A CONSTANT-FACTOR APPROXIMATION FOR MULTI-COVERING WITH DISKS*

Santanu Bhowmick,† *Kasturi Varadarajan,*‡ *and Shi-Ke Xue*§

Abstract. We consider the following multi-covering problem with disks. We are given two point sets Y (servers) and X (clients) in the plane, a coverage function $\kappa : X \to \mathbb{N}$, and a constant $\alpha \geq 1$. Centered at each server is a single disk whose radius we are free to set. The requirement is that each client $x \in X$ be covered by at least $\kappa(x)$ of the server disks. The objective function we wish to minimize is the sum of the α-th powers of the disk radii. We present a polynomial-time algorithm for this problem achieving an $O(1)$ approximation.

1 Introduction

We begin with the statement of the problem studied in this article. We are given two point sets Y (servers) and X (clients) in the plane, a coverage function $\kappa : X \to \mathbb{N}$, and a constant $\alpha \geq 1$. An assignment $r : Y \to \mathbb{R}^+$ of *radii* to the points in Y corresponds to "building" a disk of radius r_y centered at each $y \in Y$. For an integer $j \geq 0$, let us say that a point $x \in X$ is j-covered under such an assignment r if x is contained in at least j of the disks, i.e.

$$|\{y \in Y \mid ||y - x||_2 \leq r_y\}| \geq j$$

The goal is to find an assignment r that $\kappa(x)$-covers each point $x \in X$ and minimizes $\sum_{y \in Y} r_y^\alpha$. We call this the *non-uniform minimum-cost multi cover* problem (non-uniform MCMC problem).

We are interested in designing a polynomial time algorithm that outputs a solution whose cost is at most some factor $f \geq 1$ times the cost of an optimal solution. We call such an algorithm an f-approximation, and it is implicit that the algorithm is actually polynomial-time.

The version of this problem where $\kappa(x) = k$, $\forall x \in X$, for some given $k > 0$, has received particular attention. Here, all the clients have the same coverage requirement of k. We will refer to this as the *uniform MCMC* problem. In the context of the uniform MCMC, we will refer to a j-cover as an assignment of radii to the servers under which each client is j-covered.

*A preliminary version of this article appeared as [6].

†Department of Computer Science; University of Iowa; Iowa City, USA; santanu-bhowmick@uiowa.edu

‡Department of Computer Science; University of Iowa; Iowa City, USA; kasturi-varadarajan@uiowa.edu

§Department of Computer Science; Massachusetts Institute of Technology; Cambridge, USA; shikexue@mit.edu

1.1 Related Work

In the rest of this section, we will focus on the uniform MCMC problem, and be specific when remarking on generalizations to the non-uniform problem. The (uniform) MCMC problem was considered in two recent papers, motivated by fault-tolerant sensor network design that optimizes energy consumption. Abu-Affash et al. [1] considered the case $\alpha = 2$, which corresponds to minimizing the sum of the areas of the server disks. They gave an $O(k)$ approximation for the problem using mainly geometric ideas; more explicitly, the approximation factor they guarantee is $23.02 + 63.95(k-1)$. Bar-Yehuda and Rawitz [4] gave another algorithm that achieves the same approximation factor of $O(k)$ for any α. (The explicit approximation factor is $3^\alpha k$.) The central question that we investigate in this article is whether an approximation guarantee that is independent of k is possible. The problem is known to be NP-hard even for $k = 1$ and any $\alpha > 1$. This was shown by Bilò et al. [5] for $\alpha \geq 2$, and subsequently by Alt et al. [2] for any $\alpha > 1$.

There is a considerable amount of work on clustering and covering problems related to the MCMC problem, and we refer the reader to the previous papers for a detailed survey [1, 4]. Here, we offer a view of some of that work from the standpoint of techniques that may be applicable to the problem at hand. For the case $k = 1$ of the problem, constant factor approximations can be obtained using approaches based on linear programming, and in particular, the primal-dual method [9, 12]. The $O(k)$ approximation of Bar-Yehuda and Rawitz [4] for $k > 1$ can be situated in this line of work.

There has been some recent work on the geometric set multi-covering problem [10, 3]. In particular, the recent work of Bansal and Pruhs [3] addresses the following problem. We are given a set of points in the plane, a set of disks each with an arbitrary non-negative weight, and an integer k. The goal is to pick a subset of the disks so that each of the given points is covered at least k times. The objective function we want to minimize is the sum of the weights of the chosen disks. Bansal and Pruhs [3] give an $O(1)$ approximation for the problem, building on techniques developed for the case $k = 1$ [15, 8].

It would seem that the problem considered in this paper can be reduced to the problem solved by Bansal and Pruhs: for each $y \in Y$ and $x \in X$, add a disk centered at y with weight $||x - y||_2^\alpha$, and let X be the set of points that need to be covered. The reason this reduction does not work is that we have to add an additional constraint saying that we can use only one disk centered at each $y \in Y$. Notice that this additional constraint is not an issue for the case $k = 1$, since here if the returned solution uses two disks centered at the same $y \in Y$, we can simply discard the smaller one.

In the geometric set cover problems considered by [10, 15, 8, 3], the input disks are "immutable", and the complexity of the problem stems from the combinatorial geometry of the disks. For the MCMC application, it would be more fruitful to consider geometric set multi-cover problems where the algorithm is allowed to slightly enlarge the input disks. This version of covering with $k = 1$ is considered by Har-Peled and Lee [13]. For $k > 1$, however, we still have the above-mentioned difficulty of reducing MCMC to set multi-cover.

The case $k = 1$ of our MCMC problem actually admits a polynomial-time approximation scheme (PTAS) using dynamic programming on top of randomly shifted quad-trees

[11, 7]. This was shown by Lev-Tov and Peleg [14] for $\alpha = 1$, and subsequently by Bilò et al. [5] for any $\alpha \geq 1$. The difficulty with extending these results for $k = 1$ to general k is that the "density" of the solution grows with k, and therefore the number of sub-problems that the dynamic program needs to solve becomes exponential in k. It is conceivable that further discretization tricks [13] can be employed to get around this difficulty, but we have not succeeded in this effort. On the other hand, we are also not aware of any hardness result that rules out a PTAS.

1.2 Our Results

In this article, we obtain an $O(1)$ approximation for the uniform MCMC problem. That is, we demonstrate an approximation bound that is independent of k. More explicitly, our approximation guarantee is $4 \cdot (27\sqrt{2})^{\alpha}$.

Our approach revolves around the notion of an *outer cover*. This is an assignment of radii to the servers under which each client $x \in X$ is covered by a disk of radius at least $||y^k(x) - x||_2$, where $y^k(x)$ is the k-th nearest neighbor of x in Y. To motivate the notion, consider any k-cover, and in particular, the optimal one. Consider the set of disks obtained by picking, for each client $x \in X$, the largest disk covering x in the k-cover. (Several clients can "pick" the same disk.) This set of disks is seen to be an outer cover.

We provide a mechanism for extending any $(k-1)$-cover to a k-cover so that the increase in objective function cost is bounded by a constant times the cost of an optimal outer cover. This naturally leads to our algorithm in Section 4 – recursively compute a $(k-1)$-cover and then extend it to a k-cover. To bound its approximation ratio, we argue in Section 5 that the optimal solution can be partitioned into a $(k-1)$-cover and another set of disks that is almost an outer cover. Finally, we need a module for computing an approximately optimal outer cover. We show in Section 3 that an existing primal-dual algorithm for 1-covering can be generalized for this purpose.

The idea of an outer cover has its origins in the notion of *primary disks* used by Abu-Affash et al. [1]. Our work develops the idea and its significance much further, and this is partly what enables our $O(1)$ approximation bound.

Our algorithm and approximation guarantee of $O(1)$ works for the non-uniform MCMC problem as well. We therefore present our work in this slighly more general setting.

2 Preliminaries

For convenience, we solve the variant of the non-uniform MCMC problem where we have l_∞ disks rather than l_2 disks. Our input is two point sets Y and X in $\mathbb{R}^2$, a coverage function $\kappa : X \to \mathbb{N} \cup \{0\}$, and the constant $\alpha \geq 1$. (It will be useful to allow $\kappa(x)$ to be 0 for some $x \in X$.) We also assume that $\kappa(x) \leq |Y|$ for each $x \in X$, for otherwise there is no feasible solution.

We describe an algorithm for assigning a *radius* $r_y \geq 0$ for each $y \in Y$, with the guarantee that for each $x \in X$, there are at least $\kappa(x)$ points $y \in Y$ such that the l_∞ disk

of radius r_y centered at y contains x. In other words the guarantee is that for each $x \in X$,

$$|\{y \in Y \mid ||x - y||_\infty \leq r_y\}| \geq \kappa(x)$$

Our objective is to minimize $\sum_{y \in Y} r_y^\alpha$. For this optimization problem, we will show that our algorithm outputs an $O(1)$ approximation. Clearly, this also gives an $O(1)$ approximation for the original problem, where distances are measured in the l_2 norm. We will use $||\cdot||$ to denote the l_∞ norm.

For each $x \in X$, fix an ordering of the points in Y that is non-decreasing in terms of l_∞ distance to x. For $1 \leq j \leq |Y|$, let $y^j(x)$ denote the j-th point in this ordering. In other words, $y^j(x)$ is the j-th closest point in Y to x. For brevity, we denote $y^{\kappa(x)}(x)$ by $y^\kappa(x)$.

Let $\delta(p, r)$ denote the l_∞ disk of radius r centered at p. The *cost* of a set of disks is defined to be the sum of the α-th powers of the radii of the disks. The cost of an assignment of radii to the servers is defined to be the cost of the corresponding set of disks.

3 OuterCover: Algorithm to generate a preliminary cover

Given $X' \subseteq X$, Y, κ and $\alpha \geq 1$, an *outer cover* is an assignment $\rho : Y \to \mathbb{R}^+$ of radii to the servers such that for each client $x \in X'$, there is a server $y \in Y$ such that

1. The disk $\delta(y, \rho_y)$ contains x
2. Disk radius $\rho_y \geq ||x - y^\kappa(x)||$

Our goal in this section is to compute an outer cover that minimizes the cost $\sum_y \rho_y^\alpha$. In the rest of this section, we describe and analyze a procedure OuterCover(X', Y, κ, α) that returns an outer cover $\rho : Y \to \mathbb{R}^+$ whose cost is $O(1)$ times that of an optimal outer cover. Since this result is used as a black box in our algorithm for the non-uniform MCMC, the remainder of this section could be skipped on a first reading.

The procedure OuterCover(X', Y, κ, α) is implemented via a modification of the primal-dual algorithm of Charikar and Panigrahy [9]. Note that their algorithm can be viewed as solving the case where $\kappa(x) = 1$ for each $x \in X'$. As we will see, their algorithm and analysis readily generalize to the problem of computing an outer cover.

3.1 Linear Programming Formulation

We begin by formulating the problem of finding an optimal outer cover as an integer program. For each server $y_i \in Y$ and radius $r \geq 0$, let $z_i^{(r)}$ be an indicator variable that denotes whether the disk $\delta(y_i, r)$ is chosen in the outer cover.[1] For any server $y_i \in Y$ and client $x_j \in X'$, we define the *minimum eligible radius* $R_{\min}(y_i, x_j)$ to be:

$$R_{\min}(y_i, x_j) = \max(||y_i - x_j||, ||y^\kappa(x_j) - x_j||)$$

[1] For a server $y_i \in Y$, only the disks whose radius is from the set $\{||y - x|| \mid y \in Y, x \in X'\}$ will play a role in much of our algorithm. For describing the algorithm, however, it will be convenient to allow any $r \geq 0$.

A disk centered at y_i serves x_j in an outer cover if and only if its radius is at least $R_{\min}(y_i, x_j)$. Finally, let $C_i(r) = \{x_j \in X' \mid r \geq R_{\min}(y_i, x_j)\}$. The set $C_i(r)$ consists of those clients that $\delta(y_i, r)$ can serve.

The problem of computing an optimal outer cover is that of minimizing

$$\sum_{i,r} r^\alpha \cdot z_i^{(r)}, \tag{1}$$

subject to the constraints

$$\sum_{i,r:x_j \in C_i(r)} z_i^{(r)} \geq 1, \ \forall x_j \in X' \tag{2}$$

$$z_i^{(r)} \in \{0,1\}, \ \forall y_i, r. \tag{3}$$

The first constraint, equation (2), represents the condition that for every client $x_j \in X'$, at least one disk that is capable of serving it is chosen. The second constraint, equation (3), models the fact that the indicator variables $z_i^{(r)}$ can only take boolean values $\{0,1\}$. By relaxing the indicator variables to be simply non-negative, i.e.

$$z_i^{(r)} \geq 0, \ \forall y_i, r, \tag{4}$$

we get a linear program (LP), which we call the primal LP for the problem.

The dual of the above LP has a variable β_j corresponding to every client $x_j \in X'$. The dual LP seeks to maximize

$$\sum_{x_j \in X'} \beta_j, \tag{5}$$

subject to the constraints

$$\sum_{x_j \in C_i(r)} \beta_j \leq r^\alpha, \ \forall y_i, r \tag{6}$$

$$\beta_j \geq 0, \ \forall x_j \in X' \tag{7}$$

3.2 A Primal-Dual Algorithm

The primal-dual algorithm is motivated by the above linear program. The algorithm maintains a dual variable β_j for each client x_j. This variable will always be non-negative and satisfy the dual constraints (6). If at some point in the algorithm, the dual constraint (6) holds with equality for some y_i and r, the disk $\delta(y_i, r)$ is said to be *tight*. A client x_j is said to be tight if there is some tight disk $\delta(y_i, r)$ such that $x_j \in C_i(r)$. (Note that β_j is then part of the dual constraint (6) that holds with equality.)

Our algorithm, OuterCover(X', Y, κ, α), initializes each β_j to 0, which clearly satisfies (6). The goal of the while loop in lines 2 and 3, which we refer to as the *covering*

phase of the algorithm, is to ensure that each client in X' becomes tight, that is, covered by some tight disk. It follows from the termination condition of the while loop that the covering phase achieves this, provided the while loop terminates. We argue below that the while loop indeed terminates.

Algorithm 1 OuterCover(X', Y, κ, α)

1: Let $\beta_j \leftarrow 0$ for each $x_j \in X'$.
2: **while** $\exists\ x_j \in X'$ that is not tight **do**
3: Increase the non-tight variables β_j arbitrarily till some constraint in (6) that was not tight becomes tight.
4: Let T be the set of those tight disks that contain clients.
5: $F \leftarrow \varnothing$
6: **while** $T \neq \varnothing$ **do**
7: $\delta(y_i, r) \leftarrow$ The disk of largest radius in T
8: $N \leftarrow$ Set of disks that intersect $\delta(y_i, r)$
9: $F \leftarrow F \cup \{\delta(y_i, r)\}$
10: $T \leftarrow T \setminus N$
11: Assign $\rho : Y \rightarrow \mathbb{R}^+$ as follows:

$$\forall\ y_i \in Y, \rho(y_i) = \begin{cases} 3r, & \text{if } \delta(y_i, r) \in F \\ 0, & \text{if } F \text{ contains no disk centered at } y_i \end{cases}$$

Steps 4–10 constitute the *coarsening phase* of the algorithm. This phase starts with the set T of tight disks computed by the covering phase. It computes a subset $F \subseteq T$ of pairwise disjoint disks by considering the disks in T in non-increasing order of radii, and adding a disk to F if it does not intersect any previously added disk.

Step 11 constitutes the *enlargement phase*. Each disk in F is expanded by a factor of 3, and the resulting set of disks is returned by the algorithm. Because the disks in F are pairwise disjoint, it follows that F contains at most one disk centered at y_i, for any $y_i \in Y$. Thus the assignment in Step 11 is well defined.

Having described the algorithm, we now address its correctness. We begin by establishing that the covering phase terminates.

Claim 1. *The while loop in the covering phase terminates.*

Proof. This follows from two observations:

1. In each iteration of the while loop, at least one disk $\delta(y_i, r)$ that was not tight at the beginning of that iteration becomes tight. Furthermore, for such a disk $\delta(y_i, r)$, there is at least one client x_j in $C_i(r)$ that was not tight at the beginning of the iteration. Because all clients in $C_i(r)$ are tight at the end of the iteration, x_j is also tight at the end of the iteration. Thus, client x_j was not tight at the beginning of the iteration but is tight at the end of the iteration. We conclude that in each iteration, some client

that was not tight at the beginning of the iteration becomes tight at the end of the iteration.

2. Since the β_j are never decreased in the covering phase, a disk that becomes tight at some point remains tight for the rest of the phase. This implies that once a client becomes tight, it remains tight for the rest of the phase.

These two observations imply that the number of iterations in the while loop is at most $|X'|$. The claim follows. □

We now argue that there are only polynomially many disks in the set T that is computed in line 4.

Claim 2. *Let $\Gamma = \{||y - x|| \mid y \in Y, x \in X'\}$. For any disk $\delta(y_i, r) \in T$, it must be that $r \in \Gamma$.*

Proof. Let $\delta(y_i, r)$ be a disk that contains some client $x \in X'$. We will show that if $r \notin \Gamma$, then $\delta(y_i, r)$ does not become tight in the covering phase.

Let us assume $r \notin \Gamma$. Then there is an $r' \in \Gamma$ such that $r' < r$ and the interval (r', r) contains no element from Γ. From the definition of Γ, we can verify that $C_i(r) = C_i(r')$. Thus,

$$\sum_{x_j \in C_i(r)} \beta_j = \sum_{x_j \in C_i(r')} \beta_j \leq (r')^\alpha < r^\alpha,$$

where the first inequality follows from dual feasibility (6) applied to y_i and r', which the covering phase maintains. Thus the inequality

$$\sum_{x_j \in C_i(r)} \beta_j < r^\alpha$$

holds during the covering phase, and $\delta(y_i, r)$ does not become tight. □

We note that this argument also shows why it is sufficient for the algorithm to consider in its covering phase only those disks whose radius is in Γ. If (6) holds for such disks, it holds for the other disks as well.

We now argue that our algorithm returns an outer cover.

Claim 3. *The disks returned by OuterCover(X', Y, κ, α) form an outer cover.*

Proof. Consider any client $x_j \in X'$. Since x_j is tight at the end of the covering phase, there is a tight disk $\delta(y_i, r) \in T$ such that $x_j \in C_i(r)$. Thus x_j is served in case $\delta(y_i, r)$ was added to F in the coarsening phase. If $\delta(y_i, r)$ was not added to F, then it must have been intersected by some disk $\delta(y_{i'}, r')$ that was added to F, such that $r' \geq r$. Clearly, $x_j \in \delta(y_{i'}, 3r')$. Furthermore, $3r' \geq r \geq ||y^\kappa(x_j) - x_j||$. Thus, $x_j \in C_{i'}(3r')$, and x_j is served by the output of OuterCover(X', Y, κ, α). □

Finally, we bound the approximation ratio of our algorithm and conclude with the main result of this section.

Lemma 1. *The algorithm OuterCover(X', Y, κ, α) runs in polynomial time and returns an outer cover whose cost is at most 3^α times that of an optimal outer cover.*

Proof. We have already established that the algorithm returns an outer cover. It is readily seem that the running time is polynomial.

Let the set of disks in an optimal outer cover be denoted by OPT. We now show that the cost of the outer cover returned by OuterCover(X', Y, κ, α) is at most $3^\alpha \cdot \text{cost}(OPT)$. We begin by lower bounding $\text{cost}(OPT)$ in terms of the β_j. We have

$$\text{cost}(OPT) \geq \sum_{\delta(y_i,r)\in OPT} \left(\sum_{x_j \in C_i(r)} \beta_j \right) \geq \sum_{x_j \in X'} \beta_j. \tag{8}$$

The first inequality follows because the β_j satisfy (6); the second is because each client in X' is served by at least one disk in OPT, and the β_j are non-negative.

Let C denote the cost of the solution returned by OuterCover(X', Y, κ, α). We have

$$C = 3^\alpha \cdot \text{cost}(F) = 3^\alpha \sum_{\delta(y_i,r)\in F} \left(\sum_{x_j \in C_i(r)} \beta_j \right) \leq 3^\alpha \sum_{x_j \in X'} \beta_j \leq 3^\alpha \cdot \text{cost}(OPT).$$

Here, the second equality is because each disk in F is tight; since the disks in F are pairwise disjoint, each client $x_j \in X'$ is contained in at most one disk in F, from which the next inequality follows; the final inequality is due to Inequality (8).

This completes the proof. □

4 Computing a covering for the non-uniform MCMC problem

With our algorithm for computing an outer cover in place, we now address the non-uniform MCMC problem. Recall that the input is a client set X, a server set Y, a coverage function $\kappa : X \to \mathbb{N} \cup \{0\}$, and a constant α.

Given an assignment of radius r_y to each $y \in Y$, we will say that a point $x \in X$ is *j-covered* if at least j disks cover it, that is,

$$|\{y \in Y \mid ||x - y|| \leq r_y\}| \geq j.$$

We will sometimes say that x is κ-covered to mean that it is $\kappa(x)$-covered. Similarly, if we have an assignment of radii to each $y \in Y$ such that for a set of points $P \subseteq X$, every point $x \in P$ is covered by at least $\kappa(x)$ disks, we say that P is κ-covered.

Our algorithm Cover(X, Y, κ, α) for non-uniform MCMC computes an assignment of radius r_y to each server $y \in Y$ such that each client $x \in X$ is $\kappa(x)$-covered. This algorithm

Algorithm 2 Cover(X, Y, κ, α)

if $\forall x \in X, \kappa(x) = 0$ **then**
  Assign $r_y \leftarrow 0$ for each $y \in Y$, and return.
Define $\kappa'(x)$ as follows:

$$\forall x \in X, \kappa'(x) = \begin{cases} 0, & \text{if } \kappa(x) = 0 \\ \kappa(x) - 1, & \text{if } \kappa(x) > 0 \end{cases}$$

Recursively call Cover(X, Y, κ', α).
Let $X' \leftarrow \{x \in X \mid x \text{ is not } \kappa(x)\text{-covered } \}$
Call the procedure OuterCover(X', Y, κ, α) to obtain an outer cover $\rho : Y \to \mathbb{R}^+$.
Let $Y' \leftarrow Y$.
Let $\overline{Y} \leftarrow \varnothing$.
while $X' \neq \varnothing$ **do**
  Choose $\overline{y} \in Y'$.
  $\overline{Y} \leftarrow \overline{Y} \cup \{\overline{y}\}$.
  Let $\mathrm{XC}_{\overline{y}} \leftarrow \varnothing$, $\mathrm{YC}_{\overline{y}} \leftarrow \varnothing$.
  for all $x' \in X'$ **do**
    if $x' \in \delta(\overline{y}, \rho_{\overline{y}})$ and $\rho_{\overline{y}} \geq ||x' - y^\kappa(x')||$ **then**
      $\mathrm{XC}_{\overline{y}} \leftarrow \mathrm{XC}_{\overline{y}} \cup \{x'\}$.
      $\mathrm{YC}_{\overline{y}} \leftarrow \mathrm{YC}_{\overline{y}} \cup \{y^1(x'), y^2(x'), \ldots, y^\kappa(x')\}$.
  Let $\mathrm{YC}'_{\overline{y}} \subseteq \mathrm{YC}_{\overline{y}}$ be a set of at most four points such that

$$\bigcap_{y \in \mathrm{YC}'_{\overline{y}}} \delta(y, r_y) = \bigcap_{y \in \mathrm{YC}_{\overline{y}}} \delta(y, r_y).$$

  For each $y \in \mathrm{YC}'_{\overline{y}}$, increase r_y by the smallest amount that ensures $\mathrm{XC}_{\overline{y}} \subseteq \delta(y, r_y)$.
  Remove $\overline{y}$ from Y' and remove from X' any points x that are $\kappa(x)$-covered.

is recursive, and in the base case we have $\kappa(x) = 0$ for each $x \in X$. In the base case, the radius r_y is assigned to 0 for each $y \in Y$. Otherwise, we define

$$\kappa'(x) = \max\{0, \kappa(x) - 1\}, \text{ for each } x \in X,$$

and recursively call Cover(X, Y, κ', α) to compute an assignment that $\kappa'(x)$-covers each $x \in X$. We then compute $X' \subseteq X$, the set of points that are not $\kappa(x)$-covered. We compute an outer cover $\rho : Y \to \mathbb{R}^+$ for X' using the procedure OuterCover(X', Y, κ, α) described in Section 3. For any client $x \in X'$, the outer cover has a disk $\delta(y, \rho_y)$ that serves it. That is, x is contained in $\delta(y, \rho_y)$ and $\rho_y \geq ||x - y^\kappa(x)||$.

The goal of the while-loop is to increase some of the r_y to ensure that each $x \in X'$, which is currently $(\kappa(x) - 1)$-covered, is also $\kappa(x)$-covered. To do this, we iterate via the while loop over each disk $\delta(\overline{y}, \rho_{\overline{y}})$ returned by OuterCover(X', Y, κ, α). We add all points in X' that are served in the outer cover by $\delta(\overline{y}, \rho_{\overline{y}})$ to a set $\mathrm{XC}_{\overline{y}}$. That is, $\mathrm{XC}_{\overline{y}}$ consists of all $x' \in X'$ that are contained in $\delta(\overline{y}, \rho_{\overline{y}})$ and $\rho_{\overline{y}} \geq ||x' - y^\kappa(x')||$. The set $\mathrm{YC}_{\overline{y}}$ contains, for

each $x \in \mathrm{XC}_{\overline{y}}$, the $\kappa(x)$ nearest neighbors of x in Y. For purposes of analysis, we add $\overline{y}$ to a set $\overline{Y}$ as well.

Next, we identify a set $\mathrm{YC}'_{\overline{y}} \subseteq \mathrm{YC}_{\overline{y}}$ of at most 4 points such that

$$\bigcap_{y \in \mathrm{YC}'_{\overline{y}}} \delta(y, r_y) = \bigcap_{y \in \mathrm{YC}_{\overline{y}}} \delta(y, r_y).$$

Below, we argue why such a $\mathrm{YC}'_{\overline{y}}$ exists. We enlarge the radius r_y of each $y \in \mathrm{YC}'_{\overline{y}}$ by the minimum amount needed to ensure that $\mathrm{XC}_{\overline{y}} \subseteq \delta(y, r_y)$. We argue below that after this each point in $\mathrm{XC}_{\overline{y}}$ is κ-covered.

After increasing r_y for $y \in \mathrm{YC}'_{\overline{y}}$, we discard from X' all points that are now κ-covered. The discarded set contains $\mathrm{XC}_{\overline{y}}$ and possibly some other points in X'. We remove $\overline{y}$ from Y'. We go back and iterate the while loop with the new X' and Y'.

This completes our description of the algorithm. To show that it computes a κ-cover, we first need to establish two claims that we already made. For these two claims, let us fix one iteration of the while loop in lines 9–19, and let us fix $\overline{y}$ as chosen in line 10.

Claim 4. *In line 17, there exists a set $YC'_{\overline{y}} \subseteq YC_{\overline{y}}$ of at most 4 points such that*

$$\bigcap_{y \in YC'_{\overline{y}}} \delta(y, r_y) = \bigcap_{y \in YC_{\overline{y}}} \delta(y, r_y).$$

Proof. If, on the one hand, the intersection of disks $\bigcap_{y \in \mathrm{YC}_{\overline{y}}} \delta(y, r_y)$ is empty, then Helly's Theorem tells us that there is a set of at most three disks whose intersection is empty. On the other hand, if the intersection $\bigcap_{y \in \mathrm{YC}_{\overline{y}}} \delta(y, r_y)$ is non-empty, then it is a rectangle (as these are l_∞ disks) and therefore equal to the intersection of four of the disks. □

Claim 5. *After executing line 18, each point in $XC_{\overline{y}}$ is κ-covered.*

Proof. Consider any $x' \in \mathrm{XC}_{\overline{y}}$. This means that $x' \in X'$, and therefore x' is not κ-covered before the execution of line 18.

Notice that $|\mathrm{YC}_{\overline{y}}| \geq \kappa(x')$, since the $\kappa(x')$ nearest neighbors of x' are included in $\mathrm{YC}_{\overline{y}}$. Thus, before the execution of line 18, x' does not belong to $\bigcap_{y \in \mathrm{YC}_{\overline{y}}} \delta(y, r_y)$. (Otherwise, it would be κ-covered at that time.)

Therefore, x' does not belong to $\bigcap_{y \in \mathrm{YC}'_{\overline{y}}} \delta(y, r_y)$. It follows that there is at least one $y \in \mathrm{YC}'_{\overline{y}}$ such that $\delta(y, r_y)$ did not contain x' before the execution of line 18. After line 18 is executed, $\delta(y, r_y)$ does contain x'. Since x' was $(\kappa(x') - 1)$-covered before the execution of line 18, it is now $\kappa(x')$-covered. □

Finally, we argue that the algorithm computes a κ-cover.

Claim 6. *The algorithm Cover(X, Y, κ, α) computes a κ-cover.*

Proof. For this, it suffices to argue that the while loop (lines 9–19) terminates, because the termination condition is that X', the set of clients not κ-covered, is empty.

Fix a point $x \in X'$ after line 5 of the algorithm. The client x is not κ-covered at this stage. It suffices to show that in some iteration of the while loop, x gets κ-covered.

The point x is served by some disk $\delta(y, \rho_y)$ in the outer cover ρ computed in line 6. That is, $x \in \delta(y, \rho_y)$ and $\rho_y \geq ||x - y^\kappa(x)||$. Suppose that in some iteration of the while loop, $\overline{y} = y$; fix that iteration. If x has not already been κ-covered in earlier iterations, it gets added to $\mathrm{XC}_{\overline{y}}$ in that iteration. At the end of that iteration, it gets κ-covered, by Claim 5.

If there is no iteration of the while loop in which $\overline{y} = y$, then the while loop evidently terminates because $X' = \varnothing$. In this case too, x gets κ-covered in some iteration of the while loop – the iteration in which it gets removed from X'. □

5 Approximation Ratio

In this section, we bound the ratio of the cost of the solution returned by $\mathrm{Cover}(X, Y, \kappa, \alpha)$ and the cost of the optimal solution. For this purpose, the following lemma is central. It bounds the increase in cost incurred by $\mathrm{Cover}(X, Y, \kappa, \alpha)$ in going from a κ'-cover to a κ-cover by a constant times the cost of the outer cover ρ for X'.

Lemma 2. *The increase in the objective function $\sum_{y \in Y} r_y^\alpha$ from the time $\mathrm{Cover}(X, Y, \kappa', \alpha)$ completes to the time $\mathrm{Cover}(X, Y, \kappa, \alpha)$ completes is $4 \cdot 3^\alpha \cdot \sum_{y \in Y} \rho_y^\alpha$.*

Proof. Let us fix an $\overline{y} \in \overline{Y}$, and focus on the iteration when $\overline{y}$ was added to $\overline{Y}$. Notice that there is exactly one such iteration, since $\overline{y}$ is removed from Y' in the iteration it gets added to $\overline{Y}$.

We will bound the increase in cost during this iteration. For this, we need two claims.

Claim 7. *For any $x' \in XC_{\overline{y}}$, we have*

$$||\overline{y} - x'|| \leq \rho_{\overline{y}}$$

Proof. Recall that x' is in $\mathrm{XC}_{\overline{y}}$ because $x' \in \delta(\overline{y}, \rho_{\overline{y}})$. □

Claim 8. *For any $y' \in YC_{\overline{y}}$, we have*

$$||y' - \overline{y}|| \leq 2 * \rho_{\overline{y}}$$

Proof. Let y' be added to $\mathrm{YC}_{\overline{y}}$ when $x' \in X'$ was added to $\mathrm{XC}_{\overline{y}}$. Hence

$$\begin{aligned} ||y' - x'|| &\leq ||x' - y^\kappa(x')|| \\ &\leq \rho_{\overline{y}}, \end{aligned}$$

since $\delta(\overline{y}, \rho_{\overline{y}})$ serves x' in the outer cover (line 14 of Algorithm 2). Also, since $x' \in \delta(\overline{y}, \rho_{\overline{y}})$, $||x' - \overline{y}|| \leq \rho_{\overline{y}}$. Therefore,

$$\begin{aligned} ||y' - \overline{y}|| &\leq ||y' - x'|| + ||x' - \overline{y}|| \\ &\leq \rho_{\overline{y}} + \rho_{\overline{y}} \\ &= 2\rho_{\overline{y}} \end{aligned}$$

□

Fix a $y \in \mathrm{YC}'_{\overline{y}}$. If r_y was increased in this iteration, it now equals $||y - x'||$ for some $x' \in \mathrm{XC}_{\overline{y}}$. By the above two claims,

$$\begin{aligned} ||y - x'|| &\leq ||y - \overline{y}|| + ||\overline{y} - x'|| \\ &\leq 3 * \rho_{\overline{y}} \end{aligned}$$

Thus the increase in r_y^α is at most $3^\alpha(\rho_{\overline{y}})^\alpha$. Since r_y is increased in this iteration only for $y \in \mathrm{YC}'_{\overline{y}}$, and $|\mathrm{YC}'_{\overline{y}}| \leq 4$, the increase in the objective function $\sum_{y \in Y} r_y^\alpha$ (in the iteration of the while loop under consideration) is at most $4 \cdot 3^\alpha \cdot (\rho_{\overline{y}})^\alpha$.

We conclude that the increase in $\sum_{y \in Y} r_y^\alpha$ over all the iterations of the while loop is at most

$$4 \cdot 3^\alpha \cdot \sum_{\overline{y} \in \overline{Y}} (\rho_{\overline{y}})^\alpha = 4 \cdot 3^\alpha \cdot \sum_{y \in Y} \rho_y^\alpha.$$

□

We can now bound the approximation ratio of the algorithm.

Lemma 3. *Let $r' : Y \to \mathbb{R}^+$ be any assignment of radii to the points in Y under which each point $x \in X$ is $\kappa(x)$-covered. Then the cost of the output of Cover(X, Y, κ, α) is at most $c * \sum_{y \in Y} {r'_y}^\alpha$, where $c = 4 \cdot 27^\alpha$.*

Proof. Our proof is by induction on $\max_{x \in X} \kappa(x)$. For the base case, where $\kappa(x) = 0$ for each $x \in X$, the claim in the theorem clearly holds.

Let $D = \{\delta(y, r'_y) \mid y \in Y\}$ be the set of disks corresponding to the assignment r'. Our proof strategy is to show that there is a subset $D_\kappa \subseteq D$ such that

1. The cost increase incurred by Cover(X, Y, κ, α) in going from the κ'-cover to the κ-cover is at most c times the cost of the disks in D_κ. (Recall that the *cost* of a set of disks is the sum of the α-th powers of the radii of the disks.)
2. The set of disks, $D \setminus D_\kappa$, $\kappa'(x)$-covers any point $x \in X$.

By the induction hypothesis, the cost of the κ'-cover computed by Cover(X, Y, κ, α') is at most c times the cost of the disks in $D \setminus D_\kappa$. As the increase in cost incurred by Cover(X, Y, κ, α) in turning the κ'-cover to a κ-cover is at most c times the cost of the disks in D_κ, the theorem follows.

We now describe how D_k is computed, and then establish that it has the above two properties. For each $x' \in X'$, let largest(x') be the largest disk from D that contains x'. Since x' is $\kappa(x')$-covered by D, we note that the radius of largest(x') is at least $||x' - y^{\kappa}(x')||$. Let

$$D'_{\kappa} = \{\text{largest}(x') \mid x' \in X'\}.$$

Sort the disks in D'_{κ} by decreasing (non-increasing) radii. Let $B \leftarrow \varnothing$ initially. For each disk $d \in D'_{\kappa}$ in the sorted order, perform the following operation: add d to B if d does not intersect any disk already in B.

Let D_{κ} be the set B at the end of this computation. (See Figure 1.) Since no two disks in D_{κ} intersect, and D κ-covers any point in X, it follows that $D \setminus D_{\kappa}$ κ'-covers any point in X. This establishes Property 2 of D_{κ}.

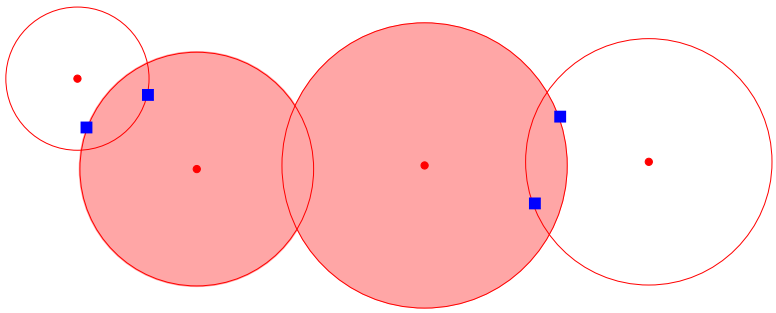

Figure 1: The set D of disks, and the clients in X', shown as blue squares. Here, $\kappa(x) = 2$ for each client x. The set D'_{κ} consist of the two shaded disks, and the set D_{κ} contains only the larger of these two disks. For this illustration we use ℓ_2 disks.

We now turn to Property 1. For this, consider L_{κ}, the set of disks obtained by increasing the radius of each disk in D_{κ} by a factor of 3. We argue that L_{κ} is an outer cover for X'. Fix any $x' \in X'$.

1. If largest$(x') \in D_{\kappa}$, then the corresponding disk in L_{κ} contains x' and has radius at least $||x' - y^{\kappa}(x')||$.
2. If largest$(x') \notin D_{\kappa}$, then there is an even larger disk in D_{κ} that intersects largest(x'). The corresponding disk in L_{κ} contains x' and has radius at least $||x' - y^{\kappa}(x')||$.

Since L_{κ} is an outer cover for X', and the procedure OuterCover(X', Y, κ, α) returns a 3^{α} approximation to the optimal outer cover, we infer that

$$\sum_{y \in Y} \rho_y^{\alpha} \leq 3^{\alpha} \cdot \text{cost}(L_{\kappa}) \leq 9^{\alpha} \cdot \text{cost}(D_{\kappa}).$$

Thus the cost increase incurred by Cover(X, Y, κ, α) in going from the κ'-cover to the κ-cover is, by Lemma 2, at most

$$4 \cdot 3^{\alpha} \cdot \sum_{y \in Y} \rho_y^{\alpha} \leq 4 \cdot 27^{\alpha} \cdot \text{cost}(D_{\kappa}) = c \cdot \text{cost}(D_{\kappa}).$$

This establishes Property 1, and completes the proof of the lemma. □

We conclude with a statement of the main result of this article. In this statement, cost refers to l_2 rather than l_∞ disks. Since (a) an l_2 disk of radius r is contained in the corresponding l_∞ disk of radius r, and (b) an l_∞ disk of radius r is contained in an l_2 disk of radius $\sqrt{2}r$, the approximation guarantee is increased by $(\sqrt{2})^\alpha$ when compared to Lemma 3.

Theorem 1. *Given point sets X and Y in the plane, a coverage function $\kappa : X \to \{0, 1, 2, \ldots, |Y|\}$, and $\alpha \geq 1$, the algorithm Cover(X, Y, κ, α) runs in polynomial time and computes a κ-cover of X with cost at most $4 \cdot (27\sqrt{2})^\alpha$ times that of the optimal κ-cover.*

6 Concluding Remarks

Our result generalizes to the setting where X and Y are points in $\mathbb{R}^d$, where d is any constant. The approximation guarantee is now $(2d) \cdot (27\sqrt{d})^\alpha$. To explain, the intersection of a finite family of l_∞ balls equals the intersection of a sub-family of at most $2d$ balls. That is why the 4 in the approximation guarantee of Theorem 1 becomes $2d$. In the transition from l_2 to l_∞ balls in $\mathbb{R}^d$, we lose a factor of $(\sqrt{d})^\alpha$.

This generalization naturally leads to the next question – what can we say when X and Y are points in an arbitrary metric space? Our approach confronts a significant conceptual obstacle here, since one can easily construct examples in which the cost of going from a $(k-1)$-cover to a k-cover (for the uniform MCMC) cannot be bounded by a constant times the cost of an optimal outer cover. Thus, new ideas seem to be needed for obtaining an $O(1)$ approximation for this problem. The work of [4] gives the best known guarantee of $O(k)$. For the non-uniform version, their approximation guarantee is $O(\max\{\kappa(x) \mid x \in X\})$.

Acknowledgements: This material is based on work supported by the National Science Foundation under grants CCF-0915543 and CCF-1318996.

References

[1] K. A. Affash, P. Carmi, M. J. Katz, and G. Morgenstern. Multi cover of a polygon minimizing the sum of areas. *International Journal of Computational Geometry and Applications*, 21(6):685–698, 2011.

[2] H. Alt, E. M. Arkin, H. Brönnimann, J. Erickson, S. P. Fekete, C. Knauer, J. Lenchner, J. S. B. Mitchell and K. Whittlesey. Minimum-cost coverage of point sets by disks. In *Proceedings of the Symposium on Computational Geometry* (SoCG), 2006, 449–458.

[3] N. Bansal and K. Pruhs. Weighted geometric set multi-cover via quasi-uniform sampling. In *Proceedings of the European Symposium on Algorithms* (ESA), 2012, 145–156.

[4] R. Bar-Yehuda and D. Rawitz. A note on multicovering with disks. *Computational Geometry*, 46(3):394–399, 2013.

[5] V. Bilò, I. Caragiannis, C. Kaklamanis, and P. Kanellopoulos. Geometric clustering to minimize the sum of cluster sizes. In *Proceedings of the European Symposium on Algorithms* (ESA), 2005, 460–471.

[6] S. Bhowmick, K. Varadarajan, and S. Xue. A constant-factor approximation for multi-covering with disks. In *Proceedings of the Symposium on Computational Geometry* (SoCG), 2013, 243–248.

[7] T. M. Chan. Polynomial-time approximation schemes for packing and piercing fat objects. *Journal of Algorithms*, 46(2):178–189, 2003.

[8] T. M. Chan, E. Grant, J. Könemann, and M. Sharpe. Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms* (SODA), 2012, 1576–1585.

[9] M. Charikar and R. Panigrahy. Clustering to minimize the sum of cluster diameters. *Journal of Computer and System Sciences*, 68(2): 417–441, 2004.

[10] C. Chekuri, K. L. Clarkson, and S. Har-Peled. On the set multi-cover problem in geometric settings. In *Proceedings of the Symposium on Computational Geometry* (SoCG), 2009, 341–350.

[11] T. Erlebach, K. Jansen, and E. Seidel. Polynomial-time approximation schemes for geometric intersection graphs. *SIAM Journal on Computing*, 34(6):1302–1323, 2005.

[12] A. Freund and D. Rawitz. Combinatorial interpretations of dual fitting and primal fitting. *Proceedings of the International Workshop on Approximation and Online Algorithms* (WAOA), 2003, 137–150.

[13] S. Har-Peled and M. Lee. Weighted geometric set cover problems revisited. *Journal of Computational Geometry*, 3(1):65–85, 2012.

[14] N. Lev-Tov and D. Peleg. Polynomial time approximation schemes for base station coverage with minimum total radii. *Computer Networks*, 47(4):489–501, 2005.

[15] K. Varadarajan. Weighted geometric set cover via quasi-uniform sampling. In *Proceedings of the ACM Symposium on Theory of Computing* (STOC), 2010, 641–648.