On the Complexity of Anchored Rectangle Packing

Antonios Antoniadis
Universität des Saarlandes,
Saarbrücken, Germany
Max-Planck-Institut für Informatik,
Saarbrücken, Germany
aantonia@mpi-inf.mpg.de

Felix Biermeier
Universität Hamburg, Germany
felix.biermeier@uni-hamburg.de

Andrés Cristi
Universidad de Chile, Santiago, Chile
andres.cristi@ing.uchile.cl

Christoph Damerius
Universität Hamburg, Germany
christoph.damerius@uni-hamburg.de

Ruben Hoeksma
Universität Bremen, Germany
University of Twente, Enschede, The Netherlands
hoeksma@uni-bremen.de

Dominik Kaaser
Universität Hamburg, Germany
dominik.kaaser@uni-hamburg.de

Peter Kling
Universität Hamburg, Germany
peter.kling@uni-hamburg.de

Lukas Nölke
Universität Bremen, Germany
noelke@uni-bremen.de

Abstract

In the *Anchored Rectangle Packing (ARP)* problem, we are given a set of points P in the unit square $[0,1]^2$ and seek a maximum-area set of axis-aligned interior-disjoint rectangles $\mathcal{S}$, each of which is anchored at a point $p \in P$. In the most prominent variant – *Lower-Left-Anchored Rectangle Packing (LLARP)* – rectangles are anchored in their lower-left corner. Freedman [19, Unsolved Problem 11, page 345] conjectured in 1969 that, if $(0,0) \in P$, then there is a LLARP that covers an area of at least 0.5. Somewhat surprisingly, this conjecture remains open to this day, with the best known result covering an area of 0.091 [11]. Maybe even more surprisingly, it is not known whether LLARP – or any ARP-problem with only one anchor – is NP-hard.

In this work, we first study the *Center-Anchored Rectangle Packing (CARP)* problem, where rectangles are anchored in their center. We prove NP-hardness and provide a PTAS. In fact, our PTAS applies to any ARP problem where the anchor lies in the interior of the rectangles. Afterwards, we turn to the LLARP problem and investigate two different resource-augmentation settings: In the first we allow an ε-perturbation of the input P, whereas in the second we permit an ε-overlap between rectangles. For the former setting, we give an algorithm that covers at least as much area as an optimal solution of the original problem. For the latter, we give an $(1-\varepsilon)$-approximation.

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

Keywords and phrases anchored rectangle, rectangle packing, resource augmentation, PTAS, NP, hardness

Digital Object Identifier 10.4230/LIPIcs.ESA.2019.8

Acknowledgements We thank the organizers of the Bremen-Hamburg workshop on algorithms, combinatorics, and optimization for organizing the open problem session where we first learned of the Freedman conjecture. In particular, we thank Nicole Megow from University of Bremen for posing the conjecture during that session and for valuable suggestions. Parts of this work were a result of the UHH & UChile TCS Workshop 2018.

1 Introduction

The *lower-left-anchored rectangle packing (LLARP)* problem was first posed in 1969 by Freedman [19]: Given the unit square $U := [0,1]^2$ in the plane and a finite set of points $P \subseteq U$, find a maximum-area set of axis-aligned and interior-disjoint rectangles $\mathcal{S}$, such that for each $p \in P$, there is exactly one (possibly trivial) rectangle $R \in \mathcal{S}$ that has p in its lower-left corner. It was conjectured [19, 20] that the area covered is at least 0.5 as long as $(0,0) \in P$. The first and so far only result that achieves a constant lower bound covers an area of at least 0.091 [11].

A natural generalization of the LLARP problem looks at anchors different from the lower-left corner. Such an *anchoring* can be defined by a pair $(\alpha, \beta) \in [0,1]^2$. The (α, β)-ARP problem then looks for a maximum-area rectangle set as above but with rectangles anchored in the relative position (α, β) (see Section 2). For example, $(0,0)$-ARP is LLARP, while $(1/2, 1/2)$-ARP requires all rectangles to have a $p \in P$ at their center. We refer to the latter as *center-anchored rectangle packing (CARP)* problem.

Our Contribution. Even though the formulation of LLARP and Freedman's conjecture [19], date back to 1969 it is still not even known whether its decision variant is NP-hard. This is why, in this work, we focus on the complexity of related ARP problems and introduce a resource augmentation setting for LLARP.

When looking at the complexity of CARP, a difference that stands out compared to LLARP is that CARP allows one to simulate "non-expandable" points: By putting four points at the corners of a tiny ε-sized square, none of their rectangles' side lengths can exceed ε. One can think of these four points as one input point that cannot be used as an anchor but still restricts the expansion of other rectangles. These non-expandable points turn out to be a valuable asset, as they can be used to build walls and inject additional geometry into the problem. We exploit this in an elaborate construction that encodes maximum independent set into a CARP instance, proving NP-hardness of CARP (cf. Section 4). The construction of non-expandable points seems difficult or even impossible for LLARP and is the main obstacle in transferring our NP-hardness proof to that setting.

The NP-hardness is complemented by a polynomial-time approximation scheme (PTAS), which extends also to any anchoring $(\alpha, \beta) \in (0,1)^2$. The PTAS is based on a carefully constructed input instance for a related problem called *maximum weight independent set of rectangles* (MWISR) and the usage of a known PTAS in a resource augmentation setting of MWISR (cf. Section 3).

With respect to the classical LLARP problem, we initiate the investigation of the problem with resource augmentation. We study two such settings. In the first, the algorithm is allowed to slightly perturb the input points. In the second, the rectangle set produced by the algorithm may have some (bounded) overlap. In both cases, the resulting solution is compared to an optimal solution without these augmentations. For the first setting, we develop an algorithm that produces an anchored rectangle set of total area no less than that of an optimal solution of LLARP without resource augmentation (cf. Section 5). Our analysis is combinatorial in nature and consists of transforming the optimal solution to a feasible solution for a perturbed instance by using a specific linear program with totally unimodular incidence matrix. For the second setting, we provide an algorithm that covers at least $(1-\varepsilon)$ times the area of an optimal solution (cf. Section 5). Our algorithm is based on the one for the first setting.

Further Related Work. The best known polynomial time algorithm for the LLARP problem is due to Dumitrescu and Tóth [11] and covers an area of at least 0.091. Since the optimal solution cannot cover more than an area of 1 (the whole unit square), their analysis also implies a 0.091-approximation. In the setting where, instead of arbitrary rectangles, squares must be used, Balas et al. [5] achieve an approximation ratio of 1/3. They also consider a setting where the algorithm can choose for each rectangle from multiple anchors (the four corners), giving a $(7/12 - \varepsilon)$-approximation algorithm, where $\varepsilon = |P|^{-1}$, and a QPTAS for rectangles, as well as a 9/47-approximation algorithm and a PTAS for squares. The QPTAS and PTAS extend to the lower-left anchored variants. Recently, Akitaya et al. [4] gave the first NP-hardness result for an ARP variant where only squares may be packed and each square can be anchored at any of its four corners. They also show that, for any instance consisting of finitely many input points inside U, the union of all feasible anchored square packings covers an area of at least 1/2. Finally, if rectangles can be anchored at any of their four corners but input points are restricted to the boundary of U, Biedl et al. [9] give a polynomial-time algorithm (based on maximum independent set for a specific class of graphs).

ARP problems fall within the more general setting of packing axis-aligned and interior-disjoint rectangles in a rectangular container. This setting captures several important and well studied optimization problems. See, for example, the (NP-hard) problems 2D-knapsack and strip packing [2, 6], as well as maximum area independent set of rectangles [3, 7, 8]. Similar problems have also been formulated by Radó and Rado [16, 13, 14, 15]. These problems differ from ARP in that the size of the packed objects is part of the input and not controllable and, in some cases, there is no anchoring of the rectangles.

2 Preliminaries

We start with some general notation. Consider a point $p \in \mathbb{R}^2$ and an axis-aligned rectangle $R \subseteq \mathbb{R}^2$ with its lower-left corner in $(x, y) \in \mathbb{R}^2$. Let w and h denote the width and height of R, respectively. For a pair $(\alpha, \beta) \in [0, 1]^2$ (called *anchoring*), we say R is (α, β)-*anchored* in p if $p = (x + \alpha \cdot w, y + \beta \cdot h)$. We also say that p (α, β)-*spans* R. If the anchoring (α, β) is clear from the context we omit it. We use $\mathcal{A}(R) = w \cdot h$ to denote the area of rectangle R. Similarly, given a set $\mathcal{S}$ of rectangles we define $\mathcal{A}(\mathcal{S})$ as the area of $\bigcup_{R \in \mathcal{S}} R$. We use $U = [0, 1]^2$ to denote the unit square. If not stated otherwise, any rectangle is assumed to be axis-aligned.

The Anchored Rectangle Packing Problem. We now define the *anchored rectangle packing (ARP)* problem with respect to an anchoring $(\alpha, \beta) \in [0, 1]^2$. An instance consists of a finite set $P \subseteq U$ of n points. A *valid rectangle set* (also *solution*) $\mathcal{S}$ for the ARP instance P is a set of interior-disjoint rectangles that contains one (possibly zero-sized) rectangle R_p for each $p \in P$ such that R_p is (α, β)-anchored in p and does not contain any point from $P \setminus \{ p \}$ in its interior. The goal is to find a solution $\mathcal{S}$ for P that covers as much area as possible. We use $\mathcal{S}^*_{\mathrm{ARP}(P)}$ to denote a valid rectangle set of maximum area $\mathrm{OPT}_{\mathrm{ARP}(P)} \coloneqq \mathcal{A}(\mathcal{S}^*_{\mathrm{ARP}(P)})$ and omit ARP and/or P if it is clear from context.

We will mostly consider two specific anchorings. When using the anchoring $(0, 0)$, we refer to the problem by the name *lower-left-anchored rectangle packing (LLARP)*. When the anchoring is $(1/2, 1/2)$, we refer to the problem by the name *center-anchored rectangle packing (CARP)*.

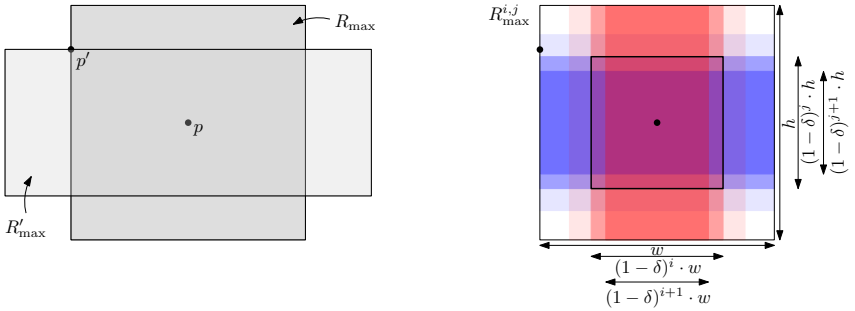


Figure 1 Maximal rectangles $R_{\max}$ and $R'_{\max}$ and candidate rectangles $R^{i,j}_{\max}$ derived from $R_{\max}$ for the anchoring $(1/2, 1/2)$.

3 A PTAS for ARP with Fractional Anchorings

In the following, we give a PTAS for the anchored rectangle packing problem for any anchoring $(\alpha, \beta) \in (0,1)^2$. It is based on a result for *maximum weight independent set of rectangles (MWISR)*[1] that uses resource augmentation similar to that in Theorem 8. An instance of MWISR consists of a set of n axis-aligned, weighted rectangles. The goal is to compute a maximum-weight subset of pairwise interior-disjoint rectangles. In the relaxed variant δ-*MWISR* (for fixed $\delta > 0$), rectangles must be non-overlapping only *after* shrinking them by a factor of $1 - \delta$. Here, Adamaszek et al. [1] give an algorithm that, in time $n^{(\delta\varepsilon)^{-\mathcal{O}(1/\varepsilon)}}$, computes a solution which is within a $(1+\varepsilon)$-factor of the optimum MWISR. While the original algorithm considers shrinking only around the rectangles' centers, this can be generalized such that it also works when shrinking around an arbitrary anchoring $(\alpha, \beta) \in (0,1)^2$. However, this does not generalize to anchors on the boundaries (in particular, not to LLARP), since [1] requires shrinking around a rectangle's center, this way the shrunken rectangle would no longer contain its anchor.

▶ **Theorem 1.** *For any fixed anchoring $(\alpha, \beta) \in (0,1)^2$, the anchored rectangle packing problem admits a polynomial-time approximation scheme.*

Proof. Consider an instance $P \subseteq U$ of (α, β)-ARP with n input-points and fix $\varepsilon > 0$. Denote by ε', δ positive constants which we fix later and define $N := \lceil \log_{1-\delta}(\delta/n) \rceil + 1$.

For each point pair $p, p' \in P$, there are at most two inclusion-wise maximal rectangles that are anchored in p and have p' on their boundary, one where p' is on the left (right) side of the rectangle and one where p' is on its bottom (top) side. Thus, there are at most $2n$ inclusion-wise maximal rectangles that are anchored in a point $p \in P$ and at most $2n^2$ overall. For any such *maximal rectangle* $R_{\max}$, construct N^2 *candidate rectangles* $(R^{i,j}_{\max})_{i,j=0}^{N-1}$ by scaling (around the anchor) by a factor of $(1-\delta)^i$ along the horizontal and $(1-\delta)^j$ along the vertical axis. Denote the set of all such candidate rectangles by $\mathcal{C}$. See Figure 1 for an illustration.

In order to obtain a $(1+\varepsilon)$-approximation for the ARP instance P, we first apply the algorithm for δ-MWISR [1] to the MWISR instance $(\mathcal{C}, \mathcal{A})$ (i.e., all candidate rectangles weighted by their area). This yields a rectangle set $\mathcal{S}'$ with $\mathcal{A}(\mathcal{S}') \geq (1+\varepsilon')^{-1} \cdot \mathrm{OPT}_{\mathrm{MWISR}}$,

[1] A similar connection to MWISR has recently been used [5] to obtain a PTAS for packing squares and a QPTAS for packing rectangles when the anchors are allowed to lie in any of the four corners (and may be different for each rectangle).

where $\mathrm{OPT}_{\mathrm{MWISR}}$ denotes the area covered by an optimal solution for the instance $(\mathcal{C}, \mathcal{A})$ of MWISR. Note that each rectangle $R \in \mathcal{S}'$ is anchored in some point $p \in P$. By definition of δ-MWISR, scaling each rectangle in $\mathcal{S}'$ by a factor of $1-\delta$ in each dimension around its anchor produces a set $\mathcal{S}$ of pairwise interior-disjoint rectangles with

$$\mathcal{A}(\mathcal{S}) \geq (1-\delta)^2(1+\varepsilon')^{-1} \cdot \mathrm{OPT}_{\mathrm{MWISR}}. \tag{1}$$

Moreover, since the scaling happens around the anchors, the rectangles are still anchored in their associated input-points. It follows that, after possibly adding some trivial rectangles, the set $\mathcal{S}$ is a solution for the ARP instance P. Note that for fixed δ and ε', the total time spent to obtain $\mathcal{S}$ is polynomial in n. The bottleneck here is the computation of $\mathcal{S}'$ from the $|\mathcal{C}| \leq 2n^2 \cdot N^2$ candidate rectangles. By [1, Theorem 1], this can be done in time $n^{(\delta\varepsilon')^{-O(1/\varepsilon')}}$.

We now bound the area $\mathrm{OPT}_{\mathrm{ARP}}$ of an optimal (α,β)-ARP solution $\mathcal{S}^*_{\mathrm{ARP}}$ in terms of $\mathrm{OPT}_{\mathrm{MWISR}}$. To this end, fix a rectangle $R^*_p \in \mathcal{S}^*_{\mathrm{ARP}}$ with anchor point $p \in P$. This rectangle is contained in at least one maximal candidate rectangle $R_{\max}$. Let $i, j \in \{0, 1, \ldots, N-1\}$ be maximal such that $R^*_p \subseteq R^{i,j}_{\max}$ and note that $\mathcal{A}(R^*_p) \leq \mathcal{A}(R_{\max}) \cdot (1-\delta)^{i+j}$. We say that R^*_p is *negligible* if $i = N-1$ or $j = N-1$. For each non-negligible $R^*_p \in \mathcal{S}^*_{\mathrm{ARP}}$, define $R_p := R^{i+1,j+1}_{\max}$ and denote by $\mathcal{S}''$ the set of all such rectangles. We consider the contribution of non-negligible and negligible rectangles to $\mathrm{OPT}_{\mathrm{ARP}}$ separately.

- By construction, the contribution of non-negligible rectangles is at most $(1-\delta)^{-2} \cdot \mathcal{A}(\mathcal{S}'')$. Furthermore, since $\mathcal{S}'' \subseteq \mathcal{C}$ and as the rectangles in $\mathcal{S}''$ are pairwise non-overlapping (as a shrunken subset of an ARP solution), $\mathcal{S}''$ is a solution to the MWISR instance $\mathcal{C}$. This implies that $\mathcal{A}(\mathcal{S}'') \leq \mathrm{OPT}_{\mathrm{MWISR}}$, which bounds the contribution of non-negligible rectangles to $\mathrm{OPT}_{\mathrm{ARP}}$ by $(1-\delta)^{-2} \cdot \mathrm{OPT}_{\mathrm{MWISR}}$.
- To bound the contribution of negligible rectangles, fix a negligible $R^*_p \in \mathcal{S}^*_{\mathrm{ARP}}$ and note that $\mathcal{A}(R^*_p) \leq \mathcal{A}(R_{\max}) \cdot (1-\delta)^{N-1} \leq \mathrm{OPT}_{\mathrm{MWISR}} \cdot (1-\delta)^{N-1} \leq \mathrm{OPT}_{\mathrm{MWISR}} \cdot \delta/n$, where the penultimate inequality holds since $\{R_{\max}\}$ is a valid MWISR solution. Since there are at most n negligible rectangles in $\mathcal{S}^*_{\mathrm{ARP}}$, they contribute at most $\delta \cdot \mathrm{OPT}_{\mathrm{MWISR}}$.

Combined, we get that $\mathrm{OPT}_{\mathrm{ARP}} \leq \left(\delta + (1-\delta)^{-2}\right) \cdot \mathrm{OPT}_{\mathrm{MWISR}}$. Using Equation (1), this implies

$$\mathcal{A}(\mathcal{S}) \geq (1-\delta)^2(1+\varepsilon')^{-1} \cdot \left(\delta + (1-\delta)^{-2}\right)^{-1} \cdot \mathrm{OPT}_{\mathrm{ARP}} = (1+\varepsilon)^{-1} \cdot \mathrm{OPT}_{\mathrm{ARP}}, \tag{2}$$

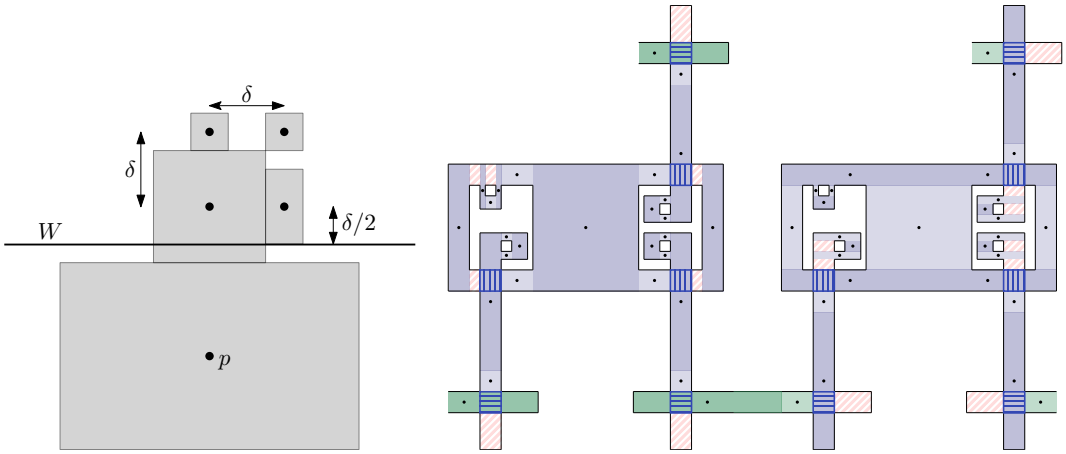
where the equality holds for small enough δ and appropriately chosen ε'. ◀

4 CARP is NP-Complete

In the decision variant of the center-anchored rectangle packing problem, we are given a value $A > 0$ and a finite point set $P \subseteq U$. The goal is to decide whether there is a valid rectangle set $\mathcal{S}$ for P that covers an area of at least A. In this section we prove that this problem is NP-complete.

A useful observation is that if the input set contains four points in the corners of a tiny square with side length $\delta > 0$, then these points restrict the expansion of other points' rectangles but cannot expand their own rectangles beyond side length 2δ (see Figure 2a). This enables us to build walls that encode the graph structure of a suitable variant of maximum independent set (MIS).

▶ **Theorem 2.** *The center-anchored rectangle packing problem is NP-complete.*



(a) Building a wall W to the north of point p by adding four wall points.

(b) Two node gadgets directly connected via a path gadget. The left node gadget is contracted, the right one expanded.

Figure 2 Illustrations of the wall construction and a normalized solution for a pair of node gadgets.

The problem is easily seen to be in NP. Thus, the main challenge in proving Theorem 2 lies in the aforementioned reduction. Since the gadgets used in the construction as well as their interplay are somewhat involved, we give a high-level overview in Section 4.1. In Section 4.2, we describe in more detail how a suitable CARP instance is constructed from a given MIS instance. The analysis of this construction is given in Section 4.3.

4.1 Overview of the Reduction

The reduction is from the problem maximum independent set in 3-regular planar graphs (MIS3P), which is known to be NP-hard [12]. Given a MIS3P instance G, we use a result by Tamassia [18] to construct an orthogonal embedding of G. We then replace each node of G by a *node gadget* and each edge by a series of *path gadgets*. The number of path gadgets per edge corresponds (roughly) to the number of turns that it takes in the orthogonal embedding.

Gadgets & Rectangle Sets. Figure 2b illustrates the gadgets via a simple example where two node gadgets are directly[2] connected by a path gadget. Black dots represent the normal input points used to form center-anchored rectangles. Black lines represent walls built by placing four δ-spaced *wall points* at the orthogonal projections of each normal input point onto its closest wall in each direction. The horizontally striped blue areas connecting node gadget and path gadget are called *conflict areas*. The vertically striped blue areas at the entrance to the inner parts of the node gadget are called *compensation areas*. The shaded areas represent a valid example rectangle set for the (non-wall) input points. Diagonally striped red areas remain uncovered in the depicted solution.

Note that the rectangle sets of the two node gadgets in the depicted solution differ from one another. If, in a given solution, the points of a node gadget span rectangles exactly as in the right part of Figure 2b we say the node gadget is *expanded*. In particular, an expanded node gadget covers all three adjacent conflict areas. In the left part of Figure 2b, we see

[2] In our actual constructions, node gadgets are always connected via a chain of several path gadgets. Figure 2b neglects this technical detail in order to keep the illustration simple and instructive.

one possible example of a so called *contracted* node gadget. Such contracted node gadgets neither cover a neighboring conflict area (which is then covered by the adjacent path gadget), nor the neighboring compensation area (which is then not covered at all). A central part of our construction is that the latter case (where we loose an area of size b^2) happens only if the adjacent path gadget leads to another contracted gadget.

Normalized Solutions & Structural Properties. Our construction allows to prove the existence of a well-structured, almost optimal solution. We call a solution *normalized* if each wall point is assigned a $(\delta \times \delta)$-square. In a normalized solution, non-wall points cannot expand their rectangles beyond adjacent walls. Additionally, we show the following properties.

- Lemma 4: There is a normalized solution approximating an optimal solution up to a factor of $1 - O(\delta/\varepsilon^2)$. Choosing δ suitably small allows us to consider normalized solutions only.
- Lemma 5: We call a solution normalized *with oriented paths* if the rectangles spanned by non-wall points are such that any series of connected path gadgets is alternatingly covered by "big" and "small" rectangles. An optimal normalized solution can be transformed into one with oriented paths without decreasing the covered area.
- Lemma 6: Given an optimal normalized solution with oriented paths, one can transform the rectangles spanned by non-wall points of node gadgets such that each node gadget is either expanded or contracted. This transformation changes only rectangles spanned by node gadgets and does not decrease the covered area.

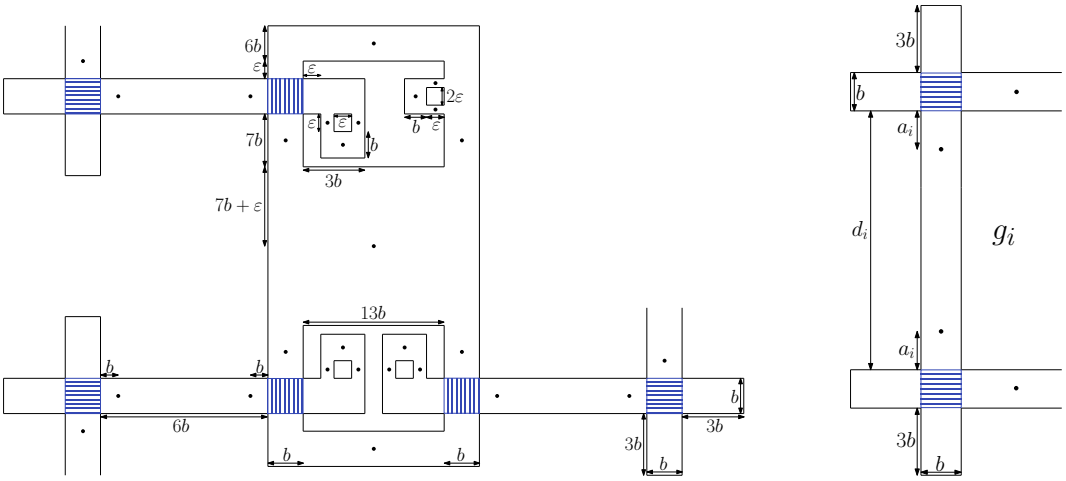
From this, we obtain a bijection between normalized solutions $\mathcal{S}$ with oriented paths and only either expanded or contracted node gadgets and independent sets $I \subseteq V(G)$. Nodes $u \in I$ correspond to expanded node gadgets and nodes $u \notin I$ to contracted node gadgets.

Covered Area & Size of Independent Set. We derive a formula for the area covered by $\mathcal{S}$ in terms of the size $|I|$ of the independent set. When a contracted node gadget does not cover one (or multiple) of its compensation areas, we charge these areas to the neighboring conflict area (which has the same size, b^2). See Figure 3 for in illustration of compensation and conflict areas. By this charging, the area covered by the inner part of any node gadget (whether expanded or contracted) differs only by the number of covered ε-stripes (small diagonally striped red areas within the node gadgets of size $O(\varepsilon \cdot b) \ll b^2$). For each path gadget, we always loose an area of size $3b^2$ at one of the two ends. Additionally, for each series of path gadgets connecting two contracted node gadgets, we loose an area b^2 of one of the conflict areas (using the charging).

Thus, ignoring negligible areas from the ε-stripes and wall-points, $\mathcal{S}$ covers an area of size $\mathcal{A}(\mathcal{S}) = A_G + 3|I| \cdot b^2$. While A_G depends only on the graph G and its embedding, the second term stems from an independent set node being the only means of covering both critical areas of an incident edge. Thus, a solution covering the maximum area yields a maximum independent set and vice versa.

4.2 Construction of the CARP Instance

We start by formalizing the ideas of walls, depicted as black lines in our images. Fix a small $\delta > 0$ (whose value we specify later) and consider a point set P, a point $p \in P$, and a wall W (an axis aligned line segment), which is closest either in x- or y-direction. We modify P by adding four wall points that form an (axis aligned) square of side length δ on the (w.r.t. p) opposite side of the wall W. We do this in such a way that p and two of the wall points lie on a straight line and the wall point closer to p is at a distance of $\delta/2$ to W. See Figure 2a for an illustration.



(a) A node gadget. **(b)** A path gadget.

Figure 3 Detailed illustration of node and path gadgets with the respective side lengths. Conflict areas are horizontally striped. Compensation areas are vertically striped.

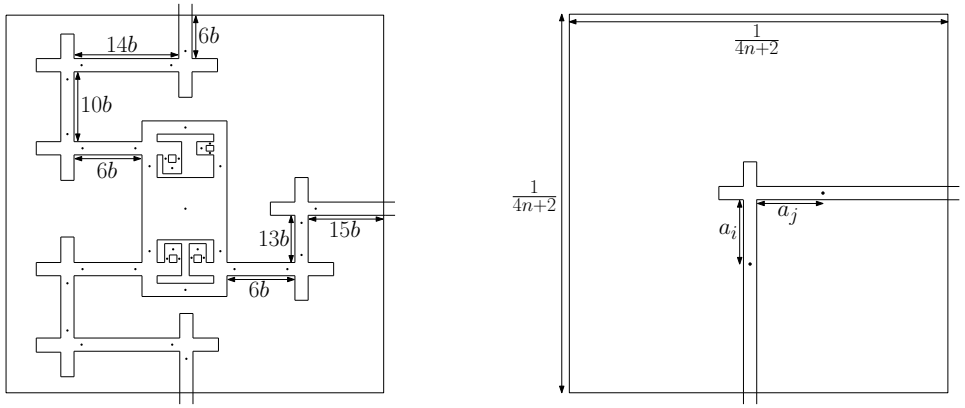
▶ **Observation 3.** *Consider a valid solution for P after adding the four wall points for p. The rectangles spanned by p and by the wall points cannot transgress the wall by more than $\delta/2$. Moreover, the rectangles spanned by the four wall points cover an area*[3] *of at most $4\delta^2$.*

Gadgets. We now formally describe the node gadgets (representing nodes of the MIS3P instance) and the path gadgets (representing edges of the MIS3P instance). Figure 3a shows the details of a node gadget. Each input point is restricted by at most four walls. A node gadget has three outgoing paths each connected to a path gadget (corresponding to an incident edge). Figure 3b shows the details of a path gadget. Path gadgets are connected either to other path gadgets via a 90° turn (the crossings in the figure) or to a node gadget. There are two global parameters, the width of a path b and a small value $\varepsilon > 0$. We choose the former such that we can fit each gadget into a small *tile* whose size depends on the MIS3P instance, and we choose the latter small enough to ensure that certain non-coverable areas of node gadgets are negligible. A path gadget g_i has two additional parameters, $d_i > 0$ (roughly the length of the straight edge it represents) and $a_i := d_i/4 - b$ (specifying the exact placement of the non-wall points depending on d_i).

Orthogonal Embedding & Connecting the Gadgets. Next, consider an MIS3P instance G (i.e., a 3-regular planar graph) consisting of n nodes. We embed G in a $(4n+2) \times (4n+2)$ grid. This can be done in polynomial time (a simple application of a result from [18]). We then partition the unit square U into a grid of equal-sized tiles of width and height $1/(4n+2)$. Depending on how the embedding behaves at a grid point $(i, j) \in \{1, \ldots, 4n+2\}^2$, we construct tile (i, j) from a suitable blueprint:

- *Grid point contains a node:* Use an appropriate rotation of the tile in Figure 4a.
- *Grid point contains an edge bend:* Use an appropriate rotation of the tile from Figure 4b.
- *Grid point contains part of an edge:* Insert tiles to form a straight path of width b, either from south to north or from east to west (determined by the orientation of the edge part).

[3] While the rectangles may cover all of the square formed by the wall points, an area of size δ^2, precisely $\frac{3}{4}$ of each rectangle's area lies outside this square. Thus, the total covered area is at most $\delta^2 + 3\delta^2$.



(a) Node gadget inside a tile of U. **(b)** Bend inside a tile connecting two path gadgets.

Figure 4 Two tiles used in our hardness construction. Further tiles include rotations of these as well as straight horizontal and vertical paths. Both sides of each tile have size $64b + 2\varepsilon$.

Note that outgoing path gadgets always leave a tile at the center of a tile boundary and that all parts are properly connected.

It remains to fix the parameters δ, b, and ε. These are chosen such that all parts of a node gadget tile (Figure 4a) fit exactly into the $1/(4n+2) \times 1/(4n+2)$ tiles and $\delta \ll \varepsilon \ll b$. The chosen lengths (Figure 3) guarantee that tile parts do not overlap. Our analysis also requires $2a_i \geq b$ (or, equivalently, $d_i \geq 6b$) for every path gadget g_i. One can easily verify that this is the case (see Section 4.2).

We now bound the size of the constructed CARP instance P_G for a graph G consisting of n nodes. Each tile of a node gadget contains 38 points. For each bend tile we add 2 points. Since there are n nodes and at most $3n + 2$ bends [18, Lemma 7], the instance P contains at most $44n + 4$ non-wall points. For each non-wall point we add at most 16 wall points. Thus, the size of P is bounded by $704n + 64 = \Theta(n)$, implying that P_G can be constructed in polynomial time.

4.3 Analysis of the Constructed CARP Instance

Fix a MIS3P instance $G = (V, E)$ consisting of n nodes and let P_G denote the construction from the previous section. Let $W \subseteq P_G$ be the set of all wall points in P_G. Recall that a solution $\mathcal{S}$ for the CARP instance P_G is called normalized if all points in W span $(\delta \times \delta)$-rectangles. Note that in such a solution the rectangles spanned by points from $P_G \setminus W$ cannot overlap the walls. An *optimal normalized solution* is a normalized solution of maximum size. In the following, we state the key results of the analysis. Due to space constraints, we do not give the proofs in all detail here. For the complete proofs, we refer the reader to the full version of this paper.

Our first lemma shows that there exists a normalized solution that approximates an optimal solution up to a factor of $1 - O(\delta/\varepsilon^2)$.

▶ **Lemma 4.** *There exists a normalized solution for CARP on the instance P_G of area at least $(1 - 2\delta/\varepsilon^2) \cdot \mathrm{OPT}_{CARP(P_G)}$.*

The next pair of lemmas form the center of our arguments. They show the existence of optimal normalized solutions exhibiting useful structural properties for both the path and node gadgets.

▶ **Lemma 5.** *There exists an optimal normalized solution in which the two rectangles spanned by a path gadget g_i's two non-wall points have width (or, depending on orientation, height) b, and one of them has height (width) $2a_i$, the other one $2(a_i + 4b)$. In particular, the rectangles of each path gadget cover exactly one of the gadget's two conflict areas.*

▶ **Lemma 6.** *There exists an optimal normalized solution such that the following holds:*

- *Each node gadget is either contracted or expanded.*
- *No two adjacent node gadgets are expanded.*
- *For each pair of contracted adjacent node gadgets there is a (uniquely identified) uncovered area of size b^2 inside one of the node gadgets.*

The previous two lemmas hold simultaneously. NP-hardness follows from these results as described in Section 4.1.

5 Resource Augmentation

In this section, we investigate the lower-left-anchored rectangle packing problem. We study two different types of resource augmentation. In the first type, which we call *perturbation-augmentation*, each solution rectangle does not need to be anchored exactly in the corresponding point but instead its anchor may be up to an ε distance away from that point (see Figure 5, center). We define an *ε-grid* $\Gamma = (V, \mathcal{L})$ as a set of points $V = \{\, (x, y) \in U \mid x = \varepsilon \cdot k_x \ \wedge\ y = \varepsilon \cdot k_y \ \wedge\ k_x, k_y \in \mathbb{N} \,\}$ together with a family of grid-cells $\mathcal{L} = \{\, [x, x+\varepsilon] \times [y, y+\varepsilon] \subseteq U \mid (x, y) \in V \,\}$. The perturbation-augmentation allows us to focus on solutions where all vertices of rectangles are points on an ε-grid Γ, thereby allowing us to enumerate all possible sets of interior-disjoint, axis-aligned rectangles with vertices in V. We call such a solution a *grid-point solution*. We show that (i) there exists a polynomial-time algorithm that finds the optimal grid-point solution and (ii) the optimal grid-point solution covers at least as much area as an optimal (unrelaxed) solution.

Before formally stating the theorem, we introduce some notation. Let $\mathrm{dist}_\infty(x, y)$ denote the distance of two points $x, y \in \mathbb{R}^2$ in the ℓ_∞-norm. For a set $X \subseteq \mathbb{R}^2$ and a point $y \in \mathbb{R}^2$, we write $\mathrm{dist}_\infty(y, X) = \inf_{x \in X} \mathrm{dist}_\infty(y, x)$ and $\overline{X} = \mathbb{R}^2 \setminus X$ for the set's complement. We say that a set of interior-disjoint rectangles $\{R_p\}_{p \in P}$ is *ε-valid*, if $\mathrm{dist}_\infty(p, \ell(R_p)) < \varepsilon$ and $\mathrm{dist}_\infty\big(p, \overline{R_p}\big) < \varepsilon$ for all $p \in P$, where $\ell(R_p)$ denotes the lower-left corner of R_p.

▶ **Theorem 7.** *For every $\varepsilon > 0$ and a finite point-set $P \subseteq U$, there exists a polynomial time algorithm that computes an ε-valid set of interior-disjoint rectangles that cover an area of at least* $\mathrm{OPT}_{LLARP(P)}$.

The second type of resource augmentation we call *overlap-augmentation*. It relaxes the condition that the rectangles need to be disjoint. More specifically, each pair of rectangles is allowed to overlap by a thin strip of width at most δ. Note that this implies that a rectangle may also contain input points as long as they are no more than a δ-distance away from the boundary (see Figure 5, right).

Again, we require additional definitions. A set of rectangles $\{R_p\}_{p \in P}$ is called a *δ-LLARP*, if the rectangle R_p is lower-left-anchored in p and $\sup_{x \in R_{p'}} \mathrm{dist}_\infty\big(x, \overline{R_p}\big) < \delta$, for all $p' \in P$. We show that we can transform an ε-valid grid-point solution into a δ-LLARP while only losing a small fraction of the covered area.[4]

[4] Naturally, any area that is covered by multiple rectangles is counted only once.

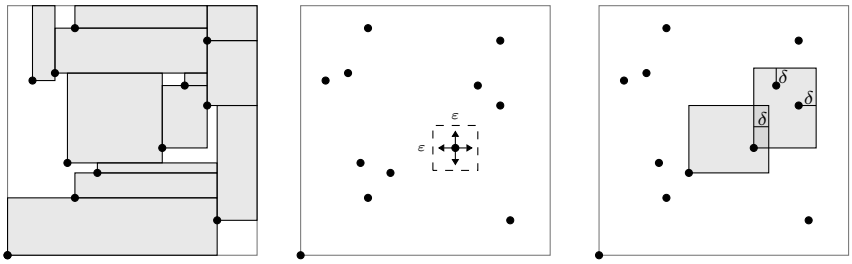


Figure 5 An optimal lower-left-anchored rectangle packing (left), and illustrations of the two kinds of resource augmentation that are used in the paper.

▶ **Theorem 8.** *For every $\varepsilon > 0$ and a finite point-set $P \subseteq U$, there exists a polynomial time algorithm that outputs an $\frac{\varepsilon}{11}$-LLARP that covers an area of at least $(1-\varepsilon) \cdot \mathrm{OPT}_{LLARP(P)}$.*

Algorithm 1 Algorithm using resource augmentation of the perturbation-type.

```
Γ ← ε-grid in U, H ← ∅
for every configuration C in Γ do
    if C is ε-valid for P then
        for every grid-point solution S := {R_q}_{q∈C} of LLARP(C) do
            H ← H ∪ {S}
return S ∈ H maximizing A(S)
```

Proof of Theorem 7. Let $\Gamma = (V, \mathcal{L})$ be an ε-grid in U and denote by k the number of grid-cells. Without loss of generality, assume that all points of P lie in the interior of one of the $O(\varepsilon^{-2})$ grid-cells. A *grid configuration* on Γ is a subset of the grid points $C \subseteq V$. We say that a grid configuration C on Γ is *ε-valid* for P if there is a surjective mapping $\varphi : P \to C$, such that $\mathrm{dist}_\infty(p, \varphi(p)) \leq \varepsilon$. Note that any LLARP for an ε-valid configuration C for P can be extended to an ε-valid solution for P by adding degenerate rectangles anchored at points in C.

Algorithm 1 enumerates all grid configurations on Γ that are ε-valid for P. Then, for each configuration C, it computes all grid-point solutions $\{R_q\}_{q \in C}$ of LLARP(C). Among all enumerated solutions, we keep one that maximizes the covered area.

First, we show that the running time of this algorithm is polynomial. Note that any grid cell has four grid points as its vertices. Since any point $p \in P$ lies in the interior of some grid cell $L \in \mathcal{L}$, an ε-valid grid configuration must contain at least one of the four vertices of L. Thus, we can construct any ε-valid grid configuration by choosing for each grid cell $L \in \mathcal{L}$ and each input point $p \in P \cap L$, a vertex of L as its image in the mapping φ. However, since whenever multiple rectangles are anchored in the same point at most one can be non-degenerate, in each $L \in \mathcal{L}$ it suffices to decide on one of at most 15 cell configurations $L \cap C$ using either 0, 1, 2, 3, or 4 of the grid cell's vertices. In particular, any number of contained input points larger than 3, yields the same set of possible cell configurations. Thus, by counting the number of input points contained in each grid cell (in time $O(n)$) and then enumerating at most 15^k grid configurations, we obtain all ε-valid grid configurations. For each such configuration C, we obtain all corresponding ε-valid grid-point solutions by picking for each $q \in C$ one of the $O(\varepsilon^{-2})$ grid points above and to the right of q as the upper-right corner of the rectangle and checking interior disjointness. This amounts to at most $O(\varepsilon^{-2\varepsilon^{-2}})$ solutions per grid configuration. Thus, in total, the running time of Algorithm 1 is linear in n and doubly exponential in $\frac{1}{\varepsilon}$.

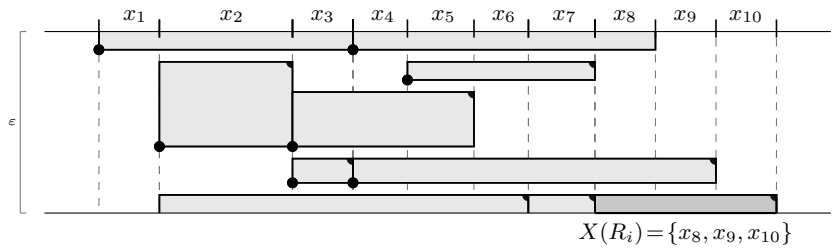


Figure 6 A row B of the ε-grid F and the corresponding rectangles of $\mathcal{S}^*_B$.

It remains to show that the computed grid-point solution covers at least as much area as an optimal lower-left-anchored rectangle packing $\mathcal{S}^*_{\text{LLARP}(P)} = \{R_1, R_2, \ldots, R_n\}$ without resource augmentation. We transform $\mathcal{S}^*_{\text{LLARP}(P)}$ to a grid-point solution by "snapping" the corners of its rectangles to Γ. We do this by first snapping the lower-left and upper-right corners of each rectangle to either of the horizontal grid-lines directly above and below the corner in question. Afterwards, the same is done analogously for the vertical grid-lines, which we omit in our discussion.

Consider one row of the grid, say $B = [0, 1] \times (y, y + \varepsilon)$, for $(x, y) \in V$ (see Figure 6). We describe how the lower-left and upper-right corners within this row can be snapped up or down without reducing the total area. Partition the row horizontally into segments $x_1, x_2, \ldots, x_r$ in-between the x-coordinates of corners of the rectangles $\mathcal{S}^*_B \coloneqq \{R_i\}_{i=1}^m \subseteq \mathcal{S}^*_{\text{LLARP}(P)}$ that have a corner in the row. We associate a variable $h_i \in [0, 1]$ with each rectangle R_i denoting the height of $R_i \cap B$ relative to the height of B. Snapping the rectangles in this row can be seen as rounding each such variable h_i either up to 1 or down to 0 (i.e., expanding the rectangle to span the full row height or collapsing it such that it does not appear in the row anymore). We aim to do this in a manner, which neither introduces overlaps nor decreases the covered area. This snapping problem is represented by the following integer linear program.

$$\max \quad \sum_{i=1,\ldots,m} h_i \cdot w(R_i) \tag{3}$$

$$s.t. \quad \sum_{i\,:\,x_j \in X(R_i)} h_i \leq 1 \qquad \text{for } j = 1, \ldots, r \tag{4}$$

$$h_i \in \{0, 1\} \qquad \text{for } i = 1, \ldots, m\,. \tag{5}$$

Here, $w(R_i)$ denotes the width of rectangle R_i, and the set $X(R_i)$ consists of all segments x_i intersecting R_i. The objective function (3) maximizes the covered area while constraints in (4) ensure that (in an integer solution) rectangles do not overlap. The binary constraints in (5) force the rectangles to either span the entire height or nothing of the row. We note that the linear program is inspired by a similar one for the demand flow problem [10].

It is easy to verify that $\mathcal{S}^*_B$ corresponds to a feasible solution for the linear programming relaxation of (3)–(5) and thus lower bounds the optimal value of the LP-relaxation. Since the sets $X(R_i)$ contain consecutive line segments, the LP satisfies the consecutive ones property [17]. Therefore, constraints (4) yield a totally unimodular matrix and the LP is integral [17]. It follows, that there is an integral solution that solves the relaxed linear program optimally and induces a partial snapping of area at least $\text{OPT}_{\text{LLARP}(P)}$. Repeating this argument for all rows and columns of Γ yields a grid-point solution with an area of at least $\text{OPT}_{\text{LLARP}(P)}$. ◀

Algorithm 2 Algorithm using resource augmentation of the overlap-type.

Compute $S_{\mathrm{I}} =$ Algorithm 1 $(P, \frac{\varepsilon}{22})$
for all $R_p \in S_{\mathrm{I}}$ **do**
  $R'_p \leftarrow$ the rectangle spanned by p and $u(R_p) + (\frac{\varepsilon}{22}, \frac{\varepsilon}{22})$
  if $u(R'_p) \notin U$ **then**
    $u(R'_p) \leftarrow \arg\min_{x \in U} ||x - u(R'_p)||$
return $S_{\mathrm{II}} = \{R'_p\}_{p \in P}$

Proof of Theorem 8. We show that Algorithm 2 satisfies the conditions of Theorem 8, that is it runs in polynomial time and outputs a $\frac{\varepsilon}{11}$-LLARP of area at least $(1-\varepsilon) \cdot \mathrm{OPT}_{\mathrm{LLARP}(P)}$. In comparison to the previous case, input points cannot be moved, but the rectangles may somewhat overlap.

Consider the $\frac{\varepsilon}{22}$-valid solution $\mathcal{S}_{\mathrm{I}} = \{R_p\}_{p \in P}$ of Algorithm 1 with input $(P, \frac{\varepsilon}{22})$. In constructing $\mathcal{S}_{\mathrm{I}}$, we move the input-points $p \in P$ to grid-points $q_p \in F$ spanning R_p, the upper-right corner of which we denote by $u(R_p)$. We transform the solution of Algorithm 1 to obtain an $\frac{\varepsilon}{11}$-LLARP.

Shift each $u(R_p)$ up and to the right by $\frac{\varepsilon}{22}$ to obtain $u(R_p)'$. The transformed solution $\mathcal{S}'_{\mathrm{I}}$ consists of the rectangles uniquely defined by the lower-left corners $p \in P$ and their corresponding upper-right corners $u(R_p)'$. Since $\mathrm{dist}_\infty(p, q_p) < \frac{\varepsilon}{22}$, moving $u(R_p)$ by $(\frac{\varepsilon}{22}, \frac{\varepsilon}{22})$ ensured that transforming the solution did not decrease its size. The overlap of rectangles is bounded by $2 \cdot \frac{\varepsilon}{22} = \frac{\varepsilon}{11}$.

However, moving the solution may cause some rectangles to protrude from the unit square. We prune this excess area, by moving the respective upper-right corners back into the square to obtain our final solution $\mathcal{S}_{\mathrm{II}}$. Due to this pruning, we lose an area of size at most $2 \cdot \frac{\varepsilon}{22} = \frac{\varepsilon}{11} \leq \varepsilon \cdot \mathrm{OPT}_{\mathrm{LLARP}(P)}$, where the inequality follows from the lower bound of $\frac{1}{11}$ on $\mathrm{OPT}_{\mathrm{LLARP}(P)}$ [11]. This implies, that $\mathcal{A}(\mathcal{S}_{\mathrm{II}}) \geq (1-\varepsilon) \cdot \mathrm{OPT}_{\mathrm{LLARP}(P)}$. ◀

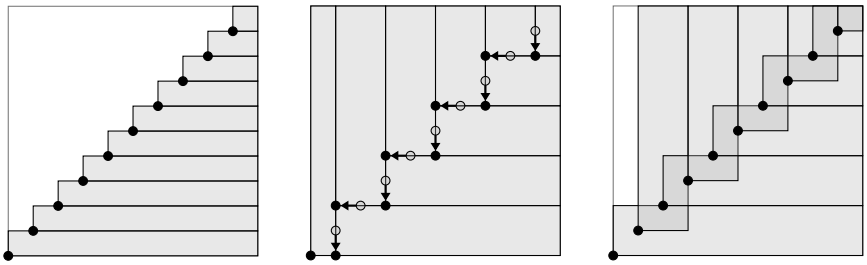


Figure 7 Algorithms Algorithm 1 and Algorithm 2 applied to a simple instance.

References

1 Anna Adamaszek, Parinya Chalermsook, and Andreas Wiese. How to Tame Rectangles: Solving Independent Set and Coloring of Rectangles via Shrinking. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *LIPIcs*, pages 43–60. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.43.

2 Anna Adamaszek and Andreas Wiese. Approximation Schemes for Maximum Weight Independent Set of Rectangles. In *54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 400–409. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.50.

3 Anna Adamaszek and Andreas Wiese. A quasi-PTAS for the Two-Dimensional Geometric Knapsack Problem. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2015)*, pages 1491–1505. SIAM, 2015. doi:10.1137/1.9781611973730.98.

4 Hugo A. Akitaya, Matthew D. Jones, David Stalfa, and Csaba D. Tóth. Maximum Area Axis-Aligned Square Packings. In *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018)*, volume 117 of *LIPIcs*, pages 77:1–77:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.77.

5 Kevin Balas, Adrian Dumitrescu, and Csaba D. Tóth. Anchored rectangle and square packings. *Discrete Optimization*, 26:131–162, 2017. doi:10.1016/j.disopt.2017.08.003.

6 Nikhil Bansal and Arindam Khan. Improved Approximation Algorithm for Two-Dimensional Bin Packing. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2014)*, pages 13–25. SIAM, 2014. doi:10.1137/1.9781611973402.2.

7 Sergey Bereg, Adrian Dumitrescu, and Minghui Jiang. Maximum Area Independent Sets in Disk Intersection Graphs. *Int. J. Comput. Geometry Appl.*, 20(2):105–118, 2010. doi:10.1142/S0218195910003220.

8 Sergey Bereg, Adrian Dumitrescu, and Minghui Jiang. On Covering Problems of Rado. *Algorithmica*, 57(3):538–561, 2010. doi:10.1007/s00453-009-9298-z.

9 Therese C. Biedl, Ahmad Biniaz, Anil Maheshwari, and Saeed Mehrabi. Packing Boundary-Anchored Rectangles. In *Proceedings of the 29th Canadian Conference on Computational Geometry (CCCG 2017)*, pages 138–143, 2017.

10 Chandra Chekuri, Marcelo Mydlarz, and F. Bruce Shepherd. Multicommodity demand flow in a tree and packing integer programs. *ACM Trans. Algorithms*, 3(3):27, 2007. doi:10.1145/1273340.1273343.

11 Adrian Dumitrescu and Csaba D. Tóth. Packing anchored rectangles. *Combinatorica*, 35(1):39–61, 2015. doi:10.1007/s00493-015-3006-1.

12 Bojan Mohar. Face Covers and the Genus Problem for Apex Graphs. *J. Comb. Theory, Ser. B*, 82(1):102–117, 2001. doi:10.1006/jctb.2000.2026.

13 Richard Rado. Some covering theorems (I). *Proc. London Math. Soc.*, 51:232–264, 1949.

14 Richard Rado. Some covering theorems (II). *Proc. London Math. Soc.*, 53:243–267, 1951.

15 Richard Rado. Some covering theorems (III). *Proc. London Math. Soc.*, 42:127–130, 1968.

16 Tibor Radó. Sur un problème relatif à un théorème de Vitali. *Fund. Math.*, 11:228–229, 1928.

17 Alexander Schrijver. *Theory of linear and integer programming.* Wiley-Interscience series in discrete mathematics and optimization. Wiley, 1999.

18 Roberto Tamassia. On Embedding a Graph in the Grid with the Minimum Number of Bends. *SIAM J. Comput.*, 16(3):421–444, 1987. doi:10.1137/0216030.

19 William Thomas Tutte, editor. *Recent Progress in Combinatorics: Proceedings of the 3rd Waterloo Conference on Combinatorics.* Academic Press, 1969.

20 Peter Winkler. Packing Rectangles. *Mathematical Mind-Benders*, pages 133–134, 2007.