Worst-Case Optimal Covering of Rectangles by Disks

Sándor P. Fekete
Department of Computer Science, TU Braunschweig, Germany
s.fekete@tu-bs.de

Utkarsh Gupta
Department of Computer Science & Engineering, IIT Bombay, India
utkarshgupta149@gmail.com

Phillip Keldenich
Department of Computer Science, TU Braunschweig, Germany
p.keldenich@tu-bs.de

Christian Scheffer
Department of Computer Science, TU Braunschweig, Germany
scheffer@ibr.cs.tu-bs.de

Sahil Shah
Department of Computer Science & Engineering, IIT Bombay, India
sahilshah00199@gmail.com

Abstract

We provide the solution for a fundamental problem of geometric optimization by giving a complete characterization of worst-case optimal disk coverings of rectangles: For any $\lambda \geq 1$, the critical covering area $A^*(\lambda)$ is the minimum value for which any set of disks with total area at least $A^*(\lambda)$ can cover a rectangle of dimensions $\lambda \times 1$. We show that there is a threshold value $\lambda_2 = \sqrt{\sqrt{7}/2 - 1/4} \approx 1.035797\ldots$, such that for $\lambda < \lambda_2$ the critical covering area $A^*(\lambda)$ is $A^*(\lambda) = 3\pi\left(\frac{\lambda^2}{16} + \frac{5}{32} + \frac{9}{256\lambda^2}\right)$, and for $\lambda \geq \lambda_2$, the critical area is $A^*(\lambda) = \pi(\lambda^2 + 2)/4$; these values are tight. For the special case $\lambda = 1$, i.e., for covering a unit square, the critical covering area is $\frac{195\pi}{256} \approx 2.39301\ldots$. The proof uses a careful combination of manual and automatic analysis, demonstrating the power of the employed interval arithmetic technique.

2012 ACM Subject Classification Theory of computation → Packing and covering problems; Theory of computation → Computational geometry

Keywords and phrases Disk covering, critical density, covering coefficient, tight worst-case bound, interval arithmetic, approximation

Digital Object Identifier 10.4230/LIPIcs.SoCG.2020.42

Related Version A full version of this paper can be found at `https://arxiv.org/abs/2003.08236` [18].

Supplementary Material The code of the automatic prover can be found at `https://github.com/phillip-keldenich/circlecover`. Furthermore, there is a video contribution [20], video available at `https://www.ibr.cs.tu-bs.de/users/fekete/Videos/Cover_full.mp4`, illustrating the algorithm and proof presented in this paper.

Acknowledgements We thank Sebastian Morr and Arne Schmidt for helpful discussions. Major parts of the research by Utkarsh Gupta and Sahil Shah were carried out during a stay at TU Braunschweig.

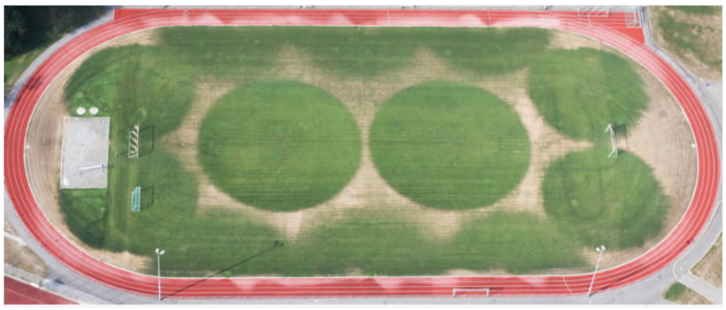

Figure 1 An incomplete covering of a rectangle by disks: Sprinklers on a soccer field during a drought. (Source: dpa [16].)

1 Introduction

Given a collection of (not necessarily equal) disks, is it possible to arrange them so that they completely cover a given region, such as a square or a rectangle? Covering problems of this type are of fundamental theoretical interest, but also have a variety of different applications, most notably in sensor networks, communication networks, wireless communication, surveillance, robotics, and even gardening and sports facility management, as shown in Fig. 1.

If the total area of the disks is small, it is clear that completely covering the region is impossible. On the other hand, if the total disk area is sufficiently large, finding a covering seems easy; however, for rectangles with large aspect ratio, a major fraction of the covering disks may be useless, so a relatively large total disk area may be required. The same issue is of clear importance for applications: What fraction of the total cost of disks can be put to efficient use for covering? This motivates the question of characterizing a critical threshold: For any given λ, find the minimum value $A^*(\lambda)$ for which any collection of disks with total area at least $A^*(\lambda)$ can cover a rectangle of dimensions $\lambda \times 1$. What is the critical covering area of $\lambda \times 1$ rectangles? In this paper we establish a complete and tight characterization.

1.1 Related Work

Like many other packing and covering problems, disk covering is typically quite difficult, compounded by the geometric complications of dealing with irrational coordinates that arise when arranging circular objects. This is reflected by the limitations of provably optimal results for the largest disk, square or triangle that can be covered by n unit disks, and hence, the "thinnest" disk covering, i.e., a covering of optimal density. As early as 1915, Neville [37] computed the optimal arrangement for covering a disk by five unit disks, but reported a wrong optimal value; much later, Bezdek[6, 7] gave the correct value for $n = 5, 6$. As recently as 2005, Fejes Tóth [45] established optimal values for $n = 8, 9, 10$. The question of incomplete coverings was raised in 2008 by Connelly, who asked how one should place n small disks of radius r to cover the largest possible area of a disk of radius $R > r$. Szalkai [44] gave an optimal solution for $n = 3$. For covering rectangles by n unit disks, Heppes and Mellissen [28] gave optimal solutions for $n \leq 5$; Melissen and Schuur [34] extended this for $n = 6, 7$. See Friedman [25] for the best known solutions for $n \leq 12$. Covering equilateral triangles by n unit disks has also been studied. Melissen [33] gave optimality results for $n \leq 10$, and conjectures for $n \leq 18$; the difficulty of these seemingly small problems is illustrated by the fact that Nurmela [38] gave conjectured optimal solutions for $n \leq 36$, improving the conjectured optimal covering for $n = 13$ of Melissen. Carmi et al. [11] considered algorithms for covering point sets by unit disks at fixed locations. There are numerous other related problems and results; for relevant surveys, see Fejes Tóth [17] (Section 8), Fejes Tóth [46] (Chapter 2), Brass et al. [10] (Chapter 2) and the book by Böröczky [9].

Even less is known for covering by non-uniform disks, with most previous research focusing on algorithmic aspects. Alt et al. [3] gave algorithmic results for minimum-cost covering of point sets by disks, where the cost function is $\sum_j r_j^\alpha$ for some $\alpha > 1$, which includes the case of total disk area for $\alpha = 2$. Agnetis et al. [2] discussed covering a line segment with variable radius disks. Abu-Affash et al. [1] studied covering a polygon minimizing the sum of areas; for recent improvements, see Bhowmick et al. [8]. Bánhelyi et al. [4] gave algorithmic results for the covering of polygons by variable disks with prescribed centers.

For relevant applications, we mention the survey by Huang and Tseng [29] for wireless sensor networks, the work by Johnson et al. [30] on covering density for sensor networks, the algorithmic results for placing a given number of base stations to cover a square [13] and a convex region by Das et al. [14]. For minimum-cost sensor coverage of planar regions, see Xu et al. [47]; for wireless communication coverage of a square, see Singh and Sengupta [42], and Palatinus and Bánhelyi [40] for the context of telecommunication networks.

The analogous question of *packing* unit disks into a square has also attracted attention. For $n = 13$, the optimal value for the densest square covering was only established in 2003 [24], while the optimal value for 14 unit disks is still unproven; densest packings of n disks in equilateral triangles are subject to a long-standing conjecture by Erdős and Oler from 1961 [39] that is still open for $n = 15$. Other mathematical work on densely packing relatively small numbers of identical disks includes [26, 32, 22, 23], and [41, 31, 27] for related experimental work. The best known solutions for packing equal disks into squares, triangles and other shapes are published on Specht's website `http://packomania.com` [43].

Establishing the critical packing density for (not necessarily equal) disks in a square was proposed by Demaine, Fekete, and Lang [15] and solved by Morr, Fekete and Scheffer [36, 21]. Using a recursive procedure for cutting the container into triangular pieces, they proved that the critical packing density of disks in a square is $\frac{\pi}{3+2\sqrt{2}} \approx 0.539$. The critical density for (not necessarily equal) disks in a disk was recently proven to be $1/2$ by Fekete, Keldenich and Scheffer [19]; see the video [5] for an overview and various animations. The critical packing density of (not necessarily equal) squares was established in 1967 by Moon and Moser [35], who used a shelf-packing approach to establish the value of $1/2$ for packing into a square.

1.2 Our Contribution

We show that there is a threshold value $\lambda_2 = \sqrt{\sqrt{7}/2 - 1/4} \approx 1.035797\ldots$, such that for $\lambda < \lambda_2$ the critical covering area $A^*(\lambda)$ is $A^*(\lambda) = 3\pi \left(\frac{\lambda^2}{16} + \frac{5}{32} + \frac{9}{256\lambda^2}\right)$, and for $\lambda \geq \lambda_2$, the critical area is $A^*(\lambda) = \pi(\lambda^2 + 2)/4$. These values are tight: For any λ, any collection of disks of total area $A^*(\lambda)$ can be arranged to cover a $\lambda \times 1$-rectangle, and for any $a(\lambda) < A^*(\lambda)$, there is a collection of disks of total area $a(\lambda)$ such that a $\lambda \times 1$-rectangle cannot be covered. (See Fig. 2 for a graph showing the (normalized) critical covering density, and Fig. 3 for examples of worst-case configurations.) The point $\lambda = \lambda_2$ is the unique real number greater than 1 for which the two bounds $3\pi \left(\frac{\lambda^2}{16} + \frac{5}{32} + \frac{9}{256\lambda^2}\right)$ and $\pi\frac{\lambda^2+2}{4}$ coincide; see Fig. 2. At this so-called *threshold value*, the worst case changes from three identical disks to two disks – the circumcircle $r_1^2 = \frac{\lambda^2+1}{4}$ and a disk $r_2^2 = \frac{1}{4}$; see Fig. 3. For the special case $\lambda = 1$, i.e., for covering a unit square, the critical covering area is $\frac{195\pi}{256} \approx 2.39301\ldots$.

The proof uses a careful combination of manual and automatic analysis, demonstrating the power of the employed interval arithmetic technique.

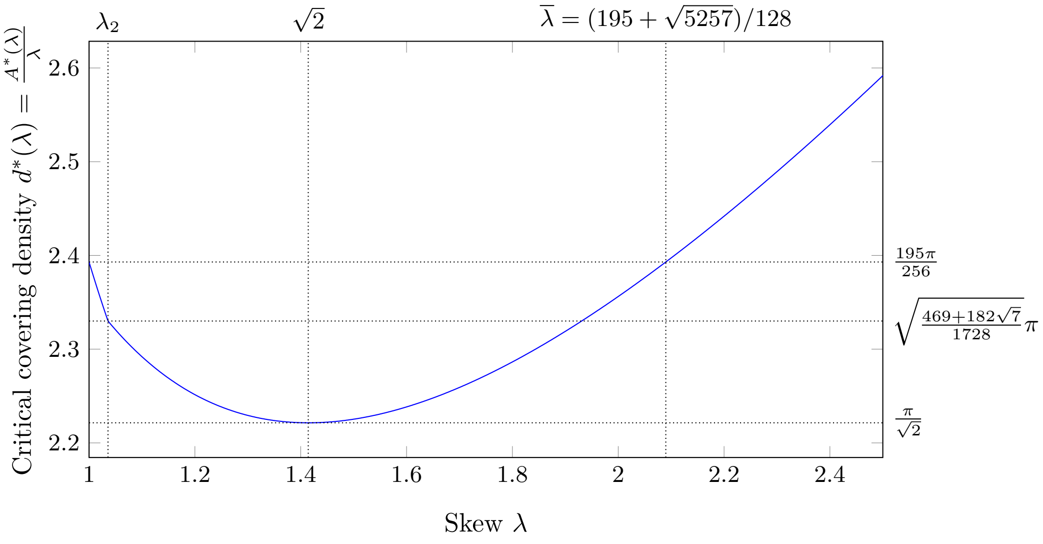

Figure 2 The critical covering density $d^*(\lambda)$ depending on λ and its values at the threshold value λ_2, the global minimum $\sqrt{2}$ and the skew $\bar{\lambda}$ at which the density becomes as bad as for the square.

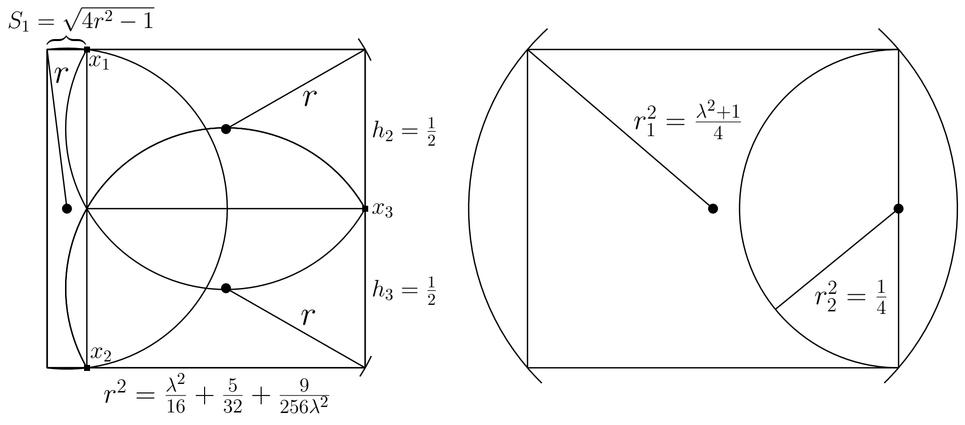

Figure 3 Worst-case configurations for small $\lambda \le \lambda_2$ (left) and for large skew $\lambda \ge \lambda_2$ (right). Shrinking r or r_1 by any $\varepsilon > 0$ in either configuration leads to an instance that cannot be covered.

2 Preliminaries

We are given a rectangular container $\mathcal{R}$, which we assume w.l.o.g. to have height 1 and some width $\lambda \ge 1$, which is called the *skew* of $\mathcal{R}$. For a collection $D = \{r_1, \dots, r_n\}$ of radii $r_1 \ge r_2 \ge \dots \ge r_n$, we want to decide whether there is a placement of n closed disks with radii $r_1, \dots, r_n$ on $\mathcal{R}$, such that every point $x \in \mathcal{R}$ is covered by at least one disk. Because we are only given radii and not center points, in a slight abuse of notation, we identify the disks with their radii and use r_i to refer to both the disk and the radius.

For any set D of disks, the *total disk area* is $A(D) := \pi \sum_{r \in D} r^2$. The *weight* of a disk of radius r is r^2, and $W(D) := \frac{A(D)}{\pi}$ is the *total weight* of D. For any rectangle $\mathcal{R}$, the *critical covering area* $A^*(\mathcal{R})$ of $\mathcal{R}$ is the minimum value for which any set D of disks with total area at least $A(D) \ge A^*(\mathcal{R})$ can cover $\mathcal{R}$. The *critical covering weight* of $\mathcal{R}$ is $W^*(\mathcal{R}) := \frac{A^*(\mathcal{R})}{\pi}$. For $\lambda \ge 1$, we define $A^*(\lambda) := A^*(\mathcal{R})$ and $W^*(\lambda) := W^*(\mathcal{R})$ for a $\lambda \times 1$ rectangle $\mathcal{R}$.

For a placement $\mathcal{P}$ of the disks in D fully covering some area A, the *covering coefficient* of $\mathcal{P}$ is the ratio $\frac{W(D)}{A}$. For $\lambda \geq 1$, the amount $E^*(\lambda) := \frac{W^*(\lambda)}{\lambda}$ of total disk weight per unit of rectangle area that is necessary for guaranteeing a possible covering is the *(critical) covering coefficient* of λ. Analogously, $d^*(\lambda) := \frac{A^*(\lambda)}{\lambda}$ is the *(critical) covering density* of λ.

For proving our result, we use Greedy Splitting for partitioning a collection of disks into two parts whose weight differs by at most the weight of the smallest disk in the heavier part: After sorting the disks by decreasing radius, we start with two empty lists and continue to place the next disk in the list with smaller total weight.

3 High-Level Description

Now we present and describe our main result: a theorem that characterizes the worst case for covering rectangles with disks. This theorem gives a closed-form solution for the *critical covering area* $A^*(\lambda)$ for any $\lambda \geq 1$; in other words, for any given rectangle $\mathcal{R}$, we determine the total disk area that is (1) sometimes necessary and (2) always sufficient to cover $\mathcal{R}$.

▶ **Theorem 1.** *Let $\lambda \geq 1$ and let $\mathcal{R}$ be a rectangle of dimensions $\lambda \times 1$. Let*

$$\lambda_2 = \sqrt{\frac{\sqrt{7}}{2} - \frac{1}{4}} \approx 1.035797\ldots, \text{ and } A^*(\lambda) = \begin{cases} 3\pi\left(\frac{\lambda^2}{16} + \frac{5}{32} + \frac{9}{256\lambda^2}\right), & \text{if } \lambda < \lambda_2, \\ \pi\frac{\lambda^2+2}{4}, & \text{otherwise.} \end{cases}$$

(1) *For any $a < A^*(\lambda)$, there is a set D^- of disks with $A(D^-) = a$ that cannot cover $\mathcal{R}$.*
(2) *Let $D = \{r_1, \ldots, r_n\} \subset \mathbb{R}$, $r_1 \geq r_2 \geq \ldots \geq r_n > 0$ be any collection of disks identified by their radii. If $A(D) \geq A^*(\lambda)$, then D can cover $\mathcal{R}$.*

The critical covering area does not depend linearly on the area λ of the rectangle; it also depends on the rectangle's skew. Fig. 2 shows a plot of the dependency of the covering density $d(\lambda)$ on λ. In the following, to simplify notation, we factor out π if possible; instead of working with the areas $A(D)$ or $A^*(\lambda)$ of the disks, we use their *weight*, i.e., their area divided by π. Similarly, we work with the covering coefficient $E^*(\lambda)$ instead of the density $d^*(\lambda)$; a lower covering coefficient corresponds to a more efficient covering.

As shown in Fig. 2, the critical covering coefficient $E^*(\lambda)$ is monotonically decreasing from $\lambda = 1$ to $\sqrt{2}$ and monotonically increasing for $\lambda > \sqrt{2}$. For a square, $E^*(1) = \frac{195}{256}$; the point $\lambda > 1$ for which the covering coefficient becomes as bad as for the square is $\overline{\lambda} := \frac{195+\sqrt{5257}}{128} \approx 2.08988\ldots$; for all $\lambda \leq \overline{\lambda}$, the covering coefficient is at most $\frac{195}{256}$.

3.1 Proof Components

The proof of Theorem 1 uses a number of components. First is a lemma that describes the worst-case configurations and shows tightness, i.e., claim (1), of Theorem 1 for all λ.

▶ **Lemma 2.** *Let $\lambda \geq 1$ and let $\mathcal{R}$ be a rectangle of dimensions $\lambda \times 1$. (1) Two disks of weight $r_1^2 = \frac{\lambda^2+1}{4}$ and $r_2^2 = \frac{1}{4}$ suffice to cover $\mathcal{R}$. (2) For any $\varepsilon > 0$, two disks of weight $r_1^2 - \varepsilon$ and r_2^2 do not suffice to cover $\mathcal{R}$. (3) Three identical disks of weight $r^2 = \frac{\lambda^2}{16} + \frac{5}{32} + \frac{9}{256\lambda^2}$ suffice to cover a rectangle $\mathcal{R}$ of dimensions $\lambda \times 1$. (4) For $\lambda \leq \lambda_2$ and any $\varepsilon > 0$, three identical disks of weight $r_-^2 := r^2 - \varepsilon$ do not suffice to cover $\mathcal{R}$.*

For large λ, the critical covering coefficient $E^*(\lambda)$ of Theorem 1 becomes worse, as large disks cannot be used to cover the rectangle efficiently. If the weight of each disk is bounded by some $\sigma \geq r_1^2$, we provide the following lemma achieving a better covering coefficient $E(\sigma)$ with $E^*(\overline{\lambda}) \leq E(\sigma) \leq E^*(\lambda)$. This coefficient is independent of the skew of $\mathcal{R}$.

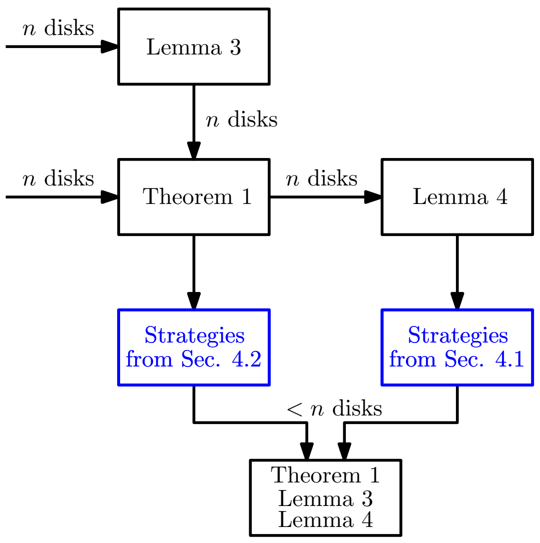

Figure 4 The inductive structure of the proof; the blue parts are computer-aided.

▶ **Lemma 3.** *Let $\hat{\sigma} := \frac{195\sqrt{5257}}{16384} \approx 0.8629$. Let $\sigma \geq \hat{\sigma}$ and $E(\sigma) := \frac{1}{2}\sqrt{\sqrt{\sigma^2+1}+1}$. Let $\lambda \geq 1$ and $D = \{r_1, \ldots, r_n\}$ be any collection of disks with $\sigma \geq r_1^2 \geq \ldots \geq r_n^2$ and $W(D) = \sum_{i=1}^{n} r_i^2 \geq E(\sigma)\lambda$. Then D can cover a rectangle $\mathcal{R}$ of dimensions $\lambda \times 1$.*

Note that $E(\hat{\sigma}) = \frac{195}{256}$, i.e. the best covering coefficient established by Lemma 3, coinciding with the critical covering coefficient of the square established by Theorem 1. Thus, we can cover any rectangle with covering coefficient $\frac{195}{256}$ if the largest disk satisfies $r_1^2 \leq \hat{\sigma}$.

The final component is the following Lemma 4, which also gives a better covering coefficient if the size of the largest disk is bounded. The bound required for Lemma 4 is smaller than for Lemma 3; in return, the covering coefficient that Lemma 4 yields is better. Note that the result of Lemma 4 is not tight.

▶ **Lemma 4.** *Let $\lambda \geq 1$ and let $\mathcal{R}$ be a rectangle of dimensions $\lambda \times 1$. Let $D = \{r_1, \ldots, r_n\}$, $0.375 \geq r_1 \geq \ldots \geq r_n > 0$ be a collection of disks. If $W(D) \geq 0.61\lambda$, or equivalently $A(D) \geq 0.61\pi\lambda \approx 1.9164\lambda$, then D suffices to cover $\mathcal{R}$.*

3.2 Proof Overview

The proofs of Theorem 1 and Lemmas 3 and 4 work by induction on the number of disks. For proving Lemma 3 for n disks, we use Theorem 1 for n disks. For proving Theorem 1 for n disks, we use Lemma 4 for n disks; Lemma 3 is only used for fewer than n disks; see Fig. 4. For proving Lemma 4 for n disks, we only use Theorem 1 and Lemma 3 for fewer than n disks. Therefore, there are no cyclic dependencies in our argument; however, we have to perform the induction for Theorem 1 and Lemmas 3 and 4 simultaneously.

Routines. The proofs of Theorem 1 and Lemma 4 are constructive; they are based on an efficient recursive algorithm that uses a set of simple *routines*. We go through the list of routines in some fixed order. For each routine, we check a sufficient criterion for the routine to work. We call these criteria *success criteria*. They only depend on the total available weight and a constant number of largest disks. If we cannot guarantee that a routine works by its success criterion, we simply disregard the routine; this means that our algorithm does not have to backtrack. We prove that, regardless of the distribution of the disks' weight, at least one success criterion is met, implying that we can always apply at least one routine. The number of routines and thus success criteria is large; this is where the need for automatic assistance comes from.

Recursion. Typical routines are recursive; they consist of splitting the collection of disks into smaller parts, splitting the rectangle accordingly, and recursing, or recursing after fixing the position of a constant number of large disks.

In the entire remaining proof, the criterion we use to guarantee that recursion works is as follows. Given a collection $D' \subsetneq D$ and a rectangular region $\mathcal{R}' \subsetneq \mathcal{R}$, we check whether the preconditions of Theorem 1 or Lemma 3 or 4 are met after appropriately scaling and rotating $\mathcal{R}'$ and the disks. Note that, due to the scaling, the radius bounds of Lemmas 3 and 4 depend on the length of the shorter side of $\mathcal{R}'$. In some cases where we apply recursion, we have more weight than necessary to satisfy the weight requirement for recursion according to Lemma 3 or 4, but these lemmas cannot be applied due to the radius bound. In that case, we also check whether we can apply Lemma 3 or 4 after increasing the length of the shorter side of $\mathcal{R}'$ as far as the disk weight allows. This excludes the case that we cannot recurse on $\mathcal{R}'$ due to the radius bound, but there is some $\mathcal{R}'' \supset \mathcal{R}'$ on which we could recurse.

3.3 Interval Arithmetic

We use interval arithmetic to prove that there always is a successful routine. In interval arithmetic, operations like addition, multiplication or taking a square root are performed on intervals $[a, b] \subset \mathbb{R}$ instead of numbers. Arithmetic operations on intervals are derived from their real counterparts as follows. The result of an operation $\circ$ in interval arithmetic is

$$[a_1, b_1] \circ [a_2, b_2] := \left[\min_{x_1 \in [a_1,b_1], x_2 \in [a_2,b_2]} x_1 \circ x_2, \max_{x_1 \in [a_1,b_1], x_2 \in [a_2,b_2]} x_1 \circ x_2 \right].$$

Thus, the result of an operation is the smallest interval that contains all possible results of $x \circ y$ for $x \in [a_1, b_1], y \in [a_2, b_2]$. Unary operations are defined analogously. For square roots, division or other operations that are not defined on all of $\mathbb{R}$, a result is undefined iff the input interval(s) contain values for which the real counterpart of the operation is undefined.

Truth values. In interval arithmetic, inequalities such as $[a_1, b_1] \leq [a_2, b_2]$ can have three possible truth values. An inequality can be *definitely true*; this means that the inequality holds for any value of $x \in [a_1, b_1], y \in [a_2, b_2]$. In the example $[a_1, b_1] \leq [a_2, b_2]$, this is the case if $b_1 \leq a_2$. An inequality can be *indeterminate*; this means that there are some values $x, x' \in [a_1, b_1], y, y' \in [a_2, b_2]$ such that the inequality holds for x, y and does not hold for x', y'. In the example $[a_1, b_1] \leq [a_2, b_2]$, this is the case if $a_1 \leq b_2$ and $b_1 > a_2$. Otherwise, an inequality is *definitely false*. An inequality that is either *definitely true* or *indeterminate* is called *possibly true*; an inequality that is either *indeterminate* or *definitely false* is called *possibly false*. These truth values can also be interpreted as intervals $[0, 0], [0, 1], [1, 1]$.

Using interval arithmetic. We apply interval arithmetic in our proof as follows. Recall that for each routine, we have a *success criterion*. These criteria only consider $\lambda \geq 1$ and the largest $k \in \mathcal{O}(1)$ disks $r_1 \geq \cdots \geq r_k$ as well as the remaining weight $R_{k+1} := \sum_{i=k+1}^{n} r_i^2$, which can be computed from λ and $r_1, \ldots, r_k$, assuming w.l.o.g. that the total disk weight $W(D)$ is exactly $W^*(\lambda)$.

If we can manually perform induction base and induction step of our result for all $\lambda \geq \hat{\lambda}$ for some finite value $\hat{\lambda}$, we can also provide an upper bound $\hat{r}_1$ for r_1 such that all cases that remain to be considered (in our induction base and induction step) correspond to a point in the $(k+1)$-dimensional space Ψ given by

$$\lambda \in [1, \hat{\lambda}], r_1 \in [0, \hat{r}_1], r_2 \in [0, r_1], \dots, r_k \in [0, r_{k-1}], \sum_{i=1}^{k} r_i^2 \leq W^*(\lambda).$$

This is due to the fact that there is nothing to prove if r_1 can cover $\mathcal{R}$ on its own; r_1 can have no more than the total disk weight $W(D)$ and $r_k \leq \cdots \leq r_2 \leq r_1$. Furthermore, observe that the induction base is just a special case with $r_i = r_{i+1} = \cdots = 0$ for some $1 < i \leq k$.

This allows subdividing (a superset of) Ψ into a large finite number of *hypercuboids* by splitting the range of each of the variables $\lambda, r_1, \dots, r_k$ into a number of smaller intervals. For each hypercuboid, we then use interval arithmetic to verify that there is a routine whose success criterion is met. If we find such a routine, we have eliminated all points in that hypercuboid from further consideration. Hypercuboids for which this does not succeed are called *critical* and must be resolved manually; note that, in particular, hypercuboids containing (tight) worst-case configurations cannot be handled by interval arithmetic. The restriction to critical hypercuboids makes the overall analysis feasible, while a manual analysis of the entire space is impractical due to the large number of routines and variables.

Implementation. We implemented the subdivision outlined above and all success criteria of our routines using interval arithmetic[1]. Because most of our success criteria use the squared radii r_i^2 instead of the radii r_i, we use λ and r_i^2 instead of r_i as variables. Moreover, for efficiency reasons, instead of the simple grid-like subdivision outlined above, we use a search-tree-like subdivision strategy where we begin by subdividing the range of λ, continue by subdividing r_1^2, followed by r_2^2, and so on. Whenever a success criterion only needs the first $i < k$ disks, we can check this criterion farther up in the tree, thus potentially avoiding visits to large parts of the search tree; see Fig. 5 for a sketch of this procedure. Even with this pruning in place, the number of hypercuboids that we have to consider is still very large; this is a result of the fact that, depending on the claim at stake, we have 5 or even 8 dimensions. Therefore, we implemented the checks for our success criteria on a CUDA-capable GPU to perform them in a massively parallel fashion.

Moreover, to provide a finer subdivision where necessary, we run our search in several *generations* (our proof uses 11 generations). Each generation yields a set of critical hypercuboids that could not be handled automatically. After each generation, for each subinterval of λ, we collect all critical hypercuboids and merge those for which the r_1^2-subintervals are overlapping by taking the smallest hypercuboid containing all points in the merged hypercuboids. This procedure typically yields only 1-3 hypercuboids per subinterval of λ. The next generation is run on each of these, starting with the bounds given by these hypercuboids.

Numerical issues. When performing computations on a computer with limited-precision floating-point numbers instead of real numbers, there can be rounding errors, underflow errors and overflow errors. Our implementation of interval arithmetic performs all operations using appropriate rounding modes; this technique is also used by the implementation of interval

[1] The source code of the implementation is available online: `https://github.com/phillip-keldenich/circlecover` .

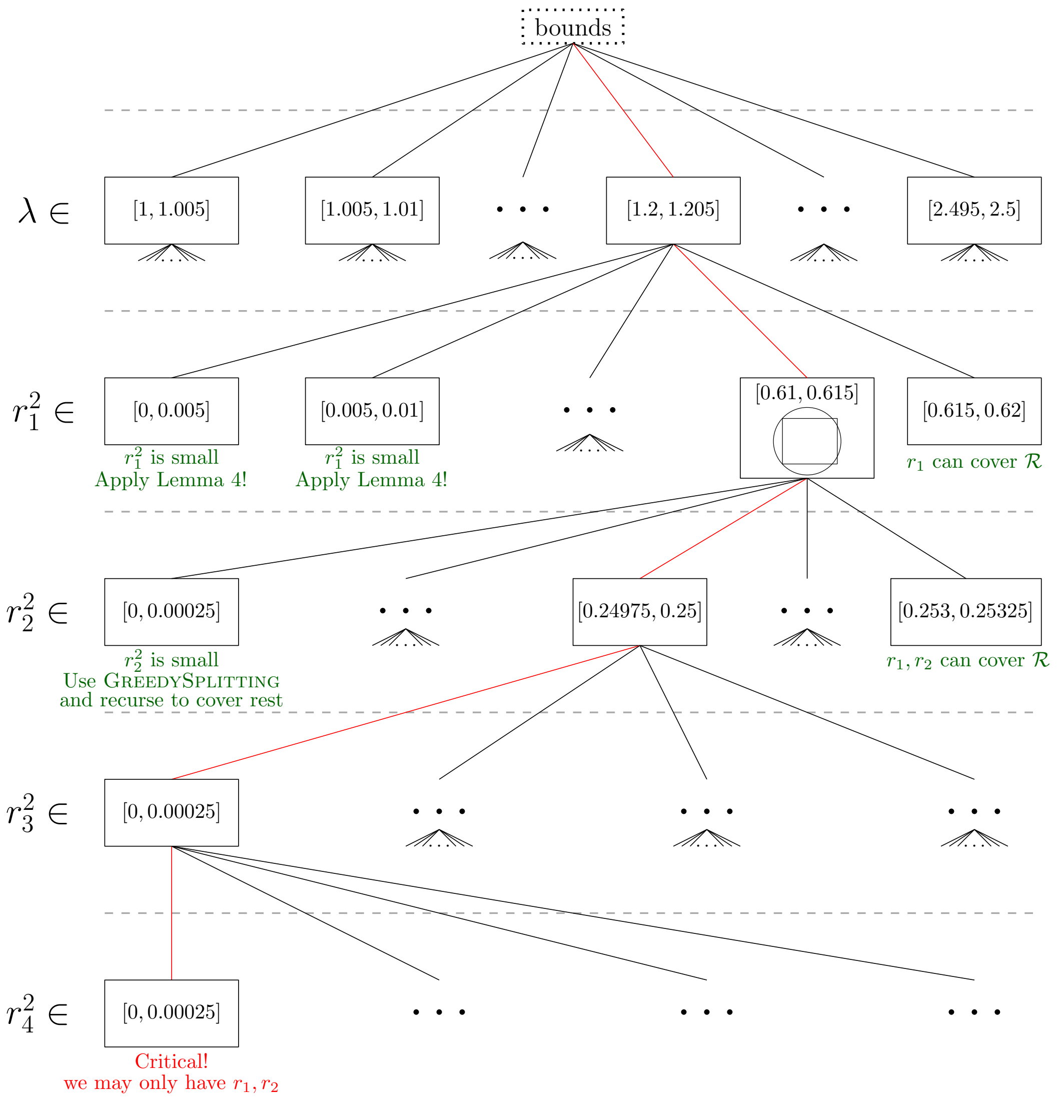

Figure 5 Sketch of our interval arithmetic-based search procedure. The red edges denote a path leading to a critical cuboid containing a tight two-disk worst-case configuration. Green text indicates that the children of the corresponding node do not have to be considered.

arithmetic in the well-known Computational Geometry Algorithms Library (CGAL) [12]. This means that any operation $\circ$ on two intervals A, B yields an interval $I \supseteq A \circ B$ to ensure that the result of any operation contains all values that are possible outcomes of $x \circ y$ for $x, y \in A, B$. This guarantees soundness of our results in the presence of numerical errors.

4 Proof Structure

In this section, we give an overview of the structure of the proofs of Theorem 1 and Lemmas 2, 3 and 4. For the proof of Lemma 2, we refer to the full version [18] of our paper. Lemma 3 is proven in Section 4.3 using a simple recursive algorithm; basically, we show that we can always split the disks using Greedy Splitting, split the rectangle accordingly, and recurse using Theorem 1. The proofs of Theorem 1 and Lemma 4 involve a larger number of routines and make use of an automatic prover based on interval arithmetic as described in Section 3.3.

4.1 Proof Structure for Lemma 4

Proving Lemma 4 means proving that, for any skew λ, any collection D of disks of radius $r_1 \leq 0.375$ and with total weight $W(D) = E\lambda$ suffices to cover $\mathcal{R}$, where $E = 0.61$ is the covering coefficient guaranteed by Lemma 4. We first reduce the number of cases that we have to consider in our induction base and induction step to a finite number. As described in Section 3.3, this requires handling the case of arbitrarily large skew λ. Finding a bound $\hat{\lambda}$ and reducing Lemma 4 for $\lambda \geq \hat{\lambda}$ to the case of $\lambda < \hat{\lambda}$ yields bounds for λ and $r_1, \ldots, r_k$ that allow a reduction to finitely many cases using interval arithmetic.

▶ **Lemma 5.** *Let $\hat{\lambda} = 2.5$. Given disks D according to the preconditions of Lemma 4 and $\lambda \geq \hat{\lambda}$, we can cover $\mathcal{R}$ using a simple recursive routine.*

Proof. The routine works as follows. We build a list of disks D_1 by adding disks in decreasing order of radius until $W(D_1) \geq E$. Due to the radius bound, this procedure always stops before all disks are used, i.e., $D_1 \subsetneq D$. Let $D_2 := D \setminus D_2$ be the remaining disks. We then place a vertical rectangular strip $\mathcal{R}_1$ of height 1 and width $\beta_{\mathcal{R}_1} := \frac{W(D_1)}{E} \geq 1$ at the left side of $\mathcal{R}$. By induction, we can recurse on $\mathcal{R}_1$ using Lemma 4 and the disks from D_1, because both side lengths are at least 1 and the efficiency we require is *exactly* E. Note that, due to adapting the width $\beta_{\mathcal{R}_1}$ according to the actual weight $W(D_1)$, we actually achieve an efficiency of E; in other words, there is no *waste* of disk weight. This means that we also require an efficiency of exactly E on the remaining rectangle $\mathcal{R}_2 := \mathcal{R} \setminus \mathcal{R}_1$. Therefore, provided that the largest disk in D_2 satisfies the size bound of Lemma 4, we can inductively apply Lemma 4 to $\mathcal{R}_2$ and D_2 and are done. This can be guaranteed by proving that the shorter side of $\mathcal{R}_2$ is at least 1 as well. We have $W(D_1) \leq E + r_1^2 \leq E + 0.375^2$ which implies $\beta_{\mathcal{R}_1} \leq 1 + \frac{0.375^2}{E} < 1.5$; therefore, $\lambda \geq 2.5$ ensures that the width of $\mathcal{R}_2$ is at least 1. ◀

As outlined in Section 3.3, the remainder of the proof of Lemma 4 is based on a list of simple covering routines and their success criteria. We prove that there always is a working routine in that list using an automatic prover based on interval arithmetic, as described in Section 3.3. This automatic prover considers the 8-dimensional space spanned by the variables λ and $r_1^2, \ldots, r_7^2$ and subdivides it into a total of more than 2^{46} hypercuboids in order to prove that there always is a working routine, i.e., no critical hypercuboids remain to be analyzed manually; this only works because the result of Lemma 4 is not tight.

In the following, we give a brief description of the routines that we use. Due to space constraints, for a detailed description of the routines, we refer to the full version of our paper [18].

Recursive splitting. Routines (S-I.1) and (S-I.2) work by splitting D into two parts, splitting $\mathcal{R}$ accordingly, and recursing on the two sub-rectangles. This split is either performed as balanced as possible using Greedy Splitting, or in an unbalanced manner; in the latter case, we choose an unbalanced split to accommodate large disks that violate the radius bound of Lemma 4 w.r.t. a rectangle of half the width of $\mathcal{R}$.

Building a strip. Routine (S-II.1) works by either covering the left or the bottom side of a rectangular strip $\mathcal{R}$; see Fig. 6. This strip uses a subset of the largest six disks and tries several configurations for placing the disks. The remaining area is covered by recursion.

Wall building. Routines (S-III.1) and (S-IV.1) are based on the idea of covering a rectangular strip of fixed length ℓ and variable width b with covering coefficient exactly E. We call this *wall building.* To achieve this covering coefficient, we stack disks of similar size on top

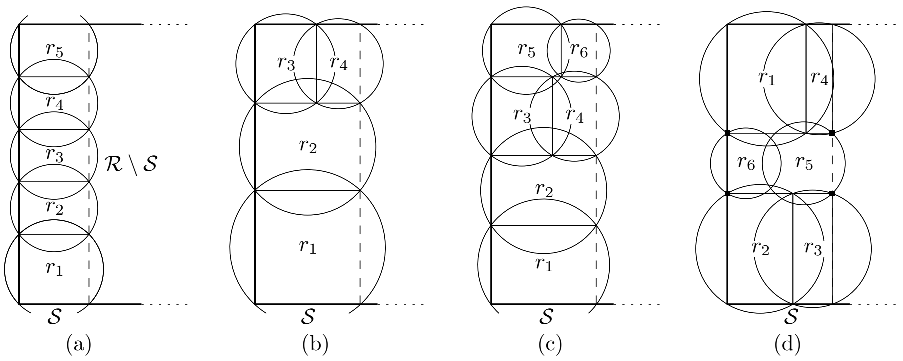

Figure 6 Some placements considered by Routine S-II.1 to build a vertical strip; horizontal strips are analogous. (a) Simply stacking a subset T of the six largest disks on top of each other. (b) Stacking r_1, r_2 on top of each other, and placing r_3, r_4 horizontally next to each other on top. (c) Same as (b), but with an additional row built from r_5, r_6. (d) Building two rows at the top and the bottom consisting of r_1, r_4 and r_2, r_3, and covering the remaining region by r_5, r_6. The points on the boundary defining the position of r_5 and r_6 are marked by squares. Note that r_5 and r_6 are not big enough to cover the entire rectangular area between the top and the bottom row.

of (or horizontally next to) each other; each disk placed in this way covers a rectangle of variable height, but width b. We provide sufficient conditions for this procedure to result in a successful covering of a strip of length ℓ. Routine (S-III.1) uses this idea to build a column of stacked disks at the left side of $\mathcal{R}$; see Fig. 7. Routine (S-IV.1) uses this idea by placing r_1 in the bottom-left corner of $\mathcal{R}$ and filling the area above r_1 with horizontal rows of disks; see Fig. 8. Intuitively speaking, these routines are necessary to handle cases in which there are large disks that interfere with recursion, but small disks, for which we do not know the weight distribution, significantly contribute to the total weight.

Using the two largest disks. Routine (S-V.1) places the two largest disks in diagonally opposite corners, each disk covering its inscribed square; see Fig. 9. The remaining area is subdivided into three rectangular regions; we cover these regions recursively, considering several ways to split the remaining disks.

Using the three largest disks. Routines (S-VI.1) and (S-VI.2) consider two different placements of the largest three disks as shown in Fig. 10.

Using the four largest disks. Routines (S-VII.1)–(S-VII.3) consider different placements of the four largest disks and recursion to cover $\mathcal{R}$; see Fig. 11.

Using the five largest disks. Routines (S-VIII.1) and (S-VIII.2) consider different placements of the five largest disks and recursion to cover $\mathcal{R}$; see Fig. 12.

Using the six largest disks. Routines (S-IX.1)–(S-IX.3) consider different placements of the six largest disks and recursion to cover $\mathcal{R}$; see Fig. 13.

Using the seven largest disks. Routines (S-X.1)–(S-X.8) consider different placements of the seven largest disks, together with recursion, to cover $\mathcal{R}$; see Figs. 14, 15 and 16.

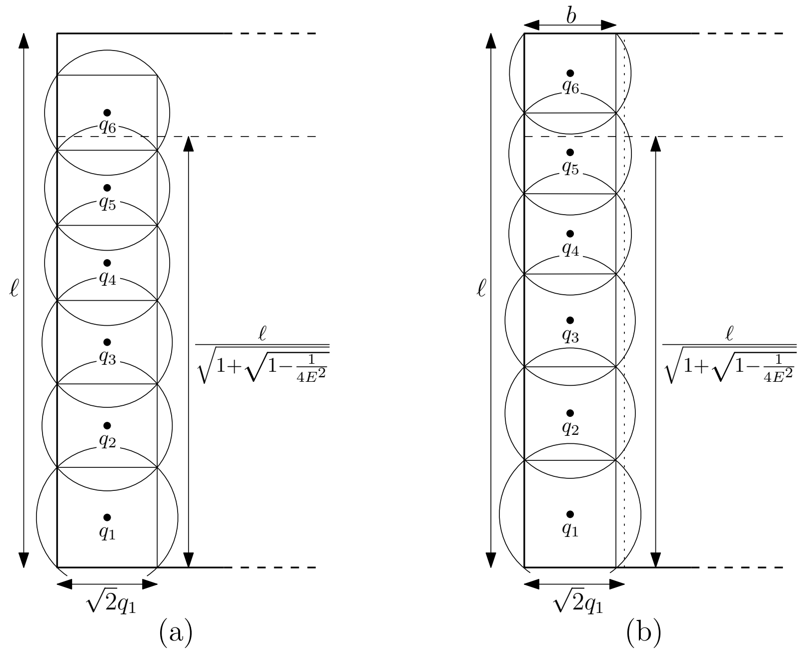

Figure 7 The wall-building procedure. (a) Using an initial guess of $b = \sqrt{2}q_1$ as width, where q_1 is the largest disk that we use, we stack disks until they exceed a certain fraction of the length ℓ. (b) We decrease b until the disks exactly cover a strip of length ℓ.

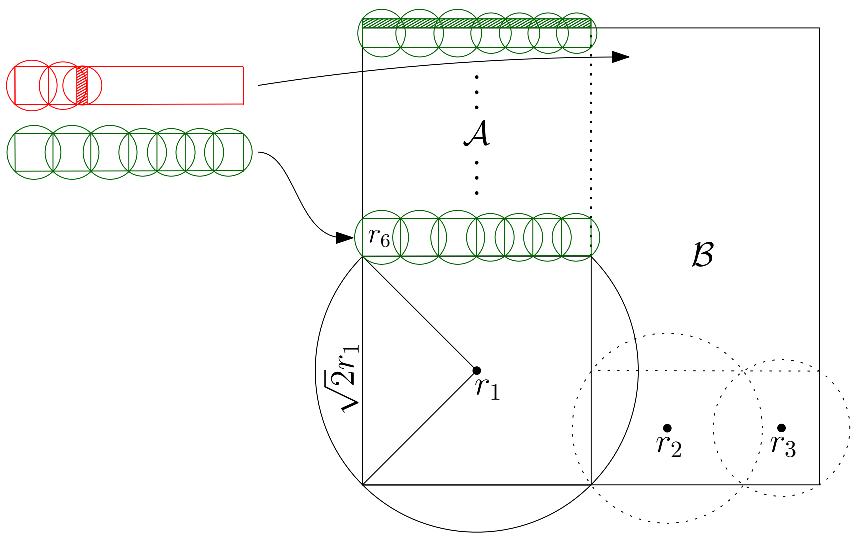

Figure 8 Routine S-IV.1 places r_1 in the bottom-left corner and tries to cover $\mathcal{A}$ using either recursion or wall building. In the latter case, whenever the disk radius drops too much while building a row of length $\ell = \sqrt{2}r_1$, we move the disks constituting this incomplete row to $\mathcal{B}$ (red). Otherwise, a complete row is built (green) and we continue with the next row. This process stops once the entire area $\mathcal{A}$ is covered, including some potential overhead (shaded green region). We compensate for the overhead by the area gained by placing r_1 covering a square. In case r_2 does not fit into $\mathcal{B}$ recursively, we try placing r_2, r_3 (or r_2, r_3, r_4) at the bottom of $\mathcal{B}$ (dotted outline).

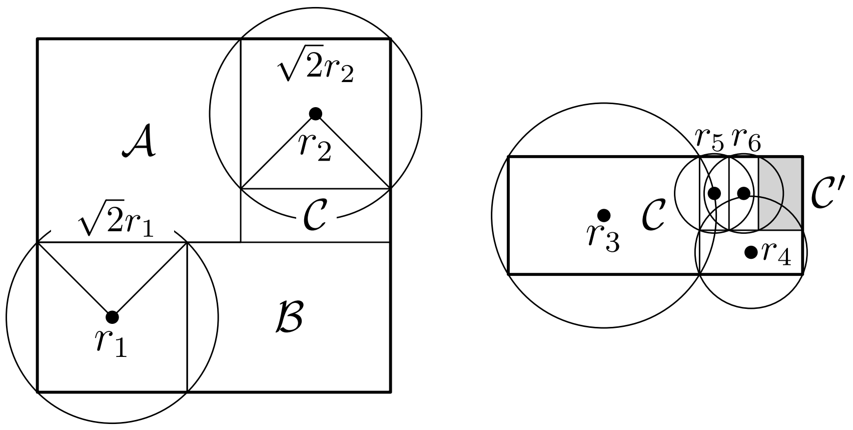

Figure 9 The routine S-V.1 places r_1 and r_2 in diagonally opposite corners, each covering a square. We cover three remaining rectangular areas $\mathcal{A}, \mathcal{B}, \mathcal{C}$ using the remaining disks (left). Regions $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$ are covered by recursion; we also consider using disks $r_3, \ldots, r_6$ to reduce $\mathcal{C}$ to $\mathcal{C}'$ (light gray) before recursing.

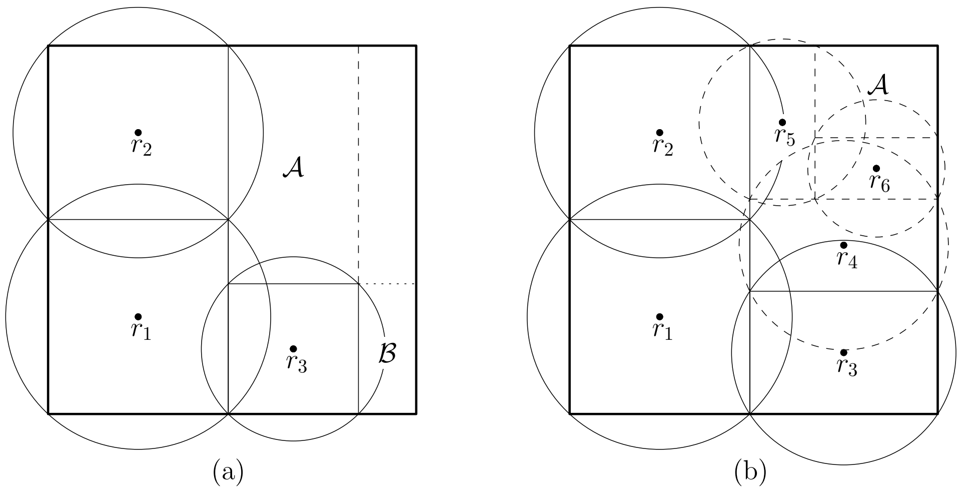

Figure 10 Two routines based on using the three largest disks. We use r_1 and r_2 to cover a vertical strip of height 1 and maximal width. (a) In Routine S-VI.1, we place r_3 to the right of r_1, covering its inscribed square at the lower left corner of the remaining rectangle; the remaining region can be subdivided into two rectangles $\mathcal{A}, \mathcal{B}$ in two ways (dashed and dotted line). (b) In Routine S-VI.2, we cover a horizontal strip of the remaining rectangle using r_3; we either recurse on the remaining rectangle directly or place some of the disks r_4, r_5, r_6 to cut off pieces of the longer side of the remaining rectangle (dashed outlines).

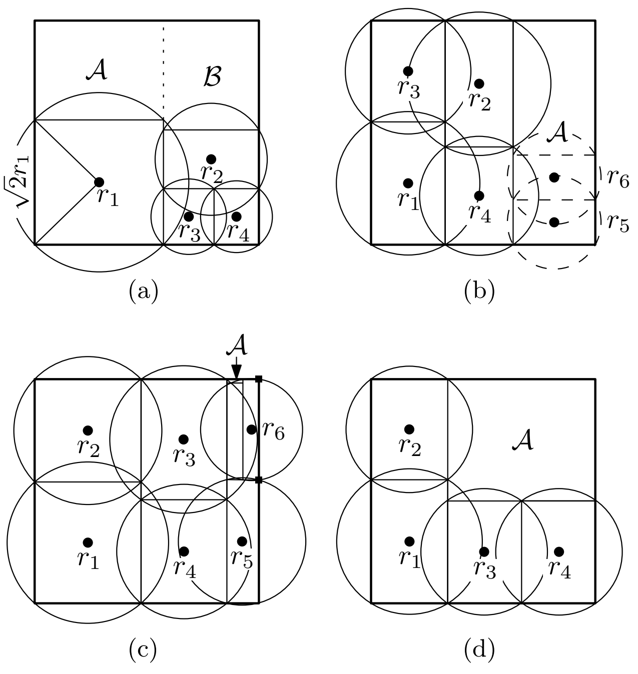

Figure 11 Covering routines that mainly rely on the four largest disks to cover $\mathcal{R}$. (a) In Routine S-VII.1, we place r_1 in the bottom-left corner, covering its inscribed square; use r_2, r_3 and r_4 to cover as much of the vertical strip remaining to the left of r_1, and recurse on $\mathcal{A}$ and $\mathcal{B}$. (b) In Routine S-VII.2, in the first case, we cover a rectangular strip using $r_1, \dots, r_4$. Either use recursion immediately on the remainder $\mathcal{A}$, or recurse after placing r_5 and possibly r_6 covering a rectangle at the bottom of $\mathcal{A}$. (c) In Routine S-VII.2, in the second case, we cover a rectangular strip using $r_1, \dots, r_4$ and place r_5 at the bottom of the remainder as in (b); however, we change the placement of r_6 to cover the remaining part of the right side of $\mathcal{R}$. The points that determine the position of r_6 are marked by black squares in the figure. We use recursion to cover the bounding box $\mathcal{A}$ of the area that remains uncovered. (d) In Routine S-VII.3, we cover an L-shaped region of $\mathcal{R}$ using the four largest disks, and recurse on the remaining region $\mathcal{A}$.

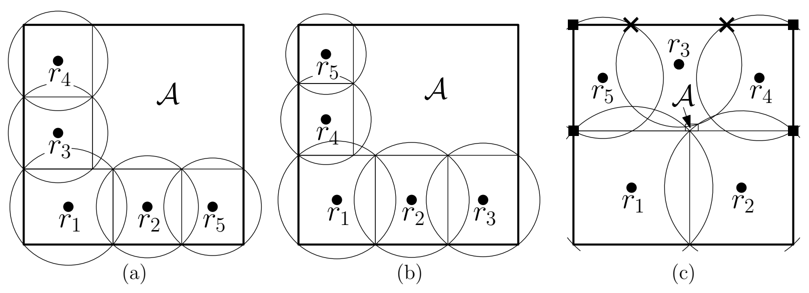

Figure 12 Routines for covering $\mathcal{R}$ using the five largest disks and recursion. According to Routine S-VIII.1, in (a) and (b), we first cover a horizontal strip of maximum height using three disks (r_1, r_2, r_3 or r_1, r_2, r_5) and then cover a vertical strip using the other two disks. (c) Routine S-VIII.2 places the five largest disks such that everything but a small region $\mathcal{A}$ is covered. The points that define the placement of r_4 and r_5 in are marked by boxes; those that define the placement of r_3 are marked $\times$. All three routines use recursion to cover $\mathcal{A}$.

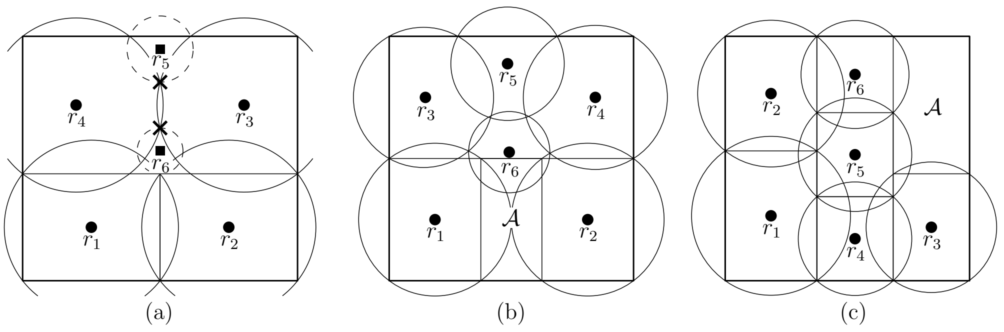

Figure 13 (a) Routine S-IX.1 covers $\mathcal{R}$ using the six largest disks. (b) In Routine S-IX.2, we also use recursion on the remaining disks to cover an additional rectangular region $\mathcal{A}$. (c) In Routine S-IX.3, we cover two vertical strips using r_1, r_2 and r_4, r_5, r_6, using r_3 and recursion to cover the remaining strip.

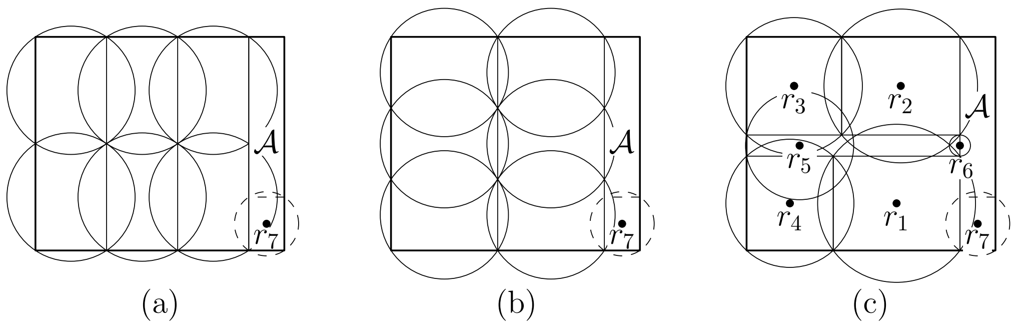

Figure 14 Routine S-X.1 considers the following three configurations to cover a strip of maximal width w. (a) Using any partition of $r_1, \ldots, r_6$ into three groups of two disks, each covering a strip of height 1 and maximal width, (b) using any partition of $r_1, \ldots, r_6$ into two groups of three disks, each covering a strip of height 1 and maximal width, or (c) using the disks r_1, r_4 and r_2, r_3 to cover strips of width w and maximal height and covering the uncovered pockets using r_5 and r_6.

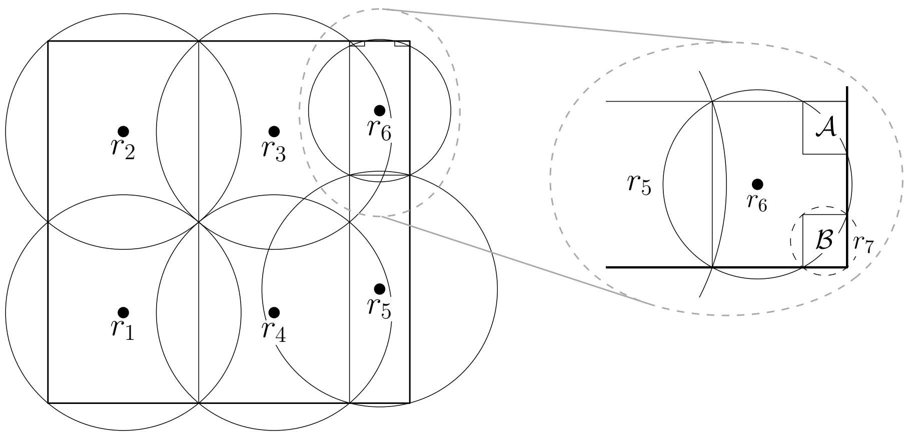

Figure 15 Routine S-X.2 covers as much width as possible using disks $r_1, \ldots, r_4$, using r_5, r_6 and r_7 on the remaining strip.

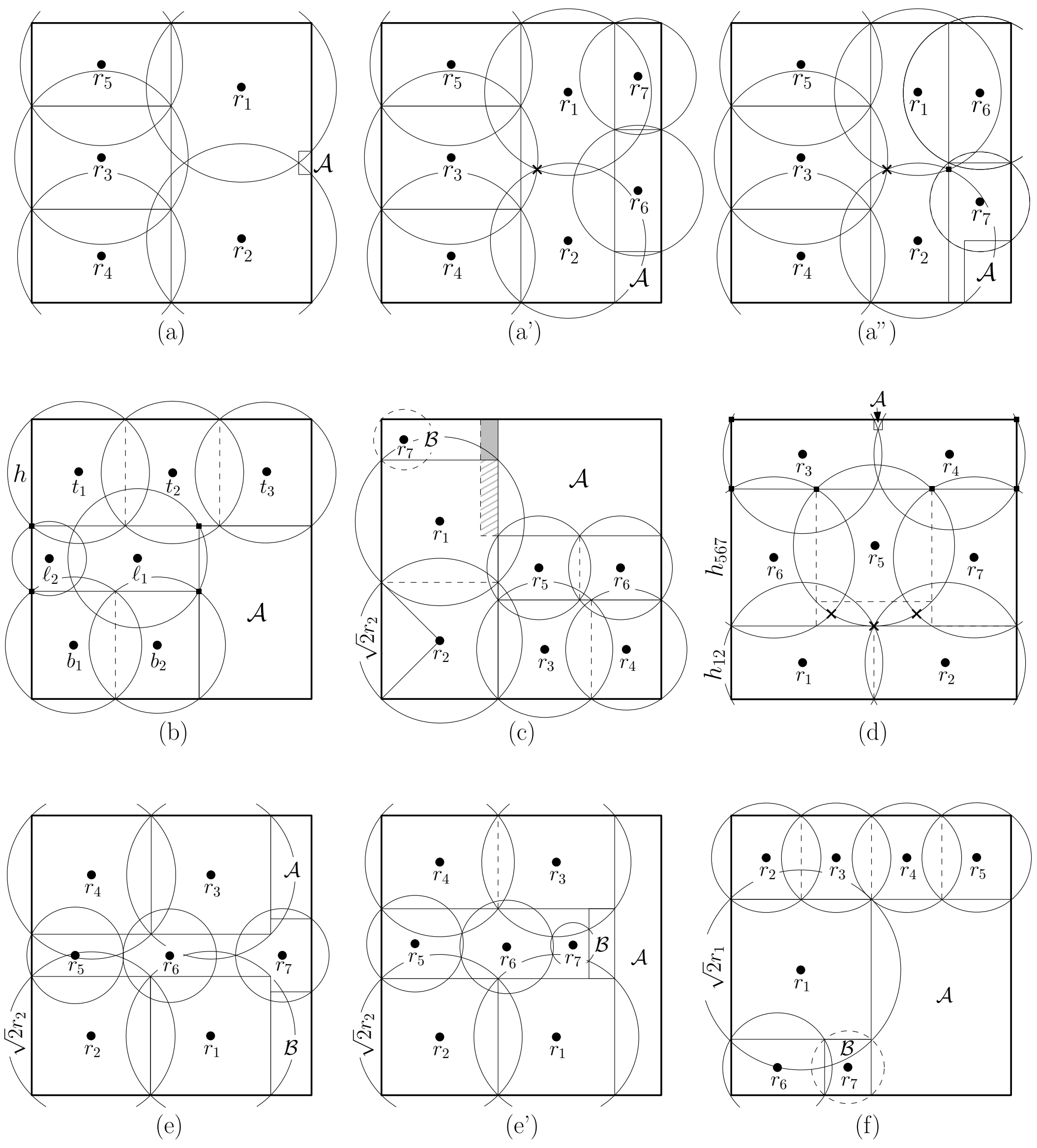

Figure 16 Routines S-X.3–S-X.8 using disks $r_1, \ldots, r_7$ and recursion to cover $\mathcal{R}$.

4.2 Proof Structure for Theorem 1

Tightness of the result claimed by Theorem 1 is proved by Lemma 2. Therefore, proving Theorem 1 means proving that, for any skew λ, any collection of disks D with $A(D) = A^*(\lambda)$ suffices to cover $\mathcal{R}$. As in the proof of Lemma 4, we begin by reducing the number of cases we have to consider to a finite number. Again, we begin by proving our result for all rectangles with sufficiently large skew.

▶ **Lemma 6.** *Let $\lambda \geq \overline{\lambda}$ and let D be a collection of disks with $W(D) = W^*(\lambda)$. We can cover $\mathcal{R}$ using the disks from D.*

Due to space restrictions, for the full proof we refer to the full version [18] of our paper. The proof is manual and uses the two simple routines Split Cover (W-I.1) and Large Disk (W-I.2); see Fig. 17. Intuitively speaking, if r_1 is small, we split D using Greedy Splitting, split $\mathcal{R}$ accordingly, and recurse on the two resulting regions. On the other hand, if r_1 is big, we cover the left side of $\mathcal{R}$ using r_1 and recurse on the remaining region.

The remainder of the proof of Theorem 1 is again based on a list of simple covering routines, which our algorithm tries to apply until it finds a working routine. We prove that there always is a working routine in the list using an automatic prover based on interval arithmetic as described in Section 3.3. After automatic analysis, several critical cases remain. We complete our proof by manually analyzing these critical cases. In the following, we give a brief description of the routines we use. Due to space constraints, for details, we refer to the full version of our paper [18].

Small disks. Because the covering coefficient guaranteed by Lemma 4 is always better than $E^*(\lambda)$, Routine (W-II.1) attempts to apply Lemma 4 directly; this works if the largest disk is not too big.

Using the largest disk. Routines (W-III.1)–(W-III.3) try several placements for the largest disk r_1; see Fig. 18.

Using the two largest disks. Routines (W-IV.1) and (W-IV.2) try several placements for the largest two disks r_1, r_2; see Fig. 19.

Using the three largest disks. Routines (W-V.1)–(W-V.5) consider several placements for the largest three disks; see Figs. 20, 21, 22, and 23.

Using the four largest disks. Routines (W-VI.1)–(W-VI.3) consider several placements for the largest four disks; see Fig. 24.

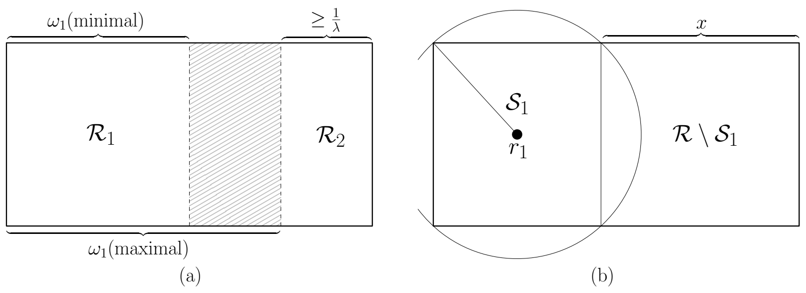

Figure 17 (a) The routine Split Cover (W-I.1) applies Greedy Splitting to the input disks, splits $\mathcal{R}$ into $\mathcal{R}_1, \mathcal{R}_2$ according to the split and recurses. The resulting split must not be too unbalanced for this routine to succeed. (b) The routine Large Disk (W-I.2) places r_1 covering a rectangle $\mathcal{S}_1$ at the right border of $\mathcal{R}$ and recurses on the remaining rectangle.

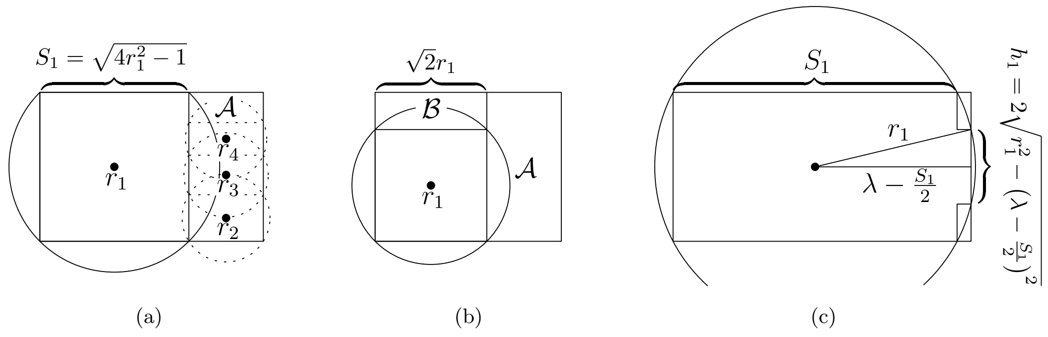
$S_1 = \sqrt{4r_1^2 - 1}$
$\mathcal{A}$
r_1
r_4
r_3
r_2
(a)
$\sqrt{2}r_1$
$\mathcal{B}$
$\mathcal{A}$
r_1
(b)
S_1
r_1
$\lambda - \frac{S_1}{2}$
$h_1 = 2\sqrt{r_1^2 - (\lambda - \frac{S_1}{2})^2}$
(c)

Figure 18 (a) In Routine W-III.1, we place r_1 covering a strip at the left side of $\mathcal{R}$ and try to recurse on $\mathcal{A}$. If that does not work, we also try to place r_2, r_3 and potentially r_4 covering horizontal strips at the bottom of the remaining rectangle before we try recursing. (b) In Routine W-III.2, we place r_1 covering its inscribed square at the bottom-left corner of $\mathcal{R}$, covering the two remaining regions $\mathcal{A}, \mathcal{B}$ recursively. (c) In Routine W-III.3, we place r_1 covering a strip at the left side of $\mathcal{R}$; if placed like this, r_1 intersects the right border of $\mathcal{R}$, only leaving two small uncovered pockets.

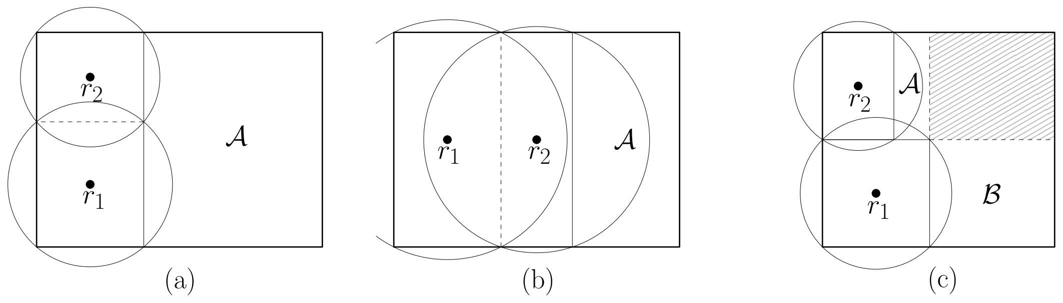

Figure 19 (a) and (b) depict Routine W-IV.1. The two largest disks are used to cover as wide a strip as possible at the left side of $\mathcal{R}$; the remaining disks are used for recursion on $\mathcal{A}$. (c) Routine W-IV.2 places r_1 covering its inscribed square and covers the remaining part of $\mathcal{R}$'s left boundary using r_2. Two regions $\mathcal{A}$ and $\mathcal{B}$ remain. The shaded area can be added to either $\mathcal{A}$ or $\mathcal{B}$; we try both options.

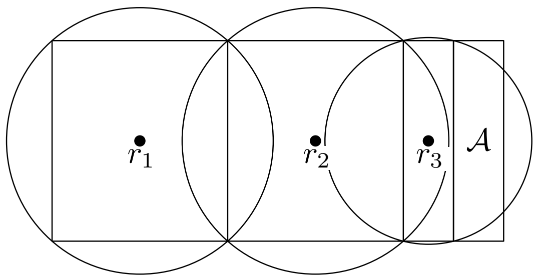

Figure 20 Routine W-V.1 places the three largest disks next to each other, each covering a vertical strip of height 1. If this does not cover the entire rectangle, we recurse on the bounding box $\mathcal{A}$ of the remaining area.

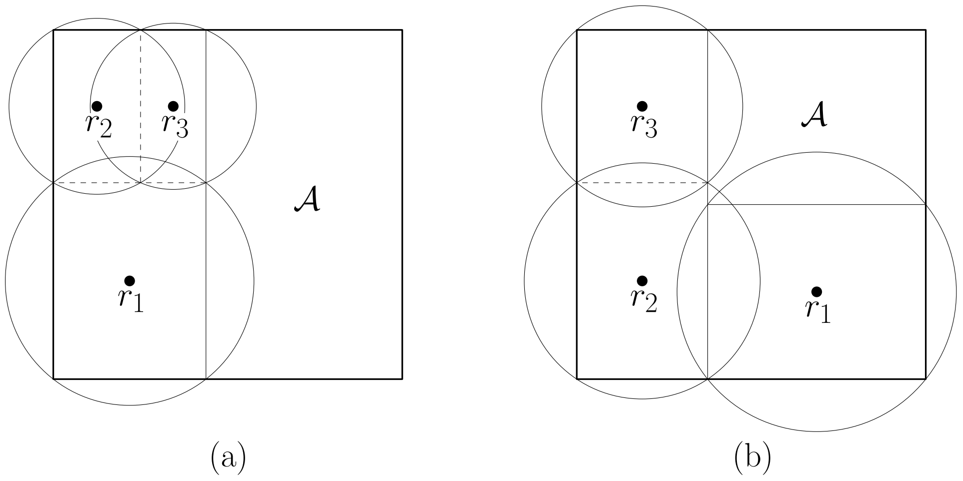

Figure 21 (a) Routine W-V.2 builds a strip of maximum possible width by placing r_1 at the bottom and r_2 besides r_3 on top. (b) Routine W-V.3 builds a vertical strip of maximum possible width by placing r_2, r_3 on top of each other, and covers the remaining part of the lower boundary using r_1.

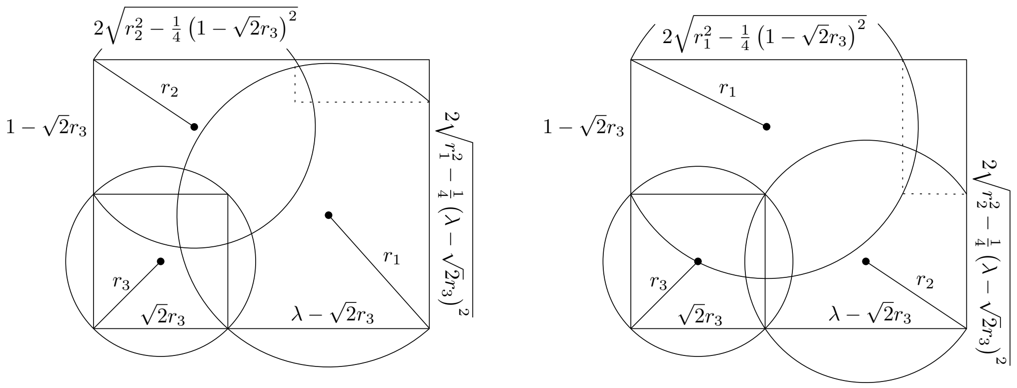
$2\sqrt{r_2^2 - \frac{1}{4}\left(1-\sqrt{2}r_3\right)^2}$
$1-\sqrt{2}r_3$
r_2
$2\sqrt{r_1^2 - \frac{1}{4}\left(\lambda-\sqrt{2}r_3\right)^2}$
r_1
r_3
$\sqrt{2}r_3$
$\lambda-\sqrt{2}r_3$
$2\sqrt{r_1^2 - \frac{1}{4}\left(1-\sqrt{2}r_3\right)^2}$
$1-\sqrt{2}r_3$
r_1
$2\sqrt{r_2^2 - \frac{1}{4}\left(\lambda-\sqrt{2}r_3\right)^2}$
r_3
r_2
$\sqrt{2}r_3$
$\lambda-\sqrt{2}r_3$

Figure 22 Routine W-V.4 covers the rectangle using the third-largest disk to cover a square at the bottom-left corner. The remaining rectangle that we recurse on is drawn with dashed outline. **Left:** Placing the largest disk to the right of the third-largest disk and the second-largest disk on top of the third-largest disk. **Right:** Placing the largest disk on top of the third-largest disk and the second-largest disk to the right of the third-largest disk.

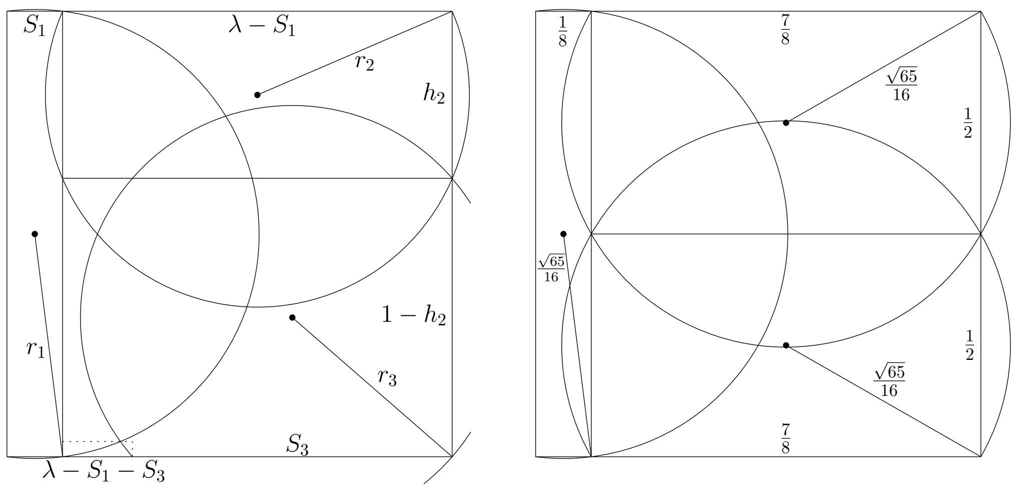

Figure 23 Left: Routine W-V.5 covers the rectangle using the largest disk to cover a strip of width S_1, using the second- and third-largest disks to cover the remaining corners. The bounding box of the uncovered pocket between the largest and third-largest disk is drawn with dashed outline. **Right:** The worst-case example for a square, consisting of three equal disks with radius $\frac{\sqrt{65}}{16}$. The covering of Routine W-V.5 converges to this covering for disks converging to this worst-case example.

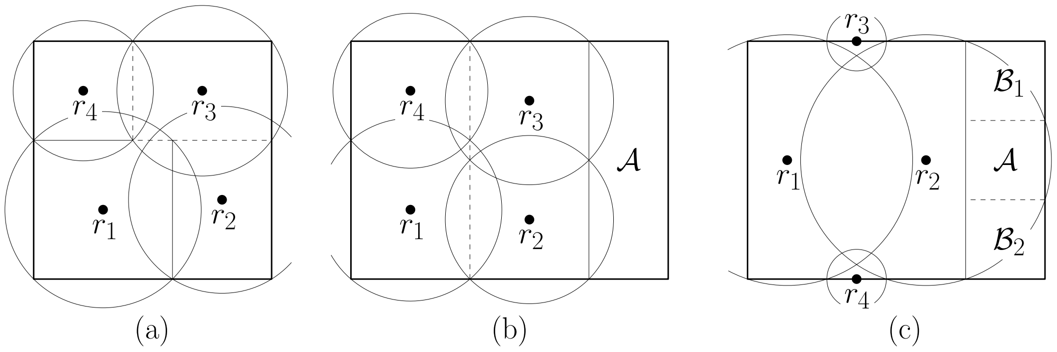

Figure 24 (a) Routine W-VI.1 covers $\mathcal{R}$ using only the four largest disks. The dashed outline depicts the rectangle that r_3 has to be able to cover. (b) Routine W-VI.2 covers a strip of maximum possible width using two groups of two disks and recurses on the remaining rectangle $\mathcal{A}$. (c) Routine W-VI.3 covers $\mathcal{R}$ by placing the two largest disks besides each other, filling the gaps between the disks using r_3, r_4. If this does not cover $\mathcal{R}$, we either recurse on the remaining strip $\mathcal{A}$ or on the bounding box of two pockets $\mathcal{B}_1, \mathcal{B}_2$ if r_2 intersects $\mathcal{R}$'s right border.

4.3 Proof of Lemma 3

In this section, we give a proof of Lemma 3.

▶ **Lemma 3.** *Let* $\hat{\sigma} := \frac{195\sqrt{5257}}{16384} \approx 0.8629$*. Let* $\sigma \geq \hat{\sigma}$ *and* $E(\sigma) := \frac{1}{2}\sqrt{\sqrt{\sigma^2+1}+1}$*. Let* $\lambda \geq 1$ *and* $D = \{r_1, \ldots, r_n\}$ *be any collection of disks with* $\sigma \geq r_1^2 \geq \ldots \geq r_n^2$ *and* $W(D) = \sum_{i=1}^{n} r_i^2 \geq E(\sigma)\lambda$*. Then* D *can cover a rectangle* $\mathcal{R}$ *of dimensions* $\lambda \times 1$*.*

Proof. In the following, let $E := E(\sigma)$; we assume w.l.o.g. that $W(D) = E\lambda$. First, we observe that $\sigma \geq \hat{\sigma}$ implies $E \geq \frac{195}{256} = E^*(\bar{\lambda})$. Because $E^*(\lambda)$ for $\lambda \geq \bar{\lambda}$ is continuous and strictly monotonically increasing, there is a unique $\Lambda(E) \geq \bar{\lambda}$ such that $E^*(\Lambda(E)) = E$, given by $\Lambda(E) := 2E + \sqrt{4E^2 - 2}$. Similarly, we observe that $\sigma(E) = E \cdot \left(\Lambda(E) - \frac{2}{\Lambda(E)}\right)$ is the inverse function of $E(\sigma)$. If $\lambda \leq \Lambda(E)$, we have $E \geq E^*(\lambda)$ and the result immediately follows from Theorem 1.

Otherwise, we apply Greedy Splitting to D. This yields a partition into two groups D_1, D_2; w.l.o.g., let D_1 be the heavier one. We split $\mathcal{R}$ into two rectangles $\mathcal{R}_1, \mathcal{R}_2$ such that $\frac{W(D_1)}{W(D_2)} = \frac{|\mathcal{R}_1|}{|\mathcal{R}_2|}$ by dividing the longer side (w.l.o.g., the width) of $\mathcal{R}$ in that ratio. After the split, we have $E = \frac{W(D)}{|\mathcal{R}|} = \frac{W(D_1)}{|\mathcal{R}_1|} = \frac{W(D_2)}{|\mathcal{R}_2|}$ and $|\mathcal{R}_2| = \frac{W(D_2)}{E}$.

If the resulting width of any $\mathcal{R}_i$ is greater than $\Lambda(E)$, we use D_i to inductively apply Lemma 3 to it. Otherwise, we apply Theorem 1; in order to do so, we must show that the skew of the narrower rectangle $\mathcal{R}_2$ is at most $\Lambda(E)$, which means proving that its width is at least $\frac{1}{\Lambda(E)}$. Because of $W(D_1) - W(D_2) \leq r_1^2 \leq \sigma$, we have $W(D_2) \geq \frac{W(D)-\sigma}{2} = \frac{E\lambda - \sigma(E)}{2}$. This implies that the area, and thus the width, of $\mathcal{R}_2$ is $\frac{W(D_2)}{E} \geq \frac{\Lambda(E) - \sigma(E)/E}{2} = \frac{1}{\Lambda(E)}$. ◀

5 Conclusion

We have given a tight characterization of the critical covering density for arbitrary rectangles. This gives rise to numerous followup questions and extensions.

As discussed (and shown in Fig. 3), the worst-case values correspond to instances with only 2 or 3 relatively large disks; if we have an upper bound R on the size of the largest disk, this gives rise to the critical covering area $A_R^*(\lambda)$ for $\lambda \times 1$-rectangles. Both from a theoretical and a practical point of view, getting some tight bounds on $A_R^*(\lambda)$ would be interesting and useful. Our results of Lemma 3 and Lemma 4 indicate possible progress in that direction; just like for unit disks, tighter results will require considerably more effort.

Establishing the critical covering density for disks and triangles is also open. We are optimistic that an approach similar to the one of this paper can be used for a solution.

Finally, *computing* optimal coverings by disks appears to be quite difficult. However, while deciding whether a given collection of disks can be packed into a unit square is known to be NP-hard [15], the complexity of deciding whether a given set of disks can be used to cover a unit square is still open. Ironically, it is the higher practical difficulty of covering by disks that makes it challenging to apply a similar idea in a straightforward manner.

References

1 A Karim Abu-Affash, Paz Carmi, Matthew J. Katz, and Gila Morgenstern. Multi cover of a polygon minimizing the sum of areas. *International Journal of Computational Geometry & Applications*, 21(06):685–698, 2011.

2 Alessandro Agnetis, Enrico Grande, Pitu B. Mirchandani, and Andrea Pacifici. Covering a line segment with variable radius discs. *Computers & Operations Research*, 36(5):1423–1436, 2009.

3 Helmut Alt, Esther M. Arkin, Hervé Brönnimann, Jeff Erickson, Sándor P. Fekete, Christian Knauer, Jonathan Lenchner, Joseph S. B. Mitchell, and Kim Whittlesey. Minimum-cost coverage of point sets by disks. In *Proc. 22nd Annu. ACM Sympos. Comput. Geom.*, pages 449–458, 2006. doi:10.1145/1137856.1137922.

4 Balázs Bánhelyi, Endre Palatinus, and Balázs L. Lévai. Optimal circle covering problems and their applications. *Central European Journal of Operations Research*, 23(4):815–832, 2015.

5 Aaron T. Becker, Sándor P. Fekete, Phillip Keldenich, Sebastian Morr, and Christian Scheffer. Packing Geometric Objects with Optimal Worst-Case Density (Multimedia Exposition). In *Proceedings 35th International Symposium on Computational Geometry (SoCG)*, pages 63:1–63:6, 2019. Video available at https://www.ibr.cs.tu-bs.de/users/fekete/Videos/PackingCirclesInSquares.mp4. doi:10.4230/LIPIcs.SoCG.2019.63.

6 K. Bezdek. *Körök optimális fedései (Optimal covering of circles)*. PhD thesis, Eötvös Lorand University, 1979.

7 Käroly Bezdek. Über einige optimale Konfigurationen von Kreisen. *Ann. Univ. Sci. Budapest Rolando Eötvös Sect. Math*, 27:143–151, 1984.

8 Santanu Bhowmick, Kasturi R. Varadarajan, and Shi-Ke Xue. A constant-factor approximation for multi-covering with disks. *JoCG*, 6(1):220–234, 2015. doi:10.20382/jocg.v6i1a9.

9 Károly Böröczky Jr. *Finite packing and covering*, volume 154. Cambridge University Press, 2004.

10 Peter Brass, William O.J. Moser, and János Pach. Density problems for packings and coverings. *Research Problems in Discrete Geometry*, pages 5–74, 2005.

11 Paz Carmi, Matthew J. Katz, and Nissan Lev-Tov. Covering points by unit disks of fixed location. In *Proc. International Symposium on Algorithms and Computation (ISAAC)*, pages 644–655. Springer, 2007.

12 CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

13 Gautam K. Das, Sandip Das, Subhas C. Nandy, and Bhabani P. Sinha. Efficient algorithm for placing a given number of base stations to cover a convex region. *Journal of Parallel and Distributed Computing*, 66(11):1353–1358, 2006.

14 Gautam K. Das, Sasanka Roy, Sandip Das, and Subhas C. Nandy. Variations of base-station placement problem on the boundary of a convex region. *International Journal of Foundations of Computer Science*, 19(02):405–427, 2008.

15 Erik D. Demaine, Sándor P. Fekete, and Robert J. Lang. Circle packing for origami design is hard. In *Origami^5: 5th International Conference on Origami in Science, Mathematics and Education*, AK Peters/CRC Press, pages 609–626, 2011. arXiv:1105.0791.

16 dpa. Rasensprenger zeichnet Kreise auf Fußballfeld, 2018.

17 Gábor Fejes Tóth. Recent progress on packing and covering. *Contemporary Mathematics*, 223:145–162, 1999.

18 Sándor P. Fekete, Utkarsh Gupta, Phillip Keldenich, Christian Scheffer, and Sahil Shah. Worst-case optimal covering of rectangles by disks. *arXiv:2003.08236 [cs.CG]*, 2020. arXiv:2003.08236.

19 Sándor P. Fekete, Phillip Keldenich, and Christian Scheffer. Packing Disks into Disks with Optimal Worst-Case Density. In *Proceedings 35th International Symposium on Computational Geometry (SoCG 2019)*, pages 35:1–35:19, 2019. doi:10.4230/LIPIcs.SoCG.2019.35.

20 Sándor P. Fekete, Phillip Keldenich, and Christian Scheffer. Covering rectangles by disks: The video. In *Proceedings of the 36th International Symposium on Computational Geometry (SoCG)*, 2020. To appear.

21 Sándor P. Fekete, Sebastian Morr, and Christian Scheffer. Split packing: Algorithms for packing circles with optimal worst-case density. *Discrete & Computational Geometry*, 2018. doi:10.1007/s00454-018-0020-2.

22 F. Fodor. The densest packing of 19 congruent circles in a circle. *Geometriae Dedicata*, 74:139–145, 1999.

23 F. Fodor. The densest packing of 12 congruent circles in a circle. *Beiträge zur Algebra und Geometrie (Contributions to Algebra and Geometry)*, 41:401–409, 2000.

24 F. Fodor. The densest packing of 13 congruent circles in a circle. *Beiträge zur Algebra und Geometrie (Contributions to Algebra and Geometry)*, 44:431–440, 2003.

25 E. Friedman. Circles covering squares web page, 2014. URL: `http://www2.stetson.edu/~efriedma/circovsqu/`.

26 M. Goldberg. Packing of 14, 16, 17 and 20 circles in a circle. *Mathematics Magazine*, 44:134–139, 1971.

27 R.L. Graham, B.D. Lubachevsky, K.J. Nurmela, and P.R.J. Östergøard. Dense packings of congruent circles in a circle. *Discrete Mathematics*, 181:139–154, 1998.

28 Aladár Heppes and Hans Melissen. Covering a rectangle with equal circles. *Periodica Mathematica Hungarica*, 34(1-2):65–81, 1997.

29 Chi-Fu Huang and Yu-Chee Tseng. A survey of solutions for the coverage problems in wireless sensor networks. *Journal of Internet Technology*, 6(1):1–8, 2005.

30 Matthew P. Johnson, Deniz Sariöz, Amotz Bar-Noy, Theodore Brown, Dinesh Verma, and Chai W. Wu. More is more: the benefits of denser sensor deployment. *ACM Transactions on Sensor Networks (TOSN)*, 8(3):22, 2012.

31 B.D. Lubachevsky and R.L. Graham. Curved hexagonal packings of equal disks in a circle. *Discrete & Computational Geometry*, 18:179–194, 1997.

32 H. Melissen. Densest packing of eleven congruent circles in a circle. *Geometriae Dedicata*, 50:15–25, 1994.

33 Hans Melissen. Loosest circle coverings of an equilateral triangle. *Mathematics Magazine*, 70(2):118–124, 1997.

34 Johannes Bernardus Marinus Melissen and Peter Cornelis Schuur. Covering a rectangle with six and seven circles. *Discrete Applied Mathematics*, 99(1-3):149–156, 2000.

35 John W. Moon and Leo Moser. Some packing and covering theorems. In *Colloquium Mathematicae*, volume 17, pages 103–110. Institute of Mathematics, Polish Academy of Sciences, 1967.

36 Sebastian Morr. Split packing: An algorithm for packing circles with optimal worst-case density. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 99–109, 2017.

37 Eric H. Neville. On the solution of numerical functional equations. *Proceedings of the London Mathematical Society*, 2(1):308–326, 1915.

38 Kari J. Nurmela. Conjecturally optimal coverings of an equilateral triangle with up to 36 equal circles. *Experimental Mathematics*, 9(2):241–250, 2000.

39 Norman Oler. A finite packing problem. *Canadian Mathematical Bulletin*, 4:153–155, 1961.

40 Endre Palatinus and Balázs Bánhelyi. Circle covering and its applications for telecommunication networks. In *8 th International Conference on Applied Informatics*, page 255, 2010.

41 G.E. Reis. Dense packing of equal circles within a circle. *Mathematics Magazine*, issue 48:33–37, 1975.

42 Williamjeet Singh and Jyotsna Sengupta. An efficient algorithm for optimizing base station site selection to cover a convex square region in cell planning. *Wireless personal communications*, 72(2):823–841, 2013.

43 Eckard Specht. Packomania, 2015. URL: `http://www.packomania.com/`.

44 Balázs Szalkai. Optimal cover of a disk with three smaller congruent disks. *Advances in Geometry*, 16(4):465–476, 2016.

45 Gábor Fejes Tóth. Thinnest covering of a circle by eight, nine, or ten congruent circles. *Combinatorial and computational geometry*, 52(361):59, 2005.

46 Gábor Fejes Tóth. Packing and covering. In *Handbook of Discrete and Computational Geometry, Third Edition*, pages 27–66. Chapman and Hall/CRC, 2017.

47 Xiaochun Xu, Sartaj Sahni, and Nageswara S.V. Rao. Minimum-cost sensor coverage of planar regions. In *FUSION*, pages 1–8, 2008.