FACHHOCHSCHULE **TRIER**
Hochschule für Technik, Wirtschaft und Gestaltung
University of Applied Sciences
Informatik

Informatik-Bericht Nr. 2008-3

Schriftenreihe Fachbereich Informatik, Fachhochschule Trier

Multiobjective Disk Cover Admits a PTAS

Christian Glaßer* Christian Reitwießner† Heinz Schmitz‡

Abstract

We introduce multiobjective disk cover problems and study their approximability. We construct a polynomial-time approximation scheme (PTAS) for the multiobjective problem where k types of points (customers) in the plane have to be covered by disks (base stations) such that the number of disks is minimized and for each type of points, the number of covered points is maximized. Our approximation scheme can be extended so that it works with the following additional features: interferences, different services for different types of customers, different shapes of supply areas, weighted customers, individual costs for base stations, and payoff for the quality of the obtained service.

Furthermore, we show that it is crucial to solve this problem in a multiobjective way, where all objectives are optimized at the same time. The constrained approach (i.e., the restriction of a multiobjective problem to a single objective) often used for such problems can significantly degrade their approximability. We can show non-approximability results for several single-objective restrictions of multiobjective disk cover problems. For example, if there are 2 types of customers, then maximizing the supplied customers of one type is not even approximable within a constant factor, unless P = NP.

1 Introduction

Geometric cover problems have received much attention in recent years, mostly due to their applicability to wireless networks. Typically, a service provider aims to deliver various kinds of services to customers and therefore has to choose base station locations such that customer locations can be covered. Various optimization problems arise in this context in a natural way. For example, for a given set P of customer locations and a set D of possible base station locations in the Euclidean plane, the Unit Disk Cover Problem tries to find a minimal subset of D such that all customers in P are covered by unit disks whose centers belong to D (hence assuming equivalent base stations and ignoring obstacles to the signal propagation) [CCJ90, CMWZ04, NV06]. In another version of the problem one tries to maximize the number of supplied customers with a given budget of base

*Julius-Maximilians-Universität Würzburg, Theoretische Informatik, Am Hubland, 97074 Würzburg, Germany. EMail: glasser@informatik.uni-wuerzburg.de

†Julius-Maximilians-Universität Würzburg, Theoretische Informatik, Am Hubland, 97074 Würzburg, Germany. EMail: reitwiessner@informatik.uni-wuerzburg.de

‡Fachhochschule Trier, Fachbereich Informatik, Schneidershof, 54293 Trier, Germany. EMail: schmitz@informatik.fh-trier.de

stations, e.g. [GRV05]. So far these problems have been studied only in terms of single-objective optimization where either disk locations or customer supply have been optimized.

In contrast, here we are interested in the complete trade-offs when both objectives are considered *at once*. These trade-offs give not only a better insight in the nature of the problem, but also allow a human decision-maker to choose an appropriate solution according to aspects that are perhaps not quantifiable or that differ from instance to instance.

This paper introduces *multiobjective* disk cover problems and presents the first study of their approximability. We want to minimize the number of base stations and simultaneously maximize the number of supplied customer locations. More than that, we allow *different types* of customers such that for each type, the number of supplied customers has to be maximized. For example, for $k = 2$ types of customers this captures the scenario where a service provider wants to optimize a wireless network with customers having subscribed to two different services. More generally, an instance of the Disc Cover Problem with k types of customers (k-DC) has k sets $P_1, \ldots, P_k$ of customer locations, a set of potential base station locations D and a disk radius r describing the range of action of a base station. We seek a valid subset of base station locations (i.e., respecting a minimum-distance constraint) such that the number of base stations is minimized, and for each type of customers, the numbers of covered customers is maximized. In practice, several additional aspects can be taken into account to obtain more realistic models. For instance, if a customer receives signals from two base stations, then interferences may have a negative effect on the quality of service. So actually, here one wants to maximize the number of points that are covered by *exactly* one disk. To capture the aspect of interference, we also investigate exact versions of k-DC, i.e., where the numbers of *uniquely* covered customers of the different types are considered (k-EDC).

Trade-offs in multiobjective optimization are captured by the notion of the so-called *Pareto curve* which is the set of all solutions whose vector of objective values is not dominated by any other solution (for an introduction see, e.g., [Ehr05]). In most interesting cases however, Pareto curves are computationally hard in the sense that we do not know polynomial-time algorithms computing them. The reason for this is that the Pareto curve may have exponential size (which is not the case in our setting), or because it comprises optimal solutions of NP-hard single-objective optimization problems (which is the case here). A reasonable approach to avoid these difficulties is to approximate the set of non-dominated solutions using the concept of the ϵ-approximate Pareto curve. Informally, for every solution S of the Pareto curve there is a solution S' in the ϵ-approximate Pareto curve that is within a factor $(1 + \epsilon)$ of S, or better, in all objectives. The question whether there exist fast approximation schemes for Pareto curves has been addressed for several multiobjective optimization problems [SO95a, SO95b, PY00]. The systematic study of the theory of multiobjective approximation was initiated by Papadimitriou and Yannakakis [PY00], see also [VY05, DY07].

Our contribution. We introduce *multiobjective* disk cover problems and study their approximability. We construct polynomial-time approximation schemes for the multiobjective problems k-DC and k-EDC where $k \geq 1$. So for each of the problems there exists an algorithm, which, given a problem instance I and some $\epsilon > 0$, outputs an ϵ-approximate Pareto curve in time polynomial in $|I|$ (Theorems 3.4 and 3.5). On the methodological side we extend the *shifting strategy* introduced by Hochbaum and Maass [HM85] to the multiobjective case.

We also discuss the possibility to extend our algorithms so that they work with the following

features: different services for different types of customers, different shapes of supply areas, weighted customers, individual costs for base stations, and payoff for the quality of the obtained service. Although we mention only problems where the number of covered points of different types have to be maximized, one can also think of an application where some types of points have to be maximized, while others are to be minimized. Only minor modifications of our algorithms are needed to take this into account as well.

Our paper also shows that we should be careful when looking for an appropriate model for a given practical problem. The choice of the right model can be crucial for a successful algorithmic solution. In our paper we see this at two places:

1. Our models contain a minimum-distance constraint for disk locations. On one hand, this constraint plays an important role in the construction of the PTAS. Without this assumption, the problem becomes more difficult such that the shifting strategy does not yield a PTAS. On the other hand, this minimum-distance constraint is actually present in practical settings: It usually makes no sense to build base stations, fire departments, drugstores, etc. *arbitrarily* close to each other, and a small constant specifying their minimum distance can always be identified. So we may add the constraint to our model and exploit it to achieve better approximation algorithms. This shows that a *too general choice* of the model can complicate the solution of the underlying practical problem.

2. The approximability of a multiobjective problem does not necessarily imply that the restriction to a single objective is approximable. The reason for this apparent contradiction is that an optimization algorithm can exploit trade-offs between the single objectives if it optimizes all objectives at the same time. We show non-approximability results for several restrictions of k-DC and k-EDC (Theorems 4.2 and 4.5). For example, for $k \geq 2$, no restriction of k-EDC to a single criterion is approximable within a constant factor (unless P = NP), while the general (multiobjective) version of k-EDC admits even a PTAS. This shows that the frequently used constrained approach (i.e., the restriction of a multiobjective problem to a single objective) can considerably degrade the approximability. In other words, also a *too restricted choice* of the model can complicate the solution of the underlying problem.

Related work. The single-objective disk cover problem was initially examined in the continuous version (i.e., no given disk centers) by Hochbaum and Maass [HM85] who construct a PTAS for this problem. The version with given disk centers has been studied by several authors [CMWZ04, NV06, CKLT07, BMCK08], but in general only constant-factor approximation results are known (which shows again the influence of the minimum-distance constraint). Calinescu, Mandoiu, Wan and Zelikovsky [CMWZ04] investigated a variant of the disk cover problem where the number of disks needed to cover all points is to be minimized. Some of their results were improved by Narayanappa and Vojtechovsky [NV06]. Very recently, the Unique Cover Problem on unit disks (the number of points covered by exactly one disk have to be maximized) and its approximability has been studied by Erlebach and van Leeuwen [EvL08]. Several other variations of the disk cover problem where a solution includes specifying radii for the individual disks were analyzed in [Cha03, EJS05, AAB$^+$06]. Cannon and Cowen [CC04] studied the single-objective problem of minimizing the number of disks where one type of customers must be covered while the other one has to be avoided. Various partial covering problems were investigated by Gandhi, Kuller and Srinivasan [GKS04]. These

problems are concerned with covering a given amount of elements while minimizing the cost of such a covering. In contrast to most of the afore mentioned papers, in [GKS04] the multiobjective version of some special covering problem on graphs is also examined.

2 Definitions

We recall some standard notations, see e.g., [PY00, VY05]. A *multiobjective optimization problem* Π has a set of valid instances $\mathcal{I}$, and for every instance $I \in \mathcal{I}$ there is a set $\mathcal{S}(I)$ of feasible and polynomially length-bounded solutions for I. As usual, we assume that $\mathcal{I}$ is decidable in polynomial time, and that there is a polynomial-time algorithms that decides on input (I, S) whether $S \in \mathcal{S}(I)$. Moreover, we have $K \geq 1$ polynomial-time computable objective functions f_i that map every $I \in \mathcal{I}$ and $S \in \mathcal{S}(I)$ to some value $f_i(I, S) \in \mathbb{N}$. Note that every optimization problem with objective functions that have values in $\mathbb{Q}$ can be transformed into an equivalent problem satisfying the previous definition. A vector $goal \in \{\min, \max\}^K$ specifies whether the i-th objective has to be minimized or maximized, respectively. So for an instance I we can evaluate every $S \in \mathcal{S}(I)$ to the K-vector $f(I, S) = (f_1(I, S), \ldots, f_K(I, S))$ of values with respect to the given objective functions.

We say a solution $S \in \mathcal{S}(I)$ *dominates* a solution $S' \in \mathcal{S}(I)$ if for all $1 \leq i \leq K$ it holds that $f_i(I, S) \leq f_i(I, S')$ if f_i is to be minimized (and $f_i(I, S) \geq f_i(I, S')$ if f_i is to be maximized), with at least one strict inequality. Denote by $P^{sol}(I) \subseteq \mathcal{S}(I)$ the *Pareto-solution set* for I, i.e., the set of all non-dominated solutions for I. The *Pareto-value set* for I is $P^{val}(I) = \{f(I, S) \mid S \in P^{sol}(I)\}$.

Let $\epsilon = (\epsilon_1, \ldots, \epsilon_K)$ be a K-vector of numbers $\epsilon_i \geq 0$. A solution $S \in \mathcal{S}(I)$ *ϵ-covers* a solution $S' \in \mathcal{S}(I)$ if for all $1 \leq i \leq K$ it holds that $f_i(I, S) \leq (1 + \epsilon_i) f_i(I, S')$ if f_i is to be minimized (and $(1 + \epsilon_i) f_i(I, S) \geq f_i(I, S')$ if f_i is to be maximized).

A set $P^{sol}_\epsilon(I) \subseteq \mathcal{S}(I)$ is an *ϵ-approximate Pareto-solution set* for I if for all $S' \in P^{sol}(I)$ there is some $S \in P^{sol}_\epsilon(I)$ that ϵ-covers S'. (Note that an ϵ-approximate Pareto-solution set can contain dominated points.) We call $P^{val}_\epsilon(I) \subseteq \mathbb{N}^K$ an *ϵ-approximate Pareto-value set* for I if $P^{val}_\epsilon(I) = \{f(I, S) \mid S \in P^{sol}_\epsilon(I)\}$ for some set $P^{sol}_\epsilon(I)$. Note that for fixed ϵ there may be more than one ϵ-approximate Pareto-solution set for I. Moreover, if $\mathcal{S}(I) \neq \emptyset$ then $P^{sol}(I) \neq \emptyset$ and $P^{sol}_\epsilon(I) \neq \emptyset$. If $\epsilon = (\delta, \ldots, \delta)$ for some $\delta > 0$ we simply write $P^{sol}_\delta(I)$ and the like.

A multiobjective optimization problem Π is *ϵ-approximable in polynomial time* if there is a polynomial-time algorithm, which on input $I \in \mathcal{I}$ outputs an ϵ-approximate Pareto-solution set $P^{sol}_\epsilon(I)$. Problem Π has a *polynomial-time approximation scheme* (PTAS) if there is an algorithm, which, given $I \in \mathcal{I}$ and $\delta > 0$, outputs an δ-approximate Pareto-solution set $P^{sol}_\delta(I)$ in time polynomial in $|I|$. APX is the class of all single-objective optimization problems that are δ-approximable for some $\delta > 0$. For some vector x denote by $|x|$ its Euclidean norm. If S is a finite set, then $|S|$ gives the cardinality of S. Both cases will be distinguishable from the context without confusion. Moreover, we use $[a, b]$ as an abbreviation for $\{a, a + 1, \ldots, b\}$.

Next we define $(k + 1)$-objective disk-cover problems. As is standard for such problems, we always want to minimize the number of disks which is the first objective in all of the following problems. The parameter $k \geq 1$ denotes the number of different types of points we want to cover. Moreover, $\varrho \in (0, 2]$ is a fixed rational constant that determines the minimal distance $\varrho \cdot r$ between different

disks of radius r. This *minimum-distance constraint* plays an important role in our model, since it is crucial for the polynomial running time of the approximation algorithm we construct in section 3. We cannot well approximate instances that essentially depend on coverings where the disks are very close to each other. This insight has an important consequences for the choice of an appropriate model: If such degenerated instances can be excluded by practical reasons (e.g., because it makes no sense to build base stations, fire departments, drugstores, etc. arbitrarily close to each other), then we should add the minimum-distance constraint to our model and exploit it to achieve a better approximability.

k-Objective Disk Cover (k-DC$_\varrho$)

Instance: k finite sets of points $P_1, \ldots, P_k \subseteq \mathbb{Z} \times \mathbb{Z}$, disk radius $r \in \mathbb{N}$, finite set of disk positions $D \subseteq \mathbb{Z} \times \mathbb{Z}$

Solution: a selection $S \subseteq D$ such that for all different $x, y \in S$, $|x - y| \geq \varrho \cdot r$

Goals: $(\min |S|, \max |C_1|, \ldots, \max |C_k|)$ where $C_i = \{x \in P_i \mid \exists y \in S, |x - y| \leq r\}$

k-Objective Exact Disk Cover (k-EDC$_\varrho$)

Instance: k finite sets of points $P_1, \ldots, P_k \subseteq \mathbb{Z} \times \mathbb{Z}$, disk radius $r \in \mathbb{N}$, finite set of disk positions $D \subseteq \mathbb{Z} \times \mathbb{Z}$

Solution: a selection $S \subseteq D$ such that for all different $x, y \in S$, $|x - y| \geq \varrho \cdot r$

Goals: $(\min |S|, \max |C_1|, \ldots, \max |C_k|)$ where $C_i = \{x \in P_i \mid \exists! y \in S, |x - y| \leq r\}$

The value of ϱ will be always clear from the context. So for simplicity we write k-DC and k-EDC instead of k-DC$_\varrho$ and k-EDC$_\varrho$.

We also discuss single-objective versions of these problems. Following Diakonikolas and Yannakakis [DY07] we define the restricted versions of multiobjective problems (also known as the ε-constraint problem [Ehr05]). Let Π be a K-objective optimization problem with objectives $(f_1, \ldots, f_K)$ and goals $(g_1, \ldots, g_K)$. The restriction to the i-th objective is the following single-objective problem.

Restriction of Π to the i-th objective (Restricted$_i$-Π)

Instance: an instance I of Π and numbers $B_1, \ldots, B_{i-1}, B_{i+1}, \ldots, B_K \in \mathbb{N}$

Solution: a solution S for I such that for $j \in [1, K] - \{i\}$ it holds that $(g_j = \max \Rightarrow f_j(I, S) \geq B_j)$ and $(g_j = \min \Rightarrow f_j(I, S) \leq B_j)$

Goal: $\max f_i(I, S)$ if $g_i = \max$, $\min f_i(I, S)$ otherwise

3 PTAS for Multiobjective Disk Cover

In this section we construct polynomial-time approximation schemes for the multiobjective problems k-DC where $k \geq 1$. To keep the exposition simple, we concentrate on the 3-objective problem 2-DC and explain a polynomial-time algorithm that computes ϵ-approximate Pareto-solution sets for this problem. Our algorithm extends the shifting strategy introduced by Hochbaum and Maass

[HM85] to the multiobjective case. For this we need a combinatorial argument (Claim 3.3) showing that this strategy works for multiple objectives. Moreover, we use dynamic programming for efficiently combining the solutions of sub-problems. At the end of the section we discuss the possibility to extend our algorithms so that they work with the following additional features: interferences, different services for different types of customers, different shapes of supply areas, weighted customers, individual costs for base stations, and payoff for the quality of the obtained service. In particular, an appropriate modification provides a PTAS for k-EDC where $k \geq 1$.

We start with the description of the algorithm. Fix some shifting parameter $l \in \mathbb{N} \setminus \{0\}$. The larger l is, the better the approximation will be. The input to the algorithm are two finite sets of points $B, G \subseteq \mathbb{Z} \times \mathbb{Z}$ (blue and green points), a disk radius $r \in \mathbb{N}$ and a finite set of disk positions $D \subseteq \mathbb{Z} \times \mathbb{Z}$. For finite $P, S \subseteq \mathbb{Z} \times \mathbb{Z}$, where S is a valid solution (it respects the minimum distance constraint), define $c(P, S) \stackrel{df}{=} |\{p \in P \mid \exists x \in S, |p - x| \leq r\}|$ as the number of points from P covered by solution S.

In the algorithm, some functions $p\colon \mathbb{N} \times \mathbb{N} \to \mathcal{P}(D) \times \mathbb{N}$ with different indices will be defined. For a given number of disks and blue points to cover, such a function provides a partial solution for this sub-problem together with the number of green points covered in this solution. For any of these functions we address their components as $(p^{sol}(k,b), p^{val}(k,b)) \stackrel{df}{=} p(k,b)$ for $k, b \in \mathbb{N}$.

2-DC-APPROX(B,G,r,D):

1. let $\{\mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_\mathtt{m}\} \stackrel{df}{=} \{\mathtt{a} \in \mathtt{rl} \cdot (\mathbb{Z} \times \mathbb{Z}) \mid (\mathtt{B} \cup \mathtt{G} \cup \mathtt{D}) \cap (\mathtt{a} + [0, 2\mathtt{rl}]^2) \neq \emptyset\}$
2. for every $\mathtt{s} \in \mathtt{r} \cdot [0,\mathtt{l})^2$ do
3. for every $\mathtt{i} \in [1,\mathtt{m}]$ do
4. $\mathtt{D}_\mathtt{i} \stackrel{df}{=} \mathtt{D} \cap (\mathtt{s} + \mathtt{a}_\mathtt{i} + [\mathtt{r}, \mathtt{rl} - \mathtt{r})^2)$
5. $\mathtt{B}_\mathtt{i} \stackrel{df}{=} \mathtt{B} \cap (\mathtt{s} + \mathtt{a}_\mathtt{i} + [2\mathtt{r}, \mathtt{rl} - 2\mathtt{r})^2)$
6. $\mathtt{G}_\mathtt{i} \stackrel{df}{=} \mathtt{G} \cap (\mathtt{s} + \mathtt{a}_\mathtt{i} + [2\mathtt{r}, \mathtt{rl} - 2\mathtt{r})^2)$
7. for every $\mathtt{k} \in [0, |\mathtt{D}_\mathtt{i}|]$ and every $\mathtt{b} \in [0, |\mathtt{B}_\mathtt{i}|]$ do
8. $\mathtt{V}_{\mathtt{k},\mathtt{b}} \stackrel{df}{=} \{\mathtt{S} \subseteq \mathtt{D}_\mathtt{i} \mid \mathtt{S}$ is a valid solution, $|\mathtt{S}| \leq \mathtt{k}, \mathtt{c}(\mathtt{B}_\mathtt{i}, \mathtt{S}) \geq \mathtt{b}\}$
9. if $\mathtt{V}_{\mathtt{k},\mathtt{b}} = \emptyset$ then $\mathtt{p}_{\mathtt{s},\mathtt{i}}(\mathtt{k},\mathtt{b}) \stackrel{df}{=} (\perp, \perp)$
10. else $\mathtt{p}_{\mathtt{s},\mathtt{i}}(\mathtt{k},\mathtt{b}) \stackrel{df}{=} (\mathtt{S},\mathtt{b})$ for $\mathtt{S} \in \mathtt{V}_{\mathtt{k},\mathtt{b}}$ such that $\mathtt{b} = \mathtt{c}(\mathtt{B}_\mathtt{i}, \mathtt{S}) = \max\{\mathtt{c}(\mathtt{B}_\mathtt{i}, \mathtt{S}') \mid \mathtt{S}' \in \mathtt{V}_{\mathtt{k},\mathtt{b}}\}$
11. done
12. done
13. for every $\mathtt{k} \in [0, |\mathtt{D}|]$ and every $\mathtt{b} \in [0, |\mathtt{B}|]$ do
14. by dynamic programming choose $\mathtt{k}_1, \mathtt{k}_2, \ldots, \mathtt{k}_\mathtt{m}, \mathtt{b}_1, \mathtt{b}_2, \ldots, \mathtt{b}_\mathtt{m} \in \mathbb{N}$ such that $\sum_{\mathtt{i}=1}^{\mathtt{m}} \mathtt{p}^{\mathtt{val}}_{\mathtt{s},\mathtt{i}}(\mathtt{k}_\mathtt{i}, \mathtt{b}_\mathtt{i})$ is maximal, $\forall_{\mathtt{i} \in [1,\mathtt{m}]}\ \mathtt{p}^{\mathtt{val}}_{\mathtt{s},\mathtt{i}}(\mathtt{k}_\mathtt{i}, \mathtt{b}_\mathtt{i}) \neq \perp$, $\sum_{\mathtt{i}=1}^{\mathtt{m}} \mathtt{k}_\mathtt{i} \leq \mathtt{k}$, and $\sum_{\mathtt{i}=1}^{\mathtt{m}} \mathtt{b}_\mathtt{i} \geq \mathtt{b}$
15. if this succeeded, let $\mathtt{p}_\mathtt{s}(\mathtt{k},\mathtt{b}) \stackrel{df}{=} (\bigcup_{\mathtt{i}=1}^{\mathtt{m}} \mathtt{p}^{\mathtt{sol}}_{\mathtt{s},\mathtt{i}}(\mathtt{k}_\mathtt{i}, \mathtt{b}_\mathtt{i}), \sum_{\mathtt{i}=1}^{\mathtt{m}} \mathtt{p}^{\mathtt{val}}_{\mathtt{s},\mathtt{i}}(\mathtt{k}_\mathtt{i}, \mathtt{b}_\mathtt{i}))$
16. done
17. done
18. $\mathtt{P} \stackrel{df}{=} \{\mathtt{p}^{\mathtt{sol}}_\mathtt{s}(\mathtt{k},\mathtt{b}) \mid \mathtt{s} \in \mathtt{r} \cdot [0,\mathtt{l})^2, \mathtt{k}, \mathtt{b} \in \mathbb{N}\}$
19. remove all dominated solutions from P
20. return P

Explanation of the algorithm. First, we want to give an overview of the algorithm. The plane is divided into a grid of squares of side length rl. In each of these squares, the problem is solved independently (i.e., a small Pareto curve is calculated). By not considering the points at the border of width r of the squares, we obtain that an optimal solution needs at least as much disks as our calculated solution to cover the points in the square. Then, these solutions are combined. This is repeated for l^2 different positions (*shifts*) of the grid and the best solution is chosen.

The algorithm starts by partitioning the plane into squares of side length rl. Of course, there are infinitely many such squares, but many of them are empty and only some are of interest. The points $a_1, a_2, \ldots, a_m$ are the lower-left corner points of squares we need to consider. Because we will shift these squares later, we also have to include squares that contain a point for some, but possibly not all shifts, and thus we look for points in a square of side length $2rl$. These points a_i are all points such that there is at least one blue point, green point or one disk position in the square of side length $2rl$ which has a_i as its lower-left corner point.

Next, the algorithm loops over all l^2 (l in each dimension) shifts s of hop size r (the radius of a disk). In line 3, we loop over every index of the rl-grid points a_i which were found worth considering at the beginning.

In lines 4 to 6, we prepare a spatially restricted sub-problem of the general problem. The expression $[0, rl)^2$ denotes the set of points in a square of side length rl. Modified to $[r, rl - r)^2$ it denotes the set of points in such a square where a border of width r is removed from every edge. We only retain those points D_i from the set of disk positions D which lie in the restriced square that is positioned at the grid point a_i and shifted by s. In this way, disk positions from different sub-problems are guaranteed to have a minimal distance of ϱr (recall that $\varrho \in (0, 2]$). We also restrict the blue and green points, but here we use a larger border of width $2r$. The points on the $2r$-border are completely ignored in every sub-problem. By this method, as we will argue later, we get an optimal solution for each square restrictred in this way. We can combine the solutions of these sub-problems to obtain a global solution.

We now calculate the whole Pareto curve of this sub-problem starting in line 7. To this end, we loop over every possible number of disks k and covered blue points b and calculate the solution $S \subseteq D_i$ that maximizes the number of covered green points g using at most k disks in positions from D_i such that at least b blue points are also covered. This can be done by exhaustive search in polynomial time, as we will explain next. We first argue that there are only polynomially many valid solutions $|V_{k,b}|$. Because all disk positions in a valid solution $S \in V_{k,b}$ must have a mutual distance of at least ϱr, virtual circles of diameter ϱr around these positions can touch each other but must not overlap. The area covered by these virtual circles is $|S| \left(\frac{\varrho r}{2}\right)^2 \pi$. Since $\frac{\varrho r}{2} \leq r$ and $S \subseteq D_i$, these virtual circles are all located in a square of side length rl, and we get $|S| \left(\frac{\varrho r}{2}\right)^2 \pi \leq (rl)^2$. Solved for the number of disks we obtain $|S| \leq \frac{4l^2}{\pi \varrho^2} \stackrel{df}{=} c$, which is a constant. Since there are only polynomially many ways to choose at most c elements from the polynomially sized set D_i, we see that $|V_{k,b}|$ is polynomial in the input size. Since c can be calculated effectively, $V_{k,b}$ can be searched exhaustively for a solution that maximizes the number of covered green points in polynomial time. If such a solution does not exist (because $V_{k,b} = \emptyset$) then both components of $p_{s,i}(k, b)$ are set to a special undefined value $\perp$, which we will need later.

After the i-loop, we have small Pareto curves $p_{s,i}$ for each sub-problem given by shift s and point

a_i. In the loop starting in line 13, we combine them into a larger Pareto curve p_s for the current shift s. To this end, we try to find a solution that maximizes the number of covered green points for a given number of disks k and blue points b using the solutions $p_{s,i}$ in line 14. We distribute the k disks and b blue points over all squares in any possible way. The number of disks available to square i is called k_i and the number of blue points that must be covered in square i is called b_i. The distribution that maximizes the total number of covered green points is chosen and the combination of the individual solutions $p_{s,i}(k_i, b_i)$ for this distribution is stored in $p_s(k, b)$.

In general, there are exponentially many ways to distribute the numbers k and b, but the search for an optimal distribution can be done efficiently by dynamic programming, as can be seen from the following algorithm. We need to consider the iterations of the k, b-loop starting in line 15 of the 2-DC-APPROX-algorithm all at once, so the following code can be used as a replacement of the lines 13–16 of the 2-DC-APPROX-algorithm. The return value of CombinePartialSolutions is the function p_s.

CombinePartialSolutions($\mathtt{p}_{\mathtt{s},1}, \mathtt{p}_{\mathtt{s},2}, \ldots, \mathtt{p}_{\mathtt{s},\mathtt{m}}$):

1. $\mathtt{p}'_1 \stackrel{df}{=} \mathtt{p}_{\mathtt{s},1}$
2. for $\mathtt{t} := 2$ to m do
3. for every $\mathtt{k} \in [0, |\mathtt{D}|]$ and every $\mathtt{b} \in [0, |\mathtt{B}|]$ do
4. find maximal $\mathtt{p}'^{\mathtt{val}}_{\mathtt{t}}(\mathtt{k}, \mathtt{b}) \stackrel{df}{=} \mathtt{p}'^{\mathtt{val}}_{\mathtt{t}-1}(\bar{\mathtt{k}}_1, \bar{\mathtt{b}}_1) + \mathtt{p}^{\mathtt{val}}_{\mathtt{s},\mathtt{t}}(\bar{\mathtt{k}}_2, \bar{\mathtt{b}}_2)$ for $\bar{\mathtt{k}}_1, \bar{\mathtt{b}}_1, \bar{\mathtt{k}}_2, \bar{\mathtt{b}}_2 \in \mathbb{N}$ such that $\bar{\mathtt{k}}_1 + \bar{\mathtt{k}}_2 \leq \mathtt{k}$, $\bar{\mathtt{b}}_1 + \bar{\mathtt{b}}_2 \geq \mathtt{b}$, $\mathtt{p}'^{\mathtt{val}}_{\mathtt{t}-1}(\bar{\mathtt{k}}_1, \bar{\mathtt{b}}_1) \neq \perp$ and $\mathtt{p}^{\mathtt{val}}_{\mathtt{s},\mathtt{t}}(\bar{\mathtt{k}}_2, \bar{\mathtt{b}}_2)) \neq \perp$
5. if this is not possible then let $\mathtt{p}'_{\mathtt{t}}(\mathtt{k}, \mathtt{b}) \stackrel{df}{=} \perp$
6. else let $\mathtt{p}'^{\mathtt{sol}}_{\mathtt{t}}(\mathtt{k}, \mathtt{b}) \stackrel{df}{=} \mathtt{p}'^{\mathtt{sol}}_{\mathtt{t}-1}(\bar{\mathtt{k}}_1, \bar{\mathtt{b}}_1) \cup \mathtt{p}^{\mathtt{sol}}_{\mathtt{s},\mathtt{t}}(\bar{\mathtt{k}}_2, \bar{\mathtt{b}}_2)$
7. done
8. done
9. return $\mathtt{p}'_{\mathtt{m}}$

In every iteration of the t-loop of CombinePartialSolutions, another square is incorporated into the Pareto curve, each time solving some kind of knapsack problem. Since there are only polynomially many combinations of $\bar{k}_1, \bar{b}_1, \bar{k}_2, \bar{b}_2$ such that the constraints in line 4 are met, p'_t can be computed in polynomial time. The correctness of this dynamic programming method follows by induction. Since m is polynomial in the input length, the computation of CombinePartialSolutions and thus also the computation of lines 13–16 in 2-DC-APPROX can be done in polynomial time.

Back at the 2-DC-APPROX-algorithm, we have an approximate Pareto curve p_s for every of the shift values after the end of the second loop over s. In Line 18, we simply put all the previously obtained solutions $p_s^{sol}(k, b)$ for all s, k, b in one set P, remove the dominated solutions in line 19 and return that set in line 20.

Correctness of the algorithm. We now argue for the correctness of the algorithm by showing that for fixed l it runs in polynomial time and that the relative error becomes arbitrarily small if l is increased.

Lemma 3.1 *For every fixed $l \geq 5$, the algorithm 2-DC-APPROX works in polynomial time.*

Proof As we have already argued in the explanation of the algorithm, every loop has polynomially many iterations. Furthermore, the more complicated subroutines in line 10 and line 14 can be computed in polynomial time by exhaustive search and dynamic programming as mentioned in the explanations. □

We argue that by choosing l large enough, the algorithm 2-DC-APPROX has an arbitrarily small relative error.

Lemma 3.2 *Fix an $l \geq 5$ and let $\epsilon \stackrel{df}{=} (0, \frac{16}{l}, \frac{16}{l})$. On input of a 2-DC instance $I = (B, G, r, D)$ the algorithm 2-DC-APPROX computes an ϵ-approximate Pareto-solution set P for I.*

Proof Choose some Pareto solution $S' \in P^{sol}(I)$ and let P be the set that is returned by 2-DC-APPROX(I). We show that $P \subseteq \mathcal{S}(I)$ and there exists an $S \in P$ that ϵ-covers S'.

Let C_b (resp., C_g) denote the blue (resp., green) points covered by the solution S'. Moreover, for a shift $s \in r \cdot [0, l)^2$ let $\beta(s)$ denote the set of points on the border-strips of s, i.e.,

$$\beta(s) = s + [-2r, 2r)^2 + \big((rl\mathbb{Z} \times \mathbb{Z}) \cup (\mathbb{Z} \times rl\mathbb{Z})\big).$$

We now use a combinatorial argument to show that there exists a shift s such that only a small fraction of the points in C_b and C_g are located on the border-strips of s.

Claim 3.3 *There is an $s \in r \cdot [0, l)^2$ such that $|C_b \cap \beta(s)| \leq \frac{16}{l}|C_b|$ and $|C_g \cap \beta(s)| \leq \frac{16}{l}|C_g|$.*

Proof Assume that for every $s \in r \cdot [0, l)^2$ it holds that $|C_b \cap \beta(s)| > \frac{16}{l}|C_b|$ or $|C_g \cap \beta(s)| > \frac{16}{l}|C_g|$. Without loss of generality,

$$|C_b \cap \beta(s)| > \tfrac{16}{l}|C_b| \text{ holds for at least one half of all shifts } s \text{ (i.e., for } \tfrac{l^2}{2} \text{ shifts).} \qquad (*)$$

We consider the number of events where a blue point is located on the border of some shift. This number can be expressed either as a sum over all points or as a sum over all shifts. This yields the following equation.

$$\sum_{x \in C_b} |\{s \in r \cdot [0, l)^2 \mid x \in \beta(s)\}| = \sum_{s \in r \cdot [0,l)^2} |C_b \cap \beta(s)|$$

For each point there exist exactly $8l - 16$ shifts $s \in r \cdot [0, l)^2$ such that the point is on the border $\beta(s)$ (since the squares are of size $rl \times rl$ and have a border of width $2r$ which consists of exactly $8l - 16$ small squares of size $r \times r$). So the left-hand side of the equation equals $|C_b| \cdot (8l - 16)$. However, by $(*)$, the right-hand side is greater than

$$\frac{l^2}{2} \cdot \frac{16}{l}|C_b| \geq |C_b| \cdot (8l - 16)$$

which is a contradiction. This proves the claim. □

We consider a run of 2-DC-APPROX on input I. In the loop 2–17, let us choose s according to Claim 3.3. Moreover, choose an arbitrary $i \in [1, m]$ in the loop 3–16 and define D_i, B_i, and G_i according to the lines 4–6. Let S_i be the set of disks in S' whose centers are in the shifted square $a_i + [r, rl - r)^2$, but not on the border-strip of width r, i.e., $S_i \stackrel{df}{=} S' \cap (s + a_i + [r, rl - r)^2)$. If the center of a disk is on the border-strip of width r of some square, then this disk cannot cover points that are in the square, but not on the border-strip of width $2r$. This shows

$$c(G_i, S_i) = c(G_i, S') \quad \text{and} \quad c(B_i, S_i) = c(B_i, S'). \tag{1}$$

Let $k_i \stackrel{df}{=} |S_i|$ and $b_i \stackrel{df}{=} c(B_i, S_i)$. Choose $k = k_i$ and $b = b_i$ in line 7. Observe that S_i belongs to the set $V_{k,b}$ that is defined in line 8. Hence, after line 10, $p^{\text{sol}}_{s,i}(k_i, b_i) \subseteq D_i$,

$$|p^{\text{sol}}_{s,i}(k_i, b_i)| \leq k_i, \tag{2}$$

$$c(R_i, p^{\text{sol}}_{s,i}(k_i, b_i)) \geq b_i, \text{ and} \tag{3}$$

$$p^{\text{val}}_{s,i}(k_i, b_i) \geq c(B_i, S_i). \tag{4}$$

Estimation (4) holds, since the solution $S_i \in V_{k_i,b_i}$ already covers $c(G_i, S_i)$ green points. So with the chosen k_i and b_i we obtain

$$\sum_{i=1}^{m} p^{\text{val}}_{s,i}(k_i, b_i) \stackrel{(4)}{\geq} \sum_{i=1}^{m} c(G_i, S_i) \stackrel{(1)}{=} c(G - \beta(s), S') = |C_g| - |C_g \cap \beta(s)| \stackrel{3.3}{>} (1 - \frac{16}{l})|C_g|. \tag{5}$$

Moreover,

$$\sum_{i=1}^{m} b_i = \sum_{i=1}^{m} c(B_i, S_i) \stackrel{(1)}{=} c(B - \beta(s), S') = |C_b| - |C_b \cap \beta(s)| \stackrel{3.3}{>} (1 - \frac{16}{l})|C_b|. \tag{6}$$

In the loop 13–16 choose $b \stackrel{df}{=} \sum_{i=1}^{m} b_i$ and $k \stackrel{df}{=} \sum_{i=1}^{m} k_i$. By (2) and (3), after line 15 it holds that $p_s(k, b) \neq (\perp, \perp)$ (i.e., the conditions in line 14 can be satisfied) and hence $p_s(k, b) = (S, g)$ for a suitable $S \subseteq D$. Observe that

$$|S| \leq k \leq |S'| \quad \text{(because of the conditions in line 14)}, \tag{7}$$

$$(1 - \frac{16}{l})|C_g| \stackrel{(5)}{<} \sum_{i=1}^{m} p^{\text{val}}_{s,i}(k_i, b_i) \leq g \stackrel{(!)}{=} c(G - \beta(s), S) \leq c(G, S), \text{ and} \tag{8}$$

$$(1 - \frac{16}{l})|C_b| \stackrel{(6)}{<} \sum_{i=1}^{m} b_i = b \stackrel{(!)}{\leq} c(B - \beta(s), S) \leq c(B, S). \tag{9}$$

In (8) and (9), the equalities marked with (!) hold, since the points in B_i and G_i are sufficiently far away from other squares so that they are only influenced by disks in a_i, but not by disks in other squares. From (7), (8), and (9) it follows that the solution S ϵ-covers our given Pareto solution S'. By the definition of P we have $S \in P$ after line 18. So after line 19 there still exists some $\tilde{S} \in P$ that ϵ-covers S'.

It remains to show that $P \subseteq \mathcal{S}(I)$, i.e., all returned solutions are valid. For this, note that after line 8 it holds that $V_{k,b} \subseteq \mathcal{S}(I)$. So after line 10, either $p^{\text{sol}}_{s,i}(k, b) = \perp$ or $p^{\text{sol}}_{s,i}(k, b) \in \mathcal{S}(I) \cap D_i$.

After line 15, $p_s^{\mathrm{sol}}(k,b) = \bigcup_{i=1}^{m} p_{s,i}^{sol}(k_i,b_i)$ where the $p_{s,i}^{sol}(k_i,b_i) \in \mathcal{S}(I) \cap D_i$. Disks in D_i are not on the border-strips of width r of the corresponding square. So for $i \neq j$ we have a minimum distance of $2r \geq \varrho \cdot r$ between disks in D_i and disks in D_j. Therefore, after line 15, $p_s^{\mathrm{sol}}(k,b) \in \mathcal{S}(I)$ (this is the point where we need the border-strips of width r for disks). So after line 18 it holds that $P \subseteq \mathcal{S}(I)$. □

Theorem 3.4 *Fix $k \geq 1$ and $\varrho \in (0,2]$. For all $\delta > 0$, k-DC is $(0,\delta,\ldots,\delta)$-approximable in polynomial time (and hence has a PTAS).*

Proof By the Lemmas 3.1 and 3.2, 2-DC is $(0,\delta,\delta)$-approximable in polynomial time for every $\delta > 0$. These lemmas and the underlying algorithm 2-DC-APPROX can be extended in a straightforward way to k types of points where $k \geq 1$. □

The multiobjective shifting strategy used in 2-DC-APPROX is a very general method that can be applied to several other multiobjective covering problems. It is easy to see that the algorithm 2-DC-APPROX can be adapted such that it takes interferences into account.

Theorem 3.5 *Fix $k \geq 1$ and $\varrho \in (0,2]$. For all $\delta > 0$, k-EDC is $(0,\delta,\ldots,\delta)$-approximable in polynomial time (and hence has a PTAS).*

Proof The proof is very similar to the proof of Theorem 3.4. One only has to observe that because of the borders, disks in one square cannot interfere with points in other squares. So also for k-EDC it holds that if disks on the borders are neglected, then the single squares can be optimized independently. □

Besides interferences, also other parameters can be added to the problem. For instance it might be the case that the single services have different operating distances. This brings us to the version of 2-DC where we have to place simultaneously two disks of different radii on the selected locations (one disk for each type of customers). 2-DC-APPROX can be easily adapted such that it gives a PTAS also for this variant of the problem. In general, we can allow even more complicated rules that determine whether or not a customer is supplied by a base station. Here the different services can have supply areas of very general shape as long as

- we can efficiently test whether a point belongs to such an area and
- the minimum distance constraint is satisfied (i.e., the distance of two base stations is at least $\varrho \cdot r$ where $2r$ is the maximal diameter of the area and ϱ is a fixed constant).

Further generalization could handle weights for the customers, individual costs for the base stations, and payoffs that depend on the quality of the service obtained by the single customers. For these scenarios and their combinations, appropriate versions of 2-DC-APPROX provide polynomial-time approximation schemes.

4 Non-Approximability of the Restricted Version

The approximability of a multiobjective problem does not necessarily imply that the single-objective restrictions of this problem are approximable. The reason for this apparent contradiction is that all solutions for a restricted version of the problem must *strictly* satisfy the additional constraints on the values of the objectives that are not optimized any more (i.e., the constraints $f_j(I,S) \geq B_j$ or $f_j(I,S) \leq B_j$ in the definition of the restricted problem). An approximation algorithm has more freedom if it can optimize all objectives at the same time, since here the algorithm can *exploit trade-offs* between the single objectives. In fact, the problems k-DC and k-EDC are examples where such trade-offs yield a significantly better approximability. In this section we will show that several restrictions of k-DC and k-EDC are not approximable within a constant factor, unless P = NP. For instance, for $k \geq 2$, no restriction of k-EDC is in APX (unless P = NP), while the general (multiobjective) version of k-EDC even admits a PTAS. Angel, Bampis, and Kononov [ABK01, ABK03], Cheng, Janiak, and Kovalyov [CJK98], and Dongarra et al. [DJSS07] discovered similar phenomena for multiobjective scheduling problems.

For our results in this section we need the NP-completeness of the following versions of geometric disk cover problems.

Disk Cover

$$\mathrm{DC} = \{(P,D,k) \,\big|\, P,D \subseteq \mathbb{Z}\times\mathbb{Z} \text{ are finite sets, } k \in \mathbb{N}, \text{ and there exists an } S \subseteq D \text{ such that } |S| \leq k \text{ and } \forall x \in P\, \exists y \in S, |x-y| \leq 2\}$$

Exact Disk Cover

$$\mathrm{EDC} = \{(P,D) \,\big|\, P,D \subseteq \mathbb{Z}\times\mathbb{Z} \text{ are finite sets and there exists an } S \subseteq D \text{ such that } \forall x \in P\, \exists! y \in S, |x-y| \leq 2\}$$

Note that there is a minimum-distance constraint for disk locations implicitly given in these definitions since we consider points from $\mathbb{Z}\times\mathbb{Z}$ and a fixed radius $r = 2$.

Theorem 4.1 DC *and* EDC *are* NP*-complete.*

Proof It is easy to see that DC and EDC belong to NP. So it suffices to show NP-hardness. For this we follow Fowler, Paterson, and Tanimoto [FPT81] who described a reduction of 3-SAT to the following disk cover problem.

$$\mathrm{DC}' = \{(P,k) \,\big|\, P \subseteq \mathbb{Z}\times\mathbb{Z} \text{ is a finite set, } k \in \mathbb{N}, \text{ and there exists an } S \subseteq \mathbb{Z}\times\mathbb{Z} \text{ such that } |S| \leq k \text{ and } \forall x \in P\, \exists y \in S, |x-y| \leq 2\}$$

We have to adapt the technique by Fowler, Paterson, and Tanimoto so that it takes the following three differences between DC′ and DC/EDC into account.

1. DC/EDC-instances contain a set D of possible disk positions, while in the case of DC′, the set S can contain arbitrary positions.

2. In solutions of DC′-instances, a point can be covered by more than one disk. In the case of EDC, each point must be covered by *exactly one* disk.

3. In DC′-instances the cardinality of S is bounded by the parameter k, while in EDC-instances S can be of arbitrary size.

We describe a polynomial-time many-one reduction from 3-SAT to DC. Let the input be a 3-CNF formula with m variables $x_1, \ldots, x_m$ and n clauses $C_1, \ldots, C_n$. The reduction will output a DC-instance (P, D, k). Figure 1 shows the high-level structure of the sets P and D, i.e., the approximate places in the plane where the points from P and D are located. The structure consists of m closed lines (*wires*) and of n bold points (*clause points*). Wires represent points and possible disk positions that are arranged such that two successive points can be covered by one disk and each disk can cover no more than two points. Each wire corresponds to one variable and each clause point corresponds to one clause in the 3-CNF formula. In addition we have to make sure that any pair of clause points has a distance that is divisible by 4 (by the structure of the regions $R_1 - R_5$, only even distances are possible at all, but a distance $\equiv 2 (\mathrm{mod}\ 4)$ would introduce an unwanted negation in a clause). This finishes the description of the reduction.

First let us analyze the parity of the number of points at certain sections of the wires. Each of the three open loops in region R_5 contains an odd number of points (more precisely, 21, 9, or 19 points). Points in the region R_4 belong to the same wire and their number is even. The points at the two vertical (resp., horizontal) lines in region R_3 belong to the same wire and their number is even. It follows that in region R_2, the loop that goes to the right contains an odd number of points. Since we are only interested in the parity, we can think of replacing this loop by the single point that is missing in the vertical line at the right-hand side. Hence, loops that lead to clause points do not change the parity of points at a wire. Together with the structure of the regions R_1 this shows that wires contain an even number of points.

There are always one or more disk locations close to neighbouring points at a wire. We say that a disk location "belongs to the gap between two points" if it has a distance ≤ 2 to each of the points. So in order to cover all points at a wire we have to place a disk at every second gap (note that this holds in particular for the crossovers R_3). There are two ways to do this: Either we place disks at every even gap or at every odd gap. These two possibilities correspond to the assignment of the variable with either 0 or 1. If some variable x_i appears negatively in some clause C_j, then in the corresponding region R_4 we drop one gap at each of the lines which negates the assignment of x_i in the region R_5. Each of the open loops in region R_5 has one gap close to the clause point. The clause point can be covered without using an extra disk if and only if we have placed a disk in at least one of these gaps. This is equivalent to saying that the clause C_j is satisfied by the assignment represented by the current choice of disks. Hence, the given formula is satisfiable if and only if we can cover all points without using extra disks.

We are left with computing the minimal number k of disks that is needed to cover all points. Note that $|P|$ be the total number of points in Figure 1. By placing one disk at every second gap, we cover exactly two points per wire with one disk. We have seen that if the formula is satisfiable, then we do not need extra disks for the clause points. Moreover, at crossovers 4 disks are used simultaneously for two wires. Therefore, if l is the number if crossovers (i.e., the number of regions

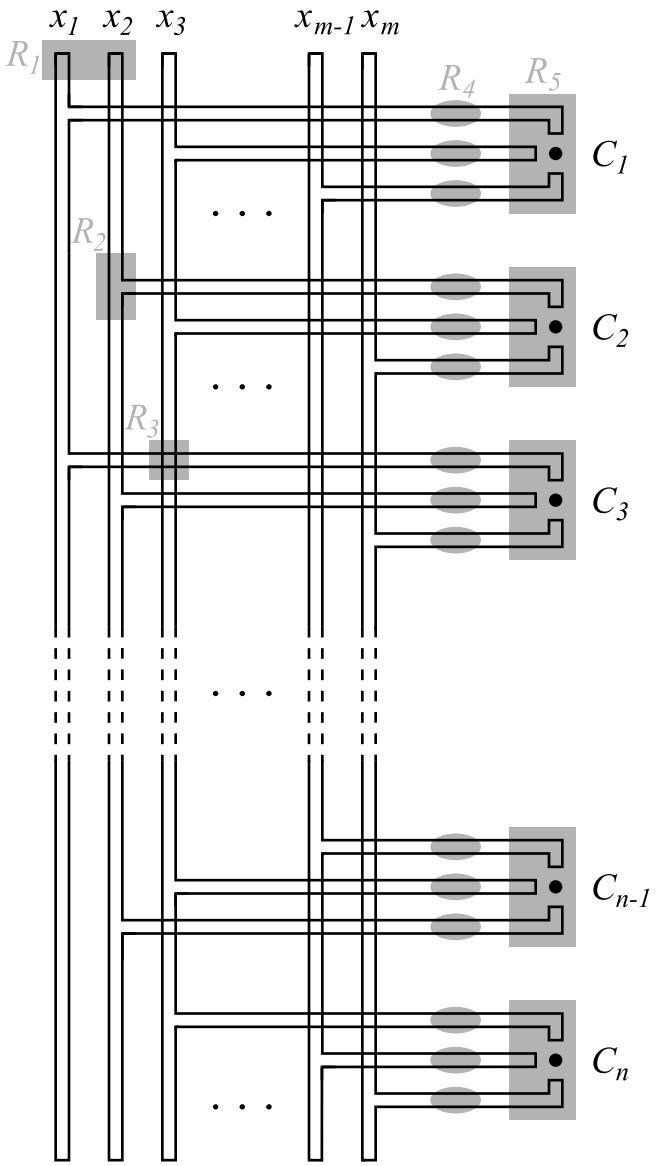


Figure 1: The high-level structure of the sets P and D that are computed by the reduction. The detailed structure of the gray regions $R_1 - R_5$ is shown in the Figures 2–6 respectively. Each closed line represents a variable and each region R_5 represents a clause of three literals. The elliptic regions R_4 determine whether a variable itself or its negation contributes to the clause.

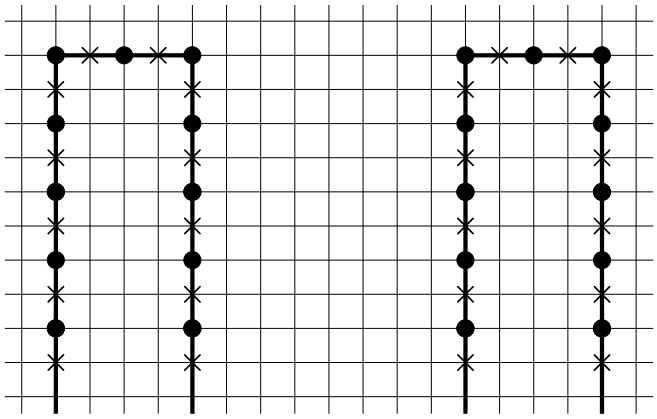


Figure 2: The detailed structure at the upper and lower ends (gray region R_1 in Figure 1). Circles represent points from P and crosses represent disk locations from D.

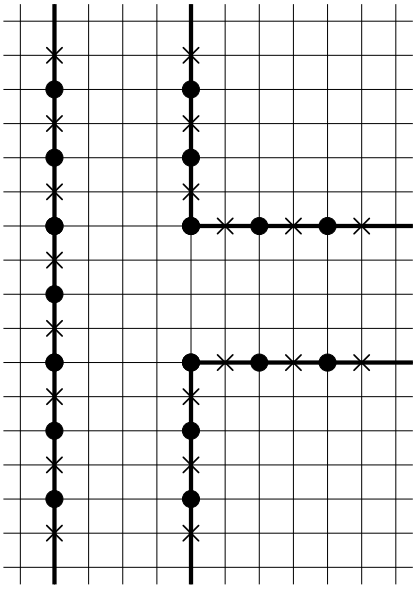


Figure 3: The detailed structure at T-junctions (gray region R_2 in Figure 1). Circles represent points from P and crosses represent disk locations from D.

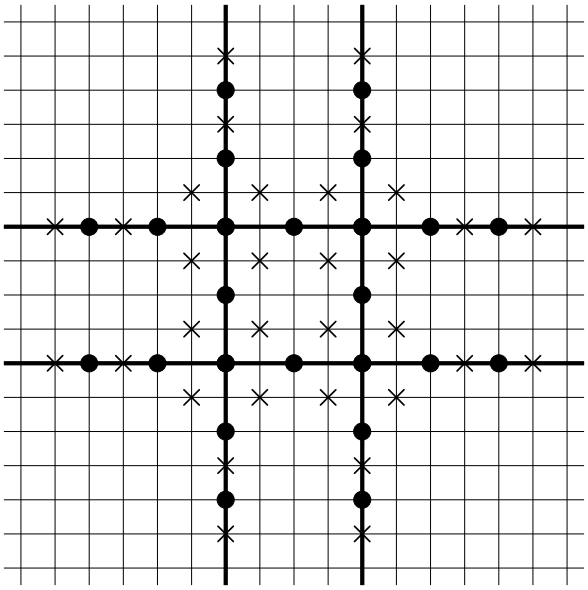
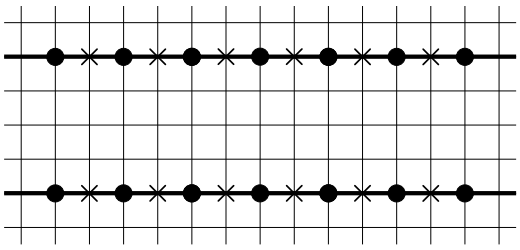
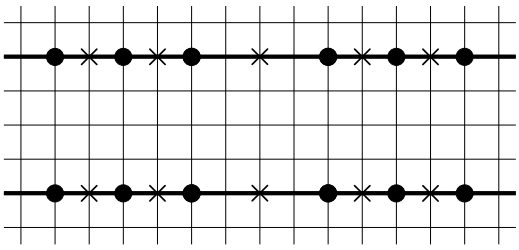


Figure 4: The detailed structure at crossovers (gray region R_3 in Figure 1). Circles represent points from P and crosses represent disk locations from D.



If x_i appears in C_j.



If $\overline{x_i}$ appears in C_j.

Figure 5: The detailed structure of the elliptic regions R_4 in Figure 1. This structure depends on whether the corresponding variable x_i appears positively or negatively in the corresponding clause C_j. Circles represent points from P and crosses represent disk locations from D.

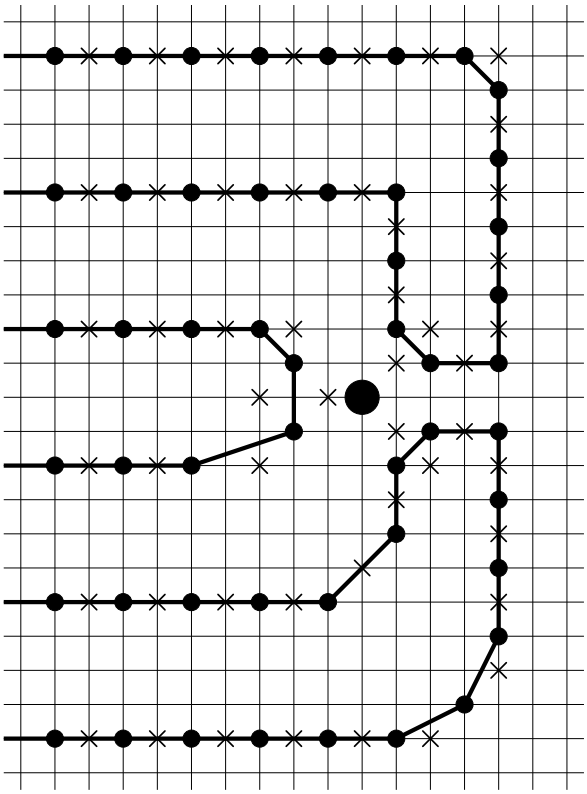


Figure 6: The detailed structure of the gray regions R_5 in Figure 1. Circles represent points from P and crosses represent disk locations from D. These regions simulate the clauses C_j. If a given assignment of the variables satisfies C_j, then the clause point (the bold point in the picture) can be covered without using an extra disk. Hence if all clauses are satisfied, then this saves exactly n disks.

R_3), then we obtain

$$k = \frac{|P| - n - 4l}{2}.$$

So the given formula is satisfiable if and only if $(P, D, k) \in \mathrm{DC}$.

The whole construction (Figure 1) is bounded to a region of size $O(n) \times O(m)$ and can be carried out in polynomial time. This shows 3-SAT$\leq^{\mathrm{p}}_{\mathrm{m}}$DC.

Finally we argue that the same reduction shows 3-SAT$\leq^{\mathrm{p}}_{\mathrm{m}}$EDC. For this, let (P, D, k) be the DC-instance computed by the reduction above. If $(P, D) \in \mathrm{EDC}$, then this must be witnessed by a collection of at most k disks, since otherwise at least one point is covered by two disks. Hence $(P, D, k) \in \mathrm{DC}$. If $(P, D, k) \in \mathrm{DC}$, then this is witnessed by placing a disk at every second gap. This ensures that each point is covered by exactly one disk. For clause points, the exact cover can be obtained, because the gaps close to the clause point have two disk locations, one that covers the clause point and one that does not. This shows 3-SAT$\leq^{\mathrm{p}}_{\mathrm{m}}$EDC. □

As we have seen in Section 3, the minimum-distance constraint helps us to find a PTAS for k-DC. Nevertheless, the problem remains difficult. Although the Pareto curve is only polynomial in size, we cannot hope to discover an algorithm that computes it exactly: For $\varrho \leq \frac{1}{2}$, an algorithm that exactly determines the Pareto curve for k-DC$_\varrho$ in polynomial time would also solve the problem DC in polynomial time. Since DC is NP-complete, this would imply P = NP.

By Theorem 3.5, the multiobjective problem k-EDC has good approximation properties (a PTAS). Now we will see (Theorem 4.2) that in contrast, the restricted versions of k-EDC vary with respect to their approximation behavior. While 1-EDC restricted to the first component is not approximable (i.e., not in APX unless P = NP), the restriction to the second component has good approximation properties (a PTAS). Even more surprising, for $k \geq 2$ the Pareto curve of k-EDC is approximable, but no restriction of k-EDC is approximable (i.e., not in APX unless P = NP). Theorem 4.5 states similar results for k-DC. For instance, while the Pareto curve of 2-EDC is approximable, the restriction of 2-EDC to the second component is not (i.e., not in APX unless P = NP).

Theorem 4.2 *Fix some $\varrho \in (0, \frac{1}{2}]$.*

1. *If* P $\neq$ NP*, then for all $k \geq 2$ and all $i \in [1, k+1]$, Restricted$_i$-k-EDC is not in* APX*.*
2. *If* P $\neq$ NP*, then Restricted$_1$-1-EDC is not in* APX*.*
3. *Restricted$_2$-1-EDC admits a PTAS.*

Proof The theorem is proved by two claims.

Claim 4.3 *If* P $\neq$ NP*, then for all $k \geq 1$, Restricted$_1$-k-EDC is not in* APX*.*

Proof Assume there exist a $\delta > 0$ and a polynomial-time algorithm $\mathcal{A}$ that δ-approximates Restricted$_1$-k-EDC. We describe a polynomial-time decision algorithm for EDC on input (P', D').

1. Simulate $\mathcal{A}$ on $(I, B_1, \ldots, B_k)$ where $B_j \stackrel{df}{=} |P'|$ for $j \in [1,k]$ and I is the following k-EDC instance: $P_j \stackrel{df}{=} P'$ for $j \in [1,k]$, $r \stackrel{df}{=} 2$, $D \stackrel{df}{=} D'$.
2. If the simulation outputs some $S \subseteq D$ such that $\forall x \in P' \, \exists! y \in S, |x-y| \leq 2$, then accept, otherwise reject.

If $(P', D') \in$ EDC, then there exists an $S' \subseteq D'$ such that $\forall x \in P' \, \exists! y \in S', |x-y| \leq 2$. For different $x, y \in S'$ we have $|x-y| \geq 1 \geq \varrho \cdot r$, since $S' \subseteq \mathbb{Z} \times \mathbb{Z}$. So S' is a valid solution for I. Hence $\mathcal{A}$ on I must output some $S \subseteq D$ such that $|S| \leq (1+\delta) \cdot |S'|$ and $|C_j| \geq B_j$ for $j \in [1,k]$ where $C_j = \{x \in P_j \mid \exists! y \in S, |x-y| \leq r\}$. In particular, $B_1 = |P'|$ and $C_1 = P'$. So $\forall x \in P' \, \exists! y \in S, |x-y| \leq 2$ and hence the algorithm accepts.

If $(P', D') \notin$ EDC, then there is no $S \subseteq D'$ such that $\forall x \in P' \, \exists! y \in S, |x-y| \leq 2$. Hence our algorithm rejects. This shows EDC $\in$ P and hence, by Theorem 4.1, P = NP. This proves Claim 4.3. □

Claim 4.4 *If* P $\neq$ NP*, then for all* $k \geq 2$ *and* $i \in [2, k+1]$*, Restricted*$_i$*-*k*-EDC is not in* APX*.*

Proof The proof is similar to the one of Claim 4.3. We only have to use the following adapted algorithm.

1. Simulate $\mathcal{A}$ on $(I, B_1, \ldots, B_k)$ where $B_1 = |D'|$ and $B_j \stackrel{df}{=} |P'|$ for $j \in [2,k]$ and I is the following k-EDC instance: $P_j \stackrel{df}{=} P'$ for $j \in [1,k]$, $r \stackrel{df}{=} 2$, $D \stackrel{df}{=} D'$.
2. If the simulation outputs some $S \subseteq D$ such that $\forall x \in P' \, \exists! y \in S, |x-y| \leq 2$, then accept, otherwise reject.

This proves Claim 4.4. □

The statements 1 and 2 of the theorem follow from the Claims 4.3 and 4.4. The third statement holds, since by Theorem 3.5, 1-EDC is $(0, \delta)$-approximable in polynomial time for every $\delta > 0$. This finishes the proof of Theorem 4.2. □

Theorem 4.5 *Fix some* $\varrho \in (0, \frac{1}{2}]$*.*

1. *Restricted*$_2$*-1-DC has a PTAS.*
2. *If* P $\neq$ NP*, then for all* $k \geq 2$ *and all* $i \in [2, k+1]$*, Restricted*$_i$*-*k*-DC is not in* APX*.*

Proof By Theorem 3.4, 1-DC is $(0, \delta)$-approximable in polynomial time for every $\delta > 0$. Hence, Restricted$_2$-1-DC has a PTAS which shows the first statement.

The proof of the second statement is similar to the one of Claim 4.3. Let $k \geq 2$ and $i \in [2, k+1]$, and assume there exist a $\delta > 0$ and a polynomial-time algorithm $\mathcal{A}$ that δ-approximates Restricted$_i$-k-DC. We describe a polynomial-time decision algorithm for DC on input (P', D', k').

1. Simulate $\mathcal{A}$ on $(I, B_1, \ldots, B_k)$ where $B_1 = k'$ and $B_j \stackrel{df}{=} |P'|$ for $j \in [1,k]$ and I is the following k-DC instance: $P_j \stackrel{df}{=} P'$ for $j \in [1,k]$, $r \stackrel{df}{=} 2$, $D \stackrel{df}{=} D'$.

2. If the simulation outputs some $S \subseteq D$ such that $|S| \leq k'$ and $\forall x \in P' \, \exists y \in S, |x-y| \leq 2$, then accept, otherwise reject.

If $(P', D', k') \in$ DC, then there exists an $S' \subseteq D'$ such that $|S'| \leq k'$ and $\forall x \in P' \, \exists y \in S', |x-y| \leq 2$. For different $x, y \in S'$ we have $|x-y| \geq 1 \geq \varrho \cdot r$, since $S' \subseteq \mathbb{Z} \times \mathbb{Z}$. So S' is a valid solution for I. Hence $\mathcal{A}$ on I must output some $S \subseteq D$ such that $(1+\delta) \cdot |C_i| \geq |P'|$, $|S| \leq B_1 = k'$, $|C_j| \geq B_{j+1}$ for $j \in [1, i-1]$, and $|C_j| \geq B_j$ for $j \in [i+1, k]$ where $C_j = \{x \in P_j \mid \exists y \in S, |x-y| \leq r\}$. So for $j \in [1,k] - \{i\}$ it holds that $C_j = P'$. So $\forall x \in P' \, \exists y \in S, |x-y| \leq 2$ and hence the algorithm accepts.

If $(P', D', k') \notin$ DC, then there is no $S \subseteq D'$ such that $\forall x \in P' \, \exists y \in S, |x-y| \leq 2$. Hence our algorithm rejects. This shows DC $\in$ P and hence, by Theorem 4.1, P = NP. □

References

[ABK01] E. Angel, E. Bampis, and A. Kononov. A FPTAS for approximating the unrelated parallel machines scheduling problem with costs. In *Proceedings 9th Annual European Symposium on Algorithms*, volume 2161 of *Lecture Notes in Computer Science*, pages 194–205. Springer, 2001.

[ABK03] E. Angel, E. Bampis, and A. Kononov. On the approximate tradeoff for bicriteria batching and parallel machine scheduling problems. *Theoretical Computer Science*, 306(1-3):319–338, 2003.

[CCJ90] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990.

[CJK98] T. C. E. Cheng, A. Janiak, and M. Y. Kovalyov. Bicriterion single machine scheduling with resource dependent processing times. *SIAM Journal on Optimization*, 8(2):617–630, 1998.

[CMWZ04] Gruia Calinescu, Ion I. Mandoiu, Peng-Jun Wan, and Alexander Zelikovsky. Selecting forwarding neighbors in wireless ad hoc networks. *MONET*, 9(2):101–111, 2004.

[DJSS07] J. Dongarra, E. Jeannot, E. Saule, and Z. Shi. Bi-objective scheduling algorithms for optimizing makespan and reliability on heterogeneous systems. In *Proceedings 19th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 280–288. ACM, 2007.

[DY07] I. Diakonikolas and M. Yannakakis. Small approximate pareto sets for bi-objective shortest paths and other problems. In *Proceedings 10th International Workshop on Approximation, Randomization, and Combinatorial Optimization*, volume 4627 of *Lecture Notes in Computer Science*, pages 74–88. Springer, 2007.

[Ehr05] M. Ehrgott. *Multicriteria Optimization*. Springer Verlag, 2005.

[FPT81] R. J. Fowler, M. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters*, 12(3):133–137, 1981.

[GRV05] C. Glaßer, S. Reith, and H. Vollmer. The complexity of base station positioning in cellular networks. *Discrete Applied Mathematics*, 148(1):1–12, 2005.

[HM85] D. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32:130–136, 1985.

[NV06] Sada Narayanappa and Petr Vojtechovsky. An improved approximation factor for the unit disk covering problem. In *CCCG*, 2006.

[PY00] C. H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *FOCS*, pages 86–92, 2000.

[SO95a] H. M. Safer and J. B. Orlin. Fast approximation schemes for multi-criteria combinatorial optimization. Working papers 3756-95, Massachusetts Institute of Technology, Sloan School of Management, 1995.

[SO95b] H. M. Safer and J. B. Orlin. Fast approximation schemes for multi-criteria flow, knapsack, and scheduling problems. Working papers 3757-95, Massachusetts Institute of Technology, Sloan School of Management, 1995.

[VY05] S. Vassilvitskii and M. Yannakakis. Efficiently computing succinct trade-off curves. *Theoretical Computer Science*, 348(2-3):334–356, 2005.