximation for k-clustering. The MCC and k-clustering are thus interesting examples of metric covering/clustering problems that admit $(1+\varepsilon)$-approximation (using $(1+\varepsilon)k$ balls in case of k-clustering), if one is willing to settle for quasi-polynomial time. From this perspective, our results are surprising, as most of the problems in general metrics are APX-hard. The MCC and k-clustering are also examples where the techniques used in fixed dimensional Euclidean spaces generalize nicely to metric spaces. This is in contrast to the *facility location problem* [3].

The algorithms that we have designed for both of the problems use similar techniques that exploit the following key property of optimal covers: there are only a "small" number of balls whose radius is "large". We can therefore afford to guess these balls by an explicit enumeration. However, there can be a "large" number of balls with "small" radius. To help 'find' these, we partition the metric space into blocks (or subsets) with at most half the original diameter, and recurse on each block. We have to pay a price for this recursion in the approximation guarantee. This price depends on the number of blocks in the partition that a small radius ball can intersect. (This is not an issue in the case $\alpha = 1$, where each ball that is not guessed intersects precisely one of the blocks [16].)

We are led to the following problem: is there a way to probabilistically partition a metric space into blocks of at most half the diameter, so that for any ball with "small" radius, the

expected number of blocks that intersect the ball can be nicely bounded? The celebrated partitioning algorithms of Bartal [6] and Fakcharoenphol, Rao, and Talwar [14] guarantee that the probability that such a ball is intersected by two or more blocks is nicely bounded. However, their bounds on the probability that a small ball is intersected do not directly imply a good bound on the expected number of blocks intersected by a small ball. Indeed, if one employs the partitioning algorithm of [14], the expected number of blocks intersected by a small ball can be quite "large" . Fortunately, the desired bound on the expectation can be shown to hold for the algorithm of Bartal [6], even though he did not study the expectation itself. We use a similar partitioning scheme and derive the expectation bound in Section 2, using an analysis that closely tracks previous work [1, 7, 20]. While the bound on the expectation is easily derived from previous work, our work is the first to study and fruitfully apply this bound.

The algorithms for MCC and k-clustering, which use the partitioning scheme of Section 2, are described in Section 3 and 4, respectively. In Section 5, we consider the approximability of a variant of the MCC where we allow α to be part of the input. For $\alpha \geq \log |X|$, we show, under standard complexity theoretic assumptions, that no polynomial (or quasi-polynomial) time algorithm for MCC can achieve an approximation factor better than $O(\log |X|)$. This partly explains the dependence on α of the running time of our algorithms.

2 The Partitioning Scheme

Let Z be a point set with an associated metric d, let $P \subseteq Z$ be a point set with at least 2 points, and $n \geq |P|$ be a parameter. For $Q \subseteq Z$, denote the maximum interpoint distance (or diameter) of Q by $\mathrm{diam}(Q)$. Consider any partition of P into subsets (or blocks) $\{P_1, P_2, \ldots, P_t\}$, where $2 \leq t \leq |P|$. Abusing notation, we will also view $\{P_1, P_2, \ldots, P_t\}$ as a sequence of blocks. We say that P_i non-terminally (resp. terminally) intersects a ball B if P_i intersects B and it is not (resp. it is) the last set in the sequence $P_1, P_2, \ldots, P_t$ that intersects B. We would like to find a partition $\{P_1, P_2, \ldots, P_t\}$ of P that ensures the following properties:

1. For each $1 \leq i \leq t$, $\mathrm{diam}(P_i) \leq \mathrm{diam}(P)/2$.
2. For any ball B (centered at some point in Z) of radius $r \leq \frac{\mathrm{diam}(P)}{16 \log n}$, the expected size of the set $\{i | P_i \cap B \neq \emptyset\}$ is at most $1 + c\frac{r}{\mathrm{diam}(P)} \log n$, where $c > 0$ is a constant. In other words, the expected number of blocks in the partition that intersect B is at most $1 + c\frac{r}{\mathrm{diam}(P)} \log n$.
3. For any ball B (centered at some point in Z) of radius $r \leq \frac{\mathrm{diam}(P)}{16 \log n}$, the expected number of blocks in the partition that non-terminally intersect B is at most $c\frac{r}{\mathrm{diam}(P)} \log n$, where $c > 0$ is a constant.

We note that the second property follows from the third, as the number of blocks that intersect ball B is at most one more than the number of blocks that non-terminally intersect B. We design a probabilistic partitioning algorithm that finds a partition with the desired properties. We refer the reader to the full version of the paper for the algorithm and its analysis [4]. We conclude by summarizing the result.

▶ **Theorem 1.** *Let Z be a point set with an associated metric d, let $P \subseteq Z$ be a point set with at least 2 points, and $n \geq |P|$ be a parameter. There is a polynomial-time probabilistic algorithm RAND-PARTITION(P) that partitions P into blocks $\{P_1, P_2, \ldots, P_t\}$ and has the following guarantees:*

1. *For each $1 \leq i \leq t$, $diam(P_i) \leq diam(P)/2$.*

2. *There is a constant $c > 0$ so that for any ball B (centered at some point in Z) of radius $r \leq \frac{diam(P)}{16 \log n}$, the expected size of the set $\{i | P_i \cap B \neq \emptyset\}$ is at most $1 + c\frac{r}{diam(P)} \log n$ and the expected number of blocks that non-terminally intersect B is at most $c\frac{r}{diam(P)} \log n$.*

3 Algorithm for MCC

We now describe our $(1+\epsilon)$-factor approximation algorithm for the MCC problem. Recall that we are given a set X of clients, a set Y of servers, and a metric d on $X \cup Y$. We wish to compute a cover for X with minimum cost. Let $m = |Y|$ and $n = |X|$.

For $P \subseteq X$, let $\text{opt}(P)$ denote some optimal cover for P. Denote by $\text{cost}(B)$ the cost of a ball B (the α-th power of B's radius) and by $\text{cost}(\mathcal{B})$ the cost $\sum_{B \in \mathcal{B}} \text{cost}(B)$ of a set $\mathcal{B}$ of balls.

To compute a cover for P, our algorithm first guesses the set $\mathcal{Q} \subseteq \text{opt}(P)$ consisting of all the large balls in $\text{opt}(P)$. As we note in the structure lemma below, we may assume that the number of large balls in $\text{opt}(P)$ is small. We then use the algorithm of Theorem 1 to partition P into $\{P_1, P_2, \ldots, P_t\}$. For each $1 \leq i \leq t$, we recursively compute a cover for the set $P_i' \subseteq P_i$ of points not covered by $\mathcal{Q}$.

To obtain an approximation guarantee for this algorithm, we use the guarantees of Theorem 1. With this overview, we proceed to the structure lemma and a complete description of the algorithm.

3.1 A Structure Lemma

It is not hard to show that for any $\gamma \geq 1$ and $P \subseteq X$ such that $\text{diam}(P)$ is at least a constant factor of $\text{diam}(X \cup Y)$, $\text{opt}(P)$ contains at most $(c/\gamma)^\alpha$ balls of radius at least $\text{diam}(P)/\gamma$. Here c is some absolute constant. The following structural lemma extends this fact.

▶ **Lemma 2.** *Let $P \subseteq X$, $0 < \lambda < 1$ and $\gamma \geq 1$, and suppose that $opt(P)$ does not contain any ball of radius greater than or equal to $2\alpha \cdot diam(P)/\lambda$. Then the number of balls in $opt(P)$ of radius greater than or equal to $diam(P)/\gamma$ is at most $c(\lambda, \gamma) := (9\alpha\gamma/\lambda)^\alpha$.*

Proof. Suppose that $\text{opt}(P)$ does not contain any ball of radius greater than or equal to $2\alpha \cdot \text{diam}(P)/\lambda$. Note that each ball in $\text{opt}(P)$ intersects P and has radius at most $2\alpha \cdot \text{diam}(P)/\lambda$. Thus the point set $\{z \in X \cup Y \mid z \in B \text{ for some } B \in \text{opt}(P)\}$ has diameter at most $\text{diam}(P) + 8\alpha \cdot \text{diam}(P)/\lambda \leq 9\alpha \cdot \text{diam}(P)/\lambda$. It follows that there is a ball centered at a point in Y, with radius at most $9\alpha \cdot \text{diam}(P)/\lambda$ that contains P.

Let t denote the number of balls in $\text{opt}(P)$ of radius greater than or equal to $\text{diam}(P)/\gamma$. By optimality of $\text{opt}(P)$, we have $t \cdot (\text{diam}(P)/\gamma)^\alpha \leq (9\alpha \cdot \text{diam}(P)/\lambda)^\alpha$. Thus $t \leq (9\alpha\gamma/\lambda)^\alpha$. ◀

3.2 The Algorithm

We may assume that the minimum distance between two points in X is 1. Let $L = 1 + \log(\text{diam}(X))$. As we want a $(1+\varepsilon)$-approximation, we fix a parameter $\lambda = \varepsilon/2L$. Let $\gamma = \frac{c \log n}{\lambda}$, where c is the constant in Theorem 1. Denote $\mathcal{D}$ to be the set of balls such that each ball is centered at a point of $y \in Y$ and has radius $r = d(x, y)$ for some $x \in X$. We note that for any $P \subseteq X$, any ball in $\text{opt}(P)$ must belong to this set. Note that $|\mathcal{D}| \leq mn$. Recall that $c(\lambda, \gamma) = (9\alpha\gamma/\lambda)^\alpha$.

With this terminology, the procedure POINT-COVER(P) described as Algorithm 1 returns a cover of $P \subseteq X$. If $|P|$ is smaller than some constant, then the procedure returns an

Algorithm 1 POINT-COVER(P)

Require: A subset $P \subseteq X$.
Ensure: A cover of the points in P.
1: **if** $|P|$ is smaller than some constant κ **then**
2: **return** a minimum solution by checking all covers with at most κ balls.
3: sol $\leftarrow$ the best cover with one ball
4: cost $\leftarrow$ cost(sol)
5: Let $\{P_1, \ldots, P_\tau\}$ be the set of nonempty subsets returned by RAND-PARTITION(P)
6: Let $\mathcal{B}$ be the set of balls in $\mathcal{D}$ having radius greater than $\frac{\text{diam}(P)}{\gamma}$
7: **for each** $\mathcal{Q} \subseteq \mathcal{B}$ of size at most $c(\lambda, \gamma)$ **do**
8: **for** $i = 1$ to τ **do**
9: Let $P_i' = \{p \in P_i \mid p \notin \bigcup_{B \in \mathcal{Q}} B\}$
10: $\mathcal{Q}' \leftarrow \mathcal{Q} \cup \bigcup_{i=1}^{\tau}$ POINT-COVER(P_i')
11: **if** cost($\mathcal{Q}'$) $<$ cost **then**
12: cost $\leftarrow$ cost($\mathcal{Q}'$)
13: sol $\leftarrow \mathcal{Q}'$
14: **return** sol

optimal solution by searching all covers with a constant number of balls. In the general case, one candidate solution is the best single ball solution. For the other candidate solutions, the procedure first computes a partition $\{P_1, \ldots, P_\tau\}$ of P, using the RAND-PARTITION(P) procedure. Here RAND-PARTITION(P) is called with $Z = X \cup Y$ and $n = |X| \geq |P|$. Then it iterates over all possible subsets of $\mathcal{D}$ of size at most $c(\lambda, \gamma)$ containing balls of radius greater than $\text{diam}(P)/\gamma$. For each such subset $\mathcal{Q}$ and $1 \leq i \leq \tau$, it computes the set $P_i' \subseteq P_i$ of points not covered by $\mathcal{Q}$. It then makes recursive calls and generates the candidate solution $\mathcal{Q} \cup \bigcup_{i=1}^{\tau}$ POINT-COVER(P_i'). Note that all the candidate solutions are actually valid covers for P. Among these candidate solutions the algorithm returns the best solution.

Our overall algorithm for MCC calls the procedure POINT-COVER(X) to get a cover of X.

3.3 Approximation Guarantee

For $P \subseteq X$, let level(P) denote the smallest non-negative integer i such that $\text{diam}(P) < 2^i$. As the minimum interpoint distance in X is 1, level(P) $= 0$ if and only if $|P| \leq 1$. Note that level(X) $\leq L$.

The following lemma bounds the quality of the approximation of our algorithm.

▶ **Lemma 3.** *POINT-COVER(P) returns a solution whose expected cost is at most* $(1 + \lambda)^l$ *cost(opt(P)), where* $l =$ *level(P).*

Proof. We prove this lemma using induction on l. If $l = 0$, then $|P| \leq 1$ and POINT-COVER(P) returns an optimal solution, whose cost is cost(opt(P)). Thus assume that $l \geq 1$ and the statement is true for subsets having level at most $l - 1$. Let $P \subseteq X$ be a point set with level(P) $= l$. If $|P|$ is smaller than the constant threshold κ, POINT-COVER(P) returns an optimal solution. So we may assume that $|P|$ is larger than this threshold. We have two cases.

Case 1: There is some ball in opt(P) whose radius is at least $2\alpha \cdot \text{diam}(P)/\lambda$. Let B denote such a ball and $r(B) \geq 2\alpha \cdot \text{diam}(P)/\lambda$ be its radius. Since $(1 + \lambda/2\alpha)r(B) \geq r(B) + \text{diam}(P)$,

the concentric ball of radius $(1+\lambda/2\alpha)r(B)$ contains P. It follows that there is a cover for P that consists of a single ball and has cost at most

$$(1+\lambda/2\alpha)^{\alpha} r(B)^{\alpha} \leq (1+\lambda)\text{cost}(\text{opt}(P)) \leq (1+\lambda)^{l}\text{cost}(\text{opt}(P)).$$

Case 2: There is no ball in $\text{opt}(P)$ whose radius is at least $2\alpha \cdot \text{diam}(P)/\lambda$. Let $\mathcal{Q}_0 \subseteq \text{opt}(P)$ contain those balls of radius at least $\text{diam}(P)/\gamma$. It follows from Lemma 2 that $|\mathcal{Q}_0| \leq c(\lambda,\gamma)$. Thus the algorithm considers a $\mathcal{Q}$ with $\mathcal{Q} = \mathcal{Q}_0$. Fix this iteration. Also fix the partition $\{P_1,\ldots,P_\tau\}$ of P computed by RAND-PARTITION(P). RAND-PARTITION ensures that $\text{diam}(P_i) \leq \text{diam}(P)/2$ for $1 \leq i \leq \tau$. Thus $\text{diam}(P_i') \leq \text{diam}(P)/2$ and the level of each P_i' is at most $l-1$. Hence by induction the expected value of $\text{cost}(\text{POINT-COVER}(P_i'))$ is at most $(1+\lambda)^{l-1}\text{cost}(\text{opt}(P_i'))$.

Let $\mathcal{S}' = \text{opt}(P) \setminus \mathcal{Q}_0$. We argue below that the expected value of $\sum_{i=1}^{\tau} \text{cost}(\text{opt}(P_i'))$ is at most $(1+\lambda)\text{cost}(\mathcal{S}')$. Assuming this, we have

$$\begin{aligned} E[\text{cost}(\mathcal{Q}_0 \cup \bigcup_{i=1}^{\tau} \text{POINT-COVER}(P_i'))] &\leq \text{cost}(\mathcal{Q}_0) + (1+\lambda)^{l-1} E[\sum_{i=1}^{\tau} \text{cost}(\text{opt}(P_i'))] \\ &\leq \text{cost}(\mathcal{Q}_0) + (1+\lambda)^{l}\text{cost}(\mathcal{S}') \\ &\leq (1+\lambda)^{l}\text{cost}(\text{opt}(P)). \end{aligned}$$

Thus POINT-COVER(P) returns a solution whose expected cost is at most $(1+\lambda)^{l}\text{cost}(\text{opt}(P))$, as desired.

We now argue that the expected value of $\sum_{i=1}^{\tau} \text{cost}(\text{opt}(P_i'))$ is at most $(1+\lambda)\text{cost}(\mathcal{S}')$. Let $\mathcal{B}_i$ consist of those balls in $\mathcal{S}'$ that intersect P_i. For $B \in \mathcal{S}'$, let $\mu(B)$ denote the number of blocks in the partition $\{P_1,\ldots,P_\tau\}$ that B intersects. Because $\mathcal{B}_i$ is a cover for P_i', we have $\text{cost}(\text{opt}(P_i')) \leq \text{cost}(\mathcal{B}_i)$. Thus

$$\sum_{i=1}^{\tau} \text{cost}(\text{opt}(P_i')) \leq \sum_{i=1}^{\tau} \text{cost}(\mathcal{B}_i) = \sum_{B \in \mathcal{S}'} \mu(B)\text{cost}(B).$$

By definition of $\mathcal{Q}_0$, any ball $B \in \mathcal{S}' = \text{opt}(P) \setminus \mathcal{Q}_0$ has radius at most $\frac{\text{diam}(P)}{\gamma} = \frac{\lambda \cdot \text{diam}(P)}{c \log n}$, where c is the constant in Theorem 1. We may assume that $c \geq 16$ and hence $\frac{\lambda \cdot \text{diam}(P)}{c \log n} \leq \frac{\text{diam}(P)}{16 \log n}$. Theorem 1 now implies that

$$E[\mu(B)] \leq 1 + \frac{c \cdot r(B) \log n}{\text{diam}(P)} \leq 1 + \frac{c \log n}{\text{diam}(P)} \cdot \frac{\lambda \cdot \text{diam}(P)}{c \log n} = 1 + \lambda.$$

Thus the expected value of $\sum_{i=1}^{\tau} \text{cost}(\text{opt}(P_i'))$ is at most

$$\sum_{B \in \mathcal{S}'} E[\mu(B)]\text{cost}(B) \leq (1+\lambda) \sum_{B \in \mathcal{S}'} \text{cost}(B) = (1+\lambda)\text{cost}(\mathcal{S}'),$$

as claimed. ◀

We conclude that the expected cost of the cover returned by POINT-COVER(X) is at most $(1+\lambda)^{L}\text{cost}(\text{opt}(X)) \leq (1+\varepsilon)\text{cost}(\text{opt}(X))$, since $\lambda = \varepsilon/2L$.

Now consider the time complexity of the algorithm. POINT-COVER(P) makes $(mn)^{O(c(\lambda,\gamma))}$ direct recursive calls on subsets of diameter at most $\text{diam}(P)/2$. Thus the overall time complexity of POINT-COVER(X) can be bounded by $(mn)^{O(c(\lambda,\gamma)L)}$. Plugging in $\lambda = \varepsilon/2L$, $\gamma = c \log n/\lambda$, and $c(\lambda,\gamma) = (9\alpha\gamma/\lambda)^{\alpha}$, we conclude

▶ **Theorem 4.** *There is an algorithm for MCC that runs in time* $(mn)^{O(\frac{\alpha L^2 \log n}{\epsilon^2})^\alpha L}$ *and returns a cover whose expected cost is at most* $(1+\varepsilon)$ *times the optimal. Here* L *is 1 plus the logarithm of the aspect ratio of* X*, that is, the ratio of the maximum and minimum interpoint distances in the client set* X*.*

Using relatively standard techniques, which we omit here, we can pre-process the input to ensure that the ratio of the maximum and minimum interpoint distances in X is upper bounded by a polynomial in $\frac{mn}{\varepsilon}$. However, this affects the optimal solution by a factor of at most $(1+\epsilon)$. After this pre-processing, we have $L = O(\log \frac{mn}{\varepsilon})$. Using the algorithm in Theorem 4 after the pre-processing, we obtain a $(1+\epsilon)$ approximation with the quasi-polynomial running time $O(2^{\log^{O(1)} mn})$. Here the $O(1)$ hides a constant that depends on α and ε.

4 Algorithm for k-clustering

Recall that in k-clustering we are given a set X of points, a metric d on X, and a positive integer k. Let $|X| = n$. For $P \subseteq X$ and integer $\kappa \geq 0$, let $\text{opt}(P, \kappa)$ denote an optimal solution of κ-clustering for P (using balls whose center can be any point in X). We reuse the notions of $\text{level}(P)$, $\text{cost}(B)$ and $\text{cost}(\mathcal{B})$ from Section 3, for a point set P, a ball B, and a set $\mathcal{B}$ of balls, respectively. Denote $\mathcal{D}$ to be the set of balls such that each ball is centered at a point of $y \in X$ and has radius $r = d(x, y)$ for some $x \in X$. We note that for any $P \subseteq X$, any ball in $\text{opt}(P, \kappa)$ must belong to this set. Note that $|\mathcal{D}| \leq n^2$.

To start with we prove a structure lemma for k-clustering.

▶ **Lemma 5.** *Let* $P \subseteq X$*,* κ *be a positive integer, and* $\gamma \geq 1$*. Then the number of balls in* $opt(P, \kappa)$ *of radius greater than or equal to* $diam(P)/\gamma$ *is at most* $c(\gamma) := \gamma^\alpha$*.*

Proof. Note that any ball centered at a point in P and having radius $\text{diam}(P)$ contains all the points of P. Now by definition of $\text{diam}(P)$ and $\mathcal{D}$, there is a point $x \in P$ such that the ball $B(x, \text{diam}(P)) \in \mathcal{D}$. Hence $\text{opt}(P, \kappa) \leq \text{diam}(P)^\alpha$.

Let t denote the number of balls in $\text{opt}(P, \kappa)$ of radius greater than or equal to $\text{diam}(P)/\gamma$. By optimality of $\text{opt}(P, \kappa)$, we have $t \cdot (\text{diam}(P)/\gamma)^\alpha \leq \text{diam}(P)^\alpha$. Thus $t \leq \gamma^\alpha$. ◀

Like in the case of MCC, we assume that the minimum distance between two points in X is 1. Let $L = 1 + \log(\text{diam}(X))$. We fix a parameter $\lambda = \varepsilon/6L$. Let $\gamma = \frac{c \log n}{\lambda}$, where c is the constant in Theorem 1.

We design a procedure $\text{CLUSTERING}(P, \kappa)$ (see Algorithm 2) that given a subset P of X and an integer κ, returns a set of at most $(1+3\lambda)^l \kappa$ balls whose union contains P, where $l = \text{level}(P)$. We overview this procedure, focussing on the differences from the procedure POINT-COVER() used to solve the MCC problem. In $\text{CLUSTERING}(P, \kappa)$, $\text{RAND-PARTITION}(P)$ is called with $Z = X$ and $n = |X| \geq |P|$. We require two properties of the partition $\{P_1, \ldots, P_\tau\}$ of P computed by $\text{RAND-PARTITION}(P)$. Let $\mathcal{Q}_0$ be the set containing the large balls of $\text{opt}(P, \kappa)$, that is, those with radius at least $\text{diam}(P)/\gamma$. Let $\mathcal{S}' = \text{opt}(P, \kappa) \setminus \mathcal{Q}_0$ denote the set of small balls, and let $\mathcal{S}'_i \subseteq \mathcal{S}'$ consist of those balls that contain at least one point in P_i that is not covered by $\mathcal{Q}_0$. We would like (a) $\sum_{i=1}^{\tau} \text{cost}(\mathcal{S}'_i) \leq (1+3\lambda)\text{cost}(\mathcal{S}')$, and (b) $\sum_{i=1}^{\tau} |\mathcal{S}'_i| \leq (1+3\lambda)|\mathcal{S}'|$. Theorem 1 ensures that each of (a) and (b) holds in expectation. However, we would like both (a) and (b) to hold simultaneously, not just in expectation. For this reason, we try $\Theta(\log n)$ independent random partitions in Line 6, ensuring that with high probability, properties (a) and (b) hold for at least one of them.

Algorithm 2 CLUSTERING(P, κ)

Require: A subset $P \subseteq X$, an integer κ.
Ensure: A set of balls whose union contains the points in P.

1: **if** $|P|$ is smaller than some constant β **then**
2: **return** a minimum solution by checking all solutions with at most $\min\{\kappa, \beta\}$ balls.
3: sol $\leftarrow$ the best solution with one ball
4: cost $\leftarrow$ cost(sol)
5: $l \leftarrow$ level(P)
6: **for all** $2\log_{3/2} n$ iterations **do**
7: Let $\{P_1, \ldots, P_\tau\}$ be the set of nonempty subsets returned by RAND-PARTITION(P)
8: Let $\mathcal{B}$ be the set of balls in $\mathcal{D}$ having radius greater than $\frac{\text{diam}(P)}{\gamma}$
9: **for each** $\mathcal{Q} \subseteq \mathcal{B}$ of size at most $c(\gamma)$ **do**
10: **for** $i = 1$ to τ **do**
11: Let $P'_i = \{p \in P_i \mid p \notin \bigcup_{B\in\mathcal{Q}} B\}$
12: **for each** $1 \le i \le \tau$ and $0 \le \kappa_1 \le (1{+}3\lambda)\kappa$ **do**
13: cluster$(P'_i, \kappa_1) \leftarrow$ CLUSTERING(P'_i, κ_1)
14: **for** $i = 0$ to $\tau - 1$ **do**
15: $R_i \leftarrow \bigcup_{j=i+1}^{\tau} P'_j$
16: **for** $\kappa_1 = 0$ to $(1{+}3\lambda)\kappa$ **do**
17: cluster$(R_{\tau-1}, \kappa_1) \leftarrow$ cluster(P'_τ, κ_1)
18: **for all** $i = \tau - 2$ to 0 and $0 \le \kappa_1 \le (1{+}3\lambda)\kappa$ **do**
19: $\kappa'_{\min} \leftarrow \arg\min_{\kappa':0\le\kappa'\le\kappa_1}$ cost(cluster(P'_{i+1}, κ') $\cup$ cluster$(R_{i+1}, \kappa_1 - \kappa')$)
20: cluster$(R_i, \kappa_1) \leftarrow$ cluster$(P'_{i+1}, \kappa'_{\min}) \cup$ cluster$(R_{i+1}, \kappa_1 - \kappa'_{\min})$
21: $\mathcal{Q}' \leftarrow \mathcal{Q} \cup$ cluster$(R_0, (1 + 3\lambda) \cdot (\kappa - |\mathcal{Q}|))$
22: **if** $|\mathcal{Q}'| \le (1{+}3\lambda)^l\kappa$ and cost$(\mathcal{Q}') <$ cost **then**
23: cost $\leftarrow$ cost$(\mathcal{Q}')$
24: sol $\leftarrow \mathcal{Q}'$
25: **return** sol

Now let us fix one of these $\Theta(\log n)$ trials where we got a partition $\{P_1, \ldots, P_\tau\}$ satisfying properties (a) and (b), and also fix an iteration in Line 9 where we have $\mathcal{Q} = \mathcal{Q}_0$. Let $P'_i \subseteq P_i$ be the points not covered by $\mathcal{Q}_0$. For each $1 \le i \le \tau$ and $0 \le \kappa_1 \le (1 + 3\lambda)\kappa$, we set cluster$(P'_i, \kappa_1)$ to be the cover obtained by recursively invoking CLUSTERING(P'_i, κ_1) (as in Line 13).

Let us call a tuple $(\kappa_1, \kappa_2, \ldots, \kappa_\tau)$ of integers *valid* if $0 \le \kappa_i \le (1 + 3\lambda)(\kappa - |\mathcal{Q}_0|)$ and $\sum_{i=1}^{\tau} \kappa_i \le (1 + 3\lambda)(\kappa - |\mathcal{Q}_0|)$. We would like to minimize $\sum_{i=1}^{\tau}$ cost(cluster(P'_i, κ_i)) over all valid tuples $(\kappa_1, \kappa_2, \ldots, \kappa_\tau)$. As there are too many valid tuples to allow explicit enumeration, we solve this optimization problem in Lines 14–21 via a dynamic programming approach.

This completes our overview. Our overall algorithm for k-clustering calls the procedure CLUSTERING(X, k). Next we give the approximation bound on the cost of the solution returned by CLUSTERING(P, κ).

▶ **Lemma 6.** *For any $P \subseteq X$ and an integer $\kappa \ge 1$, CLUSTERING(P, κ) returns a solution consisting of at most $(1 + 3\lambda)^l\kappa$ balls and with probability at least $1 - \frac{|P|-1}{n^2}$, the cost of the solution is at most $(1 + 3\lambda)^l$ cost$(opt(P, \kappa))$, where $l = level(P)$.*

We refer the reader to the full version of the paper [4] for the proof of Lemma 6. Overall, it is similar to the proof of Lemma 3, and the key differences have already been anticipated in our overview.

Since $\lambda = \varepsilon/6L$, $(1+3\lambda)^L \leq 1+\varepsilon$. Thus we conclude that with probability at least $1-\frac{1}{n}$, CLUSTERING(X,k) returns a solution with at most $(1+\varepsilon)k$ balls whose cost is at most $(1+\varepsilon)\text{cost}(\text{opt}(X,k))$.

Now consider the time complexity of the algorithm. CLUSTERING(P,κ) makes $n^{O(c(\gamma))}$ direct recursive calls on subsets of diameter at most $\text{diam}(P)/2$. Thus the overall time complexity of CLUSTERING(X,k) can be bounded by $n^{O(c(\gamma)L)}$. Plugging in $\lambda = \varepsilon/6L$, $\gamma = c\log n/\lambda$, and $c(\gamma) = \gamma^\alpha$, we conclude

▶ **Theorem 7.** *There is a randomized algorithm for k-clustering that runs in time $n^{O((\frac{L\log n}{\epsilon})^\alpha L)}$ and with probability at least $1-\frac{1}{n}$ returns a solution with at most $(1+\varepsilon)k$ balls whose cost is at most $(1+\varepsilon)$ times the optimal. Here L is 1 plus the logarithm of the aspect ratio of X, that is, the ratio of the maximum and minimum interpoint distances in the set X.*

5 Inapproximability Result

In this section we present an inapproximability result which complements the result in Section 3. In particular here we consider the case when α is not a constant. The heart of this result is a reduction from the dominating set problem. Given a graph $G=(V,E)$, a dominating set for G is a subset V' of V such that for any vertex $v \in V \setminus V'$, v is connected to at least one vertex of V' by an edge in E. The dominating set problem is defined as follows.

Dominating Set Problem (DSP)
INSTANCE: Graph $G=(V,E)$, positive integer $k \leq |V|$.
QUESTION: Is there a dominating set for G of size at most k?

The following inapproximability result is proved by Kann [21].

▶ **Theorem 8.** *There is a constant $c>0$ such that there is no polynomial-time $c\log|V|$-factor approximation algorithm for DSP assuming $\mathcal{P} \neq \mathcal{NP}$.*

The following theorem shows an inapproximability bound for MCC when $\alpha \geq \log|X|$.

▶ **Theorem 9.** *For $\alpha \geq \log|X|$, no polynomial time algorithm for MCC can achieve an approximation factor better than $c\log|X|$ assuming $\mathcal{P} \neq \mathcal{NP}$.*

Proof. To prove this theorem we show a reduction from DSP. Given an instance $(G=(V,E),k)$ of DSP we construct an instance of MCC. The instance of MCC consists of two sets of points X (clients) and Y (servers), and a metric d defined on $X \cup Y$. Let $V=\{v_1,v_2,\ldots,v_n\}$, where $n=|V|$. For each $v_i \in V$, Y contains a point y_i and X contains a point x_i. For any point $p \in X \cup Y$, $d(p,p)=0$. For $i,j \in [n]$, $d(x_i,y_j)$ is 1 if $i=j$ or the edge $(v_i,v_j) \in E$, and $d(x_i,y_j)$ is 3 otherwise. For $i,j \in [n]$ such that $i \neq j$, we set $d(x_i,x_j)=d(y_i,y_j)=2$.

Consider two nonadjacent vertices v_i and v_j. For any $x_t \in X$ such that $t \neq i,j$, $d(x_i,x_t)+d(x_t,y_j) \geq 3$. Similarly, for any $y_t \in Y$ such that $t \neq i,j$, $d(x_i,y_t)+d(y_t,y_j) \geq 3$. Thus d defines a metric. Next we will prove that G has a dominating set of size at most k iff the cost of covering the points in X using the balls around the points in Y is at most k.

Suppose G has a dominating set J of size at most k. For each vertex $v_j \in J$, build a radius 1 ball around y_j. We return this set of balls $\mathcal{B}$ as the solution of MCC. Now consider any point $x_i \in X$. If $v_i \in J$, then x_i is covered by the ball around y_i. Otherwise, there must be a vertex $v_j \in J$ such that $(v_i,v_j) \in E$. Then $d(x_i,y_j)$ is 1 and x_i is covered by the ball around y_j. Hence $\mathcal{B}$ is a valid solution of MCC with cost at most k.

Now suppose there is a solution $\mathcal{B}$ of MCC with cost at most k. If $k > |X|$, then V is a dominating set for G of size $|X| < k$. If $k \leq |X|$, our claim is that the radius of each ball in $\mathcal{B}$ is 1. Suppose one of the balls B has a radius more than 1. Then the way the instance of MCC is created the radius should be at least 3. Hence $k \geq 3^{\alpha} \geq 3^{\log |X|} > |X|$, which is a contradiction. Now consider the set of vertices J corresponding to the centers of balls in $\mathcal{B}$. It is not hard to see that J is a dominating set for G of size at most k.

Let OPT be the cost of any optimal solution of MCC for the instance (X, Y, d). Then by the properties of this reduction the size of any minimum dominating set for G is OPT. Thus if there is an approximation algorithm for MCC that gives a solution with cost $(c \log |X|)\cdot$OPT, then using the reduction we can produce a dominating set of size $(c \log |V|)\cdot$OPT. Then from Theorem 8 it follows that $\mathcal{P} = \mathcal{NP}$. This completes the proof of our theorem. ◀

6 Conclusions

One generalization of the MCC problem that has been studied [10, 9] includes fixed costs for opening the servers. As input, we are given two point sets X (clients) and Y (servers), a metric on $Z = X \cup Y$, and a *facility cost* $f_y \geq 0$ for each server $y \in Y$. The goal is to find a subset $Y' \subseteq Y$, and a set of balls $\{B_y \mid y \in Y' \text{ and } B_y \text{ is centered at } y\}$ that covers X, so as to minimize $\sum_{y \in Y'}(f_y + r(B_y)^{\alpha})$. It is not hard to see that our approach generalizes in a straightforward way to give a $(1+\varepsilon)$ approximation to this problem using quasi-polynomial running time. To keep the exposition clear, we have focussed on the MCC rather than this generalization.

The main open problem that emerges from our work is whether there one can obtain a $(1+\varepsilon)$-approximation for the k-clustering problem in quasi-polynomial time.

Acknowledgements. We would like to thank an anonymous reviewer of an earlier version of this paper for suggestions that improved the guarantees and simplified the proof of Theorem 1. We also thank other reviewers for their feedback and pointers to the literature.

References

1 Ittai Abraham, Yair Bartal, and Ofer Neimany. Advances in metric embedding theory. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 271–286. ACM, 2006.

2 Helmut Alt, Esther M. Arkin, Hervé Brönnimann, Jeff Erickson, Sándor P. Fekete, Christian Knauer, Jonathan Lenchner, Joseph S. B. Mitchell, and Kim Whittlesey. Minimum-cost coverage of point sets by disks. In *Proceedings of the 22nd ACM Symposium on Computational Geometry, Sedona, Arizona, USA, June 5-7, 2006*, pages 449–458, 2006. doi:10.1145/1137856.1137922.

3 Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for euclidean k-medians and related problems. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC'98, pages 106–113, New York, NY, USA, 1998. ACM. doi:10.1145/276698.276718.

4 Sayan Bandyapadhyay and Kasturi R. Varadarajan. Approximate clustering via metric partitioning. *CoRR*, abs/1507.02222, 2015. URL: http://arxiv.org/abs/1507.02222.

5 Sayan Bandyapadhyay and Kasturi R. Varadarajan. On variants of k-means clustering. In *32nd International Symposium on Computational Geometry, SoCG 2016, June 14-18, 2016, Boston, MA, USA*, pages 14:1–14:15, 2016. doi:10.4230/LIPIcs.SoCG.2016.14.

6 Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS'96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193. IEEE Computer Society, 1996. URL: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4141, doi:10.1109/SFCS.1996.548477.

7 Yair Bartal. Graph decomposition lemmas and their role in metric embedding methods. In *Algorithms–ESA 2004*, pages 89–97. Springer, 2004.

8 Babak Behsaz and Mohammad R. Salavatipour. On minimum sum of radii and diameters clustering. In *Algorithm Theory – SWAT 2012 – 13th Scandinavian Symposium and Workshops, Helsinki, Finland, July 4-6, 2012. Proceedings*, pages 71–82, 2012. doi:10.1007/978-3-642-31155-0_7.

9 Vittorio Bilò, Ioannis Caragiannis, Christos Kaklamanis, and Panagiotis Kanellopoulos. Geometric clustering to minimize the sum of cluster sizes. In *Algorithms – ESA 2005, 13th Annual European Symposium, Palma de Mallorca, Spain, October 3-6, 2005, Proceedings*, pages 460–471, 2005. doi:10.1007/11561071_42.

10 Moses Charikar and Rina Panigrahy. Clustering to minimize the sum of cluster diameters. *J. Comput. Syst. Sci.*, 68(2):417–441, 2004. doi:10.1016/j.jcss.2003.07.014.

11 Vincent Cohen-Addad, Philip N. Klein, and Claire Mathieu. Local search yields approximation schemes for k-means and k-median in euclidean and minor-free metrics. In *FOCS, to appear*, 2016.

12 Srinivas Doddi, Madhav V. Marathe, S. S. Ravi, David Scot Taylor, and Peter Widmayer. Approximation algorithms for clustering to minimize the sum of diameters. *Nord. J. Comput.*, 7(3):185–203, 2000.

13 Thomas Erlebach, Klaus Jansen, and Eike Seidel. Polynomial-time approximation schemes for geometric intersection graphs. *SIAM J. Comput.*, 34(6):1302–1323, 2005. doi:10.1137/S0097539702402676.

14 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004. doi:10.1016/j.jcss.2004.04.011.

15 Ari Freund and Dror Rawitz. Combinatorial interpretations of dual fitting and primal fitting. In *Approximation and Online Algorithms, First International Workshop, WAOA 2003, Budapest, Hungary, September 16-18, 2003, Revised Papers*, pages 137–150, 2003. doi:10.1007/978-3-540-24592-6_11.

16 Matt Gibson, Gaurav Kanade, Erik Krohn, Imran A. Pirwani, and Kasturi R. Varadarajan. On metric clustering to minimize the sum of radii. *Algorithmica*, 57(3):484–498, 2010. doi:10.1007/s00453-009-9282-7.

17 Matt Gibson, Gaurav Kanade, Erik Krohn, Imran A. Pirwani, and Kasturi R. Varadarajan. On clustering to minimize the sum of radii. *SIAM J. Comput.*, 41(1):47–60, 2012. doi:10.1137/100798144.

18 Sariel Har-Peled. Geometric approximation algorithms, 2011.

19 Dorit S. Hochbaum and Wolfgang Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *J. ACM*, 32(1):130–136, 1985. doi:10.1145/2455.214106.

20 Lior Kamma, Robert Krauthgamer, and Huy L Nguyên. Cutting corners cheaply, or how to remove steiner points. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1029–1040. SIAM, 2014.

21 Viggo Kann. On the approximability of np-complete optimization problems. *PhD thesis, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm*, 1992. URL: http://www.csc.kth.se/~viggo/papers/phdthesis.pdf.

22 Nissan Lev-Tov and David Peleg. Polynomial time approximation schemes for base station coverage with minimum total radii. *Computer Networks*, 47(4):489–501, 2005. doi:10.1016/j.comnet.2004.08.012.

23 Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007. doi:10.1016/j.cosrev.2007.05.001.

24 Mohammad Salavatipour Zachary Friggstad, Mohsen Rezapour. Local search yields a ptas for k-means in doubling metrics. In *FOCS, to appear*, 2016.