(Re)packing Equal Disks into Rectangle

Fedor V. Fomin
Department of Informatics, University of Bergen, Norway

Petr A. Golovach
Department of Informatics, University of Bergen, Norway

Tanmay Inamdar
Department of Informatics, University of Bergen, Norway

Meirav Zehavi
Ben-Gurion University of the Negev, Beer-Sheva, Israel

Abstract

The problem of packing of equal disks (or circles) into a rectangle is a fundamental geometric problem. (By a packing here we mean an arrangement of disks in a rectangle without overlapping.) We consider the following algorithmic generalization of the equal disk packing problem. In this problem, for a given packing of equal disks into a rectangle, the question is whether by changing positions of a small number of disks, we can allocate space for packing more disks. More formally, in the repacking problem, for a given set of n equal disks packed into a rectangle and integers k and h, we ask whether it is possible by changing positions of at most h disks to pack $n + k$ disks. Thus the problem of packing equal disks is the special case of our problem with $n = h = 0$.

While the computational complexity of packing equal disks into a rectangle remains open, we prove that the repacking problem is NP-hard already for $h = 0$. Our main algorithmic contribution is an algorithm that solves the repacking problem in time $(h + k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$, where $|I|$ is the input size. That is, the problem is fixed-parameter tractable parameterized by k and h.

2012 ACM Subject Classification Theory of computation → Packing and covering problems; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases circle packing, unit disks, parameterized complexity, fixed-parameter tractability

Digital Object Identifier 10.4230/LIPIcs.ICALP.2022.60

Category Track A: Algorithms, Complexity and Games

Funding *Fedor V. Fomin*: Supported by the Research Council of Norway via the project BWCA, grant no. 314528.
Petr A. Golovach: Supported by the Research Council of Norway via the project BWCA, grant no. 314528.
Tanmay Inamdar: Supported by the European Research Council (ERC) via grant LOPPRE, reference 819416.
Meirav Zehavi: Supported by the Israel Science Foundation (ISF) grant no. 1176/18.

1 Introduction

Packing of equal circles inside a rectangle or a square is one of the oldest packing problems. In addition to many common-life applications, like packing bottles or cans in a box [16], packings of circles have a variety of industrial applications, including circular cutting problems, communication networks, facility location, and dashboard layout. We refer to the survey of Castillo, Kampas, and Pintér [6] for an interesting overview of industrial applications of circle packings.

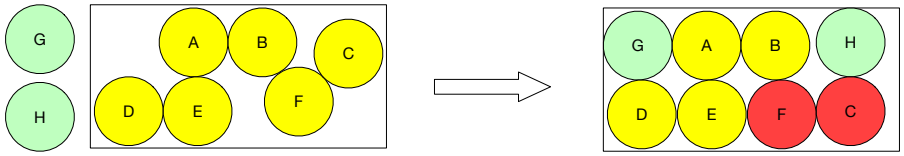

Figure 1 For a packing $\mathcal{P}$ of disks A–F, integers $h = 2$, and $k = 2$, the repacking $\mathcal{P}^*$ of $\mathcal{P}$ is obtained by relocating disks C and F, and by adding disks G and H.

The mathematical study of packing equal circles can be traced to Kepler [20]. Packing of circles also poses exciting mathematical and algorithmic challenges. After the significant efforts spent on packing for several decades [26, 29, 22, 24, 21, 28, 25, 12], optimal packings of equal circles inside a square are known only for instances of up to tens of circles [9, 27]. The computational complexity of packing of equal circles (NP-hardness or membership in NP) remains elusive. For packing circles with different radii, Demaine, Fekete, and Lang claimed NP-hardness [11]. See also the work of Abrahamsen, Miltzow, and Seiferth [1] for a generic framework for establishing $\exists\mathbb{R}$-completeness for packing problems.

Our paper establishes several results on computational and parameterized complexity of a natural generalization of packing equal circles inside a rectangle. A remark in the terminology is in order. In the literature on packing, both terms, circles and disks, could be found. While the term circle is much more popular than disk, we decided to use disks for the following reason: In our hardness proof, it is more convenient to operate with open disks. Thus all disks we consider are open and unit (that is of radius one). Let us remind, that a family of disks forms a *packing* if they are pairwise nonintersecting.[1] In our problem, we have a packing of disks in a rectangle, and the question is whether we can allocate some space for more disks by relocating a small amount of disks. More precisely, we consider the following problem. See Figure 1 for an example.

Disk Repacking

Input:	A packing $\mathcal{P}$ of n unit disks inside a rectangle R and two integers $h, k \geq 0$.
Task:	Decide whether there is a packing $\mathcal{P}^*$ of $n + k$ unit disks inside R obtained from $\mathcal{P}$ by adding k new disks and relocating at most h disks of $\mathcal{P}$ to new positions.

Thus when $n = 0$, that is, initially there are no disks inside the rectangle, this is the classical problem of packing equal circles inside a rectangle.

Related Work on Geometric Packing. Packing problems have received significant attention from the viewpoint of approximation algorithms. For the sake of illustration, let us mention a few examples. In 2D Geometric Bin Packing, which is a variant of classical Bin Packing, the goal is to pack a given collection of rectangles into the minimum number of unit square bins. Typically, it is required that the rectangles be packed in an axis-parallel manner. There has been a long series of results on this problem, culminating in the currently known best approximation given by Bansal and Khan [4]. A related problem is that of 2D Strip

[1] In the literature, it is often required for geometric packings that a packing should be maximal. In particular, for disk packing, every disk should touch either the bounding rectangle or another disk. However, in our problem, the task is to add a specified number of new disks to a given family and this makes the maximality condition in our case very artificial.

Packing problem, where the task is to pack a given set of rectangles into an infinite strip of the given width, so as to minimize the height of packing. This problem has been studied from the context of approximation [17, 19] as well as parameterized [2] algorithms. Finally, we mention the Geometric Knapsack problem, which is also closely related to Geometric Bin Packing. In Geometric Knapsack, we are given a collection of rectangles, where each rectangle has an associated profit. The goal is to pack a subset of the given rectangles (without rotation) in an axis-aligned square knapsack, so as to maximize the total profit of the packed rectangles. Currently, the best approximation is given by Galvez et al. [14]. A detailed survey of the literature on the results of these problems is beyond the scope of this work – we direct an interested reader to the cited works and references therein and the survey paper of Christensen et al. [7]. However, we would like to highlight an important difficulty in Disk Repacking– which is the focus of this work – as compared to the aforementioned geometric packing problems, namely, that packing disks in a rectangle requires the use of intricate geometric arguments as compared to packing rectilinear objects (such as rectangles) in a rectilinear container (such as a unit square, or an infinite strip).

Our Results. We show that Disk Repacking is NP-hard even if the parameter $h = 0$ – we call this special case of problem Disk Appending.

▶ **Theorem 1.** *Disk Appending is* NP*-hard when constrained to the instances* $(R, \mathcal{P}, k)$ *where* $R = [0, a] \times [0, b]$ *for positive integers* a *and* b *and the centers of all disks in* $\mathcal{P}$ *have rational coordinates. Furthermore, the problem remains* NP*-hard when it is only allowed to add new disks to* $\mathcal{P}$ *with rational coordinates of their centers.*

From the positive side, we show that Disk Repacking is FPT when parameterized by k and h. As it is common in Computational Geometry, we assume the *real RAM* computational model, that is, we are working with real numbers and assume that basic operations can be executed in unit time. We use $|I|$ to denote the input size.

▶ **Theorem 2.** *The Disk Repacking problem is* FPT *when parameterized by* $k + h$*. Specifically, it is solvable in time* $(h + k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$.

Theorem 2 also appears to be handy for approximating the maximum number of disks that can be added to a packing. In the optimization variant of Disk Repacking, called Max Disk Repacking, we are given a packing $\mathcal{P}$ of n disks in a rectangle R and an integer h, and the task is to maximize the number of new disks that can be added to the packing if we are allowed to relocate at most h disks of $\mathcal{P}$. By combining Theorem 2 with the approach of Hochbaum and Maass [18], we prove that the optimization variant of Disk Repacking admits the parameterized analog of EPTAS for the parameterization by h. More precisely, we prove the following theorem.

▶ **Theorem 3.** *For any* $0 < \varepsilon < 1$*, there exists an algorithm that, given an instance* $(\mathcal{P}, R, h)$ *of Max Disk Repacking, returns a packing* $\mathcal{P}^*$ *into* R *with at least* $n + (1 - \varepsilon) \cdot \mathsf{OPT}_h$ *disks in time* $\left(\frac{h+1}{\varepsilon}\right)^{\mathcal{O}(h/\varepsilon + 1/\varepsilon^2)} \cdot |I|^{\mathcal{O}(1)}$*, where* OPT_h *is the maximum number of disks that can be added to the input packing if we can relocate at most* h *disks.*

2 Preliminaries

Disks and rectangles. For two points A and B on the plane, we use AB to denote the line segment with endpoints in A and B. The *distance* between $A = (x_1, y_1)$ and $B = (x_2, y_2)$ or the *length* of AB, is $|AB| = \|A - B\|_2 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. The *(open unit) disk*

with a *center* $C = (c_1, c_2)$ on the plane is the set of points (x, y) satisfying the inequality $(x - c_1)^2 + (y - c_2)^2 < 1$. Whenever we write "disk" we mean an open unit disk. Throughout the paper, we assume that considered input rectangles $R = [0, a] \times [0, b]$ for some $a, b > 0$.

Parameterized Complexity. We refer to the book of Cygan et al. [10] for introduction to the area and undefined notions. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ^* is a set of strings over a finite alphabet Σ. An input of a parameterized problem is a pair (x, k), where $x \in \Sigma^*$ and $k \in \mathbb{N}$ is a *parameter*. A parameterized problem is *fixed-parameter tractable* (or FPT) if it can be solved in time $f(k) \cdot |x|^{\mathcal{O}(1)}$ for some computable function f.

Systems of Polynomial Inequalities. In our algorithms, we will need to find suitable locations for new disks that need to be added such that the locations are compatible with an existing packing. We will achieve this by solving systems of polynomial inequalities. We refer to the book of Basu, Pollack, and Roy [5] for basic tools. We use the following result.

▶ **Proposition 4** (Theorem 13.13 in [5]). *Let R be a real closed field, and let $\mathcal{P} \subseteq R[X_1, \ldots, X_\ell]$ be a finite set of s polynomials, each of degree at most d, and let*

$$(\exists X_1)(\exists X_2) \ldots (\exists X_\ell) F(X_1, X_2, \ldots, X_\ell)$$

be a sentence, where $F(X_1, \ldots, X_\ell)$ be a quantifier-free boolean formula involving $\mathcal{P}$-atoms of type $P \odot 0$, where $\odot \in \{=, \neq, >, <\}$, and P is a polynomial in $\mathcal{P}$. Then, there exists an algorithm to decide the truth of the sentence with complexity $s^{\ell+1} d^{\mathcal{O}(\ell)}$ in D, where D is the ring generated by the coefficients of the polynomials in $\mathcal{P}$.

Furthermore, a point $(X_1^*, \ldots, X_\ell^*)$ satisfying $F(X_1, \ldots, X_\ell)$ can be computed in the same time by Algorithm 13.2 (sampling algorithm) of [5] (see Theorem 13.11 of [5]). Note that because we are using the real RAM model in our algorithms, the complexity is stated with respect to the natural parameters.

3 Hardness of Disk Appending

In this section, we prove that Disk Appending is NP-hard. Due to space constraints, we only sketch the proof.

▶ **Theorem 1.** *Disk Appending is* NP*-hard when constrained to the instances $(R, \mathcal{P}, k)$ where $R = [0, a] \times [0, b]$ for positive integers a and b and the centers of all disks in $\mathcal{P}$ have rational coordinates. Furthermore, the problem remains* NP*-hard when it is only allowed to add new disks to $\mathcal{P}$ with rational coordinates of their centers.*

Sketch of proof. We reduce from the Independent Set problem. In this problem, for a given graph G and a positive integer k, the task is to decide whether G contains an independent set, that is a set of pairwise nonadjacent vertices, of size at least k. It is well-known that Independent Set is NP-complete on cubic planar graphs [15].

We only outline the main ideas of the reduction. Let G be a graph and assume that ℓ_e are positive integers given for all $e \in E(G)$. Suppose that G' is obtained from G by subdividing each edge e by $2\ell_e$ times. Then it can be shown that G has an independent set of size k if and only if G' has an independent set of size $k + \sum_{e \in E(G)} \ell_e$. We exploit this observation. Given a rectilinear embedding of a cubic planar graph G, for each vertex of G, we create a *node* area formed by surrounding disks. We can place an additional disk in such an area and this encodes the inclusion of the corresponding vertex to an independent set. Then we

join the areas created for vertices by *channels* corresponding to subdivided edges. Similarly to node areas, channels are formed by surrounding disks. Each channel contains an even number of positions where new disks can be placed, and these positions are divided into "odd" and "even" in such a way that we can put disks in either all odd or all even positions but no disks could be placed in adjacent even and odd positions. Thus node areas and channels are used to encode a graph, and then we fill the space around them by *filler* disks that prevent placing any new disk outside node areas and channels. Then placing new disks corresponds to the choice of an independent set in a subdivided graph.

To construct channels, consider four touching disks with centers A, B, C, and D shown in Figure 2 (a). Note that $h = 2 + \sqrt{3}$, $\ell = |AC| = |BC| = 2\sqrt{2 + \sqrt{3}}$, and the angle $\alpha = \pi/12$. Given disks with centers in A, B and C, every disk with its center in the triangle ABC has its center in D. Extending this, we make the following observation about the configuration of disks shown in Figure 2 (b). We call such a configuration of disks a *basic channel*.

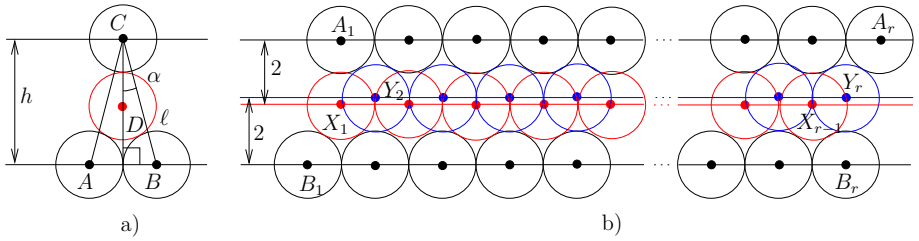

Figure 2 The basic channel; the disks shown in red and blue are not parts of the channel – they show places where new disks can be inserted.

▶ **Observation 5.** *Given disks with centers in $A_1, \ldots, A_r$ and $B_1, \ldots, B_r$ as it is shown in Figure 2 (b), any additional disk with its center properly inside the quadrilateral $A_1B_1B_rA_r$ has its center in one of the points $X_1, \ldots, X_{r-1}$ or $Y_2, \ldots, Y_r$. Furthermore, if a disk with its center in X_i (Y_i, respectively) is placed in the quadrilateral, then no other disk can have its center in Y_i or Y_{i+1} (X_{i-1} or X_i, respectively).*

It can be noted that the construction of basic channels is sufficiently robust to allow us to insert gaps between disks to adjust distances and parities. Furthermore, we can "bend" basic channels.

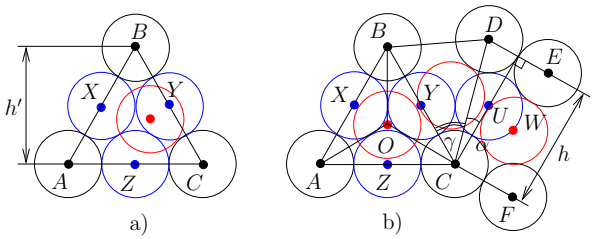

Figure 3 Node area.

For construction of node areas, consider an equilateral triangle ABC with sides of length two as shown in Figure 3 (a), $h' = 2\sqrt{3}$. Suppose that there are disks with centers in A, B and C. Then it is possible to place at most three disks with centers in the triangle ABC, and if exactly three disks are placed, then they have their centers in X, Y and Z and touch each other. Furthermore, if a disk having its center properly inside ABC is placed, then no

other disk with its center inside the triangle can be added. We exploit this property and add a basic channel as it is shown in Figure 3 (b). The point O is the center of ABC, that is, $|OA| = |OB| = |OC|$. Recall that $h = 2 + \sqrt{3}$ and $\alpha = \pi/12$. We set $\gamma = \pi/3 - \pi/12 = \pi/4$. This gives us the configuration of disks with the following properties summarized in the next observation.

▶ **Observation 6.** *Given disks with centers in A, B, C, D, E and F as it is shown in Figure 3 (b), the following is fulfilled:*

- **(i)** *at most one disk with its center in BCD can be added,*
- **(ii)** *if there is a disk with its center either in Y or U, then no other disk can have its center properly in BCD,*
- **(iii)** *if there are disks with their centers in O and W, then a disk with its center in BCD can be added,*
- **(iv)** *if there is a disk having its center properly inside ABC, then no other disk with its center inside ABC can be added.*

The node areas are connected by channels attached as it is shown in Figure 3 (b).

Note that in the described reduction, we used disks with algebraic coordinates of their centers. In particular, the crucial parameters $h = 2 + \sqrt{3}$ and $h' = 3\sqrt{3}$ are algebraic. However, we can observe that our construction is robust to allow rounding of coordinates. More precisely, we can choose a sufficiently small fixed constant $\delta > 0$ and use rational parameters $\hat{h}$ and $\hat{h}'$ such that $2 + \sqrt{3} = h < \hat{h} \leq h + \delta$ and $2\sqrt{3} = h' < \hat{h}' \leq h' + \delta$ in the construction of the channels (see Figure 2) and the node areas (see Figure 3) instead of h and h', respectively. ◀

4 An FPT algorithm for Disk Repacking

In this section, we prove that Disk Repacking is FPT when parameterized by $k + h$.

▶ **Theorem 2.** *The Disk Repacking problem is* FPT *when parameterized by $k + h$. Specifically, it is solvable in time $(h + k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$.*

Proof of Theorem 2: Overview. On a high-level, the idea behind the algorithm is as follows. We first perform a greedy procedure to ensure that all "free" areas to place disks can be intersected by a set $\mathcal{H}$ of at most k disks. Afterwards, we make use of a coloring function of $\mathcal{P}$ with the objective to color all disks in $\mathcal{P}$ that are repacked by a solution (if one exists) blue, and all disks in $\mathcal{P}$ that "closely surround" them by red. We need to ensure that, while relying on the initial greedy procedure, it would suffice to correctly color only $\mathcal{O}(h + k)$ disks. Indeed, this gives rise to the usage of a universal set, which is a "small" family of coloring functions ensured to contain, if there exists a solution, at least one coloring function that correctly colors all $\mathcal{O}(h + k)$ disks we care about.

Considering some coloring function (which expected to be "compatible" with some solution), we identify "slots" and, more generally, "containers" in its coloring pattern. In simple words, a slot is just a disk in R that does not intersect any red disk (from $\mathcal{P}$), and a container is a maximally connected region consisting of slots. We are able to prove that, if the coloring is compatible with some solution, then, for any container, either all or none of the disks in $\mathcal{P}$ that are contained in the container are repacked. This gives rise to a reduction from the problem of finding a solution compatible with a given coloring to the Knapsack problem (more precisely, an extended version of it), where each container corresponds to an item whose weight is the number of disks in $\mathcal{P}$ that it contains, and whose value is the number of disks that can be packed within it.

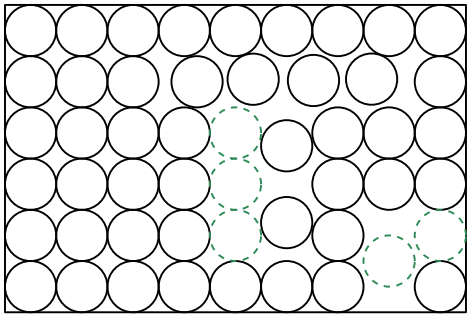

Figure 4 An instance $(\mathcal{P}, R, h = 2, k = 7)$ of Disk Repacking. The disks in $\mathcal{P}$ are colored black. The disks in some hole cover $\mathcal{H}$ are colored green (using dashed lines).

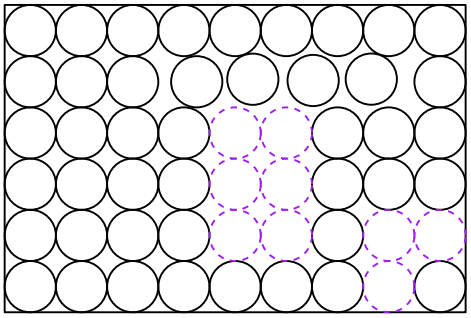

Figure 5 A solution $\mathcal{P}^*$ for the instance on the left. The disks in $\mathcal{P}^* \setminus \mathcal{P}$ are drawn in purple (using dashed lines).The set of $(\mathcal{H}, \mathcal{P}^*)$-critical disks is the set of green disks from the figure on the left and the purple disks from the figure on the right.

To execute the reduction described above, we need to be able to compute the value of each container. For this purpose, we first prove that a container can be "described" by only $\mathcal{O}(h+k)$ many disks from $\mathcal{P} \cup \mathcal{H}$; more precisely, we show that each container is the union of disks contained in R that intersect at least one out of $\mathcal{O}(h+k)$ disks in $\mathcal{P} \cup \mathcal{H}$, from which we subtract the union of some other $\mathcal{O}(h+k)$ disks from $\mathcal{P}$. Having this at hand, to compute the value of a container, we first "guess", for each disk packed by a (hypothetical) optimal packing of disks in the container, a disk from $\mathcal{P} \cup \mathcal{H}$ contained in the container (making use of its description) with whom it intersects. After that, we seek the corresponding optimal packing by making use of a system of polynomial equations (inequalities) of degree 2, $\mathcal{O}(h+k)$ variables, and $\mathcal{O}((h+k)^2)$ equations.

Proof of Theorem 2: Free areas. To execute the plan above, we start with the task of handling the "free" areas. For this, we have the following definition and immediate observation.

▶ **Definition 7** (Holes and Hole Cover). *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. The* set of holes*, denoted by* Holes*, is the set of all disks contained in R that are disjoint from all disks in $\mathcal{P}$. A set $\mathcal{H}$ of disks contained in R such that the set of holes of $(\mathcal{P} \cup \mathcal{H}, R, h, k)$ is empty is called a* hole cover*.*

▶ **Observation 8.** *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $\mathcal{H}$ be a hole cover. Then, every disk contained in R intersects at least one disk in $\mathcal{P} \cup \mathcal{H}$.*

Next, we present a definition and a lemma that will allow us to assume that we have a hole cover of small size at hand.

▶ **Definition 9** (Dense instance). *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. We say that the instance is* dense *if it has a hole cover of size smaller than k.*

▶ **Lemma 10.** *There exists a polynomial-time algorithm that, given an instance $(\mathcal{P}, R, h, k)$ of Disk Repacking, either correctly determines that $(\mathcal{P}, R, h, k)$ is a yes-instance or correctly determines that $(\mathcal{P}, R, h, k)$ is dense and returns a hole cover of size smaller than k.*

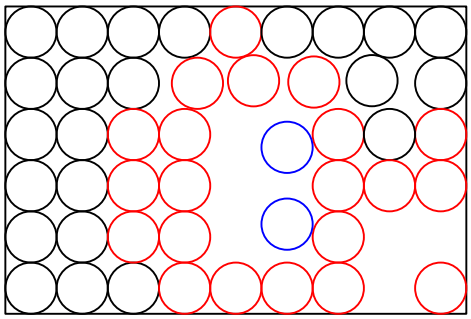

Figure 6 With respect to the instance and solution described in Figures 4 and 5, the disks $(\mathcal{H}, \mathcal{P}^*)$-forced to be blue are colored blue, and the disks $(\mathcal{H}, \mathcal{P}^*)$-forced to be red are colored red. Note that each of the disks colored black can be colored either blue or red by an $(\mathcal{H}, \mathcal{P}^*)$-compatible coloring.

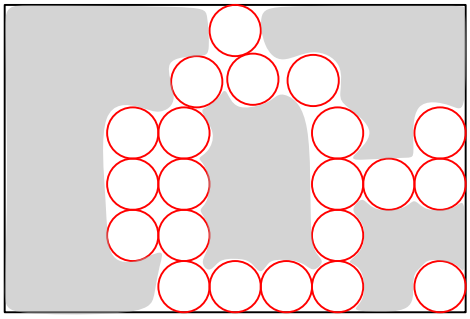

Figure 7 Consider an $(\mathcal{H}, \mathcal{P}^*)$-compatible coloring that colors blue all of the disks colored black in Figure 6. Then, we have four c-Containers, which roughly correspond to the areas colored by grey.

Proof. We perform a simple greedy procedure. Initially, $\mathcal{H} = \emptyset$. Then, as long as there exists a disk D contained in R that is disjoint from all disks in $\mathcal{P} \cup \mathcal{H}$, we add such a disk D to $\mathcal{H}$. The test for the existence of such a D can be performed by using a system of polynomial equations of degree 2 with two variables denoting the x- and y-coordinates of the center of D. For each disk in $\mathcal{P} \cup \mathcal{H}$, we have an equation enforcing that the distance between its center and the center of D is at least 2, and additionally we have two linear equations to enforce that D is contained in R. By Proposition 4, testing whether this system has a solution (which corresponds to the sought disk D) can be done is polynomial time.[2] Once the process terminates, the algorithm checks whether $|\mathcal{H}| \geq k$. If the answer is positive, then adding $\mathcal{H}$ (or, more precisely, any subset of size k of it) to $\mathcal{P}$ is a solution, and so the algorithm answers yes, and otherwise the instance is dense and the algorithm returns $\mathcal{H}$ (which witnesses that). ◀

In the two following definitions, we identify the coloring functions that will be useful.

▶ **Definition 11** ($(\mathcal{H}, \mathcal{P}^*)$-Critical Disks). *Let $(\mathcal{P}, R, h, k)$ be a yes-instance of* Disk Repacking. *Let $\mathcal{H}$ be a hole cover. Let $\mathcal{P}^*$ be a solution to $(\mathcal{P}, R, h, k)$. The set of* $(\mathcal{H}, \mathcal{P}^*)$-critical disks, *denoted by* $\mathsf{Crit}_{\mathcal{H},\mathcal{P}^*}$, *is* $(\mathcal{P}^* \setminus \mathcal{P}) \cup \mathcal{H}$.

▶ **Definition 12** ($(\mathcal{H}, \mathcal{P}^*)$-Compatible Colorings). *Let $(\mathcal{P}, R, h, k)$ be a yes-instance of* Disk Repacking. *Let $\mathcal{H}$ be a hole cover. Let $\mathcal{P}^*$ be a solution to $(\mathcal{P}, R, h, k)$. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. We say that c is* $(\mathcal{H}, \mathcal{P}^*)$-compatible *if:*

1. *For every $D \in \mathcal{P} \setminus \mathcal{P}^*$, we have that $c(D) = \mathsf{blue}$. We say that the disks in $\mathcal{P} \setminus \mathcal{P}^*$ are* $(\mathcal{H}, \mathcal{P}^*)$-forced to be blue.
2. *For every $D \in \mathcal{P} \cap \mathcal{P}^*$ whose center is at distance at most 4 from the center of some disk in $\mathsf{Crit}_{\mathcal{H},\mathcal{P}^*}$, we have that $c(D) = \mathsf{red}$. We say that the disks in $\mathcal{P} \cap \mathcal{P}^*$ whose center is at distance at most 4 from the center of some disk in $\mathsf{Crit}_{\mathcal{H},\mathcal{P}^*}$ are* $(\mathcal{H}, \mathcal{P}^*)$-forced to be red.

We proceed to show that the number of disks in $\mathcal{P}$ that should be colored "correctly" is only $\mathcal{O}(h + k)$. This is done using the following easy observation, in the following lemma.

[2] Additional details on the precise set of equations mentioned here and in other locations in this section are omitted from this extended abstract due to space constraints.

▶ **Observation 13.** *The number of pairwise disjoint disks inside a circle of radius r is at most πr^2.*

▶ **Lemma 14.** *Let $(\mathcal{P}, R, h, k)$ be a dense yes-instance of Disk Repacking. Let $\mathcal{H}$ be a hole cover of size smaller than k. Let $\mathcal{P}^*$ be a solution to $(\mathcal{P}, R, h, k)$. Then, the number of disks $(\mathcal{H}, \mathcal{P}^*)$-forced to be either blue or red is altogether bounded by $\mathcal{O}(h + k)$.*

Proof. Because $\mathcal{P}^*$ is a solution and $|\mathcal{H}| < k$, we have that $|\mathcal{P} \setminus \mathcal{P}^*| \leq h$. So, at most h disks are $(\mathcal{H}, \mathcal{P}^*)$-forced to be blue. Further, $|\mathsf{Crit}_{\mathcal{H},\mathcal{P}^*}| = |(\mathcal{P}^* \setminus \mathcal{P}) \cup \mathcal{H}| < h + 2k$. Observe that every disk in $\mathcal{P} \cap \mathcal{P}^*$ whose center is at distance at most 4 from the center of some disk in $\mathsf{Crit}_{\mathcal{H},\mathcal{P}^*}$ is contained inside a circle of radius 5 whose center is the center of some disk in $\mathsf{Crit}_{\mathcal{H},\mathcal{P}^*}$. So, due to Observation 13 and since the disks in $\mathcal{P} \cap \mathcal{P}^*$ are pairwise disjoint, there exist at most $\pi \cdot 5^2 \cdot (h + 2k) = \mathcal{O}(h + k)$ disks in $\mathcal{P} \cap \mathcal{P}^*$ whose center is at distance at most 4 from the center of some disk in $\mathsf{Crit}_{\mathcal{H},\mathcal{P}^*}$. In particular, this means that at most $\mathcal{O}(h + k)$ disks are $(\mathcal{H}, \mathcal{P}^*)$-forced to be red. This completes the proof. ◀

Proof of Theorem 2: Values of containers. Next, we present the definition of slots and containers, in which we will aim to (re)pack disks. The definition is followed by an observation and a lemma, which, in particular, state that if we try to repack at least one disk in a container, we can just repack all disks in that container.

▶ **Definition 15** (c-Slots and c-Containers)**.** *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. The set of* c-slots*, denoted by Slots_c, is the set of disks contained in R that are disjoint from all disks in $\mathcal{P}$ that are colored red by c. The set of* c-containers*, denoted by $\mathsf{Containers}_c$, is the set of maximally connected regions in the union of all disks in Slots_c.*

▶ **Observation 16.** *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. Then, the regions in $\mathsf{Containers}_c$ are pairwise disjoint.*

▶ **Lemma 17.** *Let $(\mathcal{P}, R, h, k)$ be a yes-instance of Disk Repacking. Let $\mathcal{H}$ be a hole cover. Let $\mathcal{P}^*$ be a solution to $(\mathcal{P}, R, h, k)$. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$ be $(\mathcal{H}, \mathcal{P}^*)$-compatible. Then, for every region $X \in \mathsf{Containers}_c$, either all disks in $\mathcal{P}$ contained in X belong to $\mathcal{P} \setminus \mathcal{P}^*$ or none of the disks in $\mathcal{P} \cup \mathcal{P}^*$ contained in X belongs to $(\mathcal{P} \setminus \mathcal{P}^*) \cup (\mathcal{P}^* \setminus \mathcal{P})$.*

Proof. Targeting a contradiction, suppose that there exists a disk D contained in X that belongs to $(\mathcal{P} \setminus \mathcal{P}^*) \cup (\mathcal{P}^* \setminus \mathcal{P})$ and a disk D' contained in X that belongs to $\mathcal{P} \cap \mathcal{P}^*$. Let γ be a curve connecting the centers of these disks that lies entirely inside X. By the definition of a c-container and due to Observation 8, every point of this curve contained in a disk that belongs to X and intersects a disk in $\mathcal{P}$ colored blue by c or a disk in $\mathcal{H}$. So, there must exist a point on γ that is the center of a disk D^* that intersects both a disk A contained in X that belongs to $(\mathcal{P} \setminus \mathcal{P}^*) \cup \mathcal{H}$ and a disk A' contained in X that belongs to $\mathcal{P} \cap \mathcal{P}^*$. From the definition of a c-container, A' is colored blue by c. Moreover, note that the center of A' is at distance at most 4 from the center of A, since each of the centers of A and A' is at distance at most 2 from the center of D^*. However, since c is $(\mathcal{H}, \mathcal{P}^*)$-compatible, A' is $(\mathcal{H}, \mathcal{P}^*)$-forced to be red and hence it is colored red by c. Since c cannot color a disk both blue and red, we have reached a contradiction. This completes the proof. ◀

We proceed to define the weight and value of a c-container, which will be required for the reduction of our problem to Knapsack.

▶ **Definition 18** (Weight, Validity and Value of Containers). *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. Let $X \in \mathsf{Containers}_c$. The* weight *of X is the number of disks in $\mathcal{P}$ that it contains. We say that X is* valid *if its weight is at most h. The* value *of X is the maximum number of disks that can be packed inside X.*

The following is a corollary of Lemma 17.

▶ **Corollary 19.** *Let $(\mathcal{P}, R, h, k)$ be a yes-instance of Disk Repacking. Let $\mathcal{P}^*$ be a solution to $(\mathcal{P}, R, h, k)$. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$ be $(\mathcal{H}, \mathcal{P}^*)$-compatible. Then, every disk in $(\mathcal{P} \setminus \mathcal{P}^*) \cup (\mathcal{P}^* \setminus \mathcal{P})$ is a c-slot, and it is contained in a valid c-container.*

Now, we define a way in which we can "easily" describe a container, and then prove that this way can be encoded compactly.

▶ **Definition 20** (Descriptions of Containers). *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $\mathcal{H}$ be a hole cover. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. An* $\mathcal{H}$-description *(or, for short,* description*) of a region $X \in \mathsf{Containers}_c$ is a pair $(\mathcal{D}_1, \mathcal{D}_2)$ of a subset $\mathcal{D}_1 \subseteq \mathcal{P} \cup \mathcal{H}$ and a minimal subset $\mathcal{D}_2 \subseteq \mathcal{P}$ such that X equals the set of all points in R at distance less than 2 from at least one disk in $\mathcal{D}_1$ and at least 2 from all disks in $\mathcal{D}_2$.*

▶ **Lemma 21.** *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $\mathcal{H}$ be a hole cover. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. Let $X \in \mathsf{Containers}_c$. Then, X has at least one description $(\mathcal{D}_1, \mathcal{D}_2)$. Moreover, every description $(\mathcal{D}_1, \mathcal{D}_2)$ of X satisfies $|\mathcal{D}_1| + |\mathcal{D}_2| = \mathcal{O}(h' + k')$ where h' is the weight of X, and k' is the number of disks in $\mathcal{H}$ contained in X.*

Proof. By Observation 8, every c-slot intersects at least one disk in $\{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$ and is disjoint from all disks in $\{D \in \mathcal{P} : c(D) = \mathsf{red}\}$. Further, every point in every disk in $\{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$ is contained in a c-slot. So, it is immediate that X has a description $(\mathcal{D}_1, \mathcal{D}_2)$, and that $|\mathcal{D}_1| = \mathcal{O}(h' + k')$. Due to Observation 13 and since the disks in $\mathcal{P} \cup \mathcal{H}$ are pairwise disjoint, any circle of radius 5 whose center is a center of some disk in $\{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$ can contain inside at most $\pi \cdot 5^2$ disks from $\{D \in \mathcal{P} : c(D) = \mathsf{red}\}$. Due to the minimality of $\mathcal{D}_2$ (which is a subset of $\{D \in \mathcal{P} : c(D) = \mathsf{red}\}$), every disk in it must be contained inside a circle of radius 5 whose center is a center of some disk in $\{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$. Hence, $|\mathcal{D}_2| \leq |\mathcal{D}_1| \cdot \pi \cdot 5^2 = \mathcal{O}(h' + k')$. ◀

Next, we use a description in order to efficiently compute the value of a c-container.

▶ **Lemma 22.** *There is an $(h+k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$-time algorithm that, given a dense instance $I = (\mathcal{P}, R, h, k)$ of Disk Repacking, a hole cover $\mathcal{H}$ of size smaller than k, $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$ and a valid region X with a description $(\mathcal{D}_1, \mathcal{D}_2)$, computes the value of X.*

Proof. Given $I = (\mathcal{P}, R, h, k), \mathcal{H}, c, X$ and $(\mathcal{D}_1, \mathcal{D}_2)$, the algorithm works as follows. For $\ell = h+k, h+k-1, \ldots, 1$, and for every vector $(D_1, D_2, \ldots, D_\ell) \in \mathcal{D}_1 \times \mathcal{D}_1 \times \cdots \times \mathcal{D}_1$, it tests whether there exist ℓ disks $S_1, S_2, \ldots, S_\ell$ such that, for every $i \in \{1, 2, \ldots, \ell\}$, S_i intersects D_i, is contained in R and is disjoint from all disks in $\mathcal{D}_2$. The test is done by constructing a system of polynomial equations of degree 2 with 2ℓ variables and $\ell \cdot (|\mathcal{D}_2| + 2)$ equations as follows. For every $i \in \{1, 2, \ldots, \ell\}$, we have two variables, denoting the x- and y-coordinates of the center of S_i, one equation enforcing that the distance between the center of S_i and the center of D_i is smaller than 2, $|\mathcal{D}_2|$ equations enforcing that the distance between the center of S_i and the center of each of the disks in $\mathcal{D}_2$ is at least 2, and two linear equations enforcing that S_i is contained inside R. If the answer is positive, then the algorithm returns that the value of X is ℓ and terminates; else, it proceeds to the next iteration. Observe that, when $\ell = 1$, the algorithm necessarily terminates (since X contains at least one c-slot).

The correctness of the algorithm is immediate from the definition of a description and the exhaustive search that it performs. As for the running time, first observe that, by Lemma 21 and since X is valid and $|\mathcal{H}| < k$, $|\mathcal{D}_1| + |\mathcal{D}_2| \leq \mathcal{O}(h+k)$. So, for a given ℓ, we have $|\mathcal{D}_1|^{\mathcal{O}(\ell)} = (h+k)^{\mathcal{O}(h+k)}$ choices of vectors. Now, consider the iteration corresponding to some ℓ and some vector. Then, we solve a system of polynomial equations of degree 2 with $\mathcal{O}(h+k)$ variables and $\mathcal{O}((h+k)^2)$ equations. By Proposition 4, this can be done in time $(h+k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$. Thus, the algorithm indeed runs in time $(h+k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$. ◀

The following definition captures the set of all descriptions.

▶ **Definition 23** (Blueprint). *Let $(\mathcal{P}, R, h, k)$ be an instance of* Disk Repacking. *Let $\mathcal{H}$ be a hole cover. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. An $(\mathcal{H}, c)$-*blueprint *is a collection of pairs of sets $\mathsf{Blueprint} \subseteq 2^{\mathcal{P} \cup \mathcal{H}} \times 2^{\mathcal{P}}$, where the first elements of the pair are pairwise-disjoint subsets of $\mathcal{P} \cup \mathcal{H}$, such that each region in $\mathsf{Containers}_c$ has exactly one description in $\mathsf{Blueprint}$, and every pair in $\mathsf{Blueprint}$ is a description of a region in $\mathsf{Containers}_c$.*

Next, we show how to compute blueprints.

▶ **Lemma 24.** *There exists a polynomial-time algorithm that, given an instance $(\mathcal{P}, R, h, k)$ of* Disk Repacking, *a hole cover $\mathcal{H}$, and $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$, outputs an $(\mathcal{H}, c)$-blueprint.*

Proof. We will perform a simple greedy procedure to identify, for each disk in $\{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$, the description of the region that contains it. Observe that every c-container contains at least one disk in $\{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$ (due to Observation 8 and the definition of a c-container). So, if for every disk $D \in \{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$ we will take exactly one description $(\mathcal{D}_1, \mathcal{D}_2)$ among the descriptions we identified such that D is contained in $\mathcal{D}_1$, we will obtain an $(\mathcal{H}, c)$-blueprint.

To describe the greedy procedure, consider some $D \in \{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}$. Let us first show how to attain $\mathcal{D}_1$. For this purpose, we initialize $\mathcal{D}_1 = \{D\}$. Then, for every pair of disks $A \in \mathcal{D}_1$ and $B \in (\{D \in \mathcal{P} : c(D) = \mathsf{blue}\} \cup \mathcal{H}) \setminus \mathcal{D}_1$, we test whether there exists a pair of disks C and C' that are contained in R, intersect each other, are disjoint from all disks in $\{D \in \mathcal{P} : c(D) = \mathsf{red}\}$, and such that C intersects A and C' intersects B. The test for the existence of such a C is performed by using a system of polynomial equations of degree 2 with four variables denoting the x- and y-coordinates of the centers of C and C'. For each disk in $\{D \in \mathcal{P} : c(D) = \mathsf{red}\}$, we have two equations enforcing that the distances between its center and the centers of C and C' are each at least 2. Additionally, we have three equations to enforce that the distance between the centers of C and C' is smaller than 2, the distance between the centers of C and A is smaller than 2, and the distance between the centers of C' and B is smaller than 2, as well as four linear equations to enforce that C and C' are contained in R. By Proposition 4, testing whether this system has a solution (which corresponds to the sought disks C and C') can be done is polynomial time. If the answer is positive, then we add B to $\mathcal{D}_1$. In case at least one pair (A, B) resulted in the addition of B to $\mathcal{D}_1$, then we repeat the entire loop, iterating again over all pairs (A, B) (where the domain from which they are taken is updated as a new disk was added to $\mathcal{D}_1$). Notice that we can perform at most $|\mathcal{P}|$ repetitions, and that each repetition results in at most $|\mathcal{P} \cup \mathcal{H}|^2$ many iterations, each taking polynomial time. Hence, the procedure, so far, runs in polynomial time.

Now, let us show how to attain $\mathcal{D}_2$. For this purpose, we initialize $\mathcal{D}_2 = \{D \in \mathcal{P} : c(D) = \mathsf{red}\}$. Now, for every $A \in \{D \in \mathcal{P} : c(D) = \mathsf{red}\}$, we test whether there exists a disk C that is contained in R and intersects both A and at least one disk in $\mathcal{D}_1$, and is disjoint from all disks in $\mathcal{D}_2 \setminus \{A\}$. The test can be performed by iterating over every disk $B \in \mathcal{D}_1$, and

using a system of polynomial equations of degree 2 with two variables denoting the x- and y-coordinates of the center of C. For each disk in $\mathcal{D}_2 \setminus \{A\}$, we have an equation enforcing that the distance between its center and the center of C is at least 2, and additionally we have two equations to enforce that the distance between the center of C and each of the centers of A and B is smaller than 2, as well as two linear equations to enforce that C is contained in R. By Proposition 4, testing whether this system has a solution (which corresponds to the sought disk C) can be done is polynomial time. If the answer is positive, then we remove A from $\mathcal{D}_2$. Notice that this phase of the procedure also runs in polynomial time. Moreover, the correctness of the entire procedure directly follows from the definitions of a c-container and a description. ◀

We proceed to define the (extended version of the) Knapsack problem and the instances of this problem that our reduction produces.

▶ **Definition 25** ((Extended) Knapsack)**.** *In the* (Extended) Knapsack *problem, we are given a collection of n items U, where each item $u \in U$ has a weight $w(u) \in \mathbb{N}_0$ and a value $v(u) \in \mathbb{N}_0$, and an integer $W \in \mathbb{N}_0$. The objective is to find, for every $W' \in \{0, 1, \ldots, W\}$, the maximum $V_{W'} \in \mathbb{N}_0$ for which there exists a subset of items $S \subseteq \{1, 2, \ldots, n\}$ such that $\sum_{i \in S} w(u) \leq W'$ and $\sum_{i \in S} v(u) \geq V_{W'}$.*

▶ **Definition 26** ($(\mathcal{H}, c)$-Knapsack instance)**.** *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $\mathcal{H}$ be a hole cover. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. The $(\mathcal{H}, c)$-*Knapsack instance *is the instance (U, w, v, W, V) of* Knapsack *defined as follows: U is the set of all valid regions in $\mathsf{Containers}_c$; for each $X \in U$, $w(X)$ and $v(X)$ are the weight and value of X (see Definition 18); $W = h$; $V = h + k$.*

▶ **Proposition 27** ([8])**.** *The* (Extended) Knapsack *problem is solvable in time $\mathcal{O}(|U| \cdot W)$.*

We now to prove the correspondence between our problem when we restrict the solution set to solutions compatible with a given coloring and the Knapsack problem.

▶ **Lemma 28.** *Let $(\mathcal{P}, R, h, k)$ be an instance of Disk Repacking. Let $\mathcal{H}$ be a hole cover. Let $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$. Then, there exists a solution $\mathcal{P}^*$ to $(\mathcal{P}, R, h, k)$ such that c is compatible with $\mathcal{P}^*$ if and only if for the $(\mathcal{H}, c)$-*Knapsack *instance (U, w, v, W, V), there exists $W' \in \{0, 1, \ldots, W\}$ such that $V_{W'} \geq W' + k$.*

Proof. In one direction, suppose that there exists a solution $\mathcal{P}^*$ to $(\mathcal{P}, R, h, k)$ such that c is compatible with $\mathcal{P}^*$. Let $X_1, X_2, \ldots, X_\ell$ be the c-containers that contain at least one disk from $(\mathcal{P} \setminus \mathcal{P}^*) \cup (\mathcal{P}^* \setminus \mathcal{P})$. By Observation 16, these c-containers are pairwise disjoint, by Lemma 17 and since c is compatible with $\mathcal{P}^*$, all disks in $\mathcal{P}$ contained in $X_1 \cup X_2 \cup \cdots \cup X_\ell$ belong to $\mathcal{P} \setminus \mathcal{P}^*$, and by Corollary 19 and since c is compatible with $\mathcal{P}^*$, all disks in $(\mathcal{P} \setminus \mathcal{P}^*) \cup (\mathcal{P}^* \setminus \mathcal{P})$ are contained in $X_1 \cup X_2 \cup \cdots \cup X_\ell$ and all of these c-containers are valid. So, because $\mathcal{P}^*$ can repack h disks from $\mathcal{P}$, the total weight of these c-containers must be some $W' \in \{0, 1, \ldots, h\} = \{0, 1, \ldots, W\}$, and since $\mathcal{P}^*$ also packs k additional disks, the total value of these c-containers must be at least $W' + k$ (to accommodate all of the repacked and k newly packed disks). Thus, $V_{W'} \geq W' + k$.

In the other direction, suppose that there exists $W' \in \{0, 1, \ldots, W\}$ such that $V_{W'} \geq W' + k$. This means that there exist c-containers $X_1, X_2, \ldots, X_\ell$ whose total weight is $W' \in \{0, 1, \ldots, h\}$ and whose total value is at least $W' + k$. However, because these c-containers are pairwise disjoint (by Observation 16), this means that we can construct a solution $\mathcal{P}^*$ such that c is compatible with $\mathcal{P}^*$ by repacking all the disks in $\mathcal{P}$ that are contained in $X_1, X_2, \ldots, X_\ell$ (there are at most h such disks) and, additionally, inserting k new disks, within $X_1, X_2, \ldots, X_\ell$. This completes the proof. ◀

The following is a corollary of Lemmas 22 and 24.

▶ **Corollary 29.** *There exists an $(h+k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$-time algorithm that, given a dense instance $I = (\mathcal{P}, R, h, k)$ of* Disk Repacking*, a hole cover $\mathcal{H}$ of size smaller than k and $c : \mathcal{P} \to \{\mathsf{blue}, \mathsf{red}\}$, computes the $(\mathcal{H}, c)$-*Knapsack *instance.*

To compute coloring functions, we will use the following definition and proposition.

▶ **Definition 30** ((U,k)-Universal Set)**.** *For a universe U and $k \in \mathbb{N}$, a (U,k)-*universal set *is a collection $\mathcal{C}$ of functions $f : U \to \{\mathsf{blue}, \mathsf{red}\}$ such that for every pair of disjoint sets $B, R \subseteq U$ whose union has size at most k, there exists $c \in \mathcal{C}$ that colors all integers in B blue and all integers in R red.*

▶ **Proposition 31** ([23])**.** *There exists an algorithm that, given a universe U of size n and $k \in \mathbb{N}$, constructs a (U,k)-universal set of size $2^{k+\mathcal{O}(\log^2 k)} \log n$ in time $2^{k+\mathcal{O}(\log^2 k)} n \log n$.*

Based on the definition of a universal set, we define the collection of Knapsack instances relevant to our reduction.

▶ **Definition 32** ($(\mathcal{H},\mathcal{C})$-Knapsack Collection)**.** *Let $(\mathcal{P}, R, h, k)$ be an instance of* Disk Repacking*. Let $\mathcal{H}$ be a hole cover. Let $\mathcal{C}$ be a $(\mathcal{P}, q(h+k))$-universal set, where q is the constant hidden in the $\mathcal{O}$-notation in Lemma 14. Then, the* $(\mathcal{H},\mathcal{C})$-Knapsack collection *is the collection of* Knapsack *instances that includes, for every $c \in \mathcal{C}$, the* $(\mathcal{H}, c)$-Knapsack *instance.*

The following is a corollary of Corollary 29.

▶ **Corollary 33.** *There exists an $(h+k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$-time algorithm that, given a dense instance $I = (\mathcal{P}, R, h, k)$ of* Disk Repacking*, a hole cover $\mathcal{H}$ of size smaller than k and a $(\mathcal{P}, q(h+k))$-universal set $\mathcal{C}$, computes the* $(\mathcal{H},\mathcal{C})$-Knapsack *collection.*

Next, we prove the correspondence between our problem and the collection of Knapsack instances we have just defined.

▶ **Lemma 34.** *Let $(\mathcal{P}, R, h, k)$ be an instance of* Disk Repacking*. Let $\mathcal{H}$ be a hole cover. Let $\mathcal{C}$ be a $(\mathcal{P}, q(h+k))$-universal set. Then, $(\mathcal{P}, R, h, k)$ is a yes-instance of* Disk Repacking *if and only if the* $(\mathcal{H},\mathcal{C})$-Knapsack *collection contains an instance (U, w, v, W, V) for which there exists $W' \in \{0, 1, \ldots, W\}$ such that $V_{W'} \geq W' + k$.*

Proof. In one direction, suppose that $(\mathcal{P}, R, h, k)$ is a yes-instance. By the definition of a $(\mathcal{P}, q(h+k))$-universal set and due to Lemma 14, there exists $c \in \mathcal{C}$ that is compatible with $\mathcal{P}^*$. So, the $(\mathcal{H}, c)$-Knapsack instance is contained in the $(\mathcal{H},\mathcal{C})$-Knapsack collection (U, w, v, W, V), and by Lemma 28, for this instance there exists $W' \in \{0, 1, \ldots, W\}$ such that $V_{W'} \geq W' + k$.

In the other direction, suppose that the $(\mathcal{H},\mathcal{C})$-Knapsack collection contains an instance (U, w, v, W, V) for which there exists $W' \in \{0, 1, \ldots, W\}$ such that $V_{W'} \geq W' + k$. This instance is a $(\mathcal{H}, c)$-Knapsack instance for some $c \in \mathcal{C}$. So, by Lemma 28, $(\mathcal{P}, R, h, k)$ is, in particular, a yes-instance of Disk Repacking. ◀

Proof of Theorem 2: Putting it all together. We are now ready to make the final step of the proof of Theorem 2.

The algorithm works as follows. Given an instance $(\mathcal{P}, R, h, k)$ of Disk Repacking, it calls the algorithm in Lemma 10 to either correctly determine that $(\mathcal{P}, R, h, k)$ is a yes-instance or correctly determine that $(\mathcal{P}, R, h, k)$ is dense and obtain a hole cover $\mathcal{H}$

of size smaller than k. In the first case, the algorithm is done. In the second case, the algorithm proceeds as follows. First, it calls the algorithm in Proposition 31 to obtain a $(\mathcal{P}, q(h+k))$-universal set $\mathcal{C}$. Then, it calls the algorithm in Corollary 33 to obtain the $(\mathcal{H}, \mathcal{C})$-Knapsack collection. Afterwards, it uses the algorithm of Proposition 27 to determine whether the $(\mathcal{H}, \mathcal{C})$-Knapsack collection contains an instance (U, w, v, W, V) for which there exists $W' \in \{0, 1, \ldots, W\}$ such that $V_{W'} \geq W' + k$.

The correctness of the algorithm follows from Lemma 34. The runtime bound of $(h+k)^{\mathcal{O}(h+k)} \cdot |I|^{\mathcal{O}(1)}$ follows from the runtimes bounds of the algorithms that the algorithm calls, stated in Lemma 10, Proposition 31, Corollary 33, and Proposition 27.

This concludes the proof of Theorem 2.

5 An FPT approximation for Maximum Disk Repacking

In this section, we use Theorem 2 to show that the optimization variant of Disk Repacking, where we maximize the number of added disks, admits an FPT-AS (i.e., *Fixed Parameter Tractable Approximation Scheme*, a parameterized analog of EPTAS) when parameterized by h. Let us remind that in the optimization problem, called Max Disk Repacking, we are given a packing $\mathcal{P}$ of n disks in a rectangle R and an integer h, and the task is to maximize the number of new disks that can be added to the packing if we are allowed to relocate at most h disks of $\mathcal{P}$.

We need an algorithm for the special case $h = 0$, that is, for the optimization version of Disk Appending. The algorithm is based on the shifting technique, originally introduced by Hochbaum and Maass [18] (also related to Baker's technique [3]). We use OPT for the maximum number of disks that can be added in a rectangle to complement a given packing $\mathcal{P}$.

▶ **Lemma 35.** *For any $0 < \varepsilon < 1$, there exists an algorithm that for a packing of n disks in a rectangle, returns a packing with at least $n + (1-\varepsilon) \cdot \mathsf{OPT}$ disks in time $\left(\frac{1}{\varepsilon}\right)^{\mathcal{O}(1/\varepsilon^2)} \cdot |I|^{\mathcal{O}(1)}$, where $|I|$ is the input size.*

Proof. Let $\mathcal{S}^*$, $|\mathcal{S}^*| = \mathsf{OPT}$, be the set of newly added disks in an optimal solution. Let $\ell \geq 1$ be a fixed positive integer. Recall that the instance is contained inside a bounding rectangle R. Let us assume that the bottom-left corner of R has Cartesian coordinates $(0, 0)$. For every $1 \leq i, j \leq 2\ell$, let $G_{i,j}$ be a grid of side-length $\ell \times \ell$, with origin at $(-i, -j)$. Note that there exists a pair (i, j) such that the number of disks of $\mathcal{S}^*$ that do not intersect with the boundary of the grid cells in $G_{i,j}$ is at least $(1 - \frac{1}{\ell})^2 \cdot \mathsf{OPT}$.

For any $1 \leq i, j \leq n$, and a grid cell C in $G_{i,j}$, let $\Pi(C)$ be the following subproblem. Let $\mathcal{P}(C) \subseteq \mathcal{P}$ denote the packing of the original disks that are completely contained in C, or partially intersect with C. The goal is to add the maximum number of new disks to obtain a packing $\mathcal{P}^*(C)$. Note that the number of original disks in $\mathcal{P}$, as well as the new disks that can be added inside C, is upper bounded by ℓ^2, which is a constant. Therefore, an optimal solution to $\Pi(C)$ can be found by solving a system of polynomial equations. Let $OPT_{i,j}$ denote the sum of the optimal values for the subproblems $\Pi(C)$, over all grid cells C in $G_{i,j}$.

Let $\mathcal{P}(C)$ denote the packing of the original disks that are completely contained in the cell C, or partially intersect with C. Recall that C is a square of size $\ell \times \ell$, and since $\mathcal{P}(C)$ is a packing, $|\mathcal{P}(C)| = \mathcal{O}(\ell^2)$. Furthermore, the number of new disks that can be added to C to obtain a new packing is also upper bounded by $p = \mathcal{O}(\ell^2)$. We first "guess" the number of new disks, by trying all possible values q between 1 and $p = \mathcal{O}(\ell^2)$. Now, we construct a system of polynomial equations with $2q$ variables and $q(|\mathcal{P}| + 4)$ equations, as follows.

For every new disk D_i for $1 \leq i \leq q$, we have two variables corresponding to the x and y coordinates of its center in the new packing. For every new disk D_i, we also add 4 linear equations that restrict the center to lie at a horizontal/vertical distance of at least 1 from the perimeter of the cell, so that the disk D_i lies completely within the cell C. Finally, for every disk D'_j in the original packing $\mathcal{P}$, we have an equation that enforces that the distance between the center of D_i and that of D'_j must be at least 2. Now, we solve this system of $\mathcal{O}(\ell^2)$ variables and $\mathcal{O}(\ell^4)$ equations in time $\mathcal{O}(\ell)^{\mathcal{O}(\ell^2)}$ time, using Proposition 4.

Note that since the diameter of a unit disk is 2, by an averaging argument, there exists an index $1 \leq i \leq \ell$, such that $OPT_{i,j} \geq \left(1 - \frac{1}{\ell}\right)^2 \cdot \mathsf{OPT}$. This is because, there exists an index i such that at most $\frac{1}{\ell}$ disks from $\mathcal{S}^*$ intersect the vertical lines $x = a\ell + i$ for integers a. Then, for this value of i, there exists an index j, such that at most $\frac{1}{\ell}$ fraction of the disks that are completely contained within the lines $x = a\ell + i$ intersect horizontal lines $b\ell + j$ for integers b. We direct the reader to [18] for a formal argument of this type.

Therefore, for every $1 \leq i, j \leq 2\ell$, and for every grid cell C in $G_{i,j}$, we solve the subproblem $\Pi(C)$, and return the best solution. Note that if we are looking for an $(1-\varepsilon)$-approximation to the number of newly added disks, then $(1-\varepsilon) \leq \left(1 - \frac{1}{\ell}\right)^2 \leq 1 - \frac{1}{\ell}$ That is, $\ell = 1/\varepsilon$. Thus, the running time of this algorithm is $\left(\frac{1}{\varepsilon}\right)^{\mathcal{O}(1/\varepsilon^2)} \cdot |I|^{\mathcal{O}(1)}$. ◀

Now we construct an algorithm for Max Disk Repacking in Theorem 3.

▶ **Theorem 3.** *For any $0 < \varepsilon < 1$, there exists an algorithm that, given an instance $(\mathcal{P}, R, h)$ of Max Disk Repacking, returns a packing $\mathcal{P}^*$ into R with at least $n + (1-\varepsilon) \cdot \mathsf{OPT}_h$ disks in time $\left(\frac{h+1}{\varepsilon}\right)^{\mathcal{O}(h/\varepsilon + 1/\varepsilon^2)} \cdot |I|^{\mathcal{O}(1)}$, where OPT_h is the maximum number of disks that can be added to the input packing if we can relocate at most h disks.*

Proof. Let $0 < \varepsilon < 1$. Consider an instance $(\mathcal{P}, R, h)$ of Max Disk Repacking. We find the maximum nonnegative integer $k \leq 10h/\varepsilon$ such that $(\mathcal{P}, R, h, k)$ is a yes-instance of Disk Repacking using the algorithm from Theorem 2. This can be done in $\left(\frac{h+1}{\varepsilon}\right)^{\mathcal{O}(h/\varepsilon)} \cdot |I|^{\mathcal{O}(1)}$ time. Next, we run the algorithm from Lemma 35 for (G, R) for $\varepsilon' = \frac{1}{2}\varepsilon$, i.e., assuming that relocations of disks are not allowed. The algorithm runs in $\left(\frac{1}{\varepsilon}\right)^{\mathcal{O}(1/\varepsilon^2)} \cdot |I|^{\mathcal{O}(1)}$ time and returns a solution of size k'. We set $k^* = \max\{k, k'\}$. We claim that $(1-\varepsilon)\mathsf{OPT}_h \leq k^* \leq \mathsf{OPT}_h$. The second inequality is trivial. To show that $(1-\varepsilon)\mathsf{OPT}_h \leq k^*$, we consider two cases.

Suppose that $\mathsf{OPT}_h \leq 10h/\varepsilon$. Then $\mathsf{OPT}_h = k$ as the algorithm from Theorem 2 is exact and $(1-\varepsilon)\mathsf{OPT}_h \leq \mathsf{OPT}_h = k \leq k^*$.

Assume that $\mathsf{OPT}_h > 10h/\varepsilon$. Let $\mathcal{S}$ be the set of added disks in an optimum solution for $(\mathcal{P}, R, h)$ and let $\mathcal{L} \subseteq \mathcal{P}$ be the set of relocated disks. Denote by OPT' the maximum number of disks that can be added to $\mathcal{P}$ without relocations. Observe that every disk in $\mathcal{L}$ intersects at most 5 disks of $\mathcal{S}$. Therefore, $\mathsf{OPT}' \geq |\mathcal{S}| - 5|\mathcal{L}| \geq \mathsf{OPT}_h - 5h$. By Lemma 35, $(1-\varepsilon/2)\mathsf{OPT}' \leq k'$. We obtain that $(1-\varepsilon/2)(\mathsf{OPT}_h - 5h) \leq k' \leq k^*$. Because $\mathsf{OPT}_h > 10h/\varepsilon$, $k^* \geq (1-\varepsilon/2)(\mathsf{OPT}_h - \varepsilon\mathsf{OPT}_h/2) = (1-\varepsilon/2)^2\mathsf{OPT}_h \geq (1-\varepsilon)\mathsf{OPT}_h$. This proves the claim.

We conclude that k^* is the required approximation of OPT_h. To conclude the proof, note that the algorithms from Theorem 2 and Lemma 35 can be adapted to return solutions, that is, the sets of added and relocated disks. ◀

6 Conclusion and open questions

We have shown in Theorem 1 that Disk Repacking problem is NP-hard even if $h = 0$. On the other hand, by Theorem 2, Disk Repacking is FPT when parameterized by k and h. Both theorems naturally lead to the question about parameterization by k only. The difficulty here is that even for adding one disk, one has to relocate many disks. Already for $k = 1$, we do not know, whether the problem is in P or is NP-hard.

Another natural question stemming from Theorem 2 is about kernelization of Disk Repacking. Does Disk Repacking admit a polynomial kernel with parameters k and h? (We refer to books [10, 13] for an introduction to kernelization).

Finally, approximation of Disk Repacking is an interesting research direction. In Theorem 3 we demonstrated that our FPT algorithm can be used to construct an FPT-AS with respect to h for Max Disk Repacking. We leave open the question about polynomial approximation. Another open question concerns the approximability of the minimum number of relocations h for a given k. Already for $k = 1$ finding a good approximation of h is a challenging problem.

References

1 Mikkel Abrahamsen, Tillmann Miltzow, and Nadja Seiferth. Framework for er-completeness of two-dimensional packing problems. In *61st IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1014–1021. IEEE, 2020. doi:10.1109/FOCS46700.2020.00098.

2 Pradeesha Ashok, Sudeshna Kolay, Syed Mohammad Meesum, and Saket Saurabh. Parameterized complexity of strip packing and minimum volume packing. *Theor. Comput. Sci.*, 661:56–64, 2017. doi:10.1016/j.tcs.2016.11.034.

3 Brenda S. Baker. Approximation algorithms for np-complete problems on planar graphs. *J. ACM*, 41(1):153–180, 1994. doi:10.1145/174644.174650.

4 Nikhil Bansal and Arindam Khan. Improved approximation algorithm for two-dimensional bin packing. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 13–25. SIAM, 2014. doi:10.1137/1.9781611973402.2.

5 Saugata Basu, Richard Pollack, and Marie-Françoise Roy. *Algorithms in Real Algebraic Geometry*. Springer, Berlin, Heidelberg, 2009.

6 Ignacio Castillo, Frank J Kampas, and János D Pintér. Solving circle packing problems by global optimization: numerical results and industrial applications. *European Journal of Operational Research*, 191(3):786–802, 2008.

7 Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Comput. Sci. Rev.*, 24:63–79, 2017. doi:10.1016/j.cosrev.2016.12.001.

8 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2006.

9 Hallard T Croft, Kenneth Falconer, and Richard K Guy. *Unsolved problems in geometry: unsolved problems in intuitive mathematics*, volume 2. Springer Science & Business Media, 2012.

10 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.

11 Erik D. Demaine, Sándor P. Fekete, and Robert J. Lang. Circle packing for origami design is hard. *CoRR*, abs/1008.1224, 2010. arXiv:1008.1224.

12 Sándor P. Fekete, Sebastian Morr, and Christian Scheffer. Split packing: Algorithms for packing circles with optimal worst-case density. *Discret. Comput. Geom.*, 61(3):562–594, 2019. doi:10.1007/s00454-018-0020-2.

13 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization*. Cambridge University Press, Cambridge, 2019. Theory of parameterized preprocessing.

14 Waldo Gálvez, Fabrizio Grandoni, Salvatore Ingala, Sandy Heydrich, Arindam Khan, and Andreas Wiese. Approximating geometric knapsack via l-packings. *ACM Trans. Algorithms*, 17(4):33:1–33:67, 2021. doi:10.1145/3473713.

15 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

16 Michael Goldberg. The packing of equal circles in a square. *Mathematics Magazine*, 43(1):24–30, 1970. doi:10.1080/0025570X.1970.11975991.

17 Rolf Harren, Klaus Jansen, Lars Prädel, and Rob van Stee. A $(5/3 + \epsilon)$-approximation for strip packing. *Comput. Geom.*, 47(2):248–267, 2014. doi:10.1016/j.comgeo.2013.08.008.

18 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.

19 Klaus Jansen and Malin Rau. Closing the gap for pseudo-polynomial strip packing. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *27th Annual European Symposium on Algorithms, ESA 2019, September 9-11, 2019, Munich/Garching, Germany*, volume 144 of *LIPIcs*, pages 62:1–62:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ESA.2019.62.

20 Johannes Kepler. *Strena seu de nive sexangula*. Frankfurt: Godefrid Tampach, 1611.

21 Marco Locatelli and Ulrich Raber. Packing equal circles in a square: a deterministic global optimization approach. *Discrete Applied Mathematics*, 122(1-3):139–166, 2002.

22 Costas D Maranas, Christodoulos A Floudas, and Panos M Pardalos. New results in the packing of equal circles in a square. *Discrete Mathematics*, 142(1-3):287–293, 1995.

23 Moni Naor, Leonard J. Schulman, and Aravind Srinivasan. Splitters and near-optimal derandomization. In *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, USA, 23-25 October 1995*, pages 182–191, 1995.

24 Kari J Nurmela et al. More optimal packings of equal circles in a square. *Discrete & Computational Geometry*, 22(3):439–457, 1999.

25 Kari J Nurmela and Patric RJ Östergård. Packing up to 50 equal circles in a square. *Discrete & Computational Geometry*, 18(1):111–120, 1997.

26 J Schaer. The densest packing of 9 circles in a square. *Canadian Mathematical Bulletin*, 8(3):273–277, 1965.

27 E Specht. The best known packings of equal circles in a square (up to N= 10000), 2015. URL: http://hydra.nat.uni-magdeburg.de/packing/csq/csq.html.

28 Péter Gábor Szabó, Mihaly Csaba Markót, Tibor Csendes, Eckard Specht, Leocadio G Casado, and Inmaculada García. *New approaches to circle packing in a square: with program codes*, volume 6. Springer Science & Business Media, 2007.

29 L Fejes Tóth. *Lagerungen in der Ebene auf der Kugel und im Raum*. Springer, 1953.