


# Approximation and Parameterized Algorithms for Covering with Disks of Two Types of Radii

Sayan Bandyapadhyay ✉ 

Portland State University, OR, USA

Eli Mitchell ✉ 

Department of Computer Science, Portland State University, OR, USA

---

## Abstract

We study the Discrete Covering with Two Types of Radii problem motivated by its application in wireless networks. In this problem, the goal is to assign either small-range high frequency or large-range low frequency to each access point, maximizing the number of users in high-frequency regions while ensuring that each user is in the range of an access point. Unlike other weighted covering problems, our problem requires satisfying two simultaneous objectives, which calls for novel approaches that leverage the underlying geometry of the problem. In our work, we present two new algorithms: the first is a polynomial-time  $(2.5 + \epsilon)$ -approximation, and the second is an exact algorithm for sparse instances, which is fixed-parameter tractable (FPT) in the number of large-radius disks. We also prove that such an FPT algorithm is impossible for general instances lacking sparsity, assuming the Exponential Time Hypothesis. Before our work, the best-known polynomial-time approximation factor was 4 for the problem.

Our approximation algorithm results from a fine-grained classification of points that can contribute to the gain of a solution. Based on this classification, we design two sub-algorithms with interdependent guarantees to recover the respective class of points as gain. Our algorithm exploits further properties of Delaunay triangulations to achieve the improved bound. The FPT algorithm is based on branching that utilizes the sparsity of the instances to limit the overall search space.

**2012 ACM Subject Classification** Theory of computation → Computational geometry; Mathematics of computing → Approximation algorithms

**Keywords and phrases** Covering, Disks, Approximation, FPT

**Digital Object Identifier** 10.4230/LIPIcs.WADS.2025.7

**Funding** *Sayan Bandyapadhyay*: The work has been supported by the NSF grant 2311397.

## 1 Introduction

Set cover is among the most popular optimization problems, where given a ground set of elements and a family of subsets of the ground set, the goal is to find a minimum-sized subfamily of those subsets whose union contains all elements of the ground set. The problem admits a polynomial-time (poly-time)  $O(\log n)$ -approximation, and this bound is known to be tight [16, 14]. Set cover has been studied in various restricted settings with the goal of achieving better approximation factors. One such interesting setting is *geometric covering*, where elements are points in a geometric space, and subsets are induced by geometric objects such as disks and squares. Such restricted covering problems can admit poly-time constant-approximations [11, 9] or even approximation schemes [17, 27, 10], as one can exploit the natural constraints imposed by geometric spaces and objects. Indeed, this turned out to be a very popular area of research, as many practical problems can be described in geometric terms, e.g., in areas such as sensor networks, computational biology, image processing, and VLSI design [29, 6, 15, 13, 17].



© Sayan Bandyapadhyay and Eli Mitchell;

licensed under Creative Commons License CC-BY 4.0

19th International Symposium on Algorithms and Data Structures (WADS 2025).

Editors: Pat Morin and Eunjin Oh; Article No. 7; pp. 7:1–7:14

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

While several formulations of geometric set cover are now well-studied, their real applications are sometimes hindered by practical constraints. There exist well-established techniques like local search [27, 19, 7, 1] to compute near-optimal solutions to fundamental geometric problems, but real-life constraints can invalidate the assumptions needed by these approaches. In this work, we study one such problem motivated by applications in wireless and sensor networks, which was introduced by Maheshwari et al. [22]. Informally, given a set of access points and users, the goal is to assign either low-range high frequency (i.e., high speed) or large-range low frequency to each access point, maximizing the number of users in high-frequency range while ensuring that each user is in the range of at least one access point of either kind.

We use the following notation to define the problem. For any point  $x$  and a real  $\rho > 0$ , let  $D(x, \rho)$  denote the closed disk with center  $x$  and radius  $\rho$ . In **Discrete Covering with Two Types of Radii (DC-2)**, we are given two point sets  $P$  (of  $n$  “users”) and  $A = \{a_1, a_2, \dots, a_m\}$  (of  $m$  “access points”) in the plane, and two real numbers  $\rho_1$  and  $\rho_2$ , such that  $0 < \rho_1 < \rho_2$  and  $P \subseteq \bigcup_{i=1}^m D(a_i, \rho_2)$ . The goal is to select for each  $i = 1, 2, \dots, m$ , a value  $r_i \in \{\rho_1, \rho_2\}$  such that  $P \subseteq \bigcup_{i=1}^m D(a_i, r_i)$  and the *gain*, i.e., the size of the set  $\{p \in P : \text{there is an index } 1 \leq i \leq m \text{ such that } r_i = \rho_1 \text{ and } p \in D(a_i, r_i)\}$  is maximized. Thus, we want to select for each  $1 \leq i \leq m$ , either the *small* disk (of radius  $\rho_1$ ) centered at  $a_i$  or the *large* disk (of radius  $\rho_2$ ) centered at  $a_i$ , such that the set  $P$  is covered by the chosen disks and the number of points in  $P$  each of which is contained in at least one chosen small disk is maximized.

The problem is known to be NP-hard [22] similar to most of the covering problems with disks. Bandyapadhyay et al. [2] designed a poly-time 4-approximation for DC-2. In their approach, they converted the problem to a weighted covering problem with one type of radii (large) by assigning unit weight for each user point contained in a small disk, to a representative large disk. To solve this covering problem, they compute the Delaunay triangulation of the large disk centers. In particular, this triangulation is a planar graph, and hence can be colored by 4 colors. Consequently, a vertex cover of weight at most  $\frac{3}{4}$  of the total weight can be computed, and the corresponding large disks are shown to cover all user points. Thus, at least  $\frac{1}{4}$  of the total weight of the large disks remains unused while constructing the cover, which leads to the 4-approximation. Obtaining an improved approximation factor remained an interesting open question, as many covering problems involving unit disks admit PTASes [27, 21, 10].

As mentioned in [2], DC-2 can be interpreted as a combination of two problems: **Minimum Weight Unit Disk Cover (WUDC)** and **Maximum Coverage with Unit Disks (MCUD)**. In WUDC, the goal is to select a subset of unit disks that cover all input points such that the sum of the weights of the selected disks is minimized. In MCUD, for a given  $k$ , the goal is to select a subset of  $k$  unit disks that cover the maximum number of points possible. Both of these problems are well-studied in computational geometry.

Mustafa and Ray [27] developed a PTAS for WUDC when all weights are equal, i.e., for the unweighted version. Their approach, which relies on a local search scheme, is inadequate for addressing weighted instances. Finding the optimal cover even for weighted unit disks has long been known to be strongly NP-hard [18]. Regardless, approaches have been developed in the last two decades for computing approximate solutions for weighted covering problems for broad classes of objects, such as those with low union complexity [8, 11, 30, 4]. Chan et al. [9] designed constant approximations for covering with weighted disks. Subsequently, Li and Jin [21] designed a PTAS for WUDC, utilizing a shifting strategy [18] that partitions the plane into squares and addresses a restricted version of the problem within each square. We

note that this approach cannot be used directly for DC-2 when users are covered by multiple small disks, because the assignment of user weights to small disks cannot be done arbitrarily, and so the weight of a small disk in the optimal solution is an unknown in DC-2.

Chaplick et al. [10] designed a local search PTAS for the MCUD problem. It follows that for some  $k$ , there are  $k$  small disks in any optimal solution to DC-2 such that they cover the points that contribute to the optimal gain. However, the best set of  $k$  small disks that maximizes the coverage might not lead to a feasible solution, as the remaining  $n - k$  large disks might not cover the points not in the union of the chosen small disks. Dealing with two different but dependent tasks is the main challenge in tackling DC-2. The need to satisfy one objective while optimizing another necessarily means that any decision made at the local scope can have global consequences. For example, selecting one access point for a small disk may force a neighbor to be selected as a large disk to maintain coverage, and such a dependency may further propagate to disks that are far away. As such, any solution for this kind of constrained problem must take into account its global properties while making any decision to achieve good approximation.

We also study DC-2 from the perspective of parameterized complexity. Designing parameterized algorithms for geometric problems is an active area of research that is becoming increasingly popular. A problem is called fixed-parameter tractable (FPT) in a parameter  $k$  if it can be solved in time  $f(k) \cdot n^{O(1)}$  for a computable function  $f$ , where  $n$  is the input size. A known fact is that if a problem with parameter  $k$  is W[1]-hard, then it cannot be FPT in  $k$  [12]. Marx [23] considered the problem of covering with unit squares. Here the size of a cover is a natural parameter. He proved that the problem is W[1]-hard. One consequence of this is that the optimization version does not admit an Efficient PTAS or EPTAS [23], i.e., the  $n^{O(1/\epsilon^2)}$ -time PTAS due to Hochbaum and Maass [17] cannot be improved to an EPTAS. In a different work, Marx [24] studied the parameterized complexity of common optimization problems in geometric intersection graphs and obtained a mixed set of algorithmic and hardness results. In a follow-up work, Marx and Pilipczuk [26] designed a framework for designing sub-exponential ( $n^{O(\sqrt{k})}$ ) time algorithms for a wide range of geometric facility location problems. Applying this framework, they designed an  $n^{O(\sqrt{k})}$ -time algorithm for covering with unit disks. This result is also tight, assuming ETH [24] (also see this survey [25]). The framework is based on Voronoi diagram separators. We note that this framework is not useful for DC-2, again due to the mixed nature of the problem. Kowalska and Pilipczuk [20] recently studied parameterized approximation algorithms for covering with line segments. Lastly, Banik et al. [3] studied the parameterized complexity of a conflict-free version of geometric covering.

## 1.1 Our results and contributions

In our work, we design for any  $\epsilon > 0$ , an  $(n + m)^{O(1/\epsilon^9)}$ -time approximation algorithm for DC-2 with the improved approximation factor of  $(2.5 + \epsilon)$ . The algorithm is based on a fine-grained classification of user points that may contribute to the gain of a solution. As mentioned before, the best-known  $\frac{1}{4}$  factor for DC-2 is based on a weight assignment scheme that assigns 1 weight for each user point contained in a small disk, to a large disk. Then using the Delaunay triangulation of the large disk centers, a cover by large disks can be computed where at least  $\frac{1}{4}$  of the total weight remains unused. In our work, we identify a subset of user points for which  $\frac{1}{2}$  of the total weight can be saved by using the same Delaunay triangulation based algorithm, but along with a new weight assignment scheme that exploits further properties of the triangulation. Specifically, this is the subset of user points that are contained in 2 or more small disks. Unfortunately, the  $\frac{1}{2}$  weight saving in this way is

not possible for the other user points contained in exactly 1 small disk and we still gain  $\frac{1}{4}$  of the total weight for them. However, in a separate approach, we show that the points of the latter type that contribute to an optimal solution can be recovered almost exactly by treating them as the weights on the large disks and solving an instance of WUDC. Hence, our main algorithm runs two subalgorithms and returns the larger gain. Our analysis shows that this is at least  $\frac{1}{2.5+\epsilon}$  times the optimal gain, which improves on the previous factor of  $\frac{1}{4}$ .

We also study the parameterized complexity of the problem, with  $k$  being the number of large disks to be selected. This is a natural choice of parameter, as in practice, one would like to open as few large disks as possible to increase the overall quality of the coverage. In particular, the large disks correspond to low-speed antennas, and hence, the selection of too many such disks might slow down the data transfer, leading to a subpar experience for customers. We first show that the problem is W[1]-hard when  $k$  is the number of large disks to be selected. In fact, we prove that assuming ETH, there is no  $f(k) \cdot (n + m)^{o(\sqrt{k})}$  time algorithm for the problem for any computable function  $f$ . In stark contrast, we show that the problem is fixed-parameter tractable (FPT) in  $k$  if each point is contained in, at most, a constant number of large disks. We define the notion of  $s$ -sparse instances (see Section 3 for a formal definition) and show that the problem can be solved in  $s^{O(k)}(m + n)^{O(1)}$  time on any such instance, where  $s \leq m$ . This result indicates that the W[1]-hardness arises solely from the absence of sparsity in the instances. We note that the problem remains NP-hard for  $O(1)$ -sparse instances [22].

While the  $(2.5 + \epsilon)$ -approximation works for any instance of DC-2, in practice, the FPT algorithm may yield vastly better run time when points are low-depth. DC-2 is directly motivated by wireless networks, where low-depth points frequently occur in certain environments like sparsely-populated cities. As such, we chose to describe both algorithms with the hope of improving the variety of results available for practical applications.

**Organization.** The approximation algorithm is described in Section 2. In Section 3, we describe the FPT algorithm for sparse instances. The hardness result appears in Section 4.

## 2 An Approximation Algorithm for DC-2

Our main algorithm is an aggregate of two separate algorithms. The first is based on the Delaunay triangulation of large disk centers. The second algorithm is based on a reduction to the Minimum Weight Unit Disk Cover problem. Thereafter, we combine the two algorithms to obtain our main algorithm which has the desired approximation guarantee.

A set of disks  $D'$  is said to *cover* a set of points  $P'$  if the union of the disks in  $D'$  contains the points in  $P'$ . Let  $S$  (resp.  $L$ ) be the set of small (resp. large) disks centered at the access points in  $A$ . Let  $V \subseteq P$  such that each point in  $V$  does not lie in any (small) disks of  $S$ . Wlog, we assume that each point of  $V$  is in at least two disks of  $L$ . It must be in at least one disk, otherwise there is no feasible cover. Also, if it is in exactly one disk, this disk will always be part of the solution. Let  $N \subseteq P$  such that each point in  $N$  lies in at least one (small) disk of  $S$ . Thus,  $(V, N)$  is a partition of  $P$ . Let  $N_1 \subseteq N$  such that each point in  $N_1$  lies in exactly one disk of  $S$ . Similarly, let  $N_2 \subseteq N$  such that each point in  $N_2$  lies in at least two disks of  $S$ . Thus,  $(N_1, N_2)$  is a partition of  $N$ . Let  $n_1 = |N_1|$  and  $n_2 = |N_2|$ . Fix an optimal solution of DC-2, i.e., the selected subset of disks of  $S \cup L$ . Let  $\text{OPT}$  be the set of points in  $N$  covered by the optimal solution. Let  $\text{OPT}_1 = N_1 \cap \text{OPT}$  and  $\text{OPT}_2 = N_2 \cap \text{OPT}$ . Also, fix an error parameter  $\epsilon > 0$ .

► **Observation 1.** *The gain of any solution is at most  $n_1 + n_2$ .*



## 7:6 Covering with Disks of Two Types of Radii

► **Corollary 4.** *For any point  $p \in V$ , there exist two points  $a_i, a_j \in A$  such that the edge  $\{a_i, a_j\}$  is in  $\mathcal{T}$  and  $p$  is in both large disks  $D(a_i, \rho_2)$  and  $D(a_j, \rho_2)$ .*

**Proof.** Consider the disk  $D(p, \rho_2)$ . As  $p$  is in at least two large disks, there must be at least two points of  $A$  in this disk. Then by Lemma 3, there are two points  $a_l, a_t$  in  $A \cap D(p, \rho_2)$  and a path  $\pi$  in  $\mathcal{T}$  between  $a_l$  and  $a_t$  such that  $\pi$  is contained in  $D(p, \rho_2)$ . Consider any edge  $\{a_i, a_j\}$  on  $\pi$ . Then as  $a_i, a_j$  are in  $D(p, \rho_2)$ ,  $p$  is in both large disks  $D(a_i, \rho_2)$  and  $D(a_j, \rho_2)$ . ◀

► **Corollary 5.** *For any point  $p \in N_2$ , there exist two points  $a_i, a_j \in A$  such that the edge  $\{a_i, a_j\}$  is in  $\mathcal{T}$  and  $p$  is in both small disks  $D(a_i, \rho_1)$  and  $D(a_j, \rho_1)$ .*

**Proof.** Consider the disk  $D(p, \rho_1)$ . As  $p$  is in at least two small disks, there must be at least two points of  $A$  in this disk. Then by Lemma 3, there are two points  $a_l, a_t$  in  $A \cap D(p, \rho_1)$  and a path  $\pi$  in  $\mathcal{T}$  between  $a_l$  and  $a_t$  such that  $\pi$  is contained in  $D(p, \rho_1)$ . Consider any edge  $\{a_i, a_j\}$  on  $\pi$ . Then as  $a_i, a_j$  are in  $D(p, \rho_1)$ ,  $p$  is in both small disks  $D(a_i, \rho_1)$  and  $D(a_j, \rho_1)$ . ◀

Now, we describe the pseudocode of the algorithm, which is self-explanatory.

### ■ Algorithm 1 Weighted-Extraction.

**Input:** The sets  $A, P, N_1, N_2$ , small radius  $\rho_1$  and large radius  $\rho_2$ .

**Output:** A set  $sol$  of large and small disks covering  $P$ .

```

1: Compute the Delaunay triangulation  $\mathcal{T}$  of the points in  $A$ .
2: Compute a valid four coloring of the vertices of  $\mathcal{T}$ .
3: for  $a_i \in A$  do
4:    $w_i \leftarrow 0$                                      ▷ Initialize the weights
5:   for  $p_1 \in N_1$  do
6:     if  $p_1 \in D(a_i, \rho_1)$  then  $w_i \leftarrow w_i + 1$ 
7:   for  $p_2 \in N_2$  do
8:     Compute two points  $a_i, a_j$  with the properties described in Corollary 5.
9:      $w_i \leftarrow w_i + 1$ 
10:     $w_j \leftarrow w_j + 1$ 
11: Compute the color class  $C$  of vertices (access points) of  $\mathcal{T}$  with maximum weight (w.r.t.
     $w_i : a_i \in A$ ).
12:  $sol \leftarrow \emptyset$ 
13: for  $a_i \in A$  do
14:   if  $a_i \in C$  then
15:      $sol \leftarrow sol \cup D(a_i, \rho_1)$ 
16:   else
17:      $sol \leftarrow sol \cup D(a_i, \rho_2)$ 
return  $sol$ 

```

The four coloring of a planar graph can be computed in polynomial time, for example using the algorithm in [28]. Hence, the algorithm runs in polynomial time.

► **Lemma 6.** *The solution  $sol$  computed by Weighted-Extraction covers all points of  $P$ .*

**Proof.** Note that we pick either  $D(a_i, \rho_1)$  or  $D(a_i, \rho_2)$  in the solution for each access point  $a_i$ . Thus, by Observation 2, the points in  $N$  are covered by the selected disks. Now, consider any point  $p \in V$ . We prove that  $p$  is covered in the solution. By Corollary 4, we know that there

exist two points  $a_i, a_j \in A$  such that the edge  $\{a_i, a_j\}$  is in  $\mathcal{T}$  and  $p$  is in both large disks  $D(a_i, \rho_2)$  and  $D(a_j, \rho_2)$ . Note that the points in  $C$  form an independent set of the graph  $\mathcal{T}$  by all sharing the same color. Hence, the set of points  $A \setminus C$  is a vertex cover of  $\mathcal{T}$ . It follows that at least one of  $a_i$  and  $a_j$  is in  $A \setminus C$ . As we pick large disks for all access points in  $A \setminus C$ ,  $p$  must be covered by  $D(a_i, \rho_2)$  or  $D(a_j, \rho_2)$  in the solution.  $\blacktriangleleft$

We need the following lemma to analyze the gain of the solution.

► **Lemma 7.** *Let  $W$  be the total weight of the points in  $C$ . Then the set of small disks centered at the access points in  $C$  contains at least  $W$  points of  $N$ .*

**Proof.** First, note that the points in  $C$  form an independent set of the graph  $\mathcal{T}$ . Now, consider any point  $p \in N$ . We claim that it contributes 1 unit of weight to at most one point of  $C$ . If  $p$  is in  $N_1$ , by our construction, it contributes either 1 or 0 depending on whether the center of the unique small disk that contains  $p$  is in  $C$  or not. If  $p$  is in  $N_2$ , then by our construction it contributes 1 unit each to two access points  $a_i$  and  $a_j$ . However, the edge  $\{a_i, a_j\}$  is contained in  $\mathcal{T}$ . As  $C$  is an independent set, it can contain at most one of  $a_i$  and  $a_j$ . Hence, in this case,  $p$  contributes at most 1 unit as well. It follows that each 1 unit of weight of  $C$  comes from a unique point of  $N$ . Now, by our construction, 1 unit of weight for a point  $p$  of  $N$  is assigned to an access point  $a_i$  only if  $p$  is in the small disk  $D(a_i, \rho_1)$ . Hence, for each unit of weight of the points in  $C$ , there is a unique point in  $N$  that is contained in a small disk centered at a point of  $C$ . So, the number of points contained in the set of small disks centered at the access points in  $C$  is at least  $W$ .  $\blacktriangleleft$

We note that in the above, the number of points covered by small disks may be more than  $W$ , as the weight of each point contained in a small disk centered at a point of  $C$  might not be assigned to the points of  $C$ . Hence,  $W$  is only a lower bound.

► **Corollary 8.** *The solution computed by Weighted-Extraction has a gain of at least  $\text{OPT}_1/4 + \text{OPT}_2/2$ .*

**Proof.** First, note that by our construction, the sum of the weights of all points in  $A$  is  $n_1 + 2n_2$ . Now, as we pick  $C$  to be the maximum weight color class, the total weight of the points in  $C$  must be at least the average weight, which is  $(n_1 + 2n_2)/4 = n_1/4 + n_2/2$ . By Lemma 7, the gain of the solution, that is the number of points of  $N$  covered by the small disks with centers at points in  $C$ , is at least  $n_1/4 + n_2/2$ . As  $\text{OPT}_1 \leq n_1$  and  $\text{OPT}_2 \leq n_2$ , the gain is at least  $\text{OPT}_1/4 + \text{OPT}_2/2$ .  $\blacktriangleleft$

## 2.2 The second algorithm

This algorithm has three steps. In the first step, we construct an instance of the Minimum Weight Unit Disk Cover problem from the given instance of DC-2.

In Minimum Weight Unit Disk Cover (WUDC), given a set  $\mathcal{D}$  of unit disks along with a weight function  $w : \mathcal{D} \rightarrow \mathbb{R}^+$  and a set  $T$  of points in the plane, the goal is to find a subset  $\mathcal{D}' \subseteq \mathcal{D}$ , such that  $\mathcal{D}'$  covers  $T$  and the sum of the weights of the disks in  $\mathcal{D}'$  is minimized. We refer to this sum of weights as the *cost* of the solution  $\mathcal{D}'$ .

The construction of the instance  $\mathcal{I}$  of WUDC is as follows.  $T$  is set to be  $V$ .  $\mathcal{D}$  is the set of all large disks in  $L$ . The weight of each large disk  $D(a_i, \rho_2)$  is the number of points of  $N_1$  in the corresponding small disk  $D(a_i, \rho_1)$ , i.e.,  $w(D(a_i, \rho_2)) = |N_1 \cap D(a_i, \rho_1)|$ .



In the second step of the algorithm, a  $(1 + \epsilon)$ -approximate solution  $\mathcal{D}'$  to WUDC on  $\mathcal{I}$  is computed using the PTAS in [21], which runs in time  $|\mathcal{D}|^{O(1/\epsilon^9)}$ . In the third step, a solution to DC-2 is computed as follows. First, add all the large disks in  $\mathcal{D}'$  to the solution set. Next, for each large disk  $D(a_i, \rho_2)$  in  $\mathcal{D} \setminus \mathcal{D}'$ , add the corresponding small disk  $D(a_i, \rho_1)$  to the solution set.

The above algorithm runs in  $(n + m)^{O(1/\epsilon^9)}$  time. Next, we analyze this algorithm. Let  $\text{OPT}'$  be the optimal cost of  $\mathcal{I}$ .

► **Observation 9.** *The solution computed by the above algorithm covers all points of  $P$ .*

**Proof.** We pick either  $D(a_i, \rho_1)$  or  $D(a_i, \rho_2)$  in the solution for each access point  $a_i$ . Thus, by Observation 2, the points in  $N$  are covered by the selected disks. We also see that the solution  $\mathcal{D}'$  to WUDC covers the points in  $T$ . As we include the (large) disks of  $\mathcal{D}'$  in the DC-2 solution and  $T = V$ , the points of  $V$  are also covered. ◀

► **Lemma 10.**  *$\text{OPT}'$  is at most  $n_1 - \text{OPT}_1$ .*

**Proof.** Consider the optimal solution of DC-2. The solution covers all points of  $P$  and in particular the points of  $V$ . As each point of  $V$  is not in any small disk of  $S$ , it must be covered by a selected large disk. Thus, the set of selected large disks, say  $L'$ , is a valid cover for the points of  $V$ , and so is a feasible solution to  $\mathcal{I}$ . We show that the cost of this solution is  $n_1 - \text{OPT}_1$ , proving the lemma.

Now, the gain of the DC-2 solution from the points in  $N_1$  is  $\text{OPT}_1$ . Thus the union of small disks that are not selected (corresponding to the large disks in  $L'$ ) contains  $n_1 - \text{OPT}_1$  points of  $N_1$ . By construction of  $\mathcal{I}$ , these points are the only points that are assigned to the large disks in  $L'$ . Hence, the cost of  $L'$ , that is the sum of the weights of the large disks in  $L'$  is  $n_1 - \text{OPT}_1$ . ◀

By the above lemma and the fact that  $\mathcal{D}'$  is a  $(1 + \epsilon)$ -approximate solution, we have the following observation.

► **Observation 11.** *The cost of  $\mathcal{D}'$  is at most  $(1 + \epsilon) \text{OPT}' \leq (1 + \epsilon)(n_1 - \text{OPT}_1)$ .*

► **Lemma 12.** *The gain of the DC-2 solution computed by the second algorithm is at least  $(1 + \epsilon)\text{OPT}_1 - 4\epsilon \cdot \text{OPT}$ .*

**Proof.** Note that the gain is at least the number of points of  $N_1$  in the small disks that are not selected in the DC-2 solution. Equivalently, the gain is at least the sum of the weights of the large disks that are in  $\mathcal{D} \setminus \mathcal{D}'$ . By Observation 11, it follows that the gain is at least

$$\begin{aligned} n_1 - (1 + \epsilon)(n_1 - \text{OPT}_1) &= n_1 - n_1 + \text{OPT}_1 - \epsilon n_1 + \epsilon \cdot \text{OPT}_1 \\ &\geq \text{OPT}_1 - 4\epsilon \cdot \text{OPT} + \epsilon \cdot \text{OPT}_1 \\ &= (1 + \epsilon)\text{OPT}_1 - 4\epsilon \cdot \text{OPT} \end{aligned}$$

The inequality follows, as by Corollary 8,  $\text{OPT} \geq n_1/4 + n_2/2 \geq n_1/4$ . Hence, we obtain the desired lemma. ◀

## 2.3 Combining the two algorithms

Our main algorithm is a combination of the two above-mentioned algorithms. We invoke both on the DC-2 instance and return the solution with the larger gain.

► **Lemma 13.** *The gain of the larger-gain solution is at least  $\text{OPT}/(2.5 + \epsilon)$ .*



**Proof.** Let  $0 \leq \alpha \leq 1$  be such that  $\text{OPT}_1 = \alpha \cdot \text{OPT}$ . By Lemma 8 and 12, the gain of the larger solution is at least

$$\begin{aligned} & \max\{\alpha \cdot \text{OPT}/4 + (1 - \alpha)\text{OPT}/2, (1 + \epsilon)\alpha \cdot \text{OPT} - 4\epsilon \cdot \text{OPT}\} \\ & = \max\{\text{OPT}/2 - \alpha \cdot \text{OPT}/4, (1 + \epsilon)\alpha \cdot \text{OPT} - 4\epsilon \cdot \text{OPT}\} \end{aligned}$$

This function of  $\alpha$  is minimized when

$$\begin{aligned} & \text{OPT}/2 - \alpha \cdot \text{OPT}/4 = (1 + \epsilon)\alpha \cdot \text{OPT} - 4\epsilon \cdot \text{OPT} \\ \text{Or, } & (5/4 + \epsilon)\alpha = 1/2 + 4\epsilon \\ \text{Or, } & \alpha = (2 + 16\epsilon)/(5 + 4\epsilon) \end{aligned}$$

Thus, the gain is at least  $((1 + \epsilon)(2 + 16\epsilon)/(5 + 4\epsilon) - 4\epsilon)\text{OPT} = \text{OPT}/(5/2 + O(\epsilon))$ . By scaling  $\epsilon$  by a constant, we obtain the desired bound.  $\blacktriangleleft$

We state our main result of this section in the following theorem.

► **Theorem 14.** *There is an  $(n + m)^{O(1/\epsilon^9)}$ -time  $(2.5 + \epsilon)$ -approximation algorithm for DC-2.*

### 3 An FPT Algorithm for Sparse Instances

The parameterized version of DC-2 includes a fixed parameter  $k$ , where in addition to the original goal, we are assured that  $r_i = \rho_2$  for at most  $k$  access points  $a_i$ . We refer to this problem as Par-DC-2. Here, we reuse the notation from the previous section. Let  $\text{OPT}$  be the gain of any optimal solution. Recall the sets of small disks  $S$  and large disks  $L$ , and the sets of vulnerable points  $V$  and non-vulnerable points  $N$ . An instance of Par-DC-2 is called *s-sparse* for an integer  $s \geq 2$  if each point of  $V$  is in at most  $s$  large disks of  $L$ . We note that in such an instance, a point of  $N$  can lie in more than  $s$  large (or small) disks. We prove that Par-DC-2 is fixed-parameter tractable in  $k + s$ .

For a set  $S' \subseteq S$ , let the *gain of  $S'$*  be the number of points of  $N$  contained in the disks of  $S'$ . For a subset of large disks  $L' \subseteq L$ , let  $\text{sm}(L')$  be the set of corresponding small disks.

Next, we describe our algorithm. Our algorithm is based on a recursive branching strategy. Algorithm 2 shows the pseudo-code of a recursive subroutine that is the main procedure used by our algorithm. The subroutine **Recursive-Branching**( $V', L', l$ ) takes as input a subset of points  $V' \subseteq V$  that are yet to be covered, a subset  $L' \subseteq L$  of disks already chosen to cover, and an integer  $l > 0$  that denotes the depth of recursion. The subroutine returns a True/False flag. If the flag is True, it also outputs a set of  $k$  large disks  $L'_1 \supseteq L'$  such that  $L'_1 \setminus L'$  covers  $V'$ , and the gain  $g$  corresponding to the solution  $L'_1$ , i.e., the gain of  $\text{sm}(L \setminus L'_1)$ . Inside the subroutine, we first check if  $V'$  is empty, and if that is the case,  $L'$  itself is returned as the solution along with its gain. If the depth  $l = k + 1$ , but  $V' \neq \emptyset$ , the False flag is returned, indicating there is no feasible solution along this branch, so it should be pruned here. Otherwise,  $l \leq k$  and  $V' \neq \emptyset$ . In that case, we consider any arbitrary point  $p \in V'$ . There are at most  $s$  large disks in  $L$  that cover  $p$ . We branch on each such large disk  $D$ . In each branch, we remove the points of  $V'$  that are covered by  $D$  and recurse on the set of remaining points with  $L' \cup \{D\}$  being the new set of selected disks and  $l + 1$  as the new depth. We also keep track of the largest gain solution returned by these calls and return it if the flag counter  $b$  is set to True. If  $b$  is False, we return the False flag.

The main algorithm makes a call to **Recursive-Branching**( $V, \emptyset, 1$ ). If the call returns True, it outputs the solution where the large disks of  $L'_1$  are selected. Since there are no remaining points in  $V$  to cover, the remaining small disks  $\text{sm}(L \setminus L'_1)$  may be selected to cover a number of points in  $N$  equal to the gain of  $L'_1$ . Otherwise, it concludes that there is no feasible solution with  $k$  large disks.

---

**Algorithm 2** Recursive-Branching( $V', L', l$ ).

---

**Input:** A set of points  $V' \subseteq V$ , a subset  $L' \subseteq L$ , and an integer  $l > 0$ .

**Output:** A flag True/False, and if the flag is True, a set of  $k$  large disks  $L'_1 \supset L'$  such that  $L'_1 \setminus L'$  covers  $V'$  and the gain  $g$  corresponding to  $L'_1$ ; otherwise,  $\emptyset$  and 0.

```

1: if  $V' = \emptyset$  then
2:   Let  $g_1$  be the gain of the small disks in  $S \setminus \text{sm}(L')$ 
3:   return True,  $L', g_1$ 
4: if  $l = k + 1$  then
5:   return False,  $\emptyset, 0$  ▷ Executing this line implies  $V' \neq \emptyset$ .
6: Let  $p \in V'$  be any arbitrary point.
7: Let  $L_p$  be the set of large disks in  $L$  that contain  $p$ .
8:  $b \leftarrow \text{False}$ 
9:  $g \leftarrow 0$ 
10:  $L'_1 \leftarrow \emptyset$ 
11: for each disk  $D \in L_p$  do
12:   Let  $V_D \subset V'$  be the subset of points of  $V'$  not in  $D$ 
13:    $b_D, L_D, g_D \leftarrow \text{Recursive-Branching}(V_D, L' \cup \{D\}, l + 1)$ 
14:   if  $b_D = \text{True}$  then
15:      $b \leftarrow \text{True}$ 
16:     if  $g < g_D$  then
17:        $g \leftarrow g_D$ 
18:        $L'_1 \leftarrow L_D$ 
19: if  $b = \text{False}$  then
20:   return False,  $\emptyset, 0$ 
21: else
22:   return True,  $L'_1, g$ 

```

---

Next, we analyze the algorithm. Consider the recursion tree  $\mathcal{R}$  corresponding to Recursive-Branching( $V, \emptyset, 1$ ). We define the level of the nodes in  $\mathcal{R}$  as follows. The level of the root is 1. The level of any other node is 1 plus the level of its parent. Let  $L^*$  denote the large disks selected in the optimal solution. In our analysis, we assume that  $L^*$  does not contain any redundant disk, i.e., removing any disk from  $L^*$  leaves  $V$  uncovered. If it contains such a redundant disk, then we can try our algorithm with a smaller value  $k' < k$ .

We prove the correctness of Recursive-Branching with the following lemma.

► **Lemma 15.** *For each  $1 \leq l \leq k$ , there is a node  $u$  in  $\mathcal{R}$  of level  $l$  with the property that a call is made at  $u$  to Recursive-Branching( $V_D, L' \cup \{D\}, l + 1$ ) such that  $V \setminus V_D$  is the subset of points of  $V$  covered by the disks of  $L' \cup \{D\}$ , and  $L' \cup \{D\}$  is a subset of  $L^*$  with  $l$  disks.*

**Proof.** We use induction on the level  $l \geq 1$ . In the base case, consider  $l = 1$ . The root node at level 1 selects a point  $p \in V$  and recurses for each disk  $D \in L_p$ . Now,  $p$  must be covered by a large disk  $D^* \in L^*$  that is in  $L_p$ . Consider the call Recursive-Branching( $V_{D^*}, \{D^*\}, 2$ ) made at the root. By definition,  $V \setminus V_{D^*}$  is the set of points in  $V$  that are covered by  $D^*$ . Hence, the statement is true in the base case. Now, suppose the statement is true for any level  $l < k$ . We prove that the statement is true for level  $l + 1$  as well.

By induction argument, there is a node  $u'$  of level  $l < k$  that makes a call to Recursive-Branching( $V_D, L' \cup \{D\}, l + 1$ ) such that  $V \setminus V_D$  is the subset of points of  $V$  covered by the disks of  $L' \cup \{D\}$ , and  $L' \cup \{D\}$  is a subset of  $L^*$  with  $l < k$  disks. Denote the child of  $u'$  in  $\mathcal{R}$  by  $u$  that corresponds to this call and thus receives the set of large disks  $L' \cup \{D\} \subset L^*$ .

Now,  $L' \cup \{D\}$  does not cover  $V$ , as otherwise,  $L^*$  contains a redundant disk. Thus, we have that  $V_D \neq \emptyset$ . Suppose  $p'$  is the point of  $V_D$  arbitrarily chosen during the execution of the call corresponding to  $u'$ . Then a recursive call is made for each  $D' \in L_{p'}$ . Now, by our assumption,  $p'$  is not covered by the disks in  $L' \cup \{D\} \subset L^*$ . Hence, there must be a disk  $D^* \in L^* \setminus (L' \cup \{D\})$  that covers  $p'$  and is contained in  $L_{p'}$ . Consider the call  $\text{Recursive-Branching}(V_{D^*}, L' \cup \{D\} \cup \{D^*\}, l+2)$  made at  $u$ . By definition,  $V_D \setminus V_{D^*}$  is the subset of points of  $V_D$  contained in  $D^*$ . As any point in  $V \setminus V_D$  is in a disk of  $L' \cup \{D\}$ ,  $V \setminus V_{D^*}$  is the exact subset of points of  $V$  covered by the disks of  $L' \cup \{D\} \cup \{D^*\}$ . Also,  $|L' \cup \{D\} \cup \{D^*\}| = l+1$ . Hence, the statement is also true for  $u$ . ◀

The next lemma shows that the algorithm returns a solution with optimal gain.

► **Lemma 16.**  *$\text{Recursive-Branching}(V, \emptyset, 1)$  always returns the True flag and the corresponding solution has the gain value OPT.*

**Proof.** Consider the node in  $\mathcal{R}$  of level  $k$  that makes a call to  $\text{Recursive-Branching}(V_D, L' \cup \{D\}, k+1)$  where  $L' \cup \{D\} = L^*$ . The existence of such a node follows due to Lemma 15. Now, note that  $L^*$  covers  $V$ , and thus  $V_D = \emptyset$ . Hence, this call returns True,  $L^*$ , OPT to its parent node. Now, inside each call, we keep track of the maximum gain solution returned by all recursive calls and return it to the parent. Hence, the root node always returns True and the gain of the solution returned must be at least OPT. ◀

We will show that the total number of nodes of  $\mathcal{R}$  is bounded by  $s^{O(k)}$ . As the algorithm spends only a polynomial time at each node, the desired result follows. Now, note that at each node, the algorithm can make at most  $s$  calls, as each point of  $V$  is in at most  $s$  disks of  $L$ . Moreover, as we prune any branch at level  $k+1$ , the maximum level is  $k+1$ . Thus, the total number of nodes is  $s^{O(k)}$ . We state our result in the following theorem.

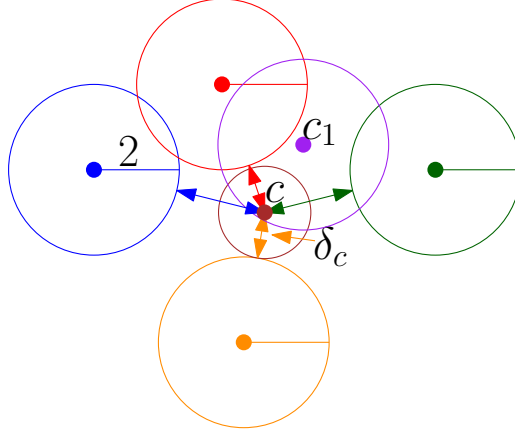
► **Theorem 17.** *Par-DC-2 can be solved in time  $s^{O(k)}(m+n)^{O(1)}$  on any  $s$ -sparse instance.*

## 4 Parameterized Hardness of DC-2

We give a reduction from Dominating set for unit disks. In this problem, we are given a set of  $n$  unit disks (radius 1)  $\mathcal{D}$  in the plane and a parameter  $k$ , and the goal is to select a subset  $\mathcal{D}' \subseteq \mathcal{D}$  of  $k$  disks such that for each disk  $D$  in  $\mathcal{D}$ ,  $D \in \mathcal{D}'$  or there is a disk  $D'$  in  $\mathcal{D}'$  such that  $D \cap D' \neq \emptyset$ .

Marx [24] proved that Dominating set for unit disks is  $W[1]$ -hard. In fact, he proved that assuming ETH, the problem cannot be solved in  $f(k) \cdot n^{o(\sqrt{k})}$  time for any computable function  $f$ . Next, we describe our reduction.

Let  $\mathcal{I}$  be the given instance of Dominating set for unit disks. Let  $C$  be the set of centers of the disks in  $\mathcal{I}$ . The distance between a point  $p$  and a set  $S$  is defined as  $\min_{q \in S} \|p - q\|$ . For any point  $c \in C$ , let  $\delta_c$  be the minimum distance between  $c$  and the disks in  $\{D(c', 2) \mid c' \in C \text{ and } c \notin D(c', 2)\}$  (see Figure 2). Let  $\delta_1 = \min_{c \in C} \delta_c$ . Also, let  $\delta_2 = \min_{c, c' \in C} \|c - c'\|$  with  $c \neq c'$ . Denote  $\delta = \min\{\delta_1, \delta_2\}$ . Also, let  $\epsilon = \delta/6$ . We set  $A = C$ ,  $P = \{(x + \epsilon, y) \mid (x, y) \in C\}$ ,  $\rho_1 = \epsilon/2$ , and  $\rho_2 = 2 + \epsilon$ . Also, we use the same parameter  $k$ . Let  $\mathcal{I}'$  be the constructed instance of Par-DC-2. The construction takes  $|\mathcal{D}|^{O(1)}$  time.



■ **Figure 2** Figure showing the calculation of  $\delta_c$ :  $D(c_1, 2)$  is not considered here, as  $c \in D(c_1, 2)$ .

► **Observation 18.** For any  $c \in C$ , the small disk  $D(c, \rho_1) \cap P = \emptyset$ .

**Proof.** Let  $c = (x, y)$ . Consider any  $p = (x_1, y_1) \in P$ . First, consider the case when  $x_1 = x + \epsilon$  and  $y_1 = y$ , then  $\|c - p\| = \epsilon > \rho_1$ . Thus,  $p \notin D(c, \rho_1)$ . Next, consider the case when  $x_1 \neq x + \epsilon$  or  $y_1 \neq y$ . This implies that there is another point  $c' = (x', y') \neq c$  such that  $p$  was constructed from  $c'$ , i.e.,  $x_1 = x' + \epsilon$  and  $y_1 = y'$ . Now, suppose  $\|c - p\| \leq \rho_1$ . Then,  $\|c - c'\| \leq \|c - p\| + \|p - c'\| \leq \rho_1 + \epsilon < \delta \leq \delta_2$ . In particular,  $\|c - c'\| < \delta_2$ . But, by the definition of  $\delta_2$ , this is a contradiction. So,  $\|c - p\| > \rho_1$  in this case, and so  $p \notin D(c, \rho_1)$ . As the two cases are exhaustive, the observation follows. ◀

The above observation implies that the gain of any solution is 0. Thus, the hardness comes from the ability to cover the points of  $P$  using  $k$  large disks.

► **Lemma 19.**  $\mathcal{I}$  has a dominating set of size  $k$  if and only if there is a feasible solution for  $\mathcal{I}'$ .

**Proof.** First, suppose  $\mathcal{I}$  has a dominating set of size  $k$ . Let  $C'$  be the set of centers of the  $k$  disks in the dominating set. We claim that the  $k$  large disks at the centers in  $C'$  cover the points in  $P$ . Consider a point  $p = (x_1, y_1) \in P$  and let  $c = (x, y) \in C$  be such that  $x_1 = x + \epsilon$  and  $y_1 = y$ . We know that there is  $c' \in C'$  such that  $D(c', 1)$  intersects  $D(c, 1)$ . Thus  $\|c - c'\| \leq 2$ . So,  $\|c' - p\| \leq \|c' - c\| + \|c - p\| \leq 2 + \epsilon$ . Hence, the large disk at  $c' \in C'$  covers  $p$ .

Next, suppose there is a feasible solution for  $\mathcal{I}'$ . Thus, there is a size  $k$  subset  $C' \subseteq A$  such that the large disks at  $C'$  cover the points in  $P$ . We claim that the set of disks  $\mathcal{D}' \subseteq \mathcal{D}$  at the centers in  $C'$  form a dominating set. Consider any disk  $D(c, 1) \in \mathcal{D}$ . Suppose  $D(c, 1) \notin \mathcal{D}'$ . Consider the point  $p = (x_1, y_1) \in P$  such that  $c = (x, y)$ ,  $x_1 = x + \epsilon$  and  $y_1 = y$ . Then there is  $c' \in C'$  such that  $D(c', \rho_2)$  covers  $p$ . We claim that  $c \in D(c', 2)$ . Suppose that is not true. Then as per the definition of  $\delta_c$ ,  $c$  is at least  $\delta_c$  distance away from  $D(c', 2)$ . Thus,  $\|c - c'\| \geq 2 + \delta_c \geq 2 + \delta = 2 + 6\epsilon$ . However,  $\|c - c'\| \leq \|c - p\| + \|p - c'\| \leq \epsilon + \rho_2 = 2 + 2\epsilon$ . We obtain a contradiction, and hence  $c \in D(c', 2)$ . Equivalently,  $D(c, 1) \cap D(c', 1) \neq \emptyset$ , and so  $D(c', 1)$  dominates  $D(c, 1)$ . Hence,  $\mathcal{D}'$  is a dominating set of size  $k$ . ◀

Based on the above discussion, we have the following theorem.

► **Theorem 20.** Par-DC-2 is W[1]-hard. Moreover, assuming ETH, Par-DC-2 cannot be solved in  $f(k) \cdot (n + m)^{o(\sqrt{k})}$  time for any computable function  $f$ .

## References

- 1 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004. doi:10.1137/S0097539702416402.
- 2 Sayan Bandyapadhyay, Anil Maheshwari, Sasanka Roy, Michiel Smid, and Kasturi R. Varadarajan. Geometric covering via extraction theorem. In Venkatesan Guruswami, editor, *15th Innovations in Theoretical Computer Science Conference, ITCS 2024, January 30 to February 2, 2024, Berkeley, CA, USA*, volume 287 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.ITCS.2024.7.
- 3 Aritra Banik, Fahad Panolan, Venkatesh Raman, Vibha Sahlot, and Saket Saurabh. Parameterized complexity of geometric covering problems having conflicts. *Algorithmica*, 82(1):1–19, 2020. doi:10.1007/S00453-019-00600-W.
- 4 Nikhil Bansal and Kirk Pruhs. Weighted geometric set multi-cover via quasi-uniform sampling. *Journal of Computational Geometry*, 7(1):221–236, 2016. doi:10.20382/JOCG.V7I1A11.
- 5 Ahmad Biniaz. Plane hop spanners for unit disk graphs: Simpler and better. *Computational Geometry: Theory and Applications*, 89, 2020. doi:10.1016/J.COMGEO.2020.101622.
- 6 Lorna Booth, Jehoshua Bruck, Massimo Franceschetti, and Ronald Meester. Covering algorithms, continuum percolation and the geometry of wireless networks. *The Annals of Applied Probability*, 13(2):722–741, 2003.
- 7 Norbert Bus, Shashwat Garg, Nabil Mustafa, and Saurabh Ray. Improved local search for geometric hitting set. In *Proc. of the 32nd International Symposium on Theoretical Aspects of Computer Science (STACS)*, 2015. doi:10.4230/LIPICS.STACS.2015.184.
- 8 Timothy M. Chan and Elyot Grant. Exact algorithms and apx-hardness results for geometric packing and covering problems. *Computational Geometry*, 47(2, Part A):112–124, 2014. Special Issue: 23rd Canadian Conference on Computational Geometry (CCCG11). doi:10.1016/j.comgeo.2012.04.001.
- 9 Timothy M. Chan, Elyot Grant, Jochen Könemann, and Malcolm Sharpe. Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1576–1585. SIAM, 2012. doi:10.1137/1.9781611973099.125.
- 10 Steven Chaplick, Minati De, Alexander Ravsky, and Joachim Spoerhase. Approximation schemes for geometric coverage problems. In *26th Annual European Symposium on Algorithms (ESA 2018)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPICS.ESA.2018.17.
- 11 Kenneth L. Clarkson and Kasturi R. Varadarajan. Improved approximation algorithms for geometric set cover. *Discrete & Computational Geometry*, 37(1):43–58, 2007. doi:10.1007/S00454-006-1273-8.
- 12 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
- 13 Ling Ding, Weili Wu, James Willson, Lidong Wu, Zaixin Lu, and Wonjun Lee. Constant-approximation for target coverage problem in wireless sensor networks. In *2012 Proceedings IEEE INFOCOM*, pages 1584–1592. IEEE, 2012. doi:10.1109/INFOCOM.2012.6195527.
- 14 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633, 2014. doi:10.1145/2591796.2591884.
- 15 Alon Efrat, Frank Hoffman, Klaus Kriegel, Christof Schultz, and Carola Wenk. Geometric algorithms for the analysis of 2d-electrophoresis gels. In *Proceedings of the fifth annual international conference on Computational biology*, pages 114–123, 2001. doi:10.1145/369133.369181.

- 16 Uriel Feige. A threshold of  $\ln n$  for approximating set cover. *J. ACM*, 45(4):634–652, 1998. doi:10.1145/285055.285059.
- 17 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.
- 18 Dorit S Hochbaum and Wolfgang Maass. Fast approximation algorithms for a nonconvex covering problem. *Journal of Algorithms*, 8(3):305–323, 1987. doi:10.1016/0196-6774(87)90012-5.
- 19 Madhukar R. Korupolu, C. Greg Plaxton, and Rajmohan Rajaraman. Analysis of a local search heuristic for facility location problems. *J. Algorithms*, 37(1):146–188, 2000. doi:10.1006/JAGM.2000.1100.
- 20 Katarzyna Kowalska and Michał Pilipczuk. Parameterized and approximation algorithms for coverings points with segments in the plane. In *41st International Symposium on Theoretical Aspects of Computer Science*, 2024. doi:10.4230/LIPIcs.STACS.2024.47.
- 21 Jian Li and Yifei Jin. A PTAS for the weighted unit disk cover problem. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP*, volume 9134 of *Lecture Notes in Computer Science*, pages 898–909. Springer, 2015. doi:10.1007/978-3-662-47672-7\_73.
- 22 Anil Maheshwari, Saeed Mehrabi, Sasanka Roy, and Michiel Smid. Covering points with concentric objects. In *Proceedings of the 32nd Canadian Conference on Computational Geometry*, pages 436–452, 2016.
- 23 Dániel Marx. Efficient approximation schemes for geometric problems? In *European Symposium on Algorithms*, pages 448–459. Springer, 2005. doi:10.1007/11561071\_41.
- 24 Dániel Marx. Parameterized complexity of independence and domination on geometric graphs. In *Parameterized and Exact Computation: Second International Workshop, IWPEC 2006, Zürich, Switzerland, September 13-15, 2006. Proceedings 2*, pages 154–165. Springer, 2006. doi:10.1007/11847250\_14.
- 25 Dániel Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008. doi:10.1093/COMJNL/BXM048.
- 26 Dániel Marx and Michał Pilipczuk. Optimal parameterized algorithms for planar facility location problems using voronoi diagrams. *ACM Transactions on Algorithms (TALG)*, 18(2):1–64, 2022. doi:10.1145/3483425.
- 27 Nabil H. Mustafa and Saurabh Ray. Improved results on geometric hitting set problems. *Discrete & Computational Geometry*, 44(4):883–895, 2010. doi:10.1007/S00454-010-9285-9.
- 28 Neil Robertson, Daniel P Sanders, Paul Seymour, and Robin Thomas. Efficiently four-coloring planar graphs. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 571–575, 1996. doi:10.1145/237814.238005.
- 29 Anju Sangwan and Rishi Pal Singh. Survey on coverage problems in wireless sensor networks. *Wireless Personal Communications*, 80:1475–1500, 2015. doi:10.1007/S11277-014-2094-3.
- 30 Kasturi Varadarajan. Weighted geometric set cover via quasi-uniform sampling. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 641–648, 2010. doi:10.1145/1806689.1806777.