Packing Squares into a Disk with Optimal Worst-Case Density

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

Vijaykrishna Gurunathan
Department of Computer Science & Engineering, IIT Bombay, India
krishnavijay1999@gmail.com

Kushagra Juneja
Department of Computer Science & Engineering, IIT Bombay, India
kuku12320@gmail.com

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

Linda Kleist
Department of Computer Science, TU Braunschweig, Germany
l.kleist@tu-bs.de

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

Abstract

We provide a tight result for a fundamental problem arising from packing squares into a circular container: The critical density of packing squares in a disk is $\delta = \frac{8}{5\pi} \approx 0.50929$. This implies that any set of (not necessarily equal) squares of total area $A \leq \frac{8}{5}$ can always be packed into a unit disk; in contrast, for any $\varepsilon > 0$ there are sets of disks of area $\frac{8}{5} + \varepsilon$ that cannot be packed. This settles the last remaining case of packing circular or square objects into a circular or square container, as the critical densities for squares in a square (0.5), circles in a square (≈ 0.539) and circles in a circle (0.5) have already been established. The proof uses a careful manual analysis, complemented by a minor automatic part that is based on interval arithmetic. Beyond the basic mathematical importance, our result is also useful as a blackbox lemma for the analysis of recursive packing algorithms.

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

Keywords and phrases Square packing, packing density, tight worst-case bound, interval arithmetic, approximation

Supplement Material `https://gitlab.ibr.cs.tu-bs.de/alg/square-in-circle-proofs.git`

Lines 498

1 Introduction

Problems of geometric packing and covering arise in a wide range of natural applications. They also have a long history of spawning many extremely demanding (and often still unsolved) mathematical challenges. These difficulties are also notable from an algorithmic perspective, as relatively straightforward one-dimensional variants of packing and covering are already NP-hard; however, deciding whether a given set of one-dimensional segments can be packed into a given interval can be checked by computing their total length. This simple

36th Symposium on Computational Geometry (SoCG 2020).
Editors: Sergio Cabello and Danny Z. Chen; Article No. 11; pp. 11:1–11:27
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

criterion is no longer available for two-dimensional, geometric packing or covering problems, for which the total volume often does not suffice to decide feasibility of a set, making it necessary to provide an explicit packing or covering.

We provide a provably optimal answer for a natural and previously unsolved case of *tight worst-case area bounds*, based on the notion of *critical packing density*: What is the largest number $\delta_p \leq 1$, such that any set S of squares with a total volume of at most δ_p can always be packed into a disk C of area 1, regardless of the individual sizes of the elements in S? We show that the correct answer is $\delta_p = \frac{8}{5\pi} \approx 0.50929$: Any set of squares of total area at most $\frac{8}{5}$ can be packed into a unit circle, and for any value $A > \frac{8}{5}$, there are sets that cannot be packed. This quantity is of mathematical importance, as it settles a number of open problems, as well as of algorithmic interest, because it provides a simple criterion for feasibility. It also settles the last remaining case of packing circular or square objects into a circular or square container, as the critical densities for squares in a square (0.5), circles in a square ($\frac{\pi}{3+\sqrt{2}} \approx 0.539$) and circles in a circle 0.5 have already been established; see Figure 1 for an overview, and the video `https://www.ibr.cs.tu-bs.de/users/fekete/Videos/PackingCirclesInSquares.mp4` for animated descriptions of the involved techniques.

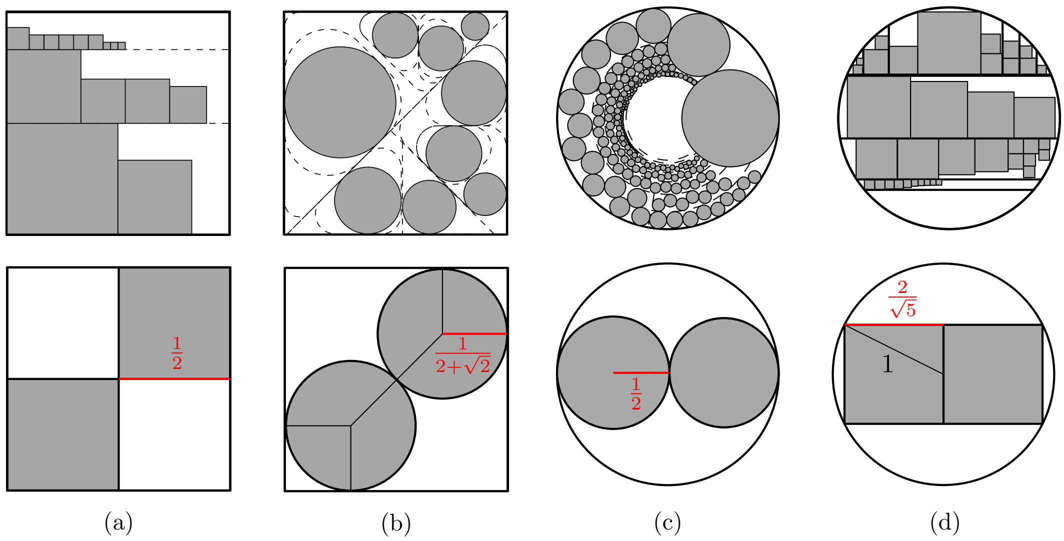

Figure 1 (a) The worst-case optimal approach Shelf Packing [9] of Moon and Moser for packing squares into a unit square and the corresponding worst-case instance. (b) The worst-case optimal packing approach of Fekete et al. [5] for packing disks into a unit square and the corresponding worst-case instance. (c) The worst-case optimal approach of Fekete et al. [4] for packing disks into a unit disk and the corresponding worst case instance. (d) Our worst-case optimal packing approach for packing squares into a unit disk and the corresponding worst-case instance.

1.1 Related Work

Problems of square packing have been studied for a long time. The decision problem whether it is possible to pack a given set of squares into the unit square was shown to be strongly NP-complete by Leung et al. [8], using a reduction from 3-Partition. Already in 1967, Moon and Moser [9] found a sufficient condition for packing squares into a square: They proved that the critical packing density for squares into a square is $\frac{1}{2} = 0.5$, so it is possible to pack a set of squares into the unit square in a shelf-like manner if their combined area does not exceed $\frac{1}{2}$. This is the *largest upper area bound* one can hope for, because two squares even infinitesimally larger than the ones shown in Figure 1(a) cannot be packed.

For the case of packing disks into a square container, Demaine, Fekete, and Lang [3] showed in 2010 that deciding whether a given set of disks can be packed is NP-hard, also by using a reduction from 3-Partition. This means that there is (most likely) no deterministic polynomial-time algorithm to decide whether a given set of disks can be packed into a given container. The problem of establishing the critical packing density for disks in a square was posed by Demaine, Fekete, and Lang [3] and resolved by Morr, Fekete and Scheffer [5, 10]. Using a recursive procedure for partitioning the container into triangular pieces, they proved that the critical packing density of disks in a square is $\frac{\pi}{3+2\sqrt{2}} \approx 0.539$.

More recently, Fekete et al. [4] established the critical packing density of disks into a disk. Employing a number of algorithmic techniques in combination with interval arithmetic and computer-assisted case checking, they proved that the critical packing density of disks in a disk is $\frac{1}{2} = 0.5$. For an animated overview, see [1], with the corresponding video available at `https://www.ibr.cs.tu-bs.de/users/fekete/Videos/PackingCirclesInSquares.mp4`.

Note that the main objective of this line of research is to compute tight worst-case bounds. For specific instances, a packing may still be possible, even if the density is higher; this also implies that proofs of infeasibility for specific instances may be trickier. However, the idea of using the total item volume for computing packing bounds can still be applied. See the work by Fekete and Schepers [6, 7], which shows how a *modified* volume for geometric objects can be computed, yielding good lower bounds for one- or higher-dimensional scenarios.

1.2 Results

We prove that the critical density for packing squares into a disk is $\frac{8}{5\pi} \approx 0.50929$: Any set of (not necessarily equal) squares with a combined area of at most $\frac{8}{5}$ can be packed into a unit circle; this is best possibly, as for any $\varepsilon > 0$ there are instances of total area $\frac{8}{5} + \varepsilon$ that cannot be packed. See Figure 1(d) for the critical configuration. Because our proof is constructive, it yields a constant-factor approximation algorithm for the smallest disk in which a given set of squares can be packed. We also sketch a proof of NP-hardness for the problem of deciding whether a given set of squares can be packed into a unit circle.

2 Preliminaries

Throughout this paper, all squares and rectangles are axis aligned. $\mathcal{D}$ denotes a unit disk. We denote by $s_1, \ldots, s_n$ a sequence of squares and simultaneously their side lengths; w.l.o.g., we consider these sequences to be sorted, i.e., $s_1 \geq \cdots \geq s_n$. *Packing* $s_1, \ldots, s_n$ into an object $\mathcal{O}$ means placing each $s_1, \ldots, s_n$ inside $\mathcal{O}$ while avoiding overlaps with previously packed squares. The *width* and the *height* of a rectangle are its dimensions regarding x- and y-coordinates.

For a given packing strategy P and an *input sequence* $s_1, \ldots, s_n$ of squares to be packed by P, the *input area* is the total area of $s_1, \ldots, s_n$. A packing strategy *stops with packing* s_{n-1} when the approach packs $s_1, \ldots, s_{n-1}$, but fails with packing s_n.

3 Complexity

We sketch a hardness proof for packing squares into a disk, as follows.

▶ **Theorem 1.** *It is* NP*-hard to decide whether a given set of squares fits into a circular container.*

The proof uses a reduction from 3-Partition; it is somewhat similar to the one by Leung et al. [8] for deciding whether a given set of squares fits into a given square container, and the one by Demaine, Fekete, and Lang in 2010 [3] for deciding whether a give set of disks fits into a given square container; see Figure 2 for an overview.

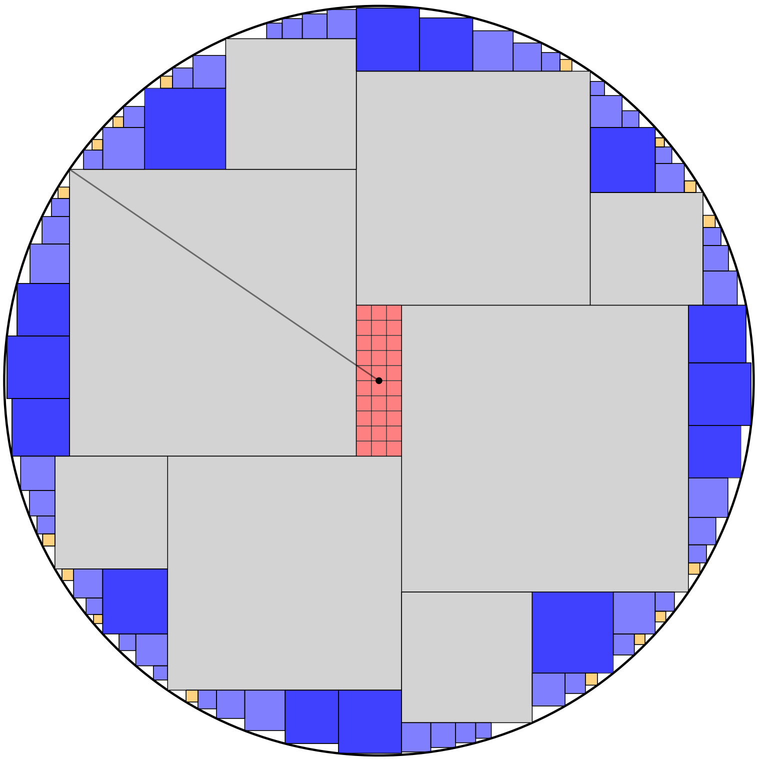

Figure 2 Overview of the 3-Partition reduction. The gray *framing* squares can only be packed in the shown manner, inducing a central rectangular pocket. The numbers of the 3-Partition instance are mapped to a set of red *number* squares of almost equal size, with small modifications of size ε_i, such that a triple (i, j, k) of red squares fits if and only if $\varepsilon_i + \varepsilon_j + \varepsilon_k \leq 0$, i.e., if there is a feasible 3-Partition. Additional recursively chosen blue and yellow *filler* squares tightly close the remaining gaps outside the framing squares, ensuring that no red square can be packed outside the central pocket. The dimensions of these filler squares are chosen such that they are either too large (shown in blue) or too small (shown in yellow) to allow a tight packing into the central pocket.

We use a set of eight *framing* squares that leave central rectangular pocket and some outside gaps. The numbers of the 3-Partition instance are mapped to a set of *number* squares of almost equal size, with small modifications of size ε_i, such that a triple (i, j, k) of red squares has total width of not exceeding the small edge of the pocket if and only if $\varepsilon_i + \varepsilon_j + \varepsilon_k \leq 0$, i.e., if there is a feasible 3-partition. For filling the gaps outside the framing squares, a set of *filler* squares are recursively constructed, so that no number square can be packed outside if all filler squares are packed outside. A detailed proof establishes the following claims.

1. The framing squares can only be packed in one canonical fashion, up to symmetries.
2. The filler squares fight tightly when packed in the described canonical manner outside the central pocket.

3. When all filler squares are packed outside the central pocket, the number squares can only be packed in the central pocket. This is possible if and only if there is a feasible 3-partition.
4. Packing a filler square inside the central pocket forces an unpackable gap that prevents an overall feasible packing.
5. The overall construction can be realized with squares of sufficiently approximated edge lengths of polynomial description size.

The first three claims are relatively straightforward. For claim **4.**, we choose the parameters such that the sequence of filler square sizes is sufficiently different from multiples of number square sizes; see Figure 2. For claim **5.**, we makes use of the limited description complexity of a 3-PARTITION instance, and sufficiently good Taylor expansion of the involved square roots. These parts are tedious, but straightforward. We omit details due to limited space, and the fact that the hardness proof is neither surprising nor central to this paper.

4 A Worst-Case Optimal Algorithm

The main result of this paper is to provide a worst-case optimal algorithm for packing squares into a unit disk.

▶ **Theorem 2.** *Every set of squares with a total area of at most $\frac{8}{5}$ can be packed into the unit disk. This is worst-case optimal, i.e., for every $\lambda > \frac{8}{5}$ there exists a set of squares with a total area of λ that cannot be packed into the unit disk.*

A proof of Theorem 2 consists of (i) a class of instances that provide the upper bound of $\frac{8}{5}$ and (ii) an algorithm that achieves the lower bound by packing any set of squares with a total area of at most $\frac{8}{5}$ into the unit disk.

The upper bound is implied by any two squares with a side length of $\sqrt{\frac{4}{5}+\varepsilon}$, for arbitrary $\varepsilon > 0$, see Figure 1(d): When placed in the unit disk, either of them must contain the disk center in its interior, so both cannot be packed simultaneously.

In the rest of the paper, we give a constructive proof for the lower bound by describing an algorithm that can pack any instance with total area $\frac{8}{5}$.

4.1 Description of the Algorithm

In the following, we consider a set of given squares with side lengths $s_1, \ldots, s_n$. We pack them in sequential order by decreasing size, and assume w.l.o.g. that $s_1 \geq \cdots \geq s_n$. Our algorithm distinguishes three types of instances:

1. All squares are small, i.e., $s_1 \leq 0.295$.
2. The first four squares are fairly large, i.e., $s_1 \leq \frac{1}{\sqrt{2}}$ and $s_1^2 + s_2^2 + s_3^2 + s_4^2 \geq \frac{8}{5} - \frac{1}{25}$.
3. All other cases.

In the first and second case, we can argue that an appropriate packing of the first four squares in combination with the known packing density of squares into a square suffices to achieve the claimed packing density for squares in a circle: In the first case, we pack all but the first four squares into a large square container by SHELF PACKING and each of the first four squares adjacent to one of the four sides as illustrated in Figure 3(a). In the second case, we pack the first four squares into a central square container, achieving high enough packed area that it suffices to pack the remaining squares into a smaller subsquare with the

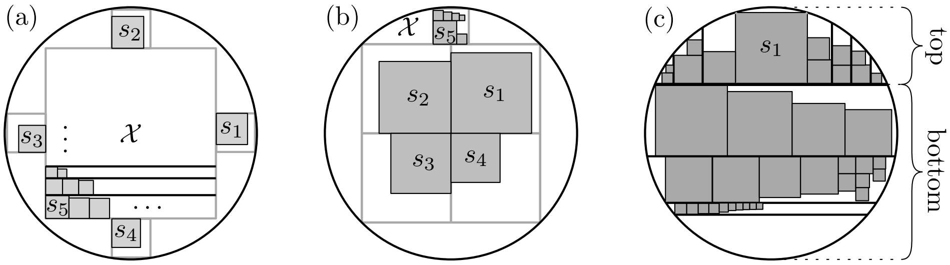

Figure 3 (a) The packing in case of $s_1 \leq 0.295$. (b) The packing in case of $s_1 \leq \frac{1}{\sqrt{2}}$ and $\frac{39}{25} \leq s_1^2 + s_2^2 + s_3^2 + s_4^2$. (c) The packing in the remaining cases is a combination of Top Packing (top) and Bottom Packing (bottom).

worst-case packing density of squares into a square. A detailed analysis of these two cases is presented in Section 6.1.

This leaves the scenario in which we have neither big nor only small squares. We deal with this case by making extensive use of shelf packing; however, the circular shape of the container prevents a straightforward application as for the rectangular container considered by Moon and Moser [9], as we may incur some gaps along the boundary of the container, and thus, some lost area. This requires a more intricate recursive approach, in which we partition the container into a number of pieces that are packed by using a sequence of axis-parallel shelves. A detailed analysis shows that this approach may only fail to pack the given set of squares if its area is larger than the critical bound.

More specifically, the largest square in the third case is packed into $\mathcal{D}$ as high as possible, see Figure 3(c) and Figure 4 for an illustration. The bottom of this square induces a horizontal split of the container into a *top* and a *bottom* part, which are then packed by two subroutines called Top Packing and Bottom Packing as described in Sections 4.2.2 and 4.2.4.

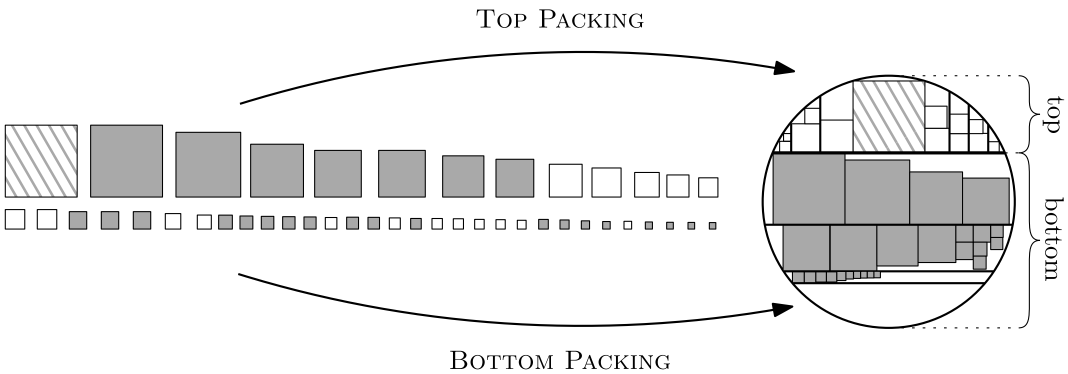

Figure 4 Our algorithm packs squares in decreasing order. The largest (hatched) square is packed as far as possible to the top, inducing a top and a bottom portion, with the empty top space consisting of two congruent pockets. Subsequent (white) squares are packed greedily into these top pockets with Top Packing (which uses shelf packing as a subroutine) if they fit; if they do not fit, they are shown in gray and packed into the bottom with Bottom Packing, which uses horizontal subcontainer slicing, and vertical shelf packing within each slice.

Technically, this yields the following description of our algorithm.

1. If $s_1 \leq 0.295$, place a square of side length $\mathcal{X} = 1.388$ concentric into $\mathcal{D}$ and place one square of side length $\mathcal{X}_i = 0.295$ to each side of $\mathcal{X}$, see Figure 3(a).

- For $i = 1, 2, 3, 4$, pack each s_i into one of the squares of side length $\mathcal{X}_i = 0.295$.
- For $i \geq 5$, use Shelf Packing for packing s_i into $\mathcal{X}$.

2. If $s_1 \leq \frac{1}{\sqrt{2}}$ and $s_1^2 + s_2^2 + s_3^2 + s_4^2 \geq \frac{39}{25}$, let $\mathcal{X}_1, \ldots, \mathcal{X}_4$ be the four equally sized maximal squares that fit into $\mathcal{D}$ and let be $\mathcal{X}$ the largest square that can be additionally packed into $\mathcal{D}$, see Figure 3(b).
- For $i = 1, 2, 3, 4$, pack each s_i into one of the squares of side length $\mathcal{X}_i$.
- For $i \geq 5$, use Shelf Packing for packing s_i into $\mathcal{X}$.

3. Otherwise
- Pack s_1 as far as possible to the top into $\mathcal{D}$.
- For $i \geq 2$,
  (3.1) if possible, use Top Packing for packing s_i,
  (3.2) otherwise, use Bottom Packing for packing s_i.

Similar to the argument by Moon and Moser for squares packed into a square container, we use careful bookkeeping to prove that this algorithm only fails to pack a square in the decreasing if the total area of all squares exceeds the critical bound, which is 8/5 for a unit disk container.

4.2 Subroutines of Our Algorithm

Our algorithm makes use of a number of different subroutines. At the lowest level, we use a refined version of the classic *shelf packing* (described in Section 4.2.1), with some adjustments accounting for the possible presence of some curved boundary. At the intermediate level, we use a routine called SubContainer Slicing (described in Section 4.2.3), which uses horizontal straight cuts to subdivide the circular container into pieces, which are then used for packing by using vertical shelves. At the highest level, we use the largest square for subdividing the circular container into a top portion, consisting of two identical pockets C_ℓ and C_r to the left and right of this square that are used for *top packing* by using axis-parallel shelves, and a bottom portion that is used for *bottom packing*, consisting of horizontal SubContainer Slicing, and vertical enhanced shelf packing within each subcontainer.

4.2.1 Refined Shelf Packing

Shelf Packing is a greedy-type packing procedures that was employed by Moon and Moser [9]. The idea is to pack objects by decreasing size; see Figure 1 (a)(Top). At each stage, there is a straight cut (shown horizontal in the figure) that subdivides the unused portion of the container from a "shelf" into which the next square is packed. The height of a shelf is determined by the first object that it accommodates. Subsequent objects are packed next to each other, without overlap, until an object no longer fits into the current shelf; in this case, we open a new shelf on top of the previous one, of height equal to the object.

In the context of our packing algorithm, we use three modifications. (1) In our descriptions and figures, we may use vertical shelves, produced by vertical cuts; the stacking within each shelf starts from the longer of the potentially two cuts that generate the subcontainer from which the shelf is cut. (2) Parts of the shelf boundaries may be circular arcs; however, in each case, we still have a supporting straight axis-parallel boundary (determined by the previous cut parallel to the shelf) and a second, orthogonal straight boundary (determined by the start of the current shelf or the previous square). (3) Our refined shelf packing uses the axis-parallel boundary line of a shelf as a support line for packing squares; in case of a collision with the circular boundary, we may adjust the x-coordinate of a square if this allows packing it. This is performed until a square no longer fits into the remaining free space.

4.2.2 Top Packing

The first and largest square s_1 is packed as high as possible into the unit disk, centered with respect to the vertical line through the disk center; see Figure 5 (a). Then the horizontal line ℓ_1 through the bottom of s_1 cuts the container into a top portion that contains s_1, with two congruent empty pockets C_ℓ and C_r left and right of s_1; each such pocket has two straight axis-parallel boundaries, b_x and b_y. We use shelf packing with shelves parallel to the shorter boundary among b_x and b_y, as shown in Figure 5 (c) and (d). If a square s_i does not fit into either pocket, it is packed into the portion below ℓ_1.

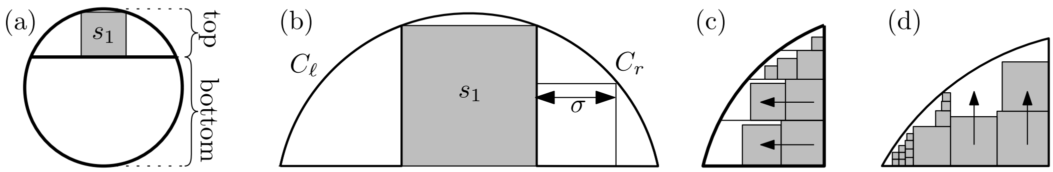

Figure 5 (a) Packing s_1 as far as possible to the top into $\mathcal{D}$. (b) The top portion of $\mathcal{D}$ with the pockets C_ℓ and C_r, and the size σ of the largest inscribed square. (c) A pocket C_ℓ where $b_x \leq b_y$, resulting in horizontal shelf packing. (d) A pocket C_ℓ where $b_x > b_y$, resulting in vertical shelf packing.

4.2.3 Subcontainer Slicing

Analogous to shelf packing, we subdivide the unused portion of the container disk into smaller pieces, by using straight horizontal cuts; see Figure 6 (Left). Also analogous is the width of a subcontainer, which is determined by the first packed square. Once a (horizontal) subcontainer is cut, it is used for enhanced shelf packing into vertical shelves, until a square no longer fits, as shown in Figure 6 (Right).

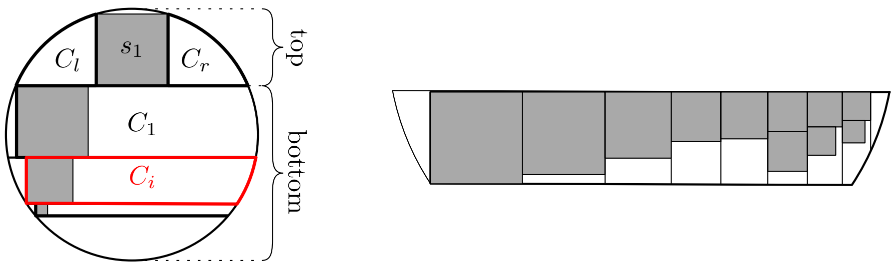

Figure 6 (Left) SubContainer Slicing partitions the lower part of $\mathcal{D}$ into subcontainers C_i, with the height corresponding to the first packed square. (Right) Within each subcontainer, SubContainer Packing places squares into C_i along vertical shelves, starting from the longer straight cut of the subcontainer.

4.2.4 Bottom Packing

A square that does not fit into the container above ℓ_1 is packed below ℓ_1. For this purpose, we use (horizontal) SubContainer Slicing, and enhanced (vertical) shelf packing within each subcontainer; see Figure 4 for the overall picture. These shelves are stacked from the longer of the two horizontal cuts, i.e., we pack away from the boundary that is closer to the disk center; see Figure 6 (Right) for packing the subcontainer.

5 Analytic Tools and Subroutines

In the following we provide a number of tools and bounds that will be used for establishing a tight worst-case performance for our algorithm.

5.1 Interval Arithmetic

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

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

In other words, the result of an operation is the smallest possible 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 in a similar manner. For the case of square roots, division or other operations that are not defined on all of $\mathbb{R}$, the result of an operation is undefined if and only if the input interval(s) contain values, for which the real counterpart of the operation is undefined.

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*.

Let $F := \{f_1(x_1, \dots, x_k) \leq r_1, \dots, f_m(x_1, \dots, x_k) \leq r_m\}$ be some given system of constraints over real variables $x_1, \dots, x_k$, where $f_1, \dots, f_m$ are real functions and $r_1, \dots, r_m \in \mathbb{R}$. Furthermore, assume that we can evaluate $f_1, \dots, f_m$ on intervals in the following sense. For each function f_i, given intervals $x_1^*, \dots, x_k^*$ for $x_1, \dots, x_k$, we can find an interval $f_i^*(x_1^*, \dots, x_k^*)$ that contains all possible outcomes of $f_i(x_1, \dots, x_k)$ for $x_1 \in x_1^*, \dots, x_k \in x_k^*$. Note that, geometrically, $x_1^* \times \dots \times x_k^*$ is a k-dimensional hypercuboid. Moreover, assume that for some such $x_1^*, \dots, x_k^*$, at least one inequality $f_i^*(x_1^*, \dots, x_k^*) \leq r_i$ from F is definitely false. Then we know that no point $(x_1, \dots, x_k) \in x_1^* \times \dots \times x_k^*$ satisfies the constraints F.

Let $\mathcal{R} \subseteq \mathbb{R}^k$ be some k-dimensional set of points. If we can cover (some superset of) $\mathcal{R}$ by hypercuboids, for each of which we can prove that at least one constraint from F is violated using interval arithmetic as outlined above, we know that F is unsatisfiable over $\mathcal{R}$. For bounded spaces $\mathcal{R}$, we can use this method to automatically prove unsatisfiability of F as follows. Conceptually, we cover $\mathcal{R}$ using a sufficiently fine k-dimensional grid and determine, for each grid cell $x_1^*, \dots, x_k^*$, that at least one inequality $f_i^*(x_1^*, \dots, x_k^*) \leq r_i$ from F is definitely false using implementations of the f_i^* on a computer. In order to improve efficiency, the grid by which we cover $\mathcal{R}$ is finer in some places than in others, reflecting that the constraints may be more strongly violated in some parts of $\mathcal{R}$ than in others.

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 arithmetic in the well-known Computational Geometry Algorithms Library (CGAL) [2]. 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.

Throughout this paper, we use interval arithmetic to prove bounds on the value of real functions for all points satisfying some constraints. For this purpose, note that proving $f(x) < r$ for all points $x \in \mathcal{R}$ satisfying constraints F is equivalent to proving unsatisfiability of $\{f(x) \geq r\} \cup F$ on $\mathcal{R}$. We use interval arithmetic as outlined above in our proofs for Lemmas 5, 16, 17, 18, 19 and 20. For the details of the statements proved in this manner, see Appendix C.

Overall, this leads to a limited number of automated proofs, for which manual checking would also be feasible, but tedious and unsatisfying. Instead, we have a clean framework that can be verified, thereby providing a clear structure to an otherwise overwhelming set of arguments. (Note that the total amount of computer checking is considerably less than what we used in our predecessor paper [4], thanks to a more systematic framework, as described in this section.)

5.2 Analysis of Classic Shelf Packing

In several places we make use of the following lemma.

▶ **Lemma 3.** *Any finite sequence of squares with largest square $x < 1/2$ is packed by* Shelf Packing *into a unit square, provided its total area A is at most $\frac{1}{2} + 2(x - \frac{1}{2})^2$.*

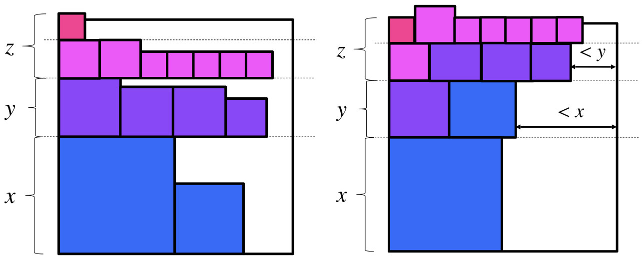

Figure 7 Establishing a density bound by shelf packing.

Proof. Consider Figure 7 (Left) and assume that the last square in the sequence cannot be packed by Shelf Packing, as shown. Then the height of the first square is x; let $y \leq x$ be the height of the second shelf. Let $x + y + z > 1$ be the total height of the arrangement when the last square of the sequence cannot be placed in a feasible shelf and is placed in an additional shelf that exceeds the height of the container. Accounting for the structure of shelf packing, we can conclude (illustrated by Figure 7 (Right)) that the total packed area A is

$$\begin{aligned} A &> z(1-y) + y(1-x) + x^2 \\ &> (1-x-y)(1-y) + y(1-x) + x^2 \\ &= (x - \frac{1}{2})^2 + (y - \frac{1}{2})^2 + \frac{1}{2} \\ &\geq \frac{1}{2} + 2(x - \frac{1}{2})^2, \end{aligned}$$

as claimed. ◀

▶ **Lemma 4** ([9]). *Every sequence $t_1 \geq \cdots \geq t_{u+1}$ of squares with a total area of at most $\frac{hw}{2}$ is packed by* Shelf Packing *into an $h \times w$-rectangle with $t_1 \leq h \leq w$.*

In our proofs, we use the contraposition of Lemma 4: If t_{u+1} is not packed by Shelf Packing into a $h \times w$-rectangle, then the area of the squares $t_1, \ldots, t_{u+1}$ exceeds $1/2 \cdot hw$.

5.3 Analysis of Top Packing

Let $t_1 \geq \cdots \geq t_{u+1}$ be an input sequence to Top Packing that stops with packing t_u.

▶ **Lemma 5.** *Let $t_1 \geq \cdots \geq t_{s+1}$ be an input sequence to* Top Packing *that stops with packing t_s. Let σ denote the side length of the largest square that fits into C_ℓ, as shown in Figure 5 (b). If $t_1 \leq \sigma$, then the total area of the squares packed by* Top Packing *is at least $0.83\sigma^2$.*

Proof. First, we show that the total area of squares packed into C_ℓ is at least $0.415\sigma^2$ separately for all cases considered by Top Packing. Without loss of generality, we assume that the bottom side side of C_ℓ is not smaller than the right, i.e., that we use horizontal shelf packing in C_ℓ. Let t_u denote the last square packed into C_ℓ before the height σ is exceeded. We distinguish different cases depending on the values of t_1, t_{u+1}, σ, u:

(i) $0.645\sigma \leq t_1 \leq \sigma$: Because of $t_1 \leq \sigma$, we know that t_1 is packed into C_ℓ. Furthermore, as $t_1 \geq 0.645\sigma$, the packed area inside C_ℓ is at least $t_1^2 \geq 0.645^2\sigma^2 > 0.415\sigma^2$.

(ii) $t_{u+1} \leq \frac{\sigma}{2\sqrt{2}}$: By applying interval arithmetic, we conclude that t_{u+1} can be packed into C_ℓ by Top Packing; in particular, t_{u+1} can always be packed into C_ℓ such that its bottom side lies on height σ and its right side is aligned with C_ℓ. Because packing t_{u+1} exceeds height σ, Lemma 3 and Observation 6 imply that the total area of t_{u+1} and the squares already packed into C_ℓ is at least $\frac{\sigma^2}{2}$.

(iii) $\frac{\sigma}{2\sqrt{2}} \leq t_1 \leq \frac{\sigma}{2}$ and $\frac{\sigma}{2\sqrt{2}} \leq t_{u+1}(\leq \frac{\sigma}{2})$: At least t_1, t_2, t_3, t_4 are packed into $\mathcal{X}_1$ by shelf packing, implying that the total packed area is at least $4\left(\frac{\sigma}{2\sqrt{2}}\right)^2 = \frac{\sigma^2}{2}$.

(iv) $\frac{\sigma}{2} \leq t_1 \leq 0.645\sigma$, $\frac{\sigma}{2\sqrt{2}} \leq t_{u+1}(\leq 0.645\sigma)$ and $u = 1$. We have $t_{u+1} = t_2$. If t_2 can be packed into C_ℓ, the total area of squares packed into C_ℓ is at least $t_1^2 + t_2^2 \geq t_1^2 + (\sigma - t_1)^2 \geq \frac{\sigma^2}{2}$, because packing t_2 exceeded height σ. In order to show that t_2 can be packed into C_ℓ, we assume w.l.o.g. $t_1 = t_2 = 0.645$ and then prove by applying interval arithmetic that the top right corner always lies inside $\mathcal{D}$.

(v) $\frac{\sigma}{2} \leq t_1 \leq 0.645\sigma$, $\frac{\sigma}{2\sqrt{2}} \leq t_{u+1}(\leq 0.645\sigma)$ and $u \geq 2$: Clearly, the number of constructed subcontainers is either one or two, because $\frac{3\sigma}{2\sqrt{2}} > \sigma$. If only one subcontainer is constructed, the subcontainer has a height of t_1, implying $t_{u+1} > \sigma - t_1$, because t_{u+1} could not be packed into the subcontainer or in the space above the first subcontainer. Hence, t_1 would be the only square packed into the subcontainer, contradicting $u \geq 2$. Thus, two subcontainers are constructed. The second subcontainer contains at least 2 squares, because the height of the second subcontainer is at most $\sigma - t_1 \leq \frac{\sigma}{2}$, i.e., at most the half of its width. Hence, the total area of squares packed into C_ℓ is at least $t_1^2 + 2t_{u+1}^2 \geq \frac{\sigma^2}{2}$.

An analogous argument shows that the total area of squares packed into C_r is at least $0.415\sigma^2$, concluding the proof. ◀

5.4 Analysis of SubContainer Packing

For the analysis of SubContainer Packing, let $C_1, \ldots, C_k$ be the subcontainers constructed by Bottom Packing and let $R_1, \ldots, R_k$ be the maximal rectangles contained in $C_1, \ldots, C_k$, respectively. For $i = 1, \ldots, k$, let h_i and w_i be the height and the width of R_i. Recall that h_i simultaneously denotes the height of C_i and the square p_i that is the first to be packed into C_i. Let t be the largest square that could be packed below C_k. We write $h_{k+1} := t$ and denote the total area of squares packed into C_i by $\|C_i\|$.

In order to use properties of Shelf Packing, we will make use of the following fact.

▶ **Observation 6.** *The total area packed by* SubContainer Packing *into* C_i *is at least the total area packed by* Shelf Packing *into* R_i.

We start by establishing several lower bounds on the area $\|C_i\|$ packed into C_i, yielding the following lower bound B_1. In our proofs, we use the fact that the square p_i of height h_i is packed into C_i.

▶ **Corollary 7.** *If* $w_i \geq 2h_i$, *then*

$$\|C_i\| \geq B_1(h_i, w_i, h_{i+1}) := \max \begin{cases} 1/2 \cdot h_i w_i + 1/4 \cdot h_i^2, & \text{(Lemma 21)} \\ h_i^2 + (w_i - h_i - h_{i+1})h_{i+1}, & \text{(Lemma 22)} \\ 1/2 \cdot h_i(w_i + h_i) - h_{i+1}^2. & \text{(Lemma 23)} \end{cases}$$

We extend the lower bound B_1 provided by Corollary 7 from $w_i \geq 2h_i$ to the general case as follows.

▶ **Lemma 8.**

$$\|C_i\| \geq B_2(h_i, w_i, h_{i+1}) := \begin{cases} h_i^2 & \text{if } w_i < h_i + h_{i+1}, \\ h_i^2 + h_{i+1}^2 & \text{if } \quad h_i + h_{i+1} \leq w_i \leq 2h_i, \\ B_1(h_i, w_i, h_{i+1}) & \text{if } \quad 2h_i < w_i. \end{cases}$$

Proof. If $w_i < h_i + h_{i+1}$, exactly one square of side length h_i is packed into C_i. If $h_i + h_{i+1} \leq w_i \leq 2h_{i+1}$, at least one square with side length of at least h_{i+1} is packed after h_i into C_i. If $2h_{i+1} < w_i$, the Lemma follows from Corollary 7. This concludes the proof. ◀

For later use, we show the following property of B_2.

▶ **Lemma 9.** *For* $u \leq v$, *we have* $B_2(h_i, w_i, u) + u^2 \leq B_2(h_i, w_i, v) + v^2$.

Proof. See Appendix A.4. ◀

Before we are able to state the last lemma of this subsection, we need to define some functions. To do so, we assume that the midpoint of $\mathcal{D}$ is the origin of the coordinate system, see Figure 8(a). Let $T(u)$ denote the side length of a maximal square s that fits into $\mathcal{D}$, when the y-coordinate of the bottom side of s coincides with u. Note that u is allowed to be negative. We have

$$T(u) = \frac{2}{\sqrt{5}}\sqrt{1 - \frac{u^2}{5}} - \frac{4u}{5}.$$

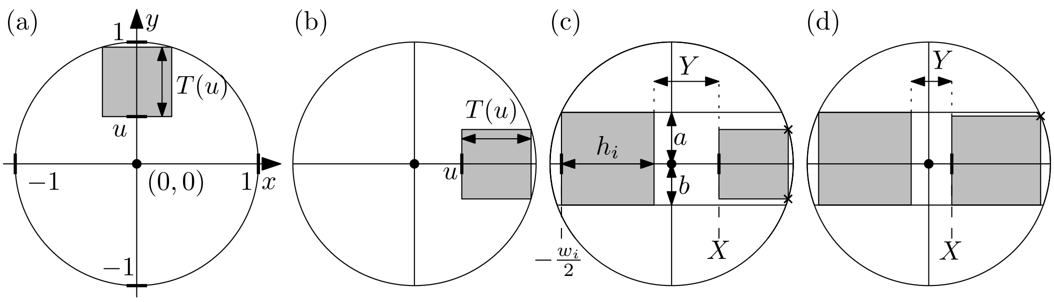

Figure 8 Definition of $T(u)$.

In particular, the inverse function $T^{-1}(s)$ yields the highest possible y-coordinate of the bottom side of a square with side length s, e.g., $y = T^{-1}(s_1)$ is the line that partitions $\mathcal{D}$ into its top and bottom part. Let a, b be the y-coordinates of the upper and the lower side of C_i and let $c := c(a, b) = \min\{a, -b\}$. The case when C_i contains the center of the disk, i.e., $a > 0 > b$, is illustrated in Figure 8(c)+(d). The maximal x-coordinate X of the left edge of a square with side length u is determined by

$$X(a,b,u) := \begin{cases} \sqrt{1-(b+u)^2} - u & \text{if } b \geq 0, \\ \sqrt{1-(a+u)^2} - u & \text{if } a < 0, \\ T^{-1}(u) & \text{if } u \leq 2c, \quad \text{(see Figure 8(c))} \\ \sqrt{1-(u-c)^2} - u & \text{otherwise.} \quad \text{(see Figure 8(d))} \end{cases}$$

$$= \begin{cases} T^{-1}(u) & \text{if } u \leq 2c, \\ \sqrt{1-(u-c)^2} - u & \text{otherwise.} \end{cases}$$

Furthermore, the x-coordinate of the right side of the first square h_i packed into subcontainer C_i is $-(\frac{w_i}{2} - h_i)$. Thus, the extent (along the x-axis) of squares packed behind h_i into C_i is lower bounded by (seeFigure 9(a))

$$Y := Y(a, h_i, w_i, h_{i+1}) := \frac{w_i}{2} - h_i + X(a, a - h_i, h_{i+1}).$$

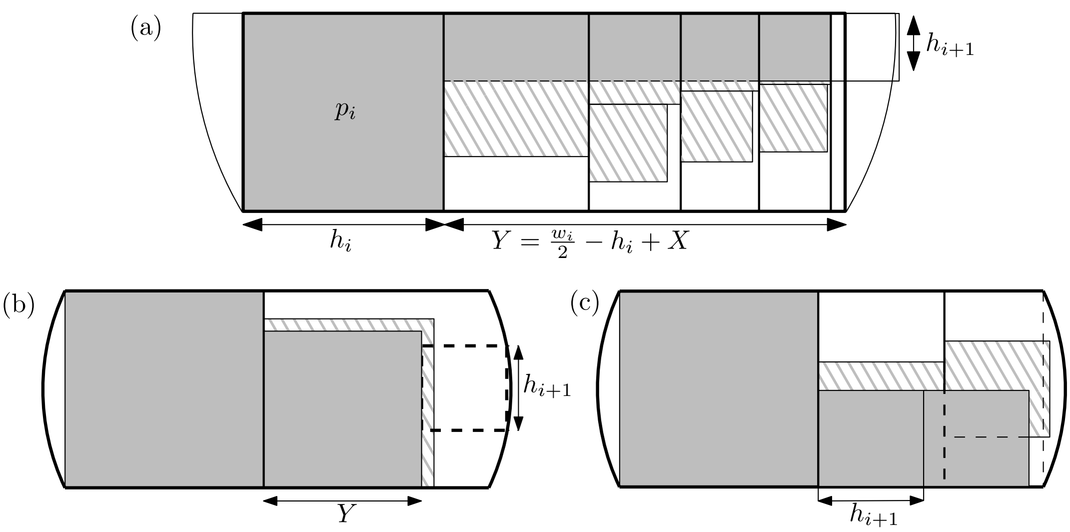

Figure 9 (a) Definition of Y. (b) The lower bound $B_3(c, h_i, w_i, h_{i+1})$ for the case of two squares packed into C_i. (c) The lower bound $B_3(c, h_i, w_i, h_{i+1})$ for the case of at least three squares packed into C_i.

▶ **Lemma 10.** *If $w_i \geq 2h_i$, we have $\|C_i\| \geq B_3(a, h_i, w_i, h_{i+1}) := h_i^2 + \max \begin{Bmatrix} Y \cdot h_{i+1} \\ \min\{Y^2, 2h_{i+1}^2\} \end{Bmatrix}$.*

Proof. The term $h_i^2 + Y h_{i+1}$ is a lower bound for $\|C_i\|$, because Y is a lower bound for the extend (along the x-axis) of squares packed behind h_i into C_i. Furthermore, $h_i^2 + Y^2$ lower bounds $\|C_i\|$ if exactly two squares are packed into C_i, see Figure 9(b), and $h_i^2 + 2h_{i+1}^2$ lower bounds $\|C_i\|$ if at least three squares are packed into C_i, see Figure 9(c). In total, we get

$B_3(a, h_i, w_i, h_{i+1})$ as a lower bound for $\|C_i\|$ if $w_i \geq 2h_i$ ensuring that at least two squares are packed into C_i. ◀

Combining Lemma 8 and Lemma 10 yields a general lower bound for $\|C_i\|$:

▶ **Corollary 11.** $\|C_i\| \geq B_4(a, h_i, w_i, h_{i+1}) := \begin{Bmatrix} h_1^2 + \min\left(\max^2(Y, 0), 2h_2^2\right), & \textit{if } w_i \leq 2h_i \\ \max\begin{Bmatrix} B_2(h_i, w_i, h_{i+1}) \\ B_3(a, h_i, w_i, h_{i+1}) \end{Bmatrix} & \textit{otherwise} \end{Bmatrix}.$

5.5 Analysis of Bottom Packing

Let C_j be the first subcontainer with its top side below the midpoint of $\mathcal{D}$. Let H_j be the vertical distance between the top side of h_j and the lowest point of $\mathcal{D}$.

▶ **Lemma 12.** *Let A_{j+1} be the area of the smallest circular segment containing $C_{j+1}, \ldots, C_k$. The total area of t_{u+1} and squares packed by* Bottom Packing *into $C_j, \ldots, C_k$ is at least $B_5(h_j, H_j) := A_{j+1} + 2h_j^2 - H_j h_j$.*

Proof. See Appendix A.5. ◀

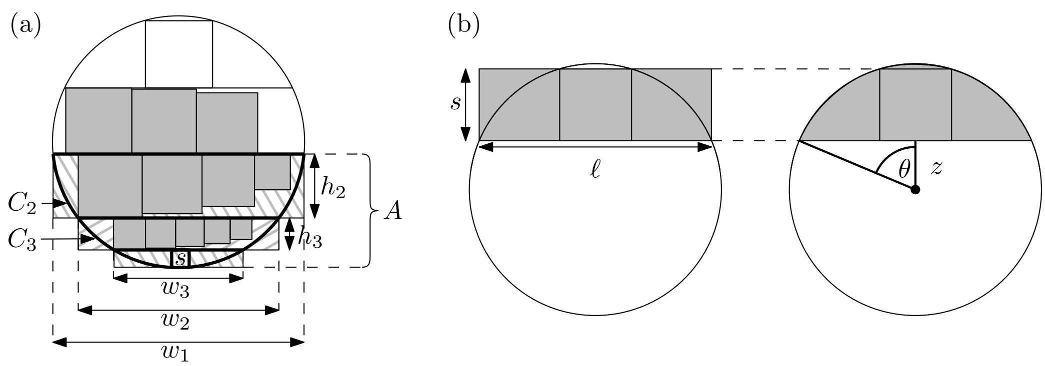

Figure 10 (a) The areas of subcontainers C_2 and C_3 are upper-bounded by the areas of the smallest enclosing rectangles (gray-white hatched rectangles) and the area below $C_k = C_3$ is lower-bounded by the area of the $s \times w_3$-rectangle (gray-white hatched rectangle) below C_3. (b) The area of the top of $\mathcal{D}$ is upper-bounded by the area of the smallest rectangle having height s and the width of the top of $\mathcal{D}$.

6 Correctness of the Algorithm

Based on the tools provided in Section 5, we can proceed to establish the main result.

6.1 Analysis of Steps 1. and 2. of the Algorithm

For the remaining analysis, we consider an input sequence $s_1 \geq \cdots \geq s_n$ to the overall algorithm, which stops with packing s_{n-1}, and fails to pack s_n.

▶ **Lemma 13.** *If $s_1 \leq 0.295$, Step 1. of the overall algorithm packs an area of at least $\frac{8}{5}$.*

Proof. Consider a configuration scaled down by a factor of 1.388^{-1}, such that $\mathcal{X}$ is the unit square. As s_5 is the first square packed by Shelf Packing into $\mathcal{X}$, Lemma 3 implies that the total area packed into $\mathcal{D}$ is at least $4s_5^2 + \frac{1}{2} + 2(s_5 - \frac{1}{2})^2$. The derivative of the function $f(x) = 4x^2 + \frac{1}{2} + 2(x - \frac{1}{2})^2$ is $12x - 2$, so $f(x)$ is minimized for $x_{\min} = 1/6$, with $f(x_{\min}) = \frac{8.027}{5} 1.388^2$, showing that this is at least $\frac{8}{5 \cdot 1.388^2}$ for $s_5 \leq \frac{0.295}{1.388}$ in the scaled configuration, which means that for $s_5 \leq 0.295$ in the original configuration, the total area of $s_1, \ldots, s_n$ is at least $\frac{8}{5}$. ◀

For the remainder of the analysis, we assume $s_1 \geq 0.295$.

▶ **Lemma 14.** *If $s_1 \leq \frac{1}{\sqrt{2}}$ and $s_1^2 + s_2^2 + s_3^2 + s_4^2 \geq \frac{39}{25}$, Step 2. of the overall algorithm packs an area of at least $\frac{8}{5}$.*

Proof. Assume the total area of $s_1, \ldots, s_n$ is at most $\frac{8}{5}$. The total area of the squares s_1, s_2, s_3, s_4 is at least $\frac{39}{25} = \frac{8}{5} - \frac{1}{25}$. Hence, the total area of the remaining squares is at most $\frac{1}{25}$. As $\mathcal{X}$ has an area of at least $\frac{2}{25}$, the remaining squares are packed by Shelf Packing into $\mathcal{X}$, contradicting that our algorithms stops without packing s_n. Thus, the total area is larger than $\frac{8}{5}$, concluding the proof. ◀

6.2 Analysis of Step 3.

For Step 3., we distinguish cases depending on the number of subcontainers.

Let σ denote the side length of the largest squares that fits into the pockets C_ℓ and C_r to the left and to the right of s_1, packed as far as possible to the top. Furthermore, we denote the total area of $s_1, \ldots, s_n$ by S and by z the largest square that can be packed below the last subcontainer constructed by Bottom Packing, implying $z < s_n$. Otherwise, our algorithm would not have stopped with packing s_{n-1}.

Due to space limitations, we present two of the (shorter) proofs to provide general idea of the arguments; full details for all lemmas can be found in the appendix.

▶ **Lemma 15.** *If no square is packed by* Bottom Packing*, then $s_1, \ldots, s_n$ are packed by* Top Packing *into $\mathcal{D}$.*

Proof. Suppose the algorithm fails when constructing a first subcontainer below the top segment. This implies that placing s_n as far as possible to the bottom yields a placement for which s_1 and s_n overlap. However, the minimum value for $s_1^2 + s_n^2$ for two overlapping squares packed into a disk container is attained for $s_1 = s_n$. This corresponds to the worst-case configuration, implying that the total area of s_1 and s_n is larger than $\frac{8}{5}$. ◀

Thus, in the remainder if the paper we assume that at least one subcontainer is constructed.

▶ **Lemma 16.** *The total input area is larger than $\frac{8}{5}$ if exactly one subcontainer is constructed.*

Proof. See Appendix B. ◀

▶ **Lemma 17.** *The total input area is larger than $\frac{8}{5}$ if exactly two subcontainers are constructed.*

Proof. We distinguish whether $s_n < \sigma$ or not.

If $s_n < \sigma$, we combine Lemmas 5, 8, and 9 and Corollary 11 with $z < s_n$ in order to lower bound S by $s_1^2 + B_4(a, h_1, w_1, h_2) + B_2(h_2, w_2, z) + z^2 + 0.83\sigma^2$, which we lower bound by $\frac{8}{5}$ by using interval arithmetic (see Lemma 32 (1)).

If $S_n \geq \sigma$, we combine Lemmas 8 and 9 and Corollary 11 with $\max\{\sigma, z\} < s_n$ in order to lower bound S by $s_1^2 + B_4(a, h_1, w_1, h_2) + B_2(h_2, w_2, \max\{\sigma, z\}) + \max\{\sigma, z\}^2$, which we lower bound by $\frac{8}{5}$ by using interval arithmetic (see Lemma 32 (2)), concluding the proof. ◀

▶ **Lemma 18.** *The total input area is larger than $\frac{8}{5}$ if exactly three subcontainers are constructed.*

Proof. See Appendix B. ◀

▶ **Lemma 19.** *The total input area is larger than $\frac{8}{5}$ if exactly four subcontainers are constructed.*

Proof. See Appendix B. ◀

▶ **Lemma 20.** *The total input area is larger than $\frac{8}{5}$ if at least five subcontainers are constructed.*

Proof. See Appendix B. ◀

By combining Lemmas 13 to 20, we conclude that in any case the total area of the input squares $s_1, \ldots, s_n$ is larger than $\frac{8}{5}$. This implies that any set of squares with a total area no larger than $\frac{8}{5}$ can be packed by our overall algorithm, concluding the proof of Theorem 2.

7 Conclusions

We have established the critical density for packing squares into a disk, based on a number of advanced techniques that are more involved than the ones used for packing squares or disks into a square. Numerous questions remain, in particular the critical density for packing squares of bounded size into a disk or the critical density of packing squares into a disk. These remain for future work; we are optimistic that some of our techniques will be useful.

References

1 A. T. Becker, S. P. Fekete, P. Keldenich, S. Morr, and C. 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`.

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

3 E. D. Demaine, S. Fekete, and R. 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.

4 S. P. Fekete, P. Keldenich, and C. 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.

5 S. P. Fekete, S. Morr, and C. Scheffer. Split packing: Algorithms for packing circles with optimal worst-case density. *Discrete & Computational Geometry*, 61(3):562–594, 2019.

6 S. P. Fekete and J. Schepers. New classes of fast lower bounds for bin packing problems. *Mathematical Programming*, 91(1):11–31, 2001.

7 S. P. Fekete and J. Schepers. A general framework for bounds for higher-dimensional orthogonal packing problems. *Mathematical Methods of Operations Research*, 60:311–329, 2004.

8 J. Y. T. Leung, T. W. Tam, C. S. Wong, G. H. Young, and F. Y. L. Chin. Packing squares into a square. *Journal of Parallel and Distributed Computing*, 10(3):271–275, 1990.

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

10 S. 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.

A Details of Section 5

In this section, we present the omitted proofs of Section 5.

A.4 Analysis of SubContainer Packing

We prove Corollary 7 with the help of the following three lemmas.

▶ **Lemma 21.** *Let $t_1 \geq \cdots \geq t_{u+1}$ be an input sequence for* SubContainer Packing *in order to pack C_i. If* SubContainer Packing *stops with packing t_u into C_i, $t_1 \leq h_i \leq w_i$, and $w_i \notin (3/2 \cdot t_1, 2t_1)$, then the total area of $t_1, \dots, t_u$ is at least $1/2 \cdot t_1 w + 1/4 \cdot t_1^2$.*

Proof. Because of $t_1 \leq h_i \leq w_i$, we know that t_1 is packed. If $w_i \leq 3/2 \cdot t_1$, then $t_1^2 \geq 1/2 \cdot t_1 w_i + 1/4 \cdot t_1^2$ and hence, the lemma holds. Thus, it remains to consider the case $w_i \geq 2t_1$. We distinguish whether $t_{u+1} \leq 1/2 \cdot t_1$ or not.

If $t_{u+1} \leq 1/2 \cdot t_1$, we consider the rectangle remaining after removing t_1. Because t_{u+1} is not packed by SubContainer Packing into C_i, Lemma 4 and Observation 6 imply that the total area of $t_2, \dots, t_{u+1}$ is at least $1/2 \cdot (w - t_1)t_1$. Consequently, the total area of $t_1, \dots, t_{u+1}$ is at least $1/2 \cdot t_1 w + 1/2 \cdot t_1^2$.

Thus, we assume w.l.o.g. $t_{u+1} > 1/2 \cdot t_1$. Let $\delta > 0$ such that $t_{u+1} = 1/2 \cdot t_1 + \delta$ implying that the total area of $t_1, \dots, t_k$ is at least $\frac{t_1 w}{2} + \frac{t_1^2}{4} + (w - t_1 - y)\delta + \frac{t_1^2}{4} - \frac{t_1 y}{2}$ where $y := (w - t_1 - \ldots - t_u)$. Thus, it suffices to show $(w - t_1 - y)\delta \geq \frac{t_1}{2}\left(y - \frac{t_1}{2}\right)$ for which we further distinguish whether $w \geq 2t_1 + \delta$ or not. If $w \geq 2t_1 + \delta$, we obtain $w - t_1 - y \geq \frac{t_1}{2}$. Furthermore, as $y \leq t_{u+1}$, we have $\delta \geq y - \frac{t_1}{2}$ implying $(w - t_1 - y)\delta \geq \frac{t_1}{2}\left(y - \frac{t_1}{2}\right)$.

Thus, we assume w.l.o.g. $t_{u+1} > \frac{t_1}{2}$. Let $\delta > 0$ such that $t_{u+1} = \frac{t_1}{2} + \delta$, implying that the total area of $t_1, \dots, t_k$ is at least $\frac{t_1 w}{2} + \frac{t_1^2}{4} + (w - t_1 - y)\delta + \frac{t_1^2}{4} - \frac{t_1 y}{2}$, where $y := (w - t_1 - \cdots - t_u)$. Thus, it suffices to show $(w - t_1 - y)\delta \geq \frac{t_1}{2}\left(y - \frac{t_1}{2}\right)$, for which we further distinguish whether $w \geq 2t_1 + \delta$ or not. If $w \geq 2t_1 + \delta$, we obtain $w - t_1 - y \geq \frac{t_1}{2}$. Furthermore, as $y \leq t_{u+1}$, we have $\delta \geq y - \frac{t_1}{2}$, implying $(w - t_1 - y)\delta \geq \frac{t_1}{2}\left(y - \frac{t_1}{2}\right)$.

Hence, we assume w.l.o.g. $2t_1 \leq w \leq 2t_1 + \delta$, clearly implying $u = 2$. The total area of t_1, t_2 is lower-bounded by $t_1^2 + t_3^2 = t_1^2 + \left(\frac{t_1}{2} + \delta\right)^2$, which is at least $\frac{t_1^2}{4} + (2t_1 + \delta)\frac{t_1}{2}$. This, in turn, is lower-bounded by $1/4 t_1^2 + \frac{1}{2} t_1 w$, because $w \leq 2t_1 + \delta$. This concludes the proof. ◀

▶ **Lemma 22.** *The total area of squares packed by* SubContainer Packing *into C_i is at least $h_i^2 + h_{i+1}(w_i - h_i - h_{i+1})$.*

Proof. The square that failed to be packed into R_i has a side length of h_{i+1}, see Figure 11. Thus, the length of that part of the bottom side of R_i that is not covered by squares packed into R_i is smaller than h_{i+1}. Denoting the first square packed into C_i by p_i, the total area packed into $C_i \setminus p_i$ is at least $h_{i+1}(w_i - h_i - h_{i+1})$, because the height of all squares packed into R_i is at least h_{i+1}, concluding the proof.

◀

▶ **Lemma 23.** *The total area of squares packed by* SubContainer Packing *into C_i is at least $1/2 \cdot h_i(w_i + h_i) - h_{i+1}^2$ if $w_i \geq 2h_i$.*

Proof. We consider the rectangle $R_i' = R_i \setminus p_i$ of height h_i and width $w_i - p_i = w_i - h_i \geq h_i$. Analogous to Observation 6, the total area packed by SubContainer Packing into $C_i \setminus p_i$ is at least the total area packed by Shelf Packing into $R_i \setminus p_i$. Consequently, Lemma 4 implies that the total area packed by SubContainer Packing into R_i' is at least $\frac{1}{2} h_i(w_i - h_i) - h_{i+1}^2$. Thus, the total area packed into C_i is at least $h_i^2 + \frac{1}{2} h_i(w_i - h_i) - h_{i+1}^2 = \frac{1}{2} h_i w_i + \frac{1}{2} h_i^2 - h_{i+1}^2$. ◀

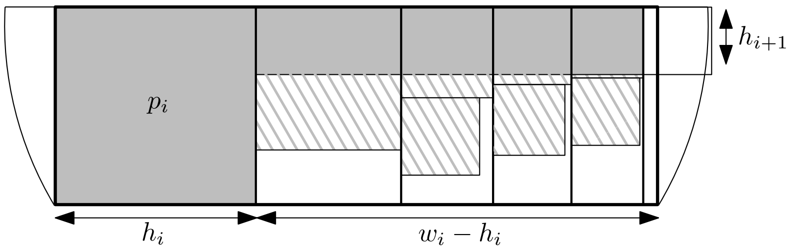

Figure 11 A lower bound (gray) for the area (hatched) packed by SUBCONTAINER PACKING into a subcontainer.

Combining Lemmas 21 to 23 yields the following lower bound B_1 for the total area $\|C_i\|$ packed into C_i:

▶ **Corollary 7.** *If $w_i \geq 2h_i$, then*

$$\|C_i\| \geq B_1(h_i, w_i, h_{i+1}) := \max \begin{cases} 1/2 \cdot h_i w_i + 1/4 \cdot h_i^2, & \textit{(Lemma 21)} \\ h_i^2 + (w_i - h_i - h_{i+1})h_{i+1}, & \textit{(Lemma 22)} \\ 1/2 \cdot h_i(w_i + h_i) - h_{i+1}^2. & \textit{(Lemma 23)} \end{cases}$$

A.4.1 Proof of Lemma 9

We now present the proof of Lemma 9.

▶ **Lemma 9.** *For $u \leq v$, we have $B_2(h_i, w_i, u) + u^2 \leq B_2(h_i, w_i, v) + v^2$.*

Proof. We distinguish whether (1) $w_i < h_i + v$, (2) $h_i + v \leq w_i < 2h_i$, or (3) $2h_i \leq w_i$ holds.

(1) $w_i < h_i + v$ implies $w_i < h_i + u$, because $u \leq v$ which implies $B_2(h_i, w_i, u) = B_2(h_i, w_i, v) = h_i^2$.

(2) $h_i + v \leq w_i < 2h_i$ implies $B_2(h_i, w_i, v) = h_i^2 + v^2$. Furthermore, by definition of B_2 (considering the relevant ranges), we have $B_2(h_i, w_i, u) \leq h_i^2 + u^2$, implying $B_2(h_i, w_i, u) + u^2 \leq B_2(h_i, w_i, v) + v^2$.

(3) $2h_i \leq w_i$: As $2h_i \leq w_i$, we have $B_2(h_i, w_i, h_{i+1}) = B_1(h_i, w_i, h_{i+1})$.

If $B_1(h_i, w_i, h_{i+1}) = \frac{h_i w_i}{2} + \frac{h_i^2}{4}$, the lemma follows, because $\frac{h_i w_i}{2} + \frac{h_i^2}{4} + h_{i+1}^2$ is monotonically increasing in h_{i+1} for $h_{i+1} \geq 0$.

If $B_1(h_i, w_i, h_{i+1}) = h_i^2 + (w_i - h_i - h_{i+1})\, h_{i+1}$, we have $B_1(h_i, w_i, h_{i+1}) + f_{i+1}^2 = h_i^2 + (w_i - h_i)\, h_{i+1}$, which is monotonically increasing in h_{i+1}, because $w_i - h_i \geq 0$.

If $B_1(h_i, w_i, h_{i+1}) = \frac{h_i w_i}{2} + \frac{h_i}{2} - h_{i+1}^2$, the lemma follows $\frac{h_i w_i}{2} + \frac{h_i}{2} - h_{i+1}^2 + h_{i+1}^2 = \frac{h_i w_i}{2} + \frac{h_i}{2}$ does not depend on h_{i+1}. This concludes the proof. ◀

A.5 Analysis of Bottom Packing

Let C_j be the first subcontainer with its top side below the midpoint of $\mathcal{D}$. Let H_j be the vertical distance between the top side of h_j and the bottom most point of $\mathcal{D}$.

▶ **Lemma 12.** *Let A_{j+1} be the area of the smallest circular segment containing $C_{j+1}, \ldots, C_k$. The total area of t_{u+1} and squares packed by* BOTTOM PACKING *into $C_j, \ldots, C_k$ is at least $B_5(h_j, H_j) := A_{j+1} + 2h_j^2 - H_j h_j$.*

Proof. Applying Lemma 22 to $C_j, \ldots, C_k$ yields that the total area of s_n and the squares packed into $C_i, \ldots, C_k$ is at least $t_{u+1}^2 + \sum_{i=j}^{k} h_i^2 + (w_i - h_i - h_{i+1})h_{i+1} =: T$.

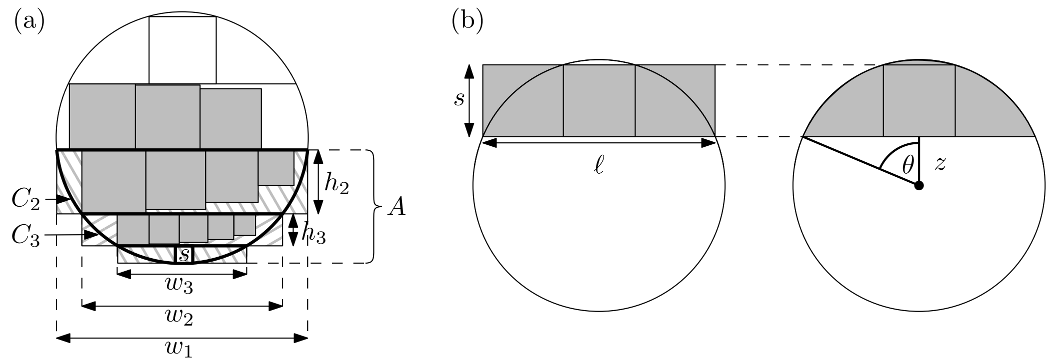

Figure 12 (a) The areas of subcontainers C_2 and C_3 are upper-bounded by the areas of the smallest enclosing rectangles (gray-white hatched rectangles) and the area below $C_k = C_3$ is lower-bounded by the area of the $s \times w_3$-rectangle (gray-white hatched rectangle) below C_3. (b) The area of the top of $\mathcal{D}$ is upper-bounded by the area of the smallest rectangle having height s and the width of the top of $\mathcal{D}$.

For $i = j+1, \ldots, k$, we upper bound the area of each subcontainer C_i by the area $h_i w_{i-1}$ of the smallest enclosing rectangle, see Figure 12(a). Furthermore, Lemma 24 implies that the area of the circular segment below C_k but containing t is upper-bounded by tw_k. Thus, A_{j+1} is at most $\sum_{i=j+1}^{k} h_i w_{i-1} + tw_k$, which is smaller than $\sum_{i=j+1}^{k} h_i w_{i-1} + t_{u+1} w_k$, because Bottom Packing fails with packing $t_{u+1} > t$. Hence, T is larger than $A_{j+1} + t_{u+1}^2 + \sum_{i=j}^{k} h_i^2 - h_i h_{i+1} - h_{i+1}^2 = A_{j+1} + t_{u+1}^2 + h_j^2 - \sum_{i=j}^{k-1} h_i h_{i+1} - h_{k+1}^2$, which is equal to $A_{j+1} + h_j^2 - \sum_{i=j}^{k-1} h_i h_{i+1}$, because $t_{u+1} = h_{k+1}$. This is at least $A_{j+1} + 2h_j^2 - H_j h_j$, because $h_i \geq h_{i+1}$. This concludes the proof. ◀

The following auxiliary Lemma 24 is used in the proof of Lemma 12.

▶ **Lemma 24.** *Consider a square $s \leq 0.6$ packed as far as possible to the top into $\mathcal{D}$, see Figure 12(b). Let ℓ be the length of the intersection of $\mathcal{D}$ with the line induced by the bottom side of s. The area B of the smallest circular segment containing s is at most ℓs, i.e., upper bounded by the area A of the $\ell \times s$-rectangle.*

Proof. Clearly, we know $A = \ell s = 2s \sin\theta$ and $B = cos^{-1}(t) - t\sqrt{1-t^2}$ with $\cos\theta = t = \sqrt{1 - \frac{s^2}{4}} - s$, using Pythagoras' theorem. Now $t = \sqrt{1 - \frac{s^2}{4}} - s \geq 1 - \frac{s^2}{4} - s \geq 1 - \frac{s}{4} - s$, because $s \leq 1$, $s \geq \frac{4}{5}(1 - \cos\theta)$. Now, we have $A = 2s \sin\theta \geq \frac{8}{5} \sin\theta(1 - \cos\theta)$, from the previous result. Rewriting B in terms of θ, we have $B = \theta - \sin\theta\cos\theta$. Using the above, $A - B \geq \frac{8}{5}\sin\theta - \frac{3}{5}\sin\theta\cos\theta - \theta =: f(\theta)$. Clearly, $f(0) = 0$ and $f^{'}(\theta) = \frac{2}{5}(1 - \cos\theta)(\cos\theta - \frac{1}{3})$. So $f(\theta) \geq 0$ for $\theta \leq cos^{-1}(\frac{1}{3})$. In the contest of this proof, the largest possible θ occurs at $s = 0.6$ and is equal to 1.209 (as compared to $\cos^{-1}(\frac{1}{3}) = 1.230$). Hence, $A - B \geq 0$ for $s \leq 0.6$. ◀

B Details of Section 6

In this section, we present the omitted proofs of Section 6.

B.2 Analysis of Step 3.

By Lemma 15, we may assume in the remainder of the paper we assume that at least one subcontainer is constructed. Recall that the rectangle R_i of the i-th subcontainer C_i is the largest rectangle inside the subcontainer with height h_i and width w_i. The square that determines the height h_{i+1} of C_{i+1} is the first square that cannot be packed into C_i. Thus, we define $h_{k+1} := s_n$, which is the first square that cannot be packed by Bottom Packing, i.e., by the overall algorithm.

In order to distinguish whether Top Packing has packed squares into the pockets C_ℓ and C_r to the left and to right of s_1, we consider the function $E(s_n)$ induced by Lemma 5 and defined as follows.

$$E(s_n) := \begin{Bmatrix} 0.83\sigma^2 & \text{if } s_n \leq \sigma, \\ 0 & \text{otherwise.} \end{Bmatrix}$$

B.2.1 Analysis for One Subcontainer

▶ **Lemma 16.** *The total input area is larger than $\frac{8}{5}$ if exactly one subcontainer is constructed.*

Proof. If $s_n \leq \sigma$, combining Lemmas 5 and 8 yields $S \geq s_1^2 + B_2(h_1, w_1, s_n) + s_n^2 + 0.83\sigma^2$, which we lower bound by $s_1^2 + B_2(h_1, w_1, z) + z^2 + 0.83\sigma^2$ by applying $z < s_n$ and Lemma 9. Finally, by applying interval arithmetic (see Lemma 31 (1)), we prove $s_1^2 + B_2(h_1, w_1, z) + z^2 + 0.83\sigma^2 > \frac{8}{5}$. Thus, we assume w.l.o.g. that $s_n > \sigma$, implying that no square is packed by Top Packing.

If $n = 2$, only s_1 is packed, implying that no subcontainer is constructed, which is a contradiction to the assumption that one subcontainer is constructed.

If $n = 3$, we have $S \geq s_1^2 + h_1^2 + s_3^2$, which we lower bound by $\frac{8}{5}$ by applying interval arithmetic (see Lemma 31 (2)).

If $n = 4$, we know that S is lower-bounded by $s_1^2 + s_4^2 + h_1^2 + Y(T^{-1}(s_1), h_1, s_4)^2$, because exactly three squares are packed, i.e., two squares are packed into C_1. We distinguish whether $s_1 > \frac{1}{\sqrt{2}}$ or not. If $s_1 > \frac{1}{\sqrt{2}}$, we use interval arithmetic (see Lemma 31 (3)) for showing $s_1^2 + s_4^2 + h_1^2 + Y(R(s_1), h_1, s_4)^2 \geq \frac{8}{5}$. If $s_1 \leq \frac{1}{\sqrt{2}}$, we have $s_1^2 + s_2^2 + s_3^2 + s_4^2 < \frac{39}{25}$. Using interval arithmetic (see Lemma 31 (4)), we show that $s_1^2 + s_2^2 + s_3^2 + s_4^2 > \frac{39}{25}$ holds, which contradicts the assumption that not all squares are packed.

If $n \geq 5$, we consider $s_1^2 + h_1^2 + 3s_n^2 \geq s_1^2 + h_1^2 + 3\max^2\{\sigma, z\}$ as a lower bound for S and use interval arithmetic (see Lemma 31 (5)) for proving $s_1^2 + h_1^2 + 3\max^2\{\sigma, z\} > \frac{8}{5}$. This concludes the proof. ◀

B.2.2 Analysis for Three Subcontainers

▶ **Lemma 18.** *The total input area is larger than $\frac{8}{5}$ if exactly three subcontainers are constructed.*

Proof. We distinguish whether $s_n < \sigma$ or not. If $s_n < \sigma$, we combine Lemmas 5, 8, and 9 and Corollary 11 with $z < s_n$ in order to lower bound S by

$$\begin{aligned} & s_1^2 + z^2 + 0.83\sigma^2 + B_4(T^{-1}(s_1), h_1, w_1, h_2) \\ & \qquad + B_4(T^{-1}(s_1) - h_1, h_2, w_2, h_3) + B_2(h_3, w_3, z), \end{aligned}$$

which we lower bound by $\frac{8}{5}$ using interval arithmetic (see Lemma 33 (1)).

If $s_n \geq \sigma$, we combine Lemmas 8 and 9 and Corollary 11 with $r < s_n$ in order to lower bound S by

$$s_1^2 + B_4(T^{-1}(s_1), h_1, w_1, h_2) + B_4(T^{-1}(s_1) - h_1, h_2, w_2, h_3) + B_2(h_3, w_3, m_3) + m_3^2,$$

where $m_3 := \max(\sigma, z)$. We then lower bound S by $\frac{8}{5}$ using interval arithmetic (see Lemma 33 (2)), thus concluding the proof. ◀

B.2.3 Analysis for Four Subcontainers

▶ **Lemma 19.** *The total input area is larger than $\frac{8}{5}$ if exactly four subcontainers are constructed.*

Proof. We distinguish whether $s_n < \sigma$ or not.

If $s_n \geq \sigma$, we combine Lemmas 8 and 9 with $r < s_n$ in order to lower bound S by $s_1^3 + \sum_{i=1}^{2} B_2(h_i, w_i, h_{i+1}) + B_2(h_4, w_4, r) + \sigma^2$, which we lower bound by $\frac{8}{5}$ by using interval arithmetic (see Lemma 34 (1)).

If $s_n < \sigma$, we distinguish whether the y-coordinate y_2 of the bottom side of C_2 is positive or not. If $y_2 < 0$, we combine Lemmas 5, 8 and 12 in order to lower bound S by $s_1^2 + 0.83\sigma^2 + B_4(T^{-1}(s_1), h_1, w_1, h_2) + B_4(T^{-1}(s_1) - h_1, h_2, w_2, h_3) + B_5(h_3, H_3)$, which we lower bound by $\frac{8}{5}$ by applying interval arithmetic (see Lemma 34 (2)). If $y_2 > 0$, we consider the first subcontainer C_{j+1}, i.e., with minimal index j, such that the y-coordinate of the top side of C_{j+1} is negative and define the following lower bound $B_6(h_2, w_2, h_{j+1})$ for the total area $\|C_2\|$ packed into C_2.

$$\|C_j\| \geq B_6(h_j, w_j, h_{j+1})$$
$$:= \begin{cases} \max\left\{ \begin{array}{c} \frac{h_j w_j}{2} + \frac{h_j^2}{4}, \\ h_j^2 + (w_j - h_j - h_{j+1})h_{j+1} \end{array} \right\} & \text{if } w_j \geq 3h_j, \ \text{(Lemmas 9 and 21)} \\ \frac{h_j w_j}{2} + \frac{w_j^2}{4} & \text{if } 2h_j \leq w_j < 3h_j, \ \text{(Lemma 21)} \\ h_j^2 & \text{if } w_j < 2h_j. \ \text{(one square)} \end{cases}$$

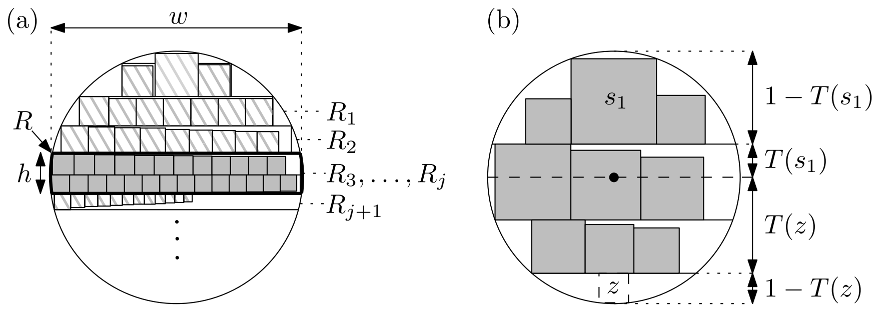

Figure 13 The case in which the first two subcontainers C_1 and C_2 lie above the midpoint of $\mathcal{D}$.

Furthermore, we consider the space R between C_2 and C_{j+1}, see Figure 13(a). Note that R does not have to be a single subcontainer but the union of the subcontainers $C_3, \ldots, C_j$. We denote the height and width of the largest rectangle $F \subset R$ by H and W and observe that the area $\|R\|$ of squares packed into R is lower-bounded by

$$B_7(H, W, h_{j+1}) \quad = \quad \begin{Bmatrix} \frac{HW}{2} + \frac{h_{j+1}^2}{4} & \text{if} & W \geq 2H, \\ \frac{HW}{2} & \text{if} & W < 2H. \end{Bmatrix}$$

In particular, if $W \geq 2H$ we have $w_i \geq 2h_i$ for all $i = 3, \ldots, j$. Lemma 21 implies that $\|R\|$ is at least $\sum_{i=3}^{j} \frac{h_i w_i}{2} + \frac{h_i^2}{4}$, which is lower-bounded by $\frac{HW}{2} + \frac{h_{j+1}^2}{4}$. As $W < 2H$, the total area packed into C_i is at least $\frac{h_i w_i}{2}$ for $i = 3, \ldots, j$. Thus, the area packed into R in total is at least $\sum_{i=3}^{j} \frac{h_i w_i}{2}$, which is lower-bounded by $\frac{HW}{2}$.

Summarizing the above, we lower-bound the total area S packed into $\mathcal{D}$ by $s_1^2 + B_2(h_1, w_1, h_2) + B_6(h_2, w_2, h_{j+1}) + B_5(h_{j+1}, H_{j+1}) + B_7(H, W, h_{j+1})$, which we lower bound by $\frac{8}{5}$ by using interval arithmetic (see Lemma 34 (3)), concluding the proof. ◀

B.2.4 Analysis for at Least Five Subcontainers

▶ **Lemma 20.** *The total input area is larger than $\frac{8}{5}$ if at least five subcontainers are constructed.*

Proof. If $s_n \leq \sigma$, we apply the same approach that we used for the case of four subcontainers in the proof of Lemma 18.

If $s_n > \sigma$, we observe that at most seven subcontainers are constructed. Next, we define some bounds on the heights h_i and the widths w_i of the subcontainers $C_1, \ldots, C_k$, see Claims 25, 26, 27, 28, 29.

▷ Claim 25. $L(h_i) := \max(\frac{2T^{-1}(s_1)-(i-1)s_1}{k+1-i}, \sigma(s_1))$ is a lower bound for h_i for $i = 1, \ldots, k$.

Proof. Consider $m := \frac{2T(s_1)-(i-1)s_1}{k+1-i}$. For the sake of contradiction, we assume $h_i < m$. The length 2 of the vertical diameter of $\mathcal{D}$ can be reformulated as follows, see Figure 13(b):

$$\begin{aligned}
2 \quad &= \quad 1 - T^{-1}(s_1) + 1 - T^{-1}(z) - \sum_{j=1}^{N} h_j \\
&\overset{z \leq s_1, h_i \leq s_1}{\leq} \quad 2 - 2T^{-1}(s_1) + \sum_{j=1}^{i-1} s_1 + \sum_{j=i}^{k} h_i \\
&\overset{h_i < m}{<} \quad 2 - 2T^{-1}(s_1) + (i-1)s_1 + \sum_{j=i}^{k} m \\
&= \quad 2 - 2T^{-1}(s_1) + (i-1)s_1 + (k+1-i)m \\
&\overset{\text{definition of } m}{=} \quad 2.
\end{aligned}$$

This is a contradiction, implying $h_i \geq m$. Furthermore, we have $s_n > \sigma$, implying $h_i \geq \sigma$ for all $i = 1, \ldots, k$. Hence, we have $h_i \geq L(h_i)$, concluding the proof of Claim 25. ◀

▷ Claim 26. $U(h_i) := \begin{Bmatrix} \max\left\{\frac{T^{-1}(s_1)+T^{-1}(\sigma)-(k-i-1)\sigma}{i}, s_1\right\} & \text{if} & 1 \leq i \leq k-1 \\ s_1 & \text{if} & i = k \end{Bmatrix}$ is an upper bound for h_i for $i = 1, \ldots, k$.

Proof. We define $m := \frac{T^{-1}(s_1)+T^{-1}(\sigma)-(k-i-1)\sigma}{i}$. For the sake of contradiction, we assume $h_i > m$. The length 2 of the vertical diameter of $\mathcal{D}$ can be reformulated as follows, see

Figure 13(b):

$$\begin{aligned}
2 &= 1 - T^{-1}(s_1) + 1 - T^{-1}(z) - \sum_{j=1}^{N} h_j \\
&\overset{h_i \leq h_j}{\geq} 1 - T^{-1}(s_1) + 1 - T^{-1}(z) + \sum_{j=1}^{i} h_i + \sum_{j=i+1}^{k} h_j \\
&= 1 - T^{-1}(s_1) + 1 - T^{-1}(z) + ih_i + \sum_{j=i+1}^{k-1} (h_j) + h_k \\
&\overset{w_j > \sigma}{\geq} 1 - T^{-1}(s_1) + ih_i + (k-i-1)\sigma + 1 - T^{-1}(z) + h_k \\
&\overset{1-T^{-1}(z)+h_k < 1-T^{-1}(h_k)}{\geq} 1 - T^{-1}(s_1) + im + (k-i-1)\sigma + 1 - T^{-1}(h_k) \\
&\overset{\text{By definition of } m}{>} 1 - T^{-1}(s_1) + im + (k-i-1)\sigma + 1 - T^{-1}(\sigma) \\
&= 2.
\end{aligned}$$

This is a contradiction, implying $h_i \leq m$. Furthermore, by definition we have $h_i < s_1$. Hence, we have $h_i \leq U(h_i)$, concluding the proof of Claim 26. ◀

▷ Claim 27. $U(H_i) = \max\left\{T^{-1}(s_1) + T^{-1}(r) - (N-i-1)\sigma, \sum_{j=1}^{i} U(h_j)\right\}$ is an upper bound on $\sum_{j=1}^{i} h_j$ for $1 \leq i \leq k-1$.

Proof. The length 2 of the vertical diameter of $\mathcal{D}$ can be reformulated as follows, see Figure 13(b):

$$\begin{aligned}
2 &= 1 - T^{-1}(s_1) + 1 - T^{-1}(z) + \sum_{j=1}^{N} h_j \\
&= 1 - T^{-1}(s_1) + 1 - T^{-1}(z) + \sum_{j=1}^{i} h_j + \sum_{j=i+1}^{k-1} (h_j) + h_k \\
&\overset{h_{k-1} \geq \sigma}{\geq} 1 - T^{-1}(s_1) + \sum_{j=1}^{i} h_j + (k-i-1)\sigma + 1 - T^{-1}(z) + h_k \\
&\overset{1-T^{-1}(z)+h_k < 1-T^{-1}(h_k)}{\geq} 1 - T^{-1}(s_1) + \sum_{j=1}^{i} h_j + (k-i-1)\sigma + 1 - T^{-1}(h_k) \\
&\overset{h_{k-1} \geq \sigma}{\geq} 1 - T^{-1}(s_1) + \sum_{j=1}^{i} h_j + (k-i-1)\sigma + 1 - T^{-1}(\sigma),
\end{aligned}$$

which is equivalent to

$$\sum_{j=1}^{i} h_j \quad \leq \quad T^{-1}(s_1) + T^{-1}(\sigma) - (k-i-1)\sigma.$$

This concludes the proof of Claim 27. ◀

▷ Claim 28. $L(H_i) := \sum_{j=1}^{i} L(h_j)$ is a lower bound on $\sum_{j=1}^{i} h_j$ for $1 \leq i \leq k$.

Proof. Follows by Claim 25. ◀

In order to define the next bound, we need to define the length $\ell(y) := 2\sqrt{1-y^2}$ of the intersection of $\mathcal{D}$ with a horizontal line with a y-coordinate of y.

▷ Claim 29. $L(w_i) := \left\{ \begin{matrix} \min\left\{ \begin{Bmatrix} \ell(T^{-1}(s_1) - L(H_i)) \\ \ell(T^{-1}(s_1) - U(H_i)) \end{Bmatrix} \right\} & \text{if} & 1 \leq i \leq N-1 \\ \sigma & \text{if} & i = N \end{matrix} \right\}$ is a lower bound on w_i.

Proof. The y-coordinate of the upper side of C_i is at most $T^{-1}(s_1) - L(H_{i-1})$. The y-coordinate of the lower side of C_i is at least $T^{-1}(s_1) - U(H_i)$. Hence, $w_i \geq \min\{\ell(T^{-1}(s_1) - L(H_i)), \ell(T^{-1}(s_1) - U(H_i))\}$, concluding the proof. ◀

Using the bounds established in Claims 25, 26, 27, 28, 29, we lower bound the total area S packed into $\mathcal{D}$ as follows:

$$\begin{aligned} S &\geq s_1^2 + \sum_{i=1}^{k} \left(h_i^2 + \max\{w_i - h_i - h_{i+1}, 0\}h_{i+1}\right) + s_n^2 \\ &\geq s_1^2 + \sum_{i=1}^{k} \left(h_i^2 + \max\{w_i - h_i, h_{i+1}\}h_{i+1} - w_{i+1}^2\right) + s_n^2 \\ &= s_1^2 + h_1^2 - s_n^2 + \sum_{i=1}^{k} \max\{w_i - h_i, h_{i+1}\} + s_n^2 \\ &\geq s_1^2 + L(h_1)^2 + \sum_{i=1}^{k} (\max\{L(w_i) - U(h_{i+1}), L(h_{i+1})\}L(h_{i+1})) =: \Lambda. \end{aligned}$$

The lower bound Λ exclusively depends on s_1. We observe that by Lemma 13 we are already allowed to assume w.l.o.g. that $s_1 \geq 0.295$. Furthermore, s_1 is upper-bounded by the root of $T^{-1}(s_1) + T^{-1}(\sigma) - (k-1)\sigma = 0$, i.e., s_1 is required to fulfill $T^{-1}(s_1) + T^{-1}(\sigma) > (k-1)\sigma$. Note that the value of σ exclusively depends on s_1. Above we already observed that at most seven subcontainers are constructed due to $s_n > \sigma$ and $s_1 \geq 0.295$, i.e., we have $k \in \{5, 6, 7\}$. Finally, by using interval arithmetic (see Lemma 35) for each case $k = 5, 6, 7$ we lower bound Λ by $\frac{8}{5}$, concluding the proof of Lemma 20. ◀

By combining Lemmas 13 to 20, we conclude that in any case the total area of the input squares $s_1, \ldots, s_n$ is larger than $\frac{8}{5}$. This implies that any set of squares with a total area no larger than $\frac{8}{5}$ can be packed by our overall algorithm, concluding the proof of Theorem 2.

C Summary of all Interval Arithmetic Proofs

In all our proofs using interval arithmetic, we include a variable for s_1, the side length of the largest square. We assume $0.295 \le s_1$; otherwise, we are in Case 1 of our algorithm where all squares are small. Moreover, we assume $s_1 \le \sqrt{8/5}$; otherwise, s_1 alone is larger than the area we have to pack. For proving Lemma 5, we use the following lemma.

▶ **Lemma 30** (Automatic Analysis for Lemma 5). *We prove the following using interval arithmetic.*

(1) *Let* $\ell_1 := \sqrt{1 - T^{-1}(s_1)^2} - \frac{s_1}{2}$ *and*

$$F_1(s_1) := \begin{cases} \left(\frac{s_1}{2} + \frac{\sigma(s_1)}{2\sqrt{2}}\right)^2 + \left(T^{-1}(s_1) + \sigma(s_1) + \frac{\sigma(s_1)}{2\sqrt{2}}\right)^2, & \text{if } s_1 > \ell_1, \\ \left(\frac{s_1}{2} + \sigma(s_1) + \frac{\sigma(s_1)}{2\sqrt{2}}\right)^2 + \left(T^{-1}(s_1) + \frac{\sigma(s_1)}{2\sqrt{2}}\right)^2, & \text{otherwise.} \end{cases}$$

Then, for all $0.295 \le s_1 \le \sqrt{8/5}$, $F_1(s_1) \le 1$.

(2) *Let* $\alpha := 0.645$ *and*

$$F_2(s_1) := \begin{cases} \left(\frac{s_1}{2} + 2\alpha \cdot \sigma(s_1)\right)^2 + \left(\alpha \cdot \sigma(s_1) + T^{-1}(s_1)\right)^2, & \text{if } s_1 \le \ell_1, \\ \left(\frac{s_1}{2} + \alpha \cdot \sigma(s_1)\right)^2 + \left(T^{-1}(s_1) + 2\alpha \cdot \sigma(s_1)\right)^2, & \text{otherwise.} \end{cases}$$

Then, for all $0.295 \le s_1 \le \sqrt{8/5}$, $F_2(s_1) \le 1$.

Let $w(y_t, h) := 2\min\left(\sqrt{1-y_t^2}, \sqrt{1-(h-y_t)^2}\right)$. Note that $w(y_t, h)$ is the maximum width of a rectangle with top side at $y = y_t$ and height h that fits into the unit disk $\mathcal{D}$. For proving Lemma 16, we use the following lemma.

▶ **Lemma 31** (Automatic Analysis for Lemma 16). *We prove the following using interval arithmetic.*

(1) *Let* $z := T(T^{-1}(s_1) + h_1)$ *and* $w_1 := w(T^{-1}(s_1), h_1)$. *Let*

$$F_3(s_1, h_1) := s_1^2 + B_2(h_1, w_1, z) + z^2 + 0.83 \cdot \sigma^2(s_1).$$

For all s_1, h_1 *with* $0 \le z \le h_1 \le s_1$ *and* $h_1 \le T^{-1}(s_1) + 1$, *we have* $F_3(s_1, h_1) > 8/5$.

(2) *Let* $F_4(s_1, h_1, s_n) := s_1^2 + h_1^2 + s_n^2$. *Let* $w_1 := w(T^{-1}(s_1), h_1)$ *and*

$$Y_1 := Y(T^{-1}(s_1), h_1, w_1, s_n).$$

For all s_1, h_1 *with* $Y_1 \le 0 \le z \le s_n \le h_1 \le s_1$ *and* $h_1 \le T^{-1}(s_1) + 1$, *we have* $F_4(s_1, h_1) > 8/5$.

(3) *Let* $F_5(s_1, h_1, s_n) := s_1^2 + h_1^2 + s_n^2 + Y_1^2$. *For all* s_1, h_1, s_n *with* $s_1 \ge \frac{1}{\sqrt{2}}$, $h_1 \le T^{-1}(s_1) + 1$, $0 \le z \le s_n \le h_1 \le s_1$ *and* $\sigma(s_1) \le s_n \le Y_1 \le h_1$, *we have* $F_5(s_1, h_1, s_n) > 8/5$.

(4) *Let* $F_6(s_1, h_1, s_n) := F_5(s_1, h_1, s_n)$. *For all* s_1, h_1, s_n *with* $0.295 \le s_1 \le \frac{1}{\sqrt{2}}$, $0 \le z \le s_n \le h_1 \le s_1$ *and* $\sigma(s_1) \le s_n \le Y_1 \le h_1 \le T^{-1}(s_1) + 1$, *we have* $F_6(s_1, h_1, s_n) > \frac{39}{25}$.

(5) *Let* $F_7(s_1, h_1) := s_1^2 + h_1^2 + 3 \cdot \max^2(\sigma(s_1), z)$. *For all* s_1, h_1 *with* $0 \le z \le h_1 \le s_1$ *and* $h_1 \le 1 + T^{-1}(s_1)$, *we have* $F_7(s_1, h_1) > 8/5$.

For proving Lemma 17, we use the following lemma.

▶ **Lemma 32** (Automatic Analysis for Lemma 17). *We prove the following using interval arithmetic.*

(1) *Let* $w_2 \coloneqq w(T^{-1}(s_1) - h_1, h_2)$ *and* $z_2 \coloneqq T(-T^{-1}(s_1) + h_1 + h_2)$. *Let*

$$F_8(s_1, h_1, h_2) \coloneqq s_1^2 + 0.83 \cdot \sigma^2(s_1) + B_4(T^{-1}(s_1), h_1, w_1, h_2) + B_2(h_2, w_2, z_2) + z_2^2.$$

For all s_1, h_1, h_2 *with* $0 \le z_2 \le h_2 \le h_1 \le s_1$, $z < \sigma(s_1)$, $h_1 \le T^{-1}(s_1) + 1$ *and* $h_2 \le 1 + T^{-1}(s_1) - h_1$, *we have* $F_8(s_1, h_1, h_2) > 8/5$.

(2) *Let* $m \coloneqq \max(z_2, \sigma(s_1))$ *and*

$$F_9(s_1, h_1, h_2) \coloneqq s_1^2 + B_4\left(T^{-1}(s_1), h_1, w_1, h_2\right) + B_2(h_2, w_2, m) + m^2.$$

For all s_1, h_1, h_2 *with* $0 \le z_2 \le h_2 \le h_1 \le s_1$, $\sigma(s_1) \le h_2$, $h_1 \le T^{-1}(s_1) + 1$ *and* $h_2 \le 1 + T^{-1}(s_1) - h_1$, *we have* $F_9(s_1, h_1, h_2) > 8/5$.

For proving Lemma 18, we use the following lemma.

▶ **Lemma 33** (Automatic Analysis for Lemma 18)**.** *We prove the following using interval arithmetic.*

(1) *Let* $z_3 \coloneqq T(-T^{-1}(s_1) + h_1 + h_2 + h_3)$, $w_3 \coloneqq w(T^{-1}(s_1) - h_1 - h_2, h_3)$ *and*

$$\begin{aligned} F_{10}(s_1, h_1, h_2, h_3) \coloneqq\, & s_1^2 + 0.83\sigma^2(s_1) + B_4(T^{-1}(s_1), h_1, w_1, h_2) + \\ & B_4(T^{-1}(s_1) - h_1, h_2, w_2, h_3) + B_2(h_3, w_3, z_3) + z_3^2. \end{aligned}$$

Then, for all s_1, h_1, h_2, h_3 *with* $0 \le z_3 \le h_3 \le h_2 \le h_1 \le s_1$, $h_1 \le T^{-1}(s_1) + 1$, $h_2 \le 1 + T^{-1}(s_1) - h_1$ *and* $h_3 \le 1 + T^{-1}(s_1) - h_1 - h_2$, *we have* $F_{10}(s_1, h_1, h_2, h_3) > 8/5$.

(2) *Let* $m_3 \coloneqq \max(\sigma(s_1), z_3)$ *and*

$$\begin{aligned} F_{11}(s_1, h_1, h_2, h_3) \coloneqq\, & s_1^2 + B_4(T^{-1}(s_1), h_1, w_1, h_2) + B_4(T^{-1}(s_1) - h_1, h_2, w_2, h_3) + \\ & B_2(h_3, w_3, m_3) + m_3^2. \end{aligned}$$

Then, for all s_1, h_1, h_2, h_3 *with* $0 \le m_3 \le h_3 \le \cdots \le h_1 \le s_1$, $h_1 \le T^{-1}(s_1) + 1$, $h_2 \le 1 + T^{-1}(s_1) - h_1$ *and* $h_3 \le 1 + T^{-1}(s_1) - h_1 - h_2$, *we have* $F_{11}(s_1, h_1, h_2, h_3) > 8/5$.

For proving Lemma 19, we use the following lemma.

▶ **Lemma 34** (Automatic Analysis for Lemma 19)**.** *Analogous to the previous lemmas, let* $w_4 \coloneqq w(T^{-1}(s_1) - h_1 - h_2 - h_3, h_4)$ *and* $z_4 \coloneqq T(-T^{-1}(s_1) + h_1 + h_2 + h_3 + h_4)$. *We prove the following using interval arithmetic.*

(1) *Let* $F_{12}(s_1, h_1, h_2, h_3, h_4) \coloneqq s_1^2 + B_2(h_4, w_4, \sigma(s_1)) + \sigma^2(s_1) + \sum_{i=1}^{3} B_2(h_i, w_i, h_{i+1})$. *Then, for all* s_1, h_1, h_2, h_3, h_4 *with* $0 \le \max(z_4, \sigma(s_1)) \le h_4 \le \cdots \le h_1 \le s_1$, $h_1 \le T^{-1}(s_1) + 1$, $h_2 \le 1 + T^{-1}(s_1) - h_1$, $h_3 \le 1 + T^{-1}(s_1) - h_1 - h_2$ *and* $h_4 \le 1 + T^{-1}(s_1) - h_1 - h_2 - h_3$, *we have* $F_{12}(s_1, h_1, h_2, h_3, h_4) > \frac{8}{5}$.

(2) *Let* $H_3 \coloneqq T^{-1}(s_1) - h_1 - h_2 + 1$ *and*

$$\begin{aligned} F_{13}(s_1, h_1, h_2, h_3) \coloneqq\, & s_1^2 + 0.83\sigma^2(s_1) + B_4(T^{-1}(s_1), h_1, w_1, h_2) + \\ & B_4(T^{-1}(s_1) - h_1, h_2, w_2, h_3) + B_5(h_3, H_3). \end{aligned}$$

Then, for all s_1, h_1, h_2, h_3 *with* $0 \le h_3 \le h_2 \le h_1 \le s_1$, $h_1 \le T^{-1}(s_1) + 1$, $h_2 \le 1 + T^{-1}(s_1) - h_1$, $h_3 \le 1 + T^{-1}(s_1) - h_1 - h_2$ *and* $H_3 \le 1$, *we have* $F_{13}(s_1, h_1, h_2, h_3) > \frac{8}{5}$.

(3) *Let*

$$\begin{aligned} F_{14}(s_1, h_1, h_2, h, y) \coloneqq\, & s_1^2 + B_2(h_1, w_1, h_2) + B_6(h_2, w_2, h) + \\ & B_7(H, W, h) + B_5(h, 1 - y), \end{aligned}$$

where $H \coloneqq T^{-1}(s_1) - h_1 - h_2 + y$ *and* $W \coloneqq w(T^{-1}(s_1) - h_1 - h_2, H)$. *Then, for all* s_1, h_1, h_2, h, y *with* $0 \le h \le h_2 \le h_1 \le s_1$, $T^{-1}(s_1) - h_1 - h_2 \ge 0$ *and* $y \le h_2$, *we have* $F_{14}(s_1, h_1, h_2, h, y) > \frac{8}{5}$.

Finally, for proving Lemma 20, we use the following lemma.

▶ **Lemma 35** (Automatic Analysis for Lemma 20). *Let* $N \in \{5,6,7\}$. *Let* $h(x) \coloneqq 2\sqrt{1-x^2}$,

$$b_i(s_1) \coloneqq \begin{cases} \max\left(\frac{2T^{-1}(s_1)-(i-1)s_1}{N+1-i}, \sigma(s_1)\right), & \text{if } 1 \le i \le N, \\ \sigma(s_1), & \text{if } i = N+1, \end{cases}$$

$$\beta_i(s_1) \coloneqq \begin{cases} \min\left(\frac{T^{-1}(s_1)+T^{-1}(\sigma(s_1))-(N-i-1)\cdot\sigma(s_1)}{i}, s_1\right), & \text{if } 1 \le i \le N-1, \\ s_1, & \text{if } i = N, \end{cases}$$

$$\rho_i(s_1) \coloneqq \sum_{j=1}^{i} b_j(s_1),$$

$$S_i(s_1) \coloneqq \min\left(\sum_{j=1}^{i} \beta_j(s_1),\ T^{-1}(s_1) + T^{-1}(\sigma(s_1)) - (N-i-1)\cdot\sigma(s_1)\right),$$

$$h_i(s_1) \coloneqq \begin{cases} \min\left(h\left(T^{-1}(s_1) - \rho_{i-1}(s_1)\right),\ h\left(T^{-1}(s_1) - S_i(s_1)\right)\right), & \text{if } 1 \le i \le N-1, \\ \sigma(s_1), & \text{if } i = N, \text{ and} \end{cases}$$

$$F_{15}(s_1) \coloneqq s_1^2 + b_1(s_1)^2 + \sum_{i=1}^{N} \max\left((h_i(s_1) - \beta_i(s_1))\cdot b_{i+1}(s_1),\ b_{i+1}^2(s_1)\right).$$

We use interval arithmetic to prove $F_{15}(s_1) > \frac{8}{5}$ *for each* $N \in \{5,6,7\}$ *and all* s_1 *with* $T^{-1}(s_1) + T^{-1}(\sigma(s_1)) > (N-1)\cdot\sigma(s_1)$ *and*

$$\forall 1 \le i \le (N-1) : \max\left(\left(T^{-1}(s_1) - \rho_{i-1}(s_1)\right)^2, \left(T^{-1}(s_1) - S_i(s_1)\right)^2\right) \le 1.$$