# Improved Approximation Algorithms for 2-Dimensional Knapsack: Packing into Multiple L-Shapes, Spirals, and More

**Waldo Gálvez**
Department of Computer Science, TU München, Germany

**Fabrizio Grandoni**
IDSIA, USI-SUPSI, Lugano, Switzerland

**Arindam Khan**
Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India

**Diego Ramírez-Romero**
Department of Mathematical Engineering, Universidad de Chile, Santiago, Chile

**Andreas Wiese**
Department of Industrial Engineering and Center for Mathematical Modeling, Universidad de Chile, Santiago, Chile

## Abstract

In the 2-Dimensional Knapsack problem (2DK) we are given a square knapsack and a collection of $n$ rectangular items with integer sizes and profits. Our goal is to find the most profitable subset of items that can be packed non-overlappingly into the knapsack. The currently best known polynomial-time approximation factor for 2DK is $17/9 + \varepsilon < 1.89$ and there is a $(3/2 + \varepsilon)$-approximation algorithm if we are allowed to rotate items by 90 degrees [Gálvez et al., FOCS 2017]. In this paper, we give $(4/3 + \varepsilon)$-approximation algorithms in polynomial time for both cases, assuming that all input data are integers polynomially bounded in $n$.

Gálvez et al.'s algorithm for 2DK partitions the knapsack into a constant number of rectangular regions plus *one* L-shaped region and packs items into those in a structured way. We generalize this approach by allowing up to a *constant* number of *more general* regions that can have the shape of an L, a U, a Z, a spiral, and more, and therefore obtain an improved approximation ratio. In particular, we present an algorithm that computes the essentially optimal structured packing into these regions.

**2012 ACM Subject Classification** Theory of computation → Approximation algorithms analysis

**Keywords and phrases** Approximation algorithms, two-dimensional knapsack, geometric packing

**Digital Object Identifier** 10.4230/LIPIcs.SoCG.2021.39

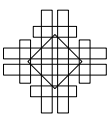
**Related Version** *Full Version*: https://arxiv.org/abs/2103.10406

**Funding** *Waldo Gálvez*: Supported by the European Research Council, Grant Agreement No. 691672, project APEG.
*Fabrizio Grandoni*: Partially supported by the SNSF Excellence Grant 200020B_182865/1.
*Andreas Wiese*: Partially supported by the ANID Fondecyt Regular grant 1200173.

## 1 Introduction

The 2-Dimensional (Geometric) Knapsack problem (2DK) is a natural geometric generalization of the fundamental (one-dimensional) Knapsack problem. In 2DK we are given a set $I$ of $n$ items $i$ which are axis-parallel rectangles specified by their width $w(i) \in \mathbb{N}$, height $h(i) \in \mathbb{N}$, and profit $p(i) \in \mathbb{N}$. Furthermore, we are given an axis-parallel square knapsack $K = [0, N] \times [0, N]$ for some $N \in \mathbb{N}$. The goal is to select a subset $I' \subseteq I$ of maximum total profit $p(I') := \sum_{i \in I'} p(i)$ that can be placed non-overlappingly inside $K$. Formally, for







each $i \in I'$ we have to define a pair $(lc(i), bc(i))$ that specifies the left and bottom coordinates of $i$, respectively, such that $i$ is placed inside $K$ as $R(i) := (lc(i), bc(i)) \times (lc(i)+w(i), bc(i)+h(i))$; we require that $R(i) \subseteq K$ and also that for any two $i, j \in I'$ it holds that $R(i) \cap R(j) = \emptyset$. We will consider also the case *with rotations*, in which each item $i \in I$ can be rotated by 90 degrees, i.e., $i$ can be replaced by a rectangle with width $h(i)$, height $w(i)$ (and profit $p(i)$). In the cardinality (or unweighted) setting of the problem all profits are 1.

2DK has several applications. For example, the rectangles can model banners out of which one wants to place the most profitable subset on a website or an advertisement board. Also, they can model pieces that one wants to cut out of some raw material like wood or steel. In addition, there are scheduling settings in which jobs need a consecutive amount of some given resource (e.g., a frequency bandwidth) for some amount of time; thus, each job can be modeled via a rectangle.

Most algorithms for 2DK and related problems work as follows: they guess a partition of the knapsack into $O_\varepsilon(1)$ rectangular boxes, for some small constant $\varepsilon > 0$. Inside each box the items are packed greedily using the Next-Fit-Decreasing-Height algorithm [18], or even simpler by stacking items on top of each other or next to each other. Implicitly, Jansen and Zhang use this strategy to obtain a $(2+\varepsilon)$-approximation algorithm for 2DK [38, 39]. The same approach, however using $(\log N)^{O_\varepsilon(1)}$ boxes, is used in a QPTAS which assumes that $N$ is quasi-polynomially bounded in $n$ [4]. Finding a PTAS for 2DK or ruling it out is a major open problem in the area. This question is also open if $N$ is polynomially bounded in $n$, i.e., for pseudo-polynomial time algorithms.

One might wonder whether a PTAS can be constructed using $O_\varepsilon(1)$ boxes only. Unfortunately, as observed in [24], essentially no better approximation ratio than 2 is achievable in this way. Hence a different type of packing is needed to breach this approximation barrier (in polynomial time). This was recently achieved by Gálvez et al. [24], where the authors pack the items into $O_\varepsilon(1)$ boxes and additionally one container with the shape of an L (which is packed with an ad-hoc, more complex algorithm).

This yields an approximation ratio of $17/9 + \varepsilon < 1.89$ (and $558/325 + \varepsilon < 1.72$ in the unweighted case). The authors also present $(3/2+\varepsilon)$- and $(4/3+\varepsilon)$-approximation algorithms for 2DK with rotations in the weighted and unweighted case, respectively.

Gálvez et al. [24] pose as an open problem how to efficiently pack items into a *constant* number of L-shaped containers, and observe that this would lead to improved approximation algorithms for 2DK. This problem was open even for just two L-shaped containers and using pseudo-polynomial time $(nN)^{O_\varepsilon(1)}$. In this paper we solve (a generalization of) this problem, and hence obtain an improved approximation ratio.

## 1.1 Our contribution

In this paper, we present a $(4/3+\varepsilon)$-approximation algorithm with a (pseudo-polynomial) running time of $(nN)^{O_\varepsilon(1)}$ for (weighted) 2DK. We also achieve improved $(4/3+\varepsilon)$- and $(5/4+\varepsilon)$-approximation algorithms for 2DK *with rotations* in the weighted and unweighted case, resp., with the same running time. See Table 1 for an overview of our and previous results in the respective settings.

Our algorithms use $O_\varepsilon(1)$ boxes and, in addition, rather than one single L-shaped container as in [24], a combination of $O_\varepsilon(1)$ containers with the shape of an L or even more complicated shapes. The latter are intuitively thin corridors with the property that if we traverse them, we change the orientation at the turns (i.e., clockwise or counter-clockwise) at most once. For example, they can have figuratively the shape of a U, a Z, or a spiral (see Figure 1). Since they are thin, they help us to distinguish the parts of $K$ that are used by items that are wide and thin (horizontal items) and items that are high and narrow (vertical items). The interaction of these types of items is a major difficulty in 2DK.

**Table 1** A summary of our approximation ratios compared to the best known results with running time $(nN)^{O_\varepsilon(1)}$ for the respective settings.

| Setting | | Known result | Our result |
|---|---|---|---|
| Without rotations | Weighted | $17/9+\varepsilon < 1.89$ [24] | $4/3+\varepsilon$ |
| | Unweighted | $558/325+\varepsilon < 1.72$ [24] | $4/3+\varepsilon$ |
| With rotations | Weighted | $3/2+\varepsilon$ [24] | $4/3+\varepsilon$ |
| | Unweighted | $4/3+\varepsilon$ [24] | $5/4+\varepsilon$ |

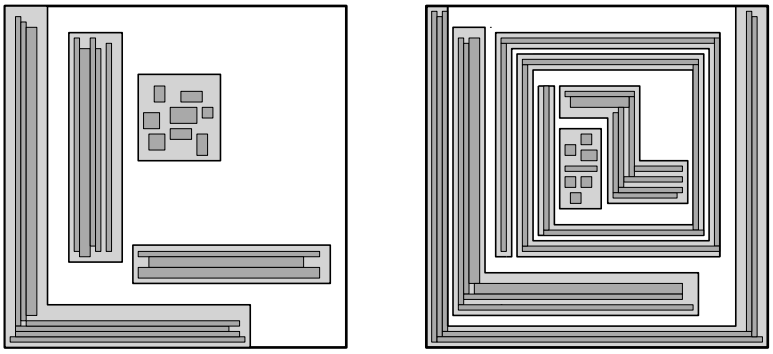


**Figure 1** Left: a packing based on a single L-shaped container and boxes as it was used in previous work. Right: A packing based on a box and cooridors with the shapes of an L, a U, a Z, and a spiral, as we use them in our algorithms.

By standard arguments (in particular building upon the corridor decomposition in [2]), it is not hard to show that we can partition $K$ into a constant number of boxes and corridors of the allowed types, so that there exists a feasible packing of a $(4/3+\varepsilon)$-approximate solution into them. The non-trivial part is how to efficiently pack items into our corridors. Here we cannot exploit the L-packing algorithm in [24]. Indeed, the latter algorithm does not seem to generalize even to two L-shaped corridors (even if $N = n^{O(1)}$), while we need to handle $\Theta_\varepsilon(1)$ corridors with possibly even more general shapes.

Our strategy is to partition each of our corridors into $O_\varepsilon(\log N)$ rectangular boxes. Using the properties of their shapes, we show that this is indeed possible by losing only a factor of $1+\varepsilon$ in the approximation guarantee. Guessing these boxes explicitly would take $N^{O_\varepsilon(\log N)}$ time which is too slow. Instead, we show how to guess the *sizes* of almost all of the boxes in time $(nN)^{O_\varepsilon(1)}$. Then, we place them into the corridors in polynomial time using a dynamic program based on color-coding, using that in total there are only $O_\varepsilon(\log N)$ boxes to place.

We remark that the above approach compromises between corridor shapes that are general enough to allow for $(4/3+\varepsilon)$ -approximate packings, and at the same time simple enough so that we can partition them into $O_\varepsilon(\log N)$ boxes that we can essentially guess in time $(nN)^{O_\varepsilon(1)}$. It is not clear how to extend our algorithm to $\Theta(\log^{1/\varepsilon} N)$, or just even $\Theta(\log^2 N)$ such boxes. However, this would allow us to exploit corridors of more general shapes (say $W$-shaped), hence achieving better approximation ratios. We leave this as an interesting open problem.

## 1.2 Related Work

The QPTAS in [4] (though with some restrictions on $N$) suggests that 2DK most likely admits a PTAS. This is already known for some relevant special cases: if the profit of each item equals its area [7], if the size of the knapsack can be slightly increased (resource augmentation) [22, 35], if all items are relatively small [21], or squares [31, 36].

One might consider packing geometric objects other than rectangles. In particular, there are constant approximation algorithms for packing triangles and also arbitrary convex polygons under resource augmentation, both assuming that arbitrary rotations are allowed [46]. Also, for circles a $(1+\varepsilon)$-approximation is known under resource augmentation in one dimension if the profit of each circle equals its area [45]. One can consider natural generalizations of 2DK to a higher number of dimensions. In particular, the 3-dimensional case, 3DK, has applications like packing containers into a ship or cargo into a truck. 3DK is known to be APX-hard [14], and constant approximation algorithms are known [19, 27]. Khan et al. [44] have given a $(2+\varepsilon)$-approximation algorithm for a generalization of 2DK, which generalizes geometric packing and vector packing.

A parameterized version of 2DK for rectangles (where the parameter is the number $k$ of packed items) is studied in [26]. The authors show that the problem is $W[1]$-hard (both with and without rotations). Furthermore, they provide an FPT $(1+\varepsilon)$-approximation for the case with rotations. Achieving a similar result for the case without rotations is open.

A packing is called a guillotine packing if all rectangles can be separated by a sequence of end-to-end (guillotine) cuts [43]. Abed et al. [1] have given QPTAS for 2DK satisfying guillotine packing constraints, assuming the input data is quasi-polynomially bounded. Recently, Khan et al. [42] have shown a pseudo polynomial-time approximation scheme for 2DK satisfying guillotine packing constraints.

In the 2-Dimensional Bin Packing problem we are given a collection of items similarly to 2DK, and copies of the same square knapsack (the bins). Our goal is to pack *all* the items using the smallest possible number of bins. The best known (asymptotic) result for this problem is due to Bansal and Khan [9]: they achieve a 1.405 approximation based on a configuration-LP. This improves a series of previous results [8, 10, 16, 35, 40].

Another closely related problem is Strip Packing. Informally, we are given a knapsack of width $N$ and infinite height, and we wish to pack *all* items so that the topmost coordinate is as small as possible. This problem admits a $(5/3+\varepsilon)$-approximation [28] (improving on [6, 18, 29, 49, 50, 51]) in the general case, $(3/2+\varepsilon)$-approximation [23] when none of the items are large, and it is NP-hard to approximate it below a factor $3/2$ by a simple reduction from Partition. However, strictly better approximation ratios can be achieved in pseudo-polynomial time $(Nn)^{O_\varepsilon(1)}$ [25, 34, 47], being $5/4+\varepsilon$ essentially the best possible ratio achievable in this setting [30, 33]. This shows that pseudo-polynomial time can make the difference for rectangle packing problems. It would be interesting to understand whether the techniques in our paper (as well as those in [4]) can be strengthened so as to run in polynomial time for arbitrary $N$. Strip Packing was also studied in the asymptotic setting [35, 40] and in the case with rotations [37].

Another related problem is the Independent Set of Rectangles problem: here we are given a collection of axis-parallel rectangles embedded in the plane, and we need to find a maximum cardinality/weight subset of non-overlapping rectangles [2, 3, 11, 17]. The problem has also been studied for squares, disks, and pseudo-disks, see e.g., [20, 32, 12].

We refer the readers to [15, 41] for surveys on geometric packing problems.

## 2 Preliminaries

We start with a classification of the input items according to their heights and widths. Let $\varepsilon > 0$. For two constants $1 \geq \varepsilon_{large} > \varepsilon_{small} > 0$ to be defined later, we classify an item $i$ as:

- *small*: if $h(i), w(i) \leq \varepsilon_{small} N$;
- *large*: if $h(i), w(i) > \varepsilon_{large} N$;

- *horizontal*: if $w(i) > \varepsilon_{large} N$ and $h(i) \leq \varepsilon_{small} N$;
- *vertical*: if $h(i) > \varepsilon_{large} N$ and $w(i) \leq \varepsilon_{small} N$;
- *intermediate*: otherwise, i.e., the length of at least one edge is in $(\varepsilon_{small} N, \varepsilon_{large} N]$.

We call *skewed* the items that are either horizontal or vertical. We let $I_{small}$, $I_{large}$, $I_{hor}$, $I_{ver}$, $I_{skew}$, and $I_{int}$ be the items which are small, large, horizontal, vertical, skewed, and intermediate, respectively. The corresponding intersection with the optimal solution $OPT$ defines the sets $OPT_{small}$, $OPT_{large}$, $OPT_{hor}$, $OPT_{ver}$, $OPT_{skew}$ and $OPT_{int}$, respectively.

In order to describe our main ideas, we will start by considering the cardinality case of the problem (i.e., $p(i) = 1$ for each item $i \in I$) without rotations. In Sections 3, 4, and 5, we will present a simplified algorithm that yields a $1.6 + \varepsilon$ approximation.

Notice that the optimum solution can contain at most $1/\varepsilon_{large}^2$ large items. Thus, unless $|OPT| \leq 1/(\varepsilon \cdot \varepsilon_{large}^2)$ (in which case we can solve the problem optimally in time $n^{O(1/(\varepsilon \cdot \varepsilon_{large}^2))}$ by complete enumeration), we can drop all large items by losing only a factor of $1 + \varepsilon$ in the approximation. Similarly, by standard shifting techniques (e.g. Lemma 2.1 in [24]), when defining $\varepsilon_{large}$ and $\varepsilon_{small}$ one can ensure that the intermediate items can be neglected by losing only a factor of $1 + \varepsilon$ in the approximation ratio (while maintaining that $\varepsilon_{large}$ and $\varepsilon_{small}$ are lower-bounded by some constant depending only on $\varepsilon$).

Hence, w.l.o.g. we can assume that all items are small or skewed. It is possible to deal with small items by standard techniques from the literature (e.g. Section 6.2.1 in [24]), however this would make our exposition much more technical without introducing substantially new ideas. Hence, for the sake of simplicity, we will assume that there are no small items, i.e., all items are skewed. We remark that, even with the mentioned restrictions, the problem is far from being trivial. In particular, the best known approximation for the considered setting is $\frac{558}{325} + \varepsilon \approx 1.72$ [24]. Our simplified algorithm has a better approximation ratio $1.6 + \varepsilon$, and it is also substantially simpler.

For the main result, which is a $(4/3 + \varepsilon)$-approximation algorithms for the general case with and without rotations, please refer to the full version of this work.

## 3 Partition into LU-corridors

Our strategy is to partition the knapsack into $O_\varepsilon(1)$ thin corridors, each having the shape of an L or a U, such that there exists a $(1.6 + \varepsilon)$-approximate solution in which each item is contained in one of these corridors (see Figure 2). In this section, we will make this precise and show that such a partition indeed exists. Our algorithm will then guess this partition in polynomial time. In Sections 4 and 5 we will show how to find the corresponding solution afterwards efficiently.

Intuitively, a *path corridor* is a polygon inside $K$ that describes a path of width at most $\varepsilon \cdot \varepsilon_{large}$ and that is allowed to have bends, see Figure 3. Formally, it is a simple rectilinear polygon within $K$ with $2k$ edges $e_0, \ldots, e_{2k-1}$ for some integer $k \geq 2$, such that for each pair of horizontal (resp., vertical) edges $e_i, e_{2k-i}$, $i \in \{1, ..., k-1\}$ there exists a vertical (resp., horizontal) line segment $\ell_i$ of length less than $\varepsilon \cdot \varepsilon_{large}$ such that both $e_i$ and $e_{2k-i}$ intersect $\ell_i$ and $\ell_i$ does not intersect any other edge, and require $e_i$ and $e_{2k-i}$ to have length at least $\varepsilon_{large}/2$. Note that $e_0$ and $e_k$ are not required to satisfy these properties. We say that such a path corridor $C$ has $s(C) := k - 1$ *subcorridors.* We say that a *box* is a path corridor with only one subcorridor, i.e., it is simply a rectangle.

Similarly, a *cycle corridor* is intuitively a path corridor in which the start and end point of the path coincide (see Figure 3). Formally, we define it to be a face bounded by two simple non-intersecting rectilinear polygons defined by edges $e_0, e_1, \ldots, e_{k-1}$ and $e'_0, e'_1, \ldots, e'_{k-1}$,

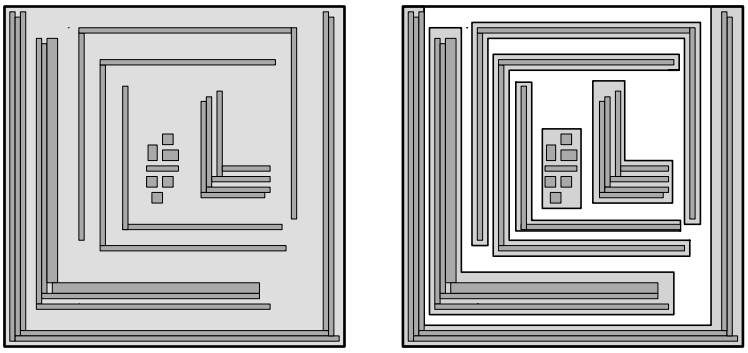


**Figure 2** Left: a packing of horizontal, vertical and small items into the knapsack. Right: a decomposition of the knapsack into box-shaped, L-shaped and U-shaped corridors which contain the previous items.

each of them of length at least $\varepsilon_{large}/2$, such that the second polygon is contained in the first one, and for each pair of corresponding horizontal (vertical) edges $e_i, e'_i$ for $i \in \{0, \ldots, k-1\}$ there is a vertical (horizontal, respectively) line segment $\ell_i$ of length less than $\varepsilon \cdot \varepsilon_{large}$ such that both edges $e_i$ and $e'_i$ intersect $\ell_i$ and $\ell_i$ does not intersect any other edge of the cycle corridor. We say that the resulting cycle corridor $C$ has $s(C) := k$ subcorridors.

We use a result from [2] that implies directly that there there exists a partition of $K$ into $O_\varepsilon(1)$ corridors and a near-optimal solution in which each item is contained in some corridor.

▶ **Lemma 1** ([2]). *There exists a solution $\overline{OPT} \subseteq OPT$ with $|OPT| \leq (1+\varepsilon)\left|\overline{OPT}\right|$ and a partition of $K$ into a set of corridors $\bar{\mathcal{C}}$ with $|\bar{\mathcal{C}}| \leq O_\varepsilon(1)$ where each corridor $C \in \bar{\mathcal{C}}$ has at most $1/\varepsilon$ subcorridors and each item $i \in \overline{OPT}$ is contained in a corridor of $\mathcal{C}$.*

Algorithmically, we could guess $\bar{\mathcal{C}}$ in time $N^{O_\varepsilon(1)}$ and then try to compute a profitable solution in which each item is contained in one corridor in $\bar{\mathcal{C}}$, i.e., mimicking $\overline{OPT}$. However, it is not clear how to compute such a solution in polynomial time. Therefore, we partition $\bar{\mathcal{C}}$ further into a set of smaller corridors $\mathcal{C}$ such that each resulting corridor has the shape of an L or a U. We will ensure that there exists a $(1.6+\varepsilon)$-approximate solution in which each item is contained in one corridor of $\mathcal{C}$. Since the corridors in $\mathcal{C}$ are simpler than the corridors in $\bar{\mathcal{C}}$, we will be able to compute in polynomial time essentially the most profitable solution in which each item is contained in a corridor of $\mathcal{C}$ (see in Sections 4 and 5).

We say that a path corridor $C$ is an *L-corridor* if $C$ has exactly two subcorridors, and then figuratively it has the shape of an L (see Figure 3). Intuitively, we define that a path corridor is a U-corridor if it has the shape of a U. Formally, let $C$ be a a path corridor with exactly three subcorridors, and hence $C$ is defined via edges $e_0, ..., e_7$. Assume w.l.o.g. that $e_2$ and $e_6$ are horizontal. We project $e_2$ and $e_6$ on the $x$-axis, let $I_2$ and $I_6$ be the resulting intervals. Then $C$ is a *U-corridor* if $I_2 \subseteq I_6$ or $I_6 \subseteq I_2$.

In order to partition $\bar{\mathcal{C}}$, we first define a partition of each path/cycle corridor $C \in \bar{\mathcal{C}}$ into $s(C)$ subcorridors (see Figure 3). We say that a *subcorridor* of a corridor $C$ is a simple polygon $P \subseteq C$ whose boundary consists of two parallel edges (in most cases these will be edges of $C$) and of two monotone axis-parallel curves, i.e., sets of axis-parallel line segments such that either for any two points $(x_1, y_1), (x_2, y_2) \in P$ where $x_1 < x_2$ we have $y_1 \leq y_2$, or for any such two points we have $y_1 \geq y_2$. We require that each vertex of $P$ has integral coordinates. Given a corridor $C$ and a set of non-overlapping items $I'$ inside $C$, we say that a partition of $C$ into a set of subcorridors is *nice for* $I'$ if each subcorridor either intersects only items from $I' \cap I_{hor}$ or only items from from $I' \cap I_{ver}$.

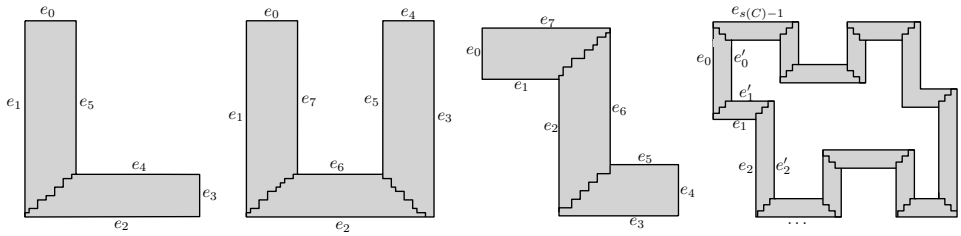



**Figure 3** An L-corridor, a U-corridor, another path corridor with two bends, and a cycle corridor. The monotone axis-parallel curves indicate the boundaries of the subcorridors.

▶ **Lemma 2** ([4], Lemma 2.4). *Let $C$ be a path/cycle corridor containing a set of items $I'$. There is a partition of $C$ into $s(C)$ subcorridors that is nice for $I'$.*

Now we take each path corridor $C \in \bar{\mathcal{C}}$ and delete the items in every third subcorridor, starting with the $\alpha$-th subcorridor for some offset $\alpha \in \{1, 2, 3\}$. Then we can divide $C$ into L-corridors such that each remaining item is contained in one of these L-corridors. Each item $i \in \overline{OPT}$ contained in $C$ is deleted only for one choice of $\alpha$, and hence there is a choice for $\alpha$ such that we lose at most one third of the profit due to this step. Now consider a cycle corridor $C \in \bar{\mathcal{C}}$. Note that $s(C)$ is even and $s(C) \geq 4$. If $s(C) = 4$ (i.e., $C$ is a ring), we delete the items in one of its four subcorridors, losing at most one quarter of the profit, and obtain a U-corridor. If $s(C) \geq 6$ and $s(C)$ is divisible by 3 we do the same operation as for path corridors, losing at most one third of the profit. For all other values of $s(C)$ we might lose a larger factor since then $s(C)$ is not divisible by 3 and $P_0$ and $P_{s(C)-1}$ are adjacent, e.g., if $s(C) = 8$. However, a case distinction shows that we can still partition $C$ into L- and U-corridors while decreasing the profit at most by a factor of 1.6.

▶ **Lemma 3.** *There exists a solution $OPT' \subseteq OPT$ with $|OPT| \leq (1.6 + \varepsilon)|OPT'|$ and a partition of $K$ into a set $\mathcal{C}$ of $O_\varepsilon(1)$ L- and U-corridors such that each item $i \in OPT'$ is contained in one corridor of $\mathcal{C}$.*

The first step in our algorithm is to guess $\mathcal{C}$ which can be done in time $N^{O_\varepsilon(1)}$. The next step is to compute a solution with at least $(1 - \varepsilon)|OPT'|$ items in which each item is contained in one corridor of $\mathcal{C}$. For this, we consider two cases separately, which are intuitively the case that $|OPT'| \geq \Omega_\varepsilon(\log N)$ and $|OPT'| \leq O_\varepsilon(\log N)$, and they are treated in Sections 4 and 5, respectively.

## 4 Packing via guessing slices

In this section, we assume that $|OPT'| > c_\varepsilon \cdot \log N$ for some constant $c_\varepsilon$ to be defined later. We describe an algorithm that computes a solution of size $(1 - \varepsilon)|OPT'|$ such that each item of this solution is contained in a corridor in $\mathcal{C}$.

First, we group the items into $O_\varepsilon(\log N)$ groups where we group the items in $I_{hor}$ according to their heights and the items in $I_{ver}$ according to their widths. Formally, for each $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$ we define $I^{(\ell)}_{hor} := \{i \in I_{hor} | h(i) \in [(1+\varepsilon)^\ell, (1+\varepsilon)^{\ell+1})\}$ and $I^{(\ell)}_{ver} := \{i \in I_{ver} | w(i) \in [(1+\varepsilon)^\ell, (1+\varepsilon)^{\ell+1})\}$. So intuitively, for each $\ell$ the items in $I^{(\ell)}_{hor}$ essentially all have the same height and the items in $I^{(\ell)}_{ver}$ essentially all have the same width. Now, for the groups $I^{(\ell)}_{hor}$, $I^{(\ell)}_{ver}$ we guess estimates $\text{opt}^{(\ell)}_{hor}, \text{opt}^{(\ell)}_{ver}$ for $|I^{(\ell)}_{hor} \cap OPT'|$, $|I^{(\ell)}_{ver} \cap OPT'|$, respectively. Even though there can be $\Theta_\varepsilon(\log N)$ of these groups and for each guessed value there are potentially $\Omega(n)$ options, we guess the estimates for *all* groups in parallel in time $(nN)^{O_\varepsilon(1)}$, adapting a technique from [13].

▶ **Lemma 4.** *In time* $(nN)^{O_\varepsilon(1)}$ *we can guess the values for all pairs* $\mathrm{opt}_{hor}^{(\ell)}, \mathrm{opt}_{ver}^{(\ell)}$ *with* $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$ *such that*

- $\sum_\ell \mathrm{opt}_{hor}^{(\ell)} + \mathrm{opt}_{ver}^{(\ell)} \geq (1-\varepsilon)|OPT'|$ *and*
- $\mathrm{opt}_{hor}^{(\ell)} \leq |OPT' \cap I_{hor}^{(\ell)}|$ *and* $\mathrm{opt}_{ver}^{(\ell)} \leq |OPT' \cap I_{hor}^{(\ell)}|$ *for each* $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$.

**Proof.** First, we guess $|OPT'|$ for which there are only $n$ options. We show now how to guess in time $N^{O_\varepsilon(1)}$ the feasible values for $\mathrm{opt}_{hor}^{(\ell)}$; a symmetric argument holds for $\mathrm{opt}_{ver}^{(\ell)}$. Let us define each $\mathrm{opt}_{hor}^{(\ell)}$ as the largest integer of the form $k_{hor}^{(\ell)} \cdot \frac{\varepsilon}{4\log_{1+\varepsilon} N}|OPT'|$ which is upper bounded by $|OPT' \cap I_{hor}^{(\ell)}|$, where $k_{hor}^{(\ell)}$ is a non-negative integer. Notice that trivially $\sum_\ell k_{hor}^{(\ell)} \leq \frac{4\log_{1+\varepsilon} N}{\varepsilon}$. We encode all such values $k_{hor}^{(\ell)}$ as a single binary string as follows: we represent each $k_{hor}^{(\ell)}$ as a string of $k_{hor}^{(\ell)}$ 0-bits followed by one 1-bit, and then chain such strings according to the index $\ell$. The final bit string encodes the solution. Notice that this string contains at most $\log_{1+\varepsilon} N + 1 + \sum_\ell k_{hor}^{(\ell)} = O(\frac{\log_{1+\varepsilon} N}{\varepsilon})$ bits, hence we can guess it in time $N^{O_\varepsilon(1)}$. The claim follows since

$$|OPT'| - \sum_\ell \left(\mathrm{opt}_{hor}^{(\ell)} + \mathrm{opt}_{ver}^{(\ell)}\right) \leq 2(\log_{1+\varepsilon} N + 1) \cdot \frac{\varepsilon}{4\log_{1+\varepsilon} N}|OPT'| \leq \varepsilon |OPT'|. \qquad \blacktriangleleft$$

**Definition of slices.** Next, for each group $I_{hor}^{(\ell)}$ we define slices that together are essentially as profitable as the items in $OPT' \cap I_{hor}^{(\ell)}$. We first order the items in $I_{hor}^{(\ell)}$ non-decreasingly by width and select the first $\frac{1}{1+\varepsilon}\mathrm{opt}_{hor}^{(\ell)}$ items. One can show easily that their total height is at most $h(OPT' \cap I_{hor}^{(\ell)})$. Let $i$ be one of these items. Intuitively, we slice $i$ horizontally into slices of height 1. Formally, for $i$ we introduce $h(i)$ items of height 1 and profit $1/(1+\varepsilon)^\ell$ each. Let $\hat{I}_{hor}^{(\ell)}$ denote the resulting set of slices. We do this procedure for each $\ell$ and a symmetric procedure for the group $I_{ver}^{(\ell)}$ for each $\ell$, resulting in a set of slices $\hat{I}_{ver}^{(\ell)}$.

▶ **Lemma 5.** *It is possible to place the slices in* $\left\{\hat{I}_{hor}^{(\ell)}, \hat{I}_{ver}^{(\ell)}\right\}_\ell$ *non-overlappingly inside* $K$ *such that each slice is contained in some corridor in* $\mathcal{C}$*. Also, we have that* $\sum_\ell \left(p(\hat{I}_{hor}^{(\ell)}) + p(\hat{I}_{ver}^{(\ell)})\right) \geq \frac{1}{1+O(\varepsilon)}|OPT'|$.

**Proof sketch.** Let $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$. Recall that $\mathrm{opt}_{hor}^{(\ell)} \leq |OPT' \cap I_{hor}^{(\ell)}|$, all items in $I_{hor}^{(\ell)}$ have the same height (up to a factor of $1+\varepsilon$), and we selected the $\frac{1}{1+\varepsilon}\mathrm{opt}_{hor}^{(\ell)}$ items in $I_{hor}^{(\ell)}$ of minimum width. Using this, one can show that the slices in $\hat{I}_{hor}^{(\ell)}$ fit into the space that is occupied by the items in $OPT' \cap I_{hor}^{(\ell)}$ in $OPT'$. Also, $\frac{1}{1+O(\varepsilon)}h(OPT' \cap I_{hor}^{(\ell)}) \leq |\hat{I}_{hor}^{(\ell)}| \leq h(OPT' \cap I_{hor}^{(\ell)})$ and each slice in $\hat{I}_{hor}^{(\ell)}$ has a profit of $1/(1+\varepsilon)^\ell$. Using this for each $\ell$ and a similar statement for the vertical items, one can prove the second claim of the lemma. $\blacktriangleleft$

Next, for each $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$ we round the widths of the slices in $\hat{I}_{hor}^{(\ell)}$ via linear grouping such that they have at most $1/\varepsilon$ different widths and we lose at most a factor of $1 + O(\varepsilon)$ in their profit due to this rounding. Formally, we sort the slices in $\hat{I}_{hor}^{(\ell)}$ non-increasingly by width and then partition them into $1/\varepsilon + 1$ groups such that each group contains $\left\lceil \frac{1}{1/\varepsilon+1}|\hat{I}_{hor}^{(\ell)}| \right\rceil$ slices (apart from possibly the last group which might contain fewer slices). Let $\hat{I}_{hor}^{(\ell)} = \hat{I}_{hor,1}^{(\ell)} \dot{\cup} ... \dot{\cup} \hat{I}_{hor,1/\varepsilon+1}^{(\ell)}$ denote the resulting partition. We drop the slices in $\hat{I}_{hor,1}^{(\ell)}$ (whose total profit is at most $\varepsilon \cdot p(\hat{I}_{hor}^{(\ell)})$). Then, for each $j \in \{2, ..., 1/\varepsilon+1\}$ we increase the width of the slices in $\hat{I}_{hor,j}^{(\ell)}$ to the width of the widest slice in $\hat{I}_{hor,j}^{(\ell)}$. By construction, the resulting slices have $1/\varepsilon$ different widths. Let $\tilde{I}_{hor}^{(\ell)}$ denote the resulting set and let $\tilde{I}_{hor}^{(\ell)} = \tilde{I}_{hor,1}^{(\ell)} \dot{\cup} ... \dot{\cup} \tilde{I}_{hor,1/\varepsilon}^{(\ell)}$ denote a partition of $\tilde{I}_{hor}^{(\ell)}$ according to the widths of the slices, i.e., for each $j \in \{1, ..., 1/\varepsilon\}$ the set $\tilde{I}_{hor,j}^{(\ell)}$ contains the rounded slices from $\hat{I}_{hor,j+1}^{(\ell)}$.

We do this procedure for each $\ell$ and a symmetric procedure for the group $I^{(\ell)}_{ver}$ for each $\ell$.

▶ **Lemma 6.** *It is possible to place the slices in* $\left\{\tilde{I}^{(\ell)}_{hor,j}, \tilde{I}^{(\ell)}_{ver,j}\right\}_{\ell,j}$ *non-overlappingly inside the knapsack such that each slice is contained in some corridor in* $\mathcal{C}$*. Also, we have* $\sum_\ell \left(p(\tilde{I}^{(\ell)}_{hor}) + p(\tilde{I}^{(\ell)}_{ver})\right) \geq \frac{1}{1+\varepsilon} \sum_\ell \left(p(\hat{I}^{(\ell)}_{hor}) + p(\hat{I}^{(\ell)}_{ver})\right)$.

**Proof sketch.** For each $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$ and $j \in \{2, ..., 1/\varepsilon + 1\}$, the slices in $\tilde{I}^{(\ell)}_{hor,j}$ fit into the space occupied by the slices in $\hat{I}^{(\ell)}_{hor,j-1}$. Also, for each $\ell$ we lost slices in $\hat{I}^{(\ell)}_{hor}$ with a total profit of at most $\varepsilon \cdot p(\hat{I}^{(\ell)}_{hor})$. A similar argumentation holds for the sets $I^{(\ell)}_{ver}$. ◀

We fix a partition of each corridor $C \in \mathcal{C}$ into subcorridors that is nice for the slices $\left\{\tilde{I}^{(\ell)}_{hor,j}, \tilde{I}^{(\ell)}_{ver,j}\right\}_{\ell,j}$. Note that we do not compute this partition explicitly but we use it for our analysis and as guidance for our algorithm. For each corridor $C \in \mathcal{C}$ or subcorridor $S$ denote by $\tilde{I}(C)$ and $\tilde{I}(S)$ the slices assigned to $C$ and $S$, resp., due to Lemma 6.

**Structuring slices inside subcorridors.** Consider a subcorridor $S$ of a corridor $C \in \mathcal{C}$. The placement of the slices inside $S$ due to Lemma 6 might be complicated. Instead, we would like to have a packing where the slices are packed *nicely*, i.e., they are stacked on top of each other if $S$ is horizontal, and side by side if $S$ is vertical (see Figure 4). This might not be possible to achieve exactly, but we construct something very similar. We prove that inside $S$ we can place $O_\varepsilon(1)$ boxes and one subcorridor $S' \subseteq S$ (which we will call sub-subcorridor in order to distinguish it from the subcorridors) that are pairwise disjoint and such that inside them we can nicely place essentially all slices from $\tilde{I}(C)$ (see Figure 4). We can guess the placement of the $O_\varepsilon(1)$ boxes inside $S$. Unfortunately, we cannot guess $S'$ directly, but we can guess the two edges that define the boundary of $S'$ together with the two axis-parallel curves. We refer to them as the *edges of* $S'$. Note that they are horizontal if $S$ (and hence $S'$) is horizontal, and vertical otherwise. We construct $S'$ such that the longer of these two edges is always also an edge of $S$ (see Figure 4).

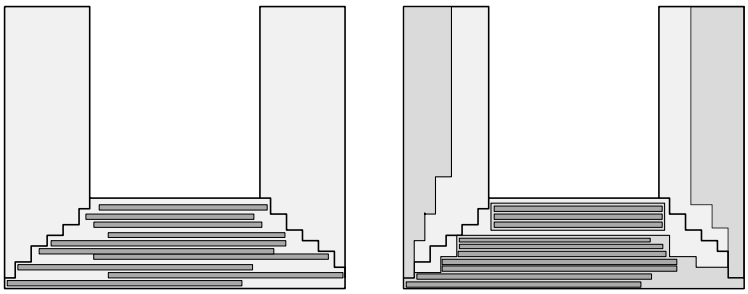


**Figure 4** Left: a subcorridor with items packed inside it. Right: A partition of each subcorridor into $O_\varepsilon(1)$ boxes and a sub-subcorridor, all containing slices which are nicely packed.

▶ **Lemma 7.** *For each horizontal/vertical subcorridor* $S$ *we can guess in time* $N^{O_\varepsilon(1)}$

- *the two edges of a horizontal/vertical sub-subcorridor* $S' \subseteq S$ *such that the longer edge of* $S'$ *coincides with the longer edge of* $S$*,*
- $O_\varepsilon(1)$ *non-overlapping boxes* $\mathcal{B}(S)$ *inside* $S$ *that are disjoint with* $S'$*,*

*such that we can nicely place slices from* $\tilde{I}(S)$ *with a total profit of* $(1-\varepsilon)p(\tilde{I}(S))$ *inside* $S'$ *and the boxes* $\mathcal{B}(S)$*.*

We apply Lemma 7 to each subcorridor $S$ of a corridor $C \in \mathcal{C}$. Let $\mathcal{S}$ and $\mathcal{B}$ denote the resulting set of sub-subcorridors and boxes, respectively. For each $\ell$, each $j$, and each $F \in \mathcal{B} \cup \mathcal{S}$ denote by $\tilde{I}^{(\ell)}_{hor,j}(F)$ and $\tilde{I}^{(\ell)}_{ver,j}(F)$ the respective slices from $\tilde{I}^{(\ell)}_{hor,j}, \tilde{I}^{(\ell)}_{ver,j}$ in $F$, respectively. Using simple slice reorderings, we can prove the following lemma.

▶ **Lemma 8.** *There is a packing of the slices in* $\left\{\tilde{I}^{(\ell)}_{hor,j}(F), \tilde{I}^{(\ell)}_{ver,j}(F)\right\}_{j,\ell,F}$ *such that for each box or sub-subcorridor* $F \in \mathcal{B} \cup \mathcal{S}$ *we can assume w.l.o.g. that*

- *horizontal/vertical items inside* $F$ *are ordered non-increasingly by width/height, starting at the longer edge of* $F$ *if* $F$ *is a sub-subcorridor, and starting at an arbitrary edge if* $F$ *is a box; ties are broken according to the input items that the slices correspond to,*
- *any two adjacent horizontal/vertical slices of the same width/height are placed exactly on top of each other/side by side.*

Therefore, we can construct this packing of the slices if we knew the cardinality of $\tilde{I}^{(\ell)}_{hor,j}(F)$ and $\tilde{I}^{(\ell)}_{ver,j}(F)$ for each $F \in \mathcal{B} \cup \mathcal{S}$ and each $\ell$ and $j$. We guess this cardinality approximately in the following lemma for each $F, \ell$ and $j$ in parallel.

▶ **Lemma 9.** *In time* $(nN)^{O_\varepsilon(1)}$ *we can guess values* $\operatorname{opt}^{(\ell)}_{hor,j}(F), \operatorname{opt}^{(\ell)}_{ver,j}(F)$ *for each* $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$, $j \in \{1, ..., 1/\varepsilon\}$, $F \in \mathcal{B} \cup \mathcal{S}$ *such that*

- $\operatorname{opt}^{(\ell)}_{hor,j}(F) \leq \left|\tilde{I}^{(\ell)}_{hor,j}(F)\right|$ *and* $\operatorname{opt}^{(\ell)}_{ver,j}(F) \leq \left|\tilde{I}^{(\ell)}_{ver,j}(F)\right|$ *for each* $\ell, j, F$ *and*
- $\sum_{F \in \mathcal{B} \cup \mathcal{S}} \operatorname{opt}^{(\ell)}_{hor,j}(F) \geq (1-\varepsilon) \sum_{F \in \mathcal{B} \cup \mathcal{S}} \left|\tilde{I}^{(\ell)}_{hor,j}(F)\right|$ *and* $\sum_{F \in \mathcal{B} \cup \mathcal{S}} \operatorname{opt}^{(\ell)}_{ver,j}(F) \geq (1-\varepsilon) \sum_{F \in \mathcal{B} \cup \mathcal{S}} \left|\tilde{I}^{(\ell)}_{ver,j}(F)\right|$ *for each* $\ell, j$.

**Proof sketch.** For each $\ell, j$, and $F$ we define $\operatorname{opt}^{(\ell)}_{hor,j}(F)$ to be the largest integral multiple of $\frac{\varepsilon}{|\mathcal{B}|+|\mathcal{S}|}\left|\tilde{I}^{(\ell)(F)}_{hor,j}\right|$ that is at most $\left|\tilde{I}^{(\ell)}_{hor,j}(F)\right|$ and note that for this value there are only $\frac{|\mathcal{B}|+|\mathcal{S}|}{\varepsilon} = O_\varepsilon(1)$ options. We define the values $\operatorname{opt}^{(\ell)}_{ver,j}(F)$ similarly. Since there are only $O_\varepsilon(\log nN)$ of these values altogether, we can guess all of them in time $2^{O_\varepsilon(\log nN)} = (nN)^{O_\varepsilon(1)}$. ◀

**Placing slices inside subcorridors.** Given the number of slices in each box and each sub-subcorridor due to Lemma 9, we compute a corresponding packing for the slices. Inside of each box we simply sort the slices by height or width, respectively, and then pack them in this order. For packing the slices inside the sub-subcorridors of a corridor $C$, recall that we do not know the precise sub-subcorridors, we know only the guessed edges due to Lemma 7. However, we can still find a packing for the slices inside of the sub-subcorridors of $C$. We start with the first sub-subcorridor $S_1$ of $C$, sort its slices by height or width, respectively (breaking ties according to the input items that the slices correspond to), and place them in this order, starting at the longer edge of $S_1$. When we do this, we push the slices as far as possible to the edge $e_0$. The resulting packing satisfies the properties of Lemma 8. If $s(C) \geq 2$ then we do the same procedure for the last sub-subcorridor $S_{s(C)}$ of $C$, and in particular we push its slices as far as possible to the edge $e_{s(C)}$. If $s(C) \in \{1, 2\}$ then we are done now. Otherwise $s(C) = 3$ since $s(C) \leq 3$ for each $C \in \mathcal{C}$ and the slices of the second sub-subcorridor $S_2$ are still not placed. We sort the slices as before and place them in this order, starting at the longer edge of $S_2$ and such that their placement satisfies the properties of Lemma 8. Since we had pushed the slices in $S_1$ and $S_3$ maximally to the edges $e_0$ and $e_k$, one can show that this is indeed possible.

**Rounding slices.** For each set $I^{(\ell)}_{hor}, I^{(\ell)}_{ver}$, their corresponding slices induce in total $O_\varepsilon(1)$ rectangular areas into which we assigned these slices: at most one for each of the $O_\varepsilon(1)$ sub-subcorridors and at most one for each of the $O_\varepsilon(1)$ boxes inside each of the $O_\varepsilon(1)$

subcorridors. For each $\ell$ we denote by $\mathcal{B}^{(\ell)}_{hor}, \mathcal{B}^{(\ell)}_{ver}$ these corresponding areas which are in fact boxes. Now the important observation is that inside the boxes $\mathcal{B}^{(\ell)}_{hor}$ we can place at least $(1-O(\varepsilon))\left|I^{(\ell)}_{hor} \cap OPT'\right| - 2|\mathcal{B}^{(\ell)}_{hor}|$ items from $I^{(\ell)}_{hor}$ as follows. Based on the slices for $I^{(\ell)}_{hor}$, we first construct a fractional packing of $\frac{1}{1+O(\varepsilon)}\text{opt}^{(\ell)}_{hor}$ items from $I^{(\ell)}_{hor}$ in which there are at most $2|\mathcal{B}^{(\ell)}_{hor}|$ items that are fractionally assigned to a box. Then we simply drop these fractional items. We use a symmetric procedure for the sets $I^{(\ell)}_{ver}$.

▶ **Lemma 10.** *For each $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$, in time $O_\varepsilon(nN)$ we can pack at least $(1-O(\varepsilon))\left|I^{(\ell)}_{hor} \cap OPT'\right| - 2|\mathcal{B}^{(\ell)}_{hor}|$ items from $I^{(\ell)}_{hor}$ into the boxes $\mathcal{B}^{(\ell)}_{hor}$. A symmetric statement holds for $I^{(\ell)}_{ver}$ and $\mathcal{B}^{(\ell)}_{ver}$ for each $\ell \in \{0, ..., \lfloor \log_{1+\varepsilon} N \rfloor\}$.*

Thus, we obtain a packing with $(1-O(\varepsilon))|OPT'| - 2\left(\sum_\ell |\mathcal{B}^{(\ell)}_{hor}| + |\mathcal{B}^{(\ell)}_{ver}|\right)$ items in total. Note that $\left(\sum_\ell |\mathcal{B}^{(\ell)}_{hor}| + |\mathcal{B}^{(\ell)}_{ver}|\right) \leq O_\varepsilon(\log N)$. Recall that we assumed that $|OPT'| > c_\varepsilon \log N$. Thus, by choosing $c_\varepsilon$ sufficiently large, we can ensure that $\left(\sum_\ell |\mathcal{B}^{(\ell)}_{hor}| + |\mathcal{B}^{(\ell)}_{ver}|\right) \leq \varepsilon \cdot |OPT'|$ and hence our packing contains at least $(1-O(\varepsilon))|OPT'|$ items in total.

▶ **Lemma 11.** *For each $\varepsilon > 0$ there is a constant $c_\varepsilon$ such that if $|OPT'| > c_\varepsilon \cdot \log N$ we can compute a solution of size $(1-\varepsilon)|OPT'|$ in time $(nN)^{O_\varepsilon(1)}$.*

## 5 Dynamic programming with color coding

Assume that $|OPT'| \leq c \cdot \log N$ for some given constant $c$ (which we will later choose to be the constant $c_\varepsilon$ defined in Section 4). We describe an algorithm that computes a solution of size $|OPT'|$ for this case in time $(nN)^{O(c)}$ such that each item of this solution is contained in a corridor in $\mathcal{C}$. Our strategy is to use color-coding [5] in order to reduce the setting of $O_\varepsilon(1)$ L- and U-corridors in $\mathcal{C}$ to the setting of only one single such corridor. Then we show how to solve this problem in polynomial time.

First, we guess $|OPT'|$. Then we color each item in $I$ randomly with one color in $\{1, ..., |OPT'|\}$. It is easy to show that with probability at least $1/e^{|OPT'|} \geq \frac{1}{N^{O(c)}}$ all items in $|OPT'|$ have different colors, in which case we say that the coloring was *successful*. If this is the case, then for each color $d \in \{1, ..., |OPT'|\}$ we can guess in time $O_\varepsilon(1)$ which corridor in $\mathcal{C}$ contains an item of $OPT'$ that we colored with color $d$. This yields $O_\varepsilon(1)^{|OPT'|} = N^{O_\varepsilon(c)}$ guesses overall. By repeating the random coloring $N^{O(c)}$ times, we can ensure that, with high probability, one of these colorings was successful. Also, we can derandomize this procedure using a $k$-perfect family of hash functions [5, 48], which yields the following lemma.

▶ **Lemma 12.** *In time $N^{O_\varepsilon(c)}$ we can guess a partition of $\{I_C\}_{C \in \mathcal{C}}$ of $I$ such that for each corridor $C \in \mathcal{C}$ the set $I_C$ contains all items from $OPT'$ that are placed inside $C$.*

### 5.1 Routine for one corridor

Recall that we are given a corridor $C \in \mathcal{C}$ and an input set $I_C$ of items colored with $\gamma \leq c \cdot \log N$ colors. W.l.o.g., let $\{1, \ldots, \gamma\}$ be these colors. Our goal is to place precisely one item per color inside $C$ such that they do not overlap. Let $OPT'_C$ denote the items of $OPT'$ placed inside $C$ and note that also $OPT'_C$ contains one item of each color.

For our $(1.6+\varepsilon)$-approximation it is sufficient to consider corridors with up to three sub-corridors; however, we will next describe a procedure that works for corridors with $k$ sub-corridors for any $k \leq 1/\varepsilon$. This extension will actually be needed to obtain a $(4/3+\varepsilon)$-approximation (see the full version).

Our strategy is to cut $C$ recursively into pieces (see Figure 5). Whenever we make a cut, we guess the items from $OPT'_C$ that are intersected by this cut and their placement in $OPT'_C$. The cut splits the considered subpart of $C$ into two pieces and we guess the colors of the items in $OPT'_C$ in each one of these pieces. Then, we recursively solve the subproblem defined by each piece. Guessing the colors ensures that we do not place an item twice, e.g., once in each of the two subproblems. We define our cuts such that there are only a polynomial number of possible arising pieces during the recursion, and we observe that for the guesses of the colors there are only $2^\gamma \leq N^c$ many options. Hence, we can embed this recursion into a polynomial time dynamic program.

**Long chords.** Formally, whenever we cut $C$ we do this along long chords defined as follows. A *long chord* is a sequence of $k$ axis-parallel line segments $f_1, ..., f_k$ that intuitively connect $e_0$ with $e_{k+1}$, i.e., such that for each $j \in \{1, ..., k\}$ each end-point of $f_j$ has integral coordinates and coincides with an endpoint of $f_{j-1}$ or $f_{j+1}$ or lies on $e_0$ or $e_{k+1}$, see Figure 5. Note that there are two special long chords that go along the edges of $C$, defined by $\ell_R := e_1, ..., e_k$ and $\ell_L := e_{k+2}, ..., e_{2k+1}$.

We can compute a set $\mathcal{L}$ containing all of the $N^{O(k)}$ long chords. We fix an (unknown) partition of $C$ into $s(C) =: k$ subcorridors $S_1, ..., S_k$ that is nice for $OPT'_C$. We are interested in the long chords $f_1, ..., f_k$ in $\mathcal{L}$ with the property that for each $j \in \{1, ..., k\}$ the line segment $f_j$ is contained in $S_j$ and it is parallel to the two parallel edges that define $S_j$ (see Figure 5). We say that such a long chord is *consistent with* $S_1, ..., S_k$ (or just *consistent* for short). Note that we do not know $S_1, ..., S_k$ and hence we cannot determine whether a given long chord is consistent or not. However, there are two key observations

- we can subdivide $C$ recursively along the long chords such that each arising piece is defined as the area enclosed by two given long chords and $e_0$ and $e_{k+1}$ (see Figure 5),
- each consistent long chord can intersect with at most $k/\varepsilon_{large}$ items in $OPT'_C$ since the corridors are thin and all input items are skewed.

**Subproblems of DP.** Therefore, we can compute a recursive partitioning of $C$ via a dynamic program. Each cell of the DP-table is defined by

- two long chords $\ell_1, \ell_2 \in \mathcal{L}$ that might intersect but that do not properly cross each other; together with (a part of) $e_0$ and $e_{k+1}$ they define a polygon $C' \subseteq C$,
- a set of $O(k/\varepsilon_{large})$ items $I'_C \subseteq I_C$ with a non-overlapping placement of them inside $C$ such that the interior of each item in $I'_C$ intersects $\ell_1$ or $\ell_2$, and
- a set of colors $\Gamma \subseteq \{1, ..., \gamma\}$.

The subproblem encoded in this cell is to place items from $I_C$ inside $C'$ such that they do not overlap with the items in $I'_C$ and such that for each color $d \in \Gamma$ we place exactly one item of color $d$. If this subproblem has a solution $OPT(\ell_1, \ell_2, I'_C, \Gamma)$, we store it in the corresponding DP-cell; otherwise we store *fail*.

To compute such a solution, consider any long chord $\ell \in \mathcal{L}$ that lies completely inside $C'$ but is not identical to $\ell_1$ or $\ell_2$ (we would like to select a consistent long chord; however, we do not know which long chords are consistent and hence we try all of them). Let us first assume that at least one such $\ell$ exists. Note that $\ell$ divides $C'$ into two smaller polygons $C'_1, C'_2$ that are surrounded by the pairs $(\ell_1, \ell)$, and $(\ell, \ell_2)$, respectively. Then, we consider

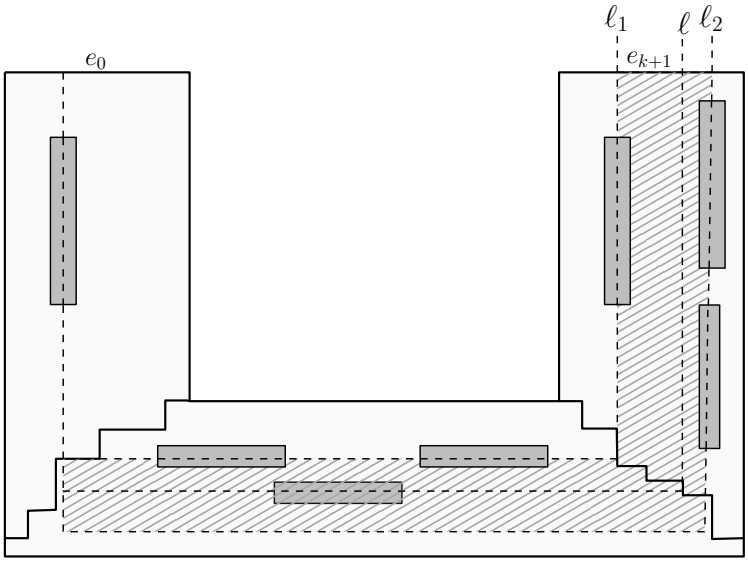



**Figure 5** A U-corridor and two consistent long chords $\ell_1$ and $\ell_2$. The long chords intersect only a constant number of items from the optimal solution (shown in the figure). There is a DP-cell that is defined by $\ell_1$, $\ell_2$, and these items, and whose corresponding area is shaded. This cell splits into smaller subproblems by defining a new long chord $\ell$ that lies in-between $\ell_1$ and $\ell_2$.

any subset of items $I''_C \subseteq I_C$ and a placement of such items inside $C'$ such that: (1) $I''_C$ are pairwise non-overlapping and not overlapping with $I'_C$, (2) they are intersected by $\ell$ in their interior, and (3) have distinct colors $\Gamma_\ell \subseteq \Gamma$. Finally, we consider any partition $\Gamma_1 \dot\cup \Gamma_2$ of the remaining colors $\Gamma \setminus \Gamma_\ell$. Let $I'_{C,1}$ and $I'_{C,2}$ be the items in $I'_C \cup I''_C$ that intersect $C'_1$ and $C'_2$, respectively. We consider the DP-cells $(\ell_1, \ell, I'_{C,1}, \Gamma_1)$ and $(\ell, \ell_2, I'_{C,2}, \Gamma_2)$ and, if none of them contains the value "fail", we store in $(\ell_1, \ell_2, I'_C, \Gamma)$ the union of $I''_C$, $OPT(\ell_1, \ell, I'_{C,1}, \Gamma_1)$, and $OPT(\ell, \ell_2, I'_{C,2}, \Gamma_2)$ (together with the placement of the corresponding items) and halt the computation for the considered DP-cell. If the above event never happens, we store "fail" in this DP-cell.

The base cases of the DP are given by pairs $\ell_1, \ell_2$ which are at most one unit apart from each other (everywhere inside $C$), so that it is not possible to define any long chord $\ell$ between $\ell_1$ and $\ell_2$ (recall that the endpoints of the line segments of the long chords have integral coordinates). Notice however that in this case at most $O(k/\varepsilon_{large})$ skewed items can fit inside $C'$, hence we can determine whether a feasible solution $OPT(\ell_1, \ell_2, I'_C, \Gamma)$ exists by enumeration in time $(nN)^{O(k/\varepsilon_{large})}$.

At the end we output the solution stored in the cell $(\ell_L, \ell_R, \emptyset, \{1, ..., \gamma\})$. We will show that this is the optimal solution for $C$. The number of DP-cells is bounded by $(nN)^{O(k/\varepsilon_{large})} \cdot 2^\gamma$ and the number of possible guesses when computing the entry of a DP-cell is bounded by $(nN)^{O(k/\varepsilon_{large})} \cdot 2^{O(\gamma)}$. This allows us to bound the running time of our DP.

▶ **Lemma 13.** *Given a path corridor $C$ with $k$ subcorridors and a set of skewed items $I_C$ with $\gamma$ distinct colors. In time $(nN)^{O(k/\varepsilon_{large})} \cdot 2^{O(\gamma)}$ we can determine whether there exists a set $I'_C \subseteq I_C$ with $\gamma$ distinct colors that fits non-overlappingly inside $C$.*

We apply Lemma 13 to each corridor $C \in \mathcal{C}$ which yields the following lemma.

▶ **Lemma 14.** *Assume that $|OPT'| \leq c \cdot \log N$ for some constant $c$. Then we can compute a solution of size $|OPT'|$ in time $(nN)^{O(c)}$.*

Now Lemmas 11 and 14 yield the following theorem.

▶ **Theorem 15.** *There is an* $(1.6 + \varepsilon)$*-approximation algorithm with a running time of* $(nN)^{O_\varepsilon(1)}$ *for unweighted instances of 2DK with only skewed items.*

## References


1 Fidaa Abed, Parinya Chalermsook, José R. Correa, Andreas Karrenbauer, Pablo Pérez-Lantero, José A. Soto, and Andreas Wiese. On guillotine cutting sequences. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015*, volume 40 of *LIPIcs*, pages 1–19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.1.

2 Anna Adamaszek, Sariel Har-Peled, and Andreas Wiese. Approximation schemes for independent set and sparse subsets of polygons. *J. ACM*, 66(4):29:1–29:40, 2019. doi:10.1145/3326122.

3 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.

4 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.

5 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

6 Brenda S. Baker, Edward G. Coffman Jr., and Ronald L. Rivest. Orthogonal packings in two dimensions. *SIAM J. Comput.*, 9(4):846–855, 1980. doi:10.1137/0209064.

7 Nikhil Bansal, Alberto Caprara, Klaus Jansen, Lars Prädel, and Maxim Sviridenko. A structural lemma in 2-dimensional packing, and its implications on approximability. In *Algorithms and Computation, 20th International Symposium, ISAAC 2009*, volume 5878 of *Lecture Notes in Computer Science*, pages 77–86. Springer, 2009. doi:10.1007/978-3-642-10631-6_10.

8 Nikhil Bansal, Alberto Caprara, and Maxim Sviridenko. A new approximation method for set covering problems, with applications to multidimensional bin packing. *SIAM J. Comput.*, 39(4):1256–1278, 2009. doi:10.1137/080736831.

9 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.

10 Alberto Caprara. Packing 2-dimensional bins in harmony. In *43rd Symposium on Foundations of Computer Science (FOCS 2002)*, pages 490–499. IEEE Computer Society, 2002. doi:10.1109/SFCS.2002.1181973.

11 Parinya Chalermsook and Julia Chuzhoy. Maximum independent set of rectangles. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009*, pages 892–901. SIAM, 2009. URL: http://dl.acm.org/citation.cfm?id=1496770.1496867.

12 Timothy M. Chan and Sariel Har-Peled. Approximation algorithms for maximum independent set of pseudo-disks. *Discret. Comput. Geom.*, 48(2):373–392, 2012. doi:10.1007/s00454-012-9417-5.

13 Chandra Chekuri and Sanjeev Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM J. Comput.*, 35(3):713–728, 2005. doi:10.1137/S0097539700382820.

14 Miroslav Chlebík and Janka Chlebíková. Hardness of approximation for orthogonal rectangle packing and covering problems. *J. Discrete Algorithms*, 7(3):291–305, 2009. doi:10.1016/j.jda.2009.02.002.

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

16 Fan Chung, Michael R. Garey, and David S. Johnson. On packing two-dimensional bins. *SIAM Journal on Algebraic Discrete Methods*, 3:66–76, 1982. doi:10.1137/0603007.

17 Julia Chuzhoy and Alina Ene. On approximating maximum independent set of rectangles. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 820–829. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.92.

18 Edward G. Coffman Jr., M. R. Garey, David S. Johnson, and Robert Endre Tarjan. Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM J. Comput.*, 9(4):808–826, 1980. doi:10.1137/0209062.

19 Florian Diedrich, Rolf Harren, Klaus Jansen, Ralf Thöle, and Henning Thomas. Approximation algorithms for 3d orthogonal knapsack. *J. Comput. Sci. Technol.*, 23(5):749–762, 2008. doi:10.1007/s11390-008-9170-7.

20 Thomas Erlebach, Klaus Jansen, and Eike Seidel. Polynomial-time approximation schemes for geometric intersection graphs. *SIAM J. Comput.*, 34(6):1302–1323, 2005. doi:10.1137/S0097539702402676.

21 Aleksei V. Fishkin, Olga Gerber, and Klaus Jansen. On efficient weighted rectangle packing with large resources. In *Algorithms and Computation, 16th International Symposium, ISAAC 2005*, volume 3827 of *Lecture Notes in Computer Science*, pages 1039–1050. Springer, 2005. doi:10.1007/11602613_103.

22 Aleksei V. Fishkin, Olga Gerber, Klaus Jansen, and Roberto Solis-Oba. Packing weighted rectangles into a square. In *Mathematical Foundations of Computer Science 2005, 30th International Symposium, MFCS 2005*, volume 3618 of *Lecture Notes in Computer Science*, pages 352–363. Springer, 2005. doi:10.1007/11549345_31.

23 Waldo Gálvez, Fabrizio Grandoni, Afrouz Jabal Ameli, Klaus Jansen, Arindam Khan, and Malin Rau. A tight $(3/2+\epsilon)$ approximation for skewed strip packing. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX-/RANDOM 2020*, volume 176 of *LIPIcs*, pages 44:1–44:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.44.

24 Waldo Gálvez, Fabrizio Grandoni, Sandy Heydrich, Salvatore Ingala, Arindam Khan, and Andreas Wiese. Approximating geometric knapsack via l-packings. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 260–271. IEEE Computer Society, 2017. Full version available at http://www.dii.uchile.cl/~awiese/2DK_full_version.pdf. doi:10.1109/FOCS.2017.32.

25 Waldo Gálvez, Fabrizio Grandoni, Salvatore Ingala, and Arindam Khan. Improved pseudo-polynomial-time approximation for strip packing. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016*, volume 65 of *LIPIcs*, pages 9:1–9:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.FSTTCS.2016.9.

26 Fabrizio Grandoni, Stefan Kratsch, and Andreas Wiese. Parameterized approximation schemes for independent set of rectangles and geometric knapsack. In *27th Annual European Symposium on Algorithms, ESA 2019*, volume 144 of *LIPIcs*, pages 53:1–53:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ESA.2019.53.

27 Rolf Harren. Approximation algorithms for orthogonal packing problems for hypercubes. *Theor. Comput. Sci.*, 410(44):4504–4532, 2009. doi:10.1016/j.tcs.2009.07.030.

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

29 Rolf Harren and Rob van Stee. Improved absolute approximation ratios for two-dimensional packing problems. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009*, volume 5687 of *Lecture Notes in Computer Science*, pages 177–189. Springer, 2009. doi:10.1007/978-3-642-03685-9_14.

30 Sören Henning, Klaus Jansen, Malin Rau, and Lars Schmarje. Complexity and inapproximability results for parallel task scheduling and strip packing. *Theory Comput. Syst.*, 64(1):120–140, 2020. doi:10.1007/s00224-019-09910-6.

31 Sandy Heydrich and Andreas Wiese. Faster approximation schemes for the two-dimensional knapsack problem. *ACM Trans. Algorithms*, 15(4):47:1–47:28, 2019. doi:10.1145/3338512.

32 Harry B. Hunt III, Madhav V. Marathe, Venkatesh Radhakrishnan, S. S. Ravi, Daniel J. Rosenkrantz, and Richard Edwin Stearns. Nc-approximation schemes for NP- and pspace-hard problems for geometric graphs. *J. Algorithms*, 26(2):238–274, 1998. doi:10.1006/jagm.1997.0903.

33 Klaus Jansen and Malin Rau. Closing the gap for pseudo-polynomial strip packing. In *27th Annual European Symposium on Algorithms, ESA 2019*, volume 144 of *LIPIcs*, pages 62:1–62:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ESA.2019.62.

34 Klaus Jansen and Malin Rau. Improved approximation for two dimensional strip packing with polynomial bounded width. *Theor. Comput. Sci.*, 789:34–49, 2019. doi:10.1016/j.tcs.2019.04.002.

35 Klaus Jansen and Roberto Solis-Oba. New approximability results for 2-dimensional packing problems. In *Mathematical Foundations of Computer Science 2007, 32nd International Symposium, MFCS 2007*, volume 4708 of *Lecture Notes in Computer Science*, pages 103–114. Springer, 2007. doi:10.1007/978-3-540-74456-6_11.

36 Klaus Jansen and Roberto Solis-Oba. A polynomial time approximation scheme for the square packing problem. In *Integer Programming and Combinatorial Optimization, 13th International Conference, IPCO 2008*, volume 5035 of *Lecture Notes in Computer Science*, pages 184–198. Springer, 2008. doi:10.1007/978-3-540-68891-4_13.

37 Klaus Jansen and Rob van Stee. On strip packing with rotations. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, STOC 2005*, pages 755–761. ACM, 2005. doi:10.1145/1060590.1060702.

38 Klaus Jansen and Guochuan Zhang. Maximizing the number of packed rectangles. In *Algorithm Theory – SWAT 2004, 9th Scandinavian Workshop on Algorithm Theory*, volume 3111 of *Lecture Notes in Computer Science*, pages 362–371. Springer, 2004. doi:10.1007/978-3-540-27810-8_31.

39 Klaus Jansen and Guochuan Zhang. On rectangle packing: maximizing benefits. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004*, pages 204–213. SIAM, 2004. URL: http://dl.acm.org/citation.cfm?id=982792.982822.

40 Claire Kenyon and Eric Rémila. A near-optimal solution to a two-dimensional cutting stock problem. *Math. Oper. Res.*, 25(4):645–656, 2000. doi:10.1287/moor.25.4.645.12118.

41 Arindam Khan. *Approximation algorithms for multidimensional bin packing.* PhD thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2016. URL: http://hdl.handle.net/1853/54371.

42 Arindam Khan, Arnab Maiti, Amatya Sharma, and Andreas Wiese. On guillotine separable packings for the two-dimensional geometric knapsack problem. In *To appear in SoCG*, 2021.

43 Arindam Khan and Madhusudhan Reddy Pittu. On guillotine separability of squares and rectangles. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020*, volume 176 of *LIPIcs*, pages 47:1–47:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.47.

44 Arindam Khan, Eklavya Sharma, and K. V. N. Sreenivas. Approximation algorithms for generalized multidimensional knapsack. *CoRR*, abs/2102.05854, 2021. arXiv:2102.05854.

45 Carla Negri Lintzmayer, Flávio Keidi Miyazawa, and Eduardo Candido Xavier. Two-dimensional knapsack for circles. In *LATIN 2018: Theoretical Informatics – 13th Latin American Symposium*, volume 10807 of *Lecture Notes in Computer Science*, pages 741–754. Springer, 2018. doi:10.1007/978-3-319-77404-6_54.

**46** Arturo I. Merino and Andreas Wiese. On the two-dimensional knapsack problem for convex polygons. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020*, volume 168 of *LIPIcs*, pages 84:1–84:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ICALP.2020.84.

**47** Giorgi Nadiradze and Andreas Wiese. On approximating strip packing with a better ratio than 3/2. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016*, pages 1491–1510. SIAM, 2016. doi:10.1137/1.9781611974331.ch102.

**48** Moni Naor, Leonard J. Schulman, and Aravind Srinivasan. Splitters and near-optimal derandomization. In *36th Annual Symposium on Foundations of Computer Science, FOCS 1995*, pages 182–191. IEEE Computer Society, 1995. doi:10.1109/SFCS.1995.492475.

**49** Ingo Schiermeyer. Reverse-fit: A 2-optimal algorithm for packing rectangles. In *Algorithms – ESA '94, Second Annual European Symposium*, volume 855 of *Lecture Notes in Computer Science*, pages 290–299. Springer, 1994. doi:10.1007/BFb0049416.

**50** Daniel Dominic Sleator. A 2.5 times optimal algorithm for packing in two dimensions. *Inf. Process. Lett.*, 10(1):37–40, 1980. doi:10.1016/0020-0190(80)90121-0.

**51** A. Steinberg. A strip-packing algorithm with absolute performance bound 2. *SIAM J. Comput.*, 26(2):401–409, 1997. doi:10.1137/S0097539793255801.