Closing the Gap for Pseudo-Polynomial Strip Packing

Klaus Jansen
Institut für Informatik, Christian-Albrechts-Universität zu Kiel, Germany
kj@informatik.uni-kiel.de

Malin Rau
L'Institute Polytechnique de Grenoble, Grenoble INP, France
Malin.Rau@inria.fr

Abstract

Two-dimensional packing problems are a fundamental class of optimization problems and Strip Packing is one of the most natural and famous among them. Indeed it can be defined in just one sentence: Given a set of rectangular axis parallel items and a strip with bounded width and infinite height, the objective is to find a packing of the items into the strip minimizing the packing height. We speak of pseudo-polynomial Strip Packing if we consider algorithms with pseudo-polynomial running time with respect to the width of the strip. It is known that there is no pseudo-polynomial time algorithm for Strip Packing with a ratio better than $5/4$ unless $\mathrm{P} = \mathrm{NP}$. The best algorithm so far has a ratio of $4/3 + \varepsilon$. In this paper, we close the gap between inapproximability result and currently known algorithms by presenting an algorithm with approximation ratio $5/4 + \varepsilon$. The algorithm relies on a new structural result which is the main accomplishment of this paper. It states that each optimal solution can be transformed with bounded loss in the objective such that it has one of a polynomial number of different forms thus making the problem tractable by standard techniques, i.e., dynamic programming. To show the conceptual strength of the approach, we extend our result to other problems as well, e.g., Strip Packing with 90 degree rotations and Contiguous Moldable Task Scheduling, and present algorithms with approximation ratio $5/4+\varepsilon$ for these problems as well.

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

Keywords and phrases Strip Packing, pseudo-polynomial, 90 degree rotation, Contiguous Moldable Task Scheduling

Digital Object Identifier 10.4230/LIPIcs.ESA.2019.62

Related Version A full version of the paper is available at `https://arxiv.org/abs/1712.04922`.

Funding Research was supported by German Research Foundation (DFG) project JA 612 /20-1.

1 Introduction

Two-dimensional packing problems typically have quite natural formulations and arise in a wide variety of contexts (see e.g. [8]). A characteristic challenge in this kind of problem is the space efficient placement of rectangles in a given area. Despite their simple description, they are usually quite hard and require sophisticated algorithmic techniques in order to reliably and efficiently find good solutions. Indeed, the study of algorithms for fundamental two-dimensional packing problems, like, e.g., Strip Packing, 2D-Knapsack, 2D-Bin Packing, or Unsplittable Flow on a Path, can be traced back to 1980 when Baker et al. [4] and Coffman et al [9] studied the first algorithms for two-dimensional packing problems. Furthermore, new results for these problems are regularly presented on top level conferences like FOCS, STOC and SODA up to today (see, e.g., [2, 5, 10, 13, 14, 18, 33, 27]). As all of these packing problems are NP-hard, they are typically studied in the context of approximation algorithms.

27th Annual European Symposium on Algorithms (ESA 2019).
Editors: Michael A. Bender, Ola Svensson, and Grzegorz Herman; Article No. 62; pp. 62:1–62:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

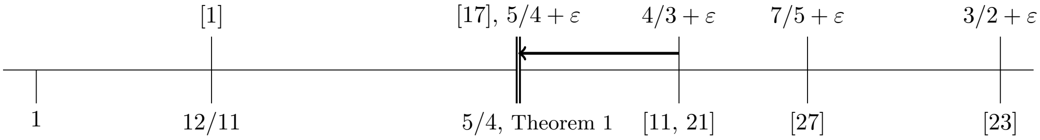


Figure 1 The upper and lower bounds for pseudo-polynomial approximations achieved so far.

We say an approximation algorithm A has an (absolute) approximation ratio α (or call it α-approximation) if for each instance I of the problem it holds that $A(I) \leq \alpha\mathrm{OPT}(I)$, where $\mathrm{OPT}(I)$ is the optimal value of the corresponding objective function.

Although there is a huge range of work related to improving the absolute approximation ratio of algorithms for Strip Packing [3, 4, 6, 9, 12, 16, 22, 24, 28, 29, 30, 31] and there have been breakthroughs for 2D-Knapsack [10] and Unsplittable Flow on a Path [14], these problems are still not understood well. In the context of Strip Packing, for instance, there is a huge gap between the best known lower and upper bound of 3/2 and $5/3+\varepsilon$ [15], respectively. Similar, for 2D-Knapsack and Unsplittable Flow on a Path, $(1+\varepsilon)$-approximation schemes might be possible while the best algorithms known so far have absolute approximation ratios of $17/9+\varepsilon$ [10] and $5/3+\varepsilon$ [14] respectively. Closing these gaps between lower and upper bounds poses a fascinating challenge.

To close these gaps, it is essential that the corresponding problem and the structure of optimal or at least good solutions, in particular, are understood well. Hence, it can be helpful to look at the problem from different angles and consider other goals regarding the approximation or the running time. One example, where this approach has already been particularly effective, is the consideration of asymptotic approximation ratios, where we allow an extra additive term, i.e., an algorithm A has an *asymptotic* approximation ratio of α if there exists a constant c such that $A(I) \leq \alpha\mathrm{OPT}+c$ for each instance I. While there has been extensive work on algorithms with asymptotic approximation ratios [3, 9, 12, 24, 31, 6, 22] the algorithm by Kenyon and Rémila [24] is particularly prominent. It has an asymptotic approximation ratio of $(1+\varepsilon)\mathrm{OPT}+\mathcal{O}(h_{\max}/\varepsilon^2)$ for each $\varepsilon > 0$ where $h_{\max}$ is the largest occurring item height. Due to its small running time (which is a polynomial in the number of jobs as well as $1/\varepsilon$) and its relatively small additive term, the techniques used in this algorithm have become the standard to handle items which have a small height compared to the value of the objective function in most of the later developed algorithms for Strip Packing and other 2-dimensional packing problems. On the other hand, for the 2-dimensional geometric Knapsack problem, the consideration of other running times (as e.g. in [2] where the considered algorithm has a pseudo- and quasi-polynomial running time, which allows the size of the Knapsack and terms of the form $2^{\log(n)^{\mathcal{O}(1)}}$ to appear as factors in the running time) have brought new insights, which ultimately led to an algorithm for this problem (presented in [10]) that has the currently best approximation ratio of $\frac{17}{9}+\varepsilon$.

In this spirit, algorithms with pseudo-polynomial running time, which allow the widths of the strip or the size of the smallest or largest item to appear in the running time with a polynomial dependence, have been considered for the Strip Packing problem to provide a better understanding of its hardness, see Figure 1 for an overview. The so far best pseudo-polynomial time algorithm has an approximation ratio of $4/3+\varepsilon$ [11, 21] while there is a lower bound of 5/4 (see [17]) on the approximation ratio for this kind of algorithms unless $\mathrm{P}=\mathrm{NP}$.

Results

Before we summarize the results presented in this paper, we define the Strip Packing problem formally. We have to pack a set $\mathcal{I}$ of n rectangular items into a given strip with width $W \in \mathbb{N}$ and infinite height. Each item $i \in \mathcal{I}$ has a width $w(i) \in \mathbb{N}_{\leq W}$ and a height $h(i) \in \mathbb{N}$. The area of an item $i \in \mathcal{I}$ is defined as $a(i) := h(i) \cdot w(i)$ and the area of a set of items $\mathcal{I}' \subseteq \mathcal{I}$ is defined as $a(\mathcal{I}') := \sum_{i \in \mathcal{I}'} a(i)$. A packing of the items is given by a mapping $\rho : I \to \mathbb{N}_{\leq W} \times \mathbb{N}, i \mapsto (x_i, y_i)$ which assigns the lower left corner of an item $i \in I$ to a position $\rho(i) = (x_i, y_i)$ in the strip. An inner point of $i \in \mathcal{I}$ (with respect to a packing ρ) is a point from the set $\text{inn}(i) := \{(x, y) \in \mathbb{R} \times \mathbb{R} \,|\, x_i < x < x_i + w(i), y_i < y < y_i + h(i)\}$. We say two items $i, j \in \mathcal{I}$ overlap if they share an inner point (i.e., $\text{inn}(i) \cap \text{inn}(j) \neq \emptyset$). A packing is feasible if no two items overlap and if $x_i + w(i) \leq W$ for all $i \in I$. The objective of the Strip Packing problem is to find a feasible packing ρ with minimal height $h(\rho) := \max\{y_i + h(i) \,|\, i \in \mathcal{I}, \rho(i) = (x_i, y_i)\}$. Given an instance I of the Strip Packing problem, we denote this minimal packing height with $\text{OPT}(I)$ and dismiss the I if the instance is clear from the context.

Analyzing the structure of solutions is a valuable tool in the development of algorithms, and this holds for approximation as well as exact algorithms. By analyzing the structure of optimal solutions and finding properties that all optimal solutions share, we aim to dramatically reduce the search space of solutions in size and gain other structural insights enabling the application of well-understood algorithmic techniques like dynamic or integer programming. This general approach is widely used in the context of two-dimensional packing problems, and there are many success stories in other areas of combinatorial optimization as well. One such example is the problem of Scheduling on Identical Machines where it lead to an approximation scheme [19] whose running time (nearly) matches the lower bound [7]. In this paper, we present an analysis of the structure of optimal solutions that consist of rectangular objects placed inside a rectangular packing area, that is restricted on one side. The structural result developed from this consideration (see Lemma 3) is particularly valuable in the design of algorithms for the Strip Packing problem as we can find a pseudo-polynomial time algorithm that matches the lower bound of $5/4$ except for a negligibly small ε.

▶ **Theorem 1.** *There is a pseudo-polynomial algorithm for Strip Packing which finds a* $(5/4 + \varepsilon)$*-approximation in* $\mathcal{O}(n \log(n)) \cdot W^{\mathcal{O}_\varepsilon(1)}$ *operations, where* $\mathcal{O}_\varepsilon$ *dismisses all factors solely dependent on* $1/\varepsilon$.

Moreover, since we consider optimal solutions with the above described properties, this result also comes in handy for the development of algorithms for the problem Contiguous Moldable Task Scheduling, which is a generalization of Strip Packing where each rectangular item can take on a bounded number of different shapes. However, when adapting the algorithm to this problem, we get a running time where $\mathcal{O}_\varepsilon(1)$ also does appear in the exponent of the number of items n, see Theorem 2. More formally in this problem, we are given a set $\mathcal{J}$ of n jobs and m identical machines. Each job $j \in \mathcal{J}$ can be scheduled on different numbers of machines given by $M_j \subseteq \{1, \dots, m\}$. Depending on the number of machines $i \in M_j$, each job $j \in \mathcal{J}$ has a specific processing time $p_j(i) \in \mathbb{N}$. A schedule S is given by three functions: $\sigma : \mathcal{J} \to \mathbb{N}$ which maps each job $j \in \mathcal{J}$ to a starting time $\sigma(j)$; $\rho : \mathcal{J} \to \{1, \dots, m\}$ which maps each job $j \in \mathcal{J}$ to the number of processors $\rho(j) \in M_j$ it is processed on; and $\varphi : \mathcal{J} \to \{1, \dots, m\}$ which maps each job $j \in \mathcal{J}$ to the first machine it is processed on. The job $j \in \mathcal{J}$ will use the machines $\varphi(j)$ to $\varphi(j) + \rho(j) - 1$ contiguously. A schedule $S = (\sigma, \rho, \varphi)$ is feasible if each machine processes at most one job at a time and its makespan is defined by $\max_{j \in \mathcal{J}} \sigma(j) + p_j(\rho(j))$. The objective is to find a feasible schedule, which minimizes the makespan. This problem and prominent variants where the jobs do not need to occupy contiguous machines have been widely studied, see e.g. [32, 25, 26, 23, 20].

This problem is a generalization of Strip Packing as it contains this problem (and Strip Packing with rotations) as a special case: We define the number of machines m as the width of the strip W and for each item $i \in \mathcal{I}$ we introduce one job i with $M_i := \{w(i)\}$ and processing time $p_i(w(i)) = h(i)$ (or introduce one job i with $M_i := \{w(i), h(i)\}$ and processing times $p_i(w(i)) = h(i)$ and $p_i(h(i)) = w(i)$ respectively). Therefore, we cannot hope for a pseudo-polynomial algorithm with a ratio better than 5/4 unless P = NP. We adapt the structure and algorithmic result to find an algorithm with an approximation ratio, which almost matches this bound.

▶ **Theorem 2.** *There is a pseudo-polynomial algorithm for the Contiguous Moldable Parallel Tasks Scheduling problem which finds a $(5/4 + \varepsilon)$-approximation in $(nm)^{\mathcal{O}_\varepsilon(1)}$ operations.*

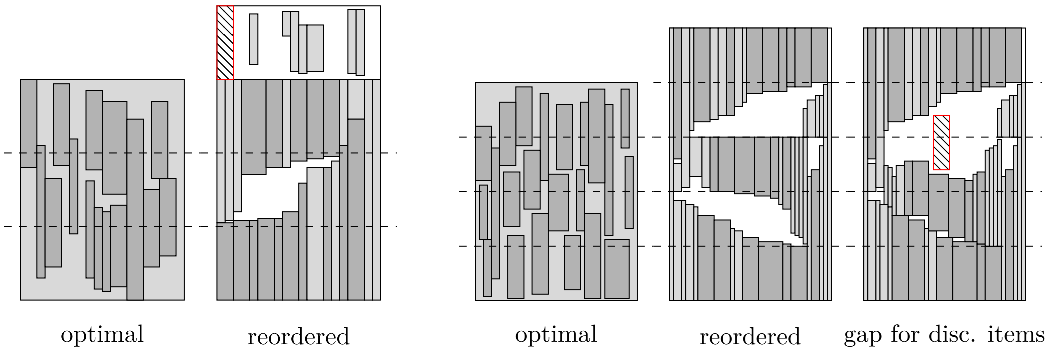
Remark that for the case where for at least one job $j \in \mathcal{J}$ we have that $|M_j| \in \Omega(m)$ the running time of this algorithm is polynomial in the input size. Furthermore, we can hope that in realistic instances the number of machines is bounded by a function in the number of jobs n. If this is the case, the mentioned algorithm is a polynomial time algorithm as well, which further motivates the consideration of pseudo-polynomial time algorithms. As the Contiguous Moldable Parallel Tasks Scheduling contains the Strip Packing with Rotations as a special case, this theorem implies a $(5/4 + \varepsilon)$-approximation with running time $(nW)^{\mathcal{O}_\varepsilon(1)}$ for this problem as well.

Methodology

We follow the general approach mentioned above. More precisely, we analyze optimal solutions and how they can be transformed carefully into well-structured solutions without too much loss in the objective. Knowing that such a transformation is always attainable, the algorithm will iterate the potential structures of the transformed optimal packings and fill the items inside this structure using dynamic and linear programming. The same basic scheme has been used for this and other packing problems before, e.g. [2, 27, 11, 21]. However, finding a suitable transformation to a well structured solution provides a challenge that depends on the problem itself (i.e. a structural result from other packing problems might not be applicable for Strip Packing) and our approach significantly differs from previous ones.

In the approaches seen before, i.e., in [27], [11] and [21], there arises a natural set of critical items, e.g., all items with height larger than $1/3$ OPT in [11] and [21]. The characteristic of this set is that the aspired approximation ratio is exceeded if we place one of these items on top of the optimal packing area. The transformation strategy used in these previous approaches is heavily dependent on the fact that there can be at most two critical items on top of each other. This allows placing all critical items in the optimal packing area while discarding some noncritical items, which are placed on top of the optimal packing later (see Figure 2a). If three critical items can be put on top of each other (which will be the case as soon as a critical item can have a height smaller than OPT/3) the described transformation will not work. To find an algorithm with ratio $4/3 - \varepsilon$, we need to overcome this major obstacle.

To construct a $(5/4 + \varepsilon)$-approximation, we introduce a new technique, in the following called **shifting and reordering**. In contrast to the previous results, our structural result does not guarantee that all critical items are packed inside the optimal packing area. Instead, we shift and reorder the items of an optimal packing such that the critical items with height larger than $1/4$OPT are aligned into three shelves using the area $W \times (5/4 + \varepsilon)$OPT (see Figure 2b).



(a) Previous reordering technique. **(b)** The new shifting and reordering technique.

Figure 2 A comparison of old and new strategies in the simplified case. Dark gray rectangles represent the critical items, while the light gray area represents the other items, which can be sliced vertically during the reordering. The hatched area represents an area where we can place the items that are sliced by the reordering.

A challenge which arises using this new strategy is the fact that by the newly introduced shifting and reordering technique a constant number of the other (non-critical) items will be sliced vertically and thus have to be discarded temporally from the packing. Although this set of discarded items also appears in previous approaches, their handling differs significantly. Since the shifting strategy extends the occupied packing area by the factor $(5/4+\varepsilon)$ with respect to its height, these discarded items cannot be placed on top of the packing as done in previous approaches, see Figure 2. Instead, the discarded items have to be placed carefully into gaps generated by the shifting and reordering step. By a careful analysis of the rearrangement, we prove that each possible structure of a rearranged optimal packing provides suitable gaps to place these items.

In Section 3, we present the central idea to find the improved structural result – the shifting and reordering technique. However, to highlight the basic steps, a simplified problem is considered. In this simplified case just the critical items have to be placed integrally while all other items are allowed be partitioned into vertical slices, which do not have to be placed contiguously.

In general (when the non critical items cannot be placed as non contiguous vertical slices) the slicing of some items will cause problems when trying to place them inside gaps generated by the new strategy, because these gaps might be thin. Hence, we cannot slice items that are too wide in some sense. Nevertheless, we may slice certain narrow items further called sliceable. To overcome this obstacle, we use a lemma from [21], which states the possibility to partition each slightly stretched packing into $\mathcal{O}_\varepsilon(1)$ rectangular areas (without removing any item). This partition provides the property that each critical item is contained in (or intersected by) area(s) exclusively containing critical and sliceable items. Up to three critical items can overlap each of the vertical borders of these areas and these overlapping items may not be shifted horizontally or vertically by our new technique. In the full version, we extend the strategy presented in Section 3 to these areas although it becomes much more involved.

Combining our new techniques to place critical items on three shelves, find suitable gaps for discarded non-critical items and handle the exclusive slicing of narrow items together enables us to prove the structural result from Lemma 3 and in Section 2, we provide a more detailed road-map of its proof. As mentioned above, the algorithm iterates all possible structures defined by the structure result and tries to place all items into this structure using linear and dynamic programming until a suitable structure is found.

The structural result applies to all optimal solutions with the property that they consist of rectangular objects placed into a rectangle that is extendable on one side. Optimal solutions of the three considered problems, i.e., Strip Packing, Strip Packing with rotations and Contiguous Moldable Task Scheduling, all have this property. Thanks to this fact, we were able to obtain algorithms which find $5/4 + \varepsilon$ approximations for each of the three problems by carefully adapting the underlying dynamic program.

2 Structural Result

In this section, we introduce the *Structural Lemma*, which presents the fundamental insight to achieve the approximation ratio $(5/4 + \varepsilon)$. Roughly speaking, the lemma states that each optimal solution can be transformed such that it has a simple structure, see Lemma 3. Due to space limitations, we cannot present the proof here and we refer to the full version. Nevertheless, we provide a high-level overview on the steps of the proof, which consists of the following two basic steps. First the given instance and a corresponding optimal solution is simplified by rounding the sizes of the items (widths and heights) as well as partitioning the set of items into parts, that can be handled almost independently. Afterward, the items in the optimal packing are reordered such that they provide the properties demanded by the lemma.

Given an optimal packing with height OPT for an instance I, we first perform some simplification steps. First, we partition the set of items by defining

- $\mathcal{L} := \{i \in \mathcal{I} \,|\, h(i) > \delta\mathrm{OPT}, w(i) \geq \delta W\}$ as the set of large items,
- $\mathcal{T} := \{i \in \mathcal{I} \,|\, h(i) \geq (1/4 + \varepsilon)\mathrm{OPT}, w(i) < \delta W\}$ as the set of tall items,
- $\mathcal{V} := \{i \in \mathcal{I} \,|\, \delta\mathrm{OPT} \leq h(i) < (1/4 + \varepsilon)\mathrm{OPT}, w(i) \leq \mu W\}$ as the set of vertical items,
- $\mathcal{M}_V := \{i \in \mathcal{I} \,|\, \varepsilon\mathrm{OPT} \leq h(i) < (1/4 + \varepsilon)\mathrm{OPT}, \mu W < w(i) \leq \delta W\}$ as the set of vertical medium items,
- $\mathcal{H} := \{i \in \mathcal{I} \,|\, h(i) \leq \mu\mathrm{OPT}, \delta W \leq w(i)\}$ as the set of horizontal items,
- $\mathcal{S} := \{i \in \mathcal{I} \,|\, h(i) \leq \mu\mathrm{OPT}, w(i) \leq \mu W\}$ as the set of small items and
- $\mathcal{M} := \{i \in \mathcal{I} \,|\, h(i) < \varepsilon\mathrm{OPT}, \mu W < w(i) \leq \delta W\} \cup \{i \in \mathcal{I} \,|\, \mu\mathrm{OPT} < h(i) \leq \delta\mathrm{OPT}\} = \mathcal{I} \setminus (\mathcal{L} \cup \mathcal{T} \cup \mathcal{V} \cup \mathcal{M}_V \cup \mathcal{H} \cup \mathcal{S})$ as the set of medium sized items,

where we chose δ and μ such that the total area of the items $\mathcal{M}_V \cup \mathcal{M}$ is small, resulting in $|\mathcal{M}_V|$ to be in $\mathcal{O}(1/(\varepsilon\delta^2))$. Afterward the heights of the items with height larger than $\delta\mathrm{OPT}$ are rounded such that there are at most $\mathcal{O}(1/(\varepsilon\delta))$ sizes and such that their y-coordinates are positioned on multiples of $\varepsilon\delta\mathrm{OPT}$.

In the next step, we discard the items $\mathcal{S} \cup \mathcal{M}$ from the packing since they can be placed later on, using the NFDH algorithm from [9]. By an adaption of a lemma in [21], we were able to show that the residual packing can be partitioned into $\mathcal{O}_\varepsilon(1)$ rectangular subareas, called boxes, that contain exactly one item from the set $\mathcal{L} \cup \mathcal{M}_V$, only items from the set $\mathcal{H}$, or only items from the set $\mathcal{T} \cup \mathcal{V}$. Furthermore, horizontal items are allowed to overlap horizontal box borders, while vertical and tall items are allowed to overlap vertical box borders. Note that in this partitioning step no item is removed from the packing or changes its position.

Remark that in [21] the items in $\mathcal{M}_V$ were handled the same as the medium sized items $\mathcal{M}$, i.e., they were simply placed on the top of the packing. However, this is not possible in our case since these items can have a height of up to $(1/4 + \varepsilon)\mathrm{OPT}$ and we need the extra height of $(1/4 + \varepsilon)\mathrm{OPT}$ to apply the shifting and reordering. Consequently, we have to think of a new strategy to handle them. Since their number is bounded by $\mathcal{O}_\varepsilon(1)$, it is possible to handle them as if they were large. This different handling of vertical medium items $\mathcal{M}_V$ is, regarding previous algorithms, one of the novelties of this result.

Next, we consider the mentioned partition of the optimal solution into rectangular axis-parallel boxes. The items in $\mathcal{L} \cup \mathcal{M}_V$ and the boxes containing horizontal items need no more attention since for each item in $\mathcal{L} \cup \mathcal{M}_V$ we can guess its position in pseudo-polynomial time and by extending the packing by a factor of $\mathcal{O}(\varepsilon)$ the horizontal items can be placed inside the boxes using a configuration LP building upon the techniques presented in [24].

We innovate the reordering of the items inside the boxes for vertical and tall items $\mathcal{T} \cup \mathcal{V}$ using the new *shifting and reordering* technique (see Section 3). Using this technique, we extend all the boxes with height larger than OPT/2 by only OPT/4, shift and reorder the items inside, and partition their area such that each subarea contains either only tall items of the same height, only vertical items, or no item. Note that the boxes can be overlapped by up to three tall items on each side (left or right). When reordering the items inside the boxes, we cannot move these overlapping items. We refer to the full version for the proof of this alteration with overlapping items.

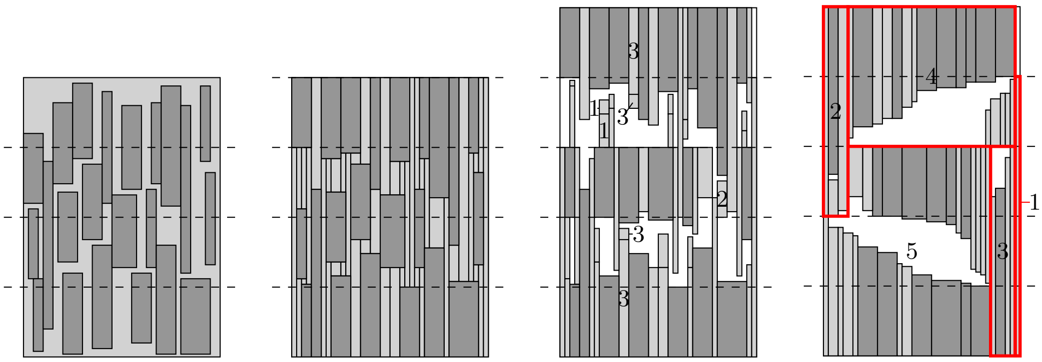
During this reordering step, we slice vertical items vertically. This slicing needs to be fixed since in the aspired Structural Lemma 3 all the items are positioned integral. We prove that by using a configuration LP to place the vertical items, we end up with only $\mathcal{O}_\varepsilon(1)$ items, that have to be placed fractionally. We place these items inside $\mathcal{O}_\varepsilon(1)$ containers of width μW and height OPT/4. An arising challenge is the placement of these containers inside the already extended packing. Other than in the previous attempts (see [27], [11], or [21]), it is not possible to place these extra boxes on top of the packing. By a careful analysis of the area added due to the shifting step, we manage to find a placement of these items inside the rearranged packing. All these considerations together are enough to prove the following structural result:

▶ **Lemma 3** (Structural Lemma). *By extending the packing height to* $(5/4 + 5\varepsilon)\mathrm{OPT}$ *each rounded optimal packing can be rearranged and partitioned into* $\mathcal{O}(1/(\delta^3\varepsilon^5))$ *boxes with the following properties:*

- *There are* $|\mathcal{L}| + |\mathcal{M}_V| = \mathcal{O}(1/(\delta^2\varepsilon))$ *boxes* $\mathcal{B}_\mathcal{L}$ *each containing exactly one item from the set* $\mathcal{L} \cup \mathcal{M}_V$ *and all items from this set are contained in these boxes.*
- *There are at most* $\mathcal{O}(1/(\delta^2\varepsilon))$ *boxes* $\mathcal{B}_\mathcal{H}$ *containing all horizontal items* $\mathcal{H}$ *with* $\mathcal{B}_\mathcal{H} \cap \mathcal{B}_\mathcal{L} = \emptyset$*. The horizontal items can overlap horizontal box borders, but never vertical box borders.*
- *There are at most* $\mathcal{O}(1/(\delta^2\varepsilon^5))$ *boxes* $\mathcal{B}_\mathcal{T}$ *containing tall items, such that each tall item* t *is contained in a box with rounded height* $h(t)$*.*
- *There are at most* $\mathcal{O}(1/(\delta^3\varepsilon^5))$ *boxes* $\mathcal{B}_\mathcal{V}$ *containing vertical items, such that each vertical item* v *is contained in a box with rounded height* $h(v)$*.*
- *There are at most* $\mathcal{O}(1/(\delta^2\varepsilon^5))$ *boxes* $\mathcal{B}_\mathcal{S}$ *for small items, such that the total area of these boxes combined with the total free area inside the horizontal boxes is at least as large as the total area of the small items.*
- *The lower and top border of each box is positioned at a multiple of* $\varepsilon\delta\mathrm{OPT}$*.*

3 Introducing the Shifting and Reordering Technique

To demonstrate the central new idea which leads to the improved structural result – the shifting and reordering technique – we consider the following simplified case. We have to pack items with a tall height integrally, while we are allowed to slice all other items vertically. We can assume that the packing, which we consider here, is the packing inside a box for $\mathcal{T}$ and $\mathcal{V}$ for the case that no tall item overlaps the box borders. Remember, in the general case, there can be such items and hence the reordering gets a little bit more complicated as in this simplified case. We will demonstrate that, in this simplified scenario, it is possible to rearrange the items such that there are a constant number of rectangular subareas, which contain only tall items with the same height.



(a) An optimal packing. **(b)** The packing after the first shift (Step 1 and 2). **(c)** The packing after the second shift (Step 3 and 4). **(d)** The final reordered packing after Step 5.

Figure 3 States of the item rearrangement. Dark rectangles represent tall items while light gray areas might contain non-tall sliced items.

Let a packing with height H be given. We define tall items as the items which have a height larger than $1/4H$. Further, assume that there is an arithmetic grid with $N+1$ horizontal grid lines with distance H/N such that each tall item starts and ends at the grid lines. For now, we can think of this grid as the integral grid with $H+1$ grid lines. Later, we can reduce the grid lines by rounding the heights of the items. We are interested in a fractional packing of the non-tall items. Therefore, we replace each non-tall item i by exactly $w(i)$ items with height $h(i)$ and width 1. This step is called slicing. We define a box as a rectangular subarea of the packing area.

▶ **Lemma 4.** *By adding at most $1/4H$ to the packing height and slicing non-tall items, we can rearrange the items such that we generate at most $3/2N$ containers which contain tall items with the same height only, and at most $9/4N+1$ container for sliced items.*

Proof. In this proof, we will present a rearrangement strategy which provides the desired properties. This strategy consists of two shifting steps and one reordering step. In the shifting steps, we shift items in the vertical direction, while in the reordering step we change the item positions horizontally. In the first shifting step, we ensure that tall items intersecting the horizontal lines $1/4H$ or $3/4H$ will touch the bottom or the top of the packing area, respectively. In the second shift, we ensure that tall items not intersecting these lines have a common upper border as well. Last, we reorder the items such that tall items with the same height are positioned next to each other if they have a common upper or lower border.

Step 1: First Shift. Note that there is no tall item completely below $1/4H$ or completely above $3/4H$ since each tall item has a height larger than $1/4H$. We shift each tall item t intersecting the horizontal line $1/4H$ down, such that its bottom border touches the bottom of the strip. The sliced items below t are shifted up exactly $h(t)$, such that they are now positioned above t. In the same way, we shift each tall item intersecting the horizontal line at $3/4H$ but not the horizontal line at $1/4H$ such that its upper border is positioned at H and shift the sliced items down accordingly, see Figure 3b.

Step 2: Introducing Pseudo Items. At this point, we introduce a set of containers for the sliced items, which we call pseudo items, see Figure 3b. We draw vertical lines at each left or right border of a tall item and erase these lines on any tall item. Each area between two consecutive lines which is bounded on top and bottom by a tall item or the packing area and contains sliced items represents a new item called pseudo item. Note that no sliced item is intersecting any box border since they are positioned on integral widths only. Furthermore, when we shift a pseudo item, we shift all sliced items included in this container as well.

When constructing the pseudo items, we consider one special case. Consider a tall item t with height larger than $3/4H$. There can be no tall item positioned above or below t, and t was shifted down. For these items, we introduce one pseudo item of height H and width $w(t)$ containing t and all sliced items above. Note that each pseudo item has a height, which is a multiple of H/N. Furthermore, note that each tall or pseudo item touching the top or the bottom border of the packing area has a height larger than $1/4H$.

Step 3: Second Shift. Next, we do a second shifting step consisting of three sub-steps. First, we shift each tall or pseudo item intersected by the horizontal line at $3/4H$ but not the horizontal line at $1/4H$ exactly $1/4H$ upwards. Second, we shift each pseudo item positioned between the horizontal lines at $1/2H$ and $3/4H$, such that their lower border is positioned at the horizontal line $3/4H$. Last, we shift each tall or pseudo item intersected by the horizontal line at $1/2H$ but not the horizontal line at $1/4H$ or $3/4H$ such that its upper border is positioned at the horizontal line $3/4H$. After this shifting, no item overlaps another item since we have shifted the items intersecting the line at $3/4H$ exactly $1/4H$, while each item below is shifted at most $1/4H$.

Step 4: Fusing Pseudo Items. After the second shift, we will fuse and shift some pseudo items. We want to establish the property that each tall and pseudo item has one border (upper or lower), which touches one of the horizontal lines at 0, $3/4H$, or $5/4H$. At the moment there can be some pseudo items between the horizontal lines $1/4H$ and $1/2H$, which do not touch one of the three lines. In the following, we study the three cases where those pseudo items can occur. These items do only exist if there is a tall item touching the bottom of the packing and another tall item above this item with a lower border at or below $1/2H$ before the second shifting step. Consider two consecutive vertical lines we had drawn to generate the pseudo items. If a tall item overlaps the vertical strip between these lines, its right and left borders either lie on the strips borders or outside of the strip.

Case 1: In the first considered case there are three tall items, t_1, t_2, and t_3 from bottom to top, which overlap the strip. In this scenario t_1 must have its lower border at 0, t_2 its upper border at $3/4H$, and t_3 its upper border at $5/4H$. As a consequence, there are at most two pseudo items: One is positioned between t_1 and t_2, and the other between t_2 and t_3. We will stack them, such that the lower border of the stack is positioned at $3/4H$ and prove that this is possible without overlapping t_3. The total height of both pseudo items is $H - h(t_1) - h(t_2) - h(t_3)$. The total area not occupied by tall items is $H - h(t_1) - h(t_2) - h(t_3) + 1/4H$ since we have added $1/4H$ to the packing height. The distance between t_1 and t_2 is at most $1/4H$ since t_1's lower border is at 0 and t_2's upper border is at $3/4H$ and both have a height larger than $1/4H$. Therefore, the distance between t_2 and t_3 is at least $H - h(t_1) - h(t_2) - h(t_3)$, see Figure 3c at the items marked with 1.

Case 2: Now consider the case where there is one tall item t_1 touching the bottom, and one tall item t_2 with height at least $1/2H$ touching $5/4H$. Obviously, t_2 has a height of at most $3/4H$. Furthermore, there is at most one pseudo item, and it has to be positioned between $1/4H$ and $1/2H$. We shift this pseudo item up until its bottom touches $1/2H$, see Figure 3c at the item marked with 2. This is possible without constructing any overlap, because the distance between t_1 and the horizontal line $1/2H$ is less than $1/4H$ and, therefore, the distance between the line $1/2H$ and the lower border of the tall item is larger than the height of the pseudo item.

After this step, we consider each tall item t with height larger than $1/2H$ touching $5/4H$. We generate a new pseudo item with width $w(t)$ and height $3/4H$, with upper border at $5/4H$ and lower border at $1/2H$, containing all pseudo items below t touching $1/2H$ with their lower border.

Case 3: In the last case we consider, there are two tall items t_1 and t_2 and two pseudo items; one of the items t_1 and t_2 touches the top of the packing or the bottom, while the other ends at $3/4H$. Hence, the distance between the tall items has to be smaller than $1/4H$. Furthermore, one of the pseudo items has to touch the top or the bottom of the packing while the other is positioned between t_1 and t_2. Since the distance between t_1 and t_2 is less than $1/4H$, one of the distances between the packing border and the lower border of t_1 or the upper border of t_2 is at least $H - h(t_1) - h(t_2)$. Therefore, we can fuse both pseudo items by shifting the one between t_1 and t_2 such that it is positioned above or below the other one, see Figure 3c at the items marked with 3.

▶ **Observation 1.** *After the shifting and fusing, each tall or pseudo item touches one of the horizontal lines at* 0*,* $3/4H$ *or* $5/4H$*.*

Step 5: Reordering the Items. In the last part of the rearrangement, we reorder the items horizontally to place pseudo and tall items with the same height next to each other. In this reordering step, we create five areas each reserved for certain items. To do so, we take vertical slices of the packing and move them to the left or the right in the strip. A vertical slice is an area of the packing with width one and height of the considered packing area, i.e. $5/4H$ in this case. While rearranging these slices, it will never happen that two items overlap. However, it can happen, that some of the tall items are placed fractionally afterward. This will be fixed in later steps.

Area 1: First, we will extract all vertical slices containing (pseudo) items with height H. Then, shifting all the remaining vertical slices to the left as much as possible, we create one box for pseudo items of height H at the right, see Figure 3d at Area 1. In this area, we sort the pseudo items such that the pseudo items containing tall items with the same height are placed next to each other. In this step, we did not place any tall item fractionally.

Area 2: Afterward, we take each vertical slice containing a (pseudo) item with height at least $1/2H$ touching the horizontal line at $5/4H$. Remember, there might be pseudo items containing a tall item t with a height between $1/2H$ and $3/4H$. We shift these slices to the left of the packing and sort them in descending order of the tall items height $h(t)$, see Figure 3d at Area 2. Afterward, we sort the pseudo items below these tall items, which are touching $1/2H$ with their bottom in ascending order of their heights, which is possible without generating any overlapping. In this step, it can happen that we slice tall items which touch the bottom of the strip. We will fix this slicing in one of the following steps, when we consider Area 5.

Area 3: Next, we look at vertical slices containing (pseudo) items t with height at least $1/2H$ touching the bottom of the strip. We shift them to the right until they touch the Area 1 and sort these slices in ascending order of the heights $h(t)$, see Figure 3d at Area 3.

Note that there are no pseudo or tall items that have their upper border positioned at $^3/_4H$ in these slices. In this step, it can happen that we slice tall items touching the top of the packing. This will be fixed in the next step.

Area 4: Look at the area above $^3/_4H$ and left of Area 2 but right of Area 1, see Figure 3d at Area 4. In this area no item overlaps the horizontal line $^3/_4H$. Therefore, we have a rectangular area where each item either touches its bottom or its top and no item is intersected by the area's borders. In [27] it was shown that, in this case, we can sort the items touching the line $^3/_4H$ in ascending order of their height and the items touching $^5/_4H$ in descending order of heights and no item will overlap another item. Now all items with the same height are placed next to each other, and thus we fixed the slicing of tall items.

Area 5: In the last step, we will reorder the remaining items. Namely the items touching the bottom of the strip left of Area 3 and the items touching the horizontal line at $^3/_4H$ with their top between Area 2 and Area 3. The items touching the bottom are sorted in descending order of their height and the items touching the horizontal line at $^3/_4H$ are sorted in ascending order regarding their heights.

▷ Claim. After the reordering of Area 5 no item overlaps another.

Proof. First, note that the items touching $^5/_4H$ have a height of at most $^3/_4H$. Therefore, no item touching the bottom having height at most $^1/_2H$ can overlap with these items. Furthermore, note that before the reordering no item was overlapping another. Let us assume there are two items b and t, which overlap at a point (x, y) after this reordering. Then all items left of x touching $^3/_4H$ have their lower border below y, while all items touching the bottom left of x have their upper border above y. Therefore, at every point left and right of (x, y) in the Area 5 there is an item overlapping it. Hence, the total width of items overlapping the horizontal line y is larger than the width of the Area 5. Therefore in the original ordering, there would have been items overlapping each other already since we did not add any items – a contradiction. As a consequence in this new ordering, no two items overlap, which concludes the proof of the claim. ◁

Analyzing the Number of Constructed Boxes. In the last part of this proof, we analyze how many boxes we have created for tall and sliced items.

▷ Claim. After the described reordering there are at most $\frac{3}{2}N$ boxes for tall items all containing items of only one height.

Proof. Each tall item with height at least $^3/_4H$ touches the bottom and we create at most one box in Area 1 for each height. Therefore, we create at most $N/4$ boxes for these items. Each tall item that has a height between $^1/_2H$ and $^3/_4H$ touches either the bottom or the horizontal line $^5/_4H$. On each of these lines, we create at most one box for items with the same height. Therefore, we create at most $2N/4$ boxes for these items. Last, each tall item with height larger than $^1/_4H$ but smaller than $^1/_2H$ either touches the bottom of the packing, the horizontal line $^3/_4H$ or the horizontal line $^5/_4H$. At each of these lines, we create at most one box for each height. Therefore, we create at most $3N/4$ of these boxes. In total, we create at most $\frac{3}{2}N$ boxes for tall items. ◁

▷ Claim 5. After the described reordering there are at most $\frac{9}{4}N + 1$ boxes containing sliced items.

Proof. Let us consider the number of boxes for sliced items. Each pseudo item's height is a multiple of H/N. Therefore, we have at most N different sizes for pseudo items. There are at most 4 boxes for each height less than $1/4H$. One is touching H with its top border in Area 1, one is touching $3/4H$ with its bottom border in Area 4, one is touching $3/4H$ with its top border in Area 5, and one is touching $1/2H$ with its bottom border in Area 2. Furthermore, there are at most 3 boxes for each size between $1/4H$ and $1/2H$. One is touching $5/4H$ with its top border in Area 4, one is touching $3/4H$ with its top border in Area 5, and one is touching 0 with its bottom border in Area 5. Additionally, there are at most 2 boxes for each pseudo item size larger than $1/2H$. One is touching $5/4H$ with its top border in Area 2, the other is touching 0 with its bottom border in Area 3. Last there is only one pseudo item with height larger than $3/4H$ in Area 1. It has height H. Since the grid is arithmetically defined, we have at most $N/4$ sizes with height at most $1/4H$, $N/4$ sizes between $1/4H$ and $1/2H$ and at most $1/4N$ sizes between $1/2H$ and $3/4H$. Therefore, we create at most $4 \cdot 1/4N + 3 \cdot 1/4N + 2 \cdot 1/4N + 1 = \frac{9}{4}N + 1$ boxes for sliced items. ◁

Since the number of boxes for tall and sliced items is as small as claimed, this concludes the proof of the Lemma 4. ◀

In this section, we have proven that in this simplified case it is possible to reorder the items such that they have a nice structure. Nevertheless, when considering the mentioned partition into rectangular subareas (called boxes) from [21] we encounter some obstacles. In each box B containing tall and vertical items there can be up to three tall items overlapping its left and right border. Especially critical to apply the above described shifting and reordering technique are the items overlapping the box at the height $h(B) - H/4$, because these items cannot be moved. The reason why we cannot move these objects is the impossibility of judging their intertwining with other objects within the respective other box(es) in which they are contained. Therefore, since the reordering technique requires them to be shifted up, a first step is to discard these items from the respective boxes by partitioning them into smaller boxes at the left and right borders of these items. However, we cannot get rid of the items overlapping the box below this horizontal line of $h(B) - H/4$, and handling these items without moving them becomes quite technical. A detailed analysis how to reorder the items inside a box if tall items overlap the borders can be found in the full version of this paper.

References

1 Anna Adamaszek, Tomasz Kociumaka, Marcin Pilipczuk, and Michal Pilipczuk. Hardness of Approximation for Strip Packing. *TOCT*, 9(3):14:1–14:7, 2017. doi:10.1145/3092026.

2 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, San Diego, CA, USA, January 4-6, 2015*, pages 1491–1505, 2015. doi:10.1137/1.9781611973730.98.

3 Brenda S. Baker, Donna J. Brown, and Howard P. Katseff. A 5/4 Algorithm for Two-Dimensional Packing. *Journal of Algorithms*, 2(4):348–368, 1981. doi:10.1016/0196-6774(81)90034-1.

4 Brenda S. Baker, Edward G. Coffman Jr., and Ronald L. Rivest. Orthogonal Packings in Two Dimensions. *SIAM Journal on Computing*, 9(4):846–855, 1980. doi:10.1137/0209064.

5 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, Portland, Oregon, USA, January 5-7, 2014*, pages 13–25, 2014. doi:10.1137/1.9781611973402.2.

6 Marin Bougeret, Pierre-François Dutot, Klaus Jansen, Christina Robenek, and Denis Trystram. Approximation Algorithms for Multiple Strip Packing and Scheduling Parallel Jobs in Platforms. *Discrete Mathematics, Algorithms and Applications*, 3(4):553–586, 2011. doi:10.1142/S1793830911001413.

7 Lin Chen, Klaus Jansen, and Guochuan Zhang. On the optimality of approximation schemes for the classical scheduling problem. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 657–668, 2014. doi:10.1137/1.9781611973402.50.

8 Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 2017. doi:10.1016/j.cosrev.2016.12.001.

9 Edward G. Coffman Jr., Michael R. Garey, David S. Johnson, and Robert Endre Tarjan. Performance Bounds for Level-Oriented Two-Dimensional Packing Algorithms. *SIAM Journal on Computing*, 9(4):808–826, 1980. doi:10.1137/0209062.

10 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, Berkeley, CA, USA, October 15-17, 2017*, pages 260–271, 2017. doi:10.1109/FOCS.2017.32.

11 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)*, pages 9:1–9:14, 2016. doi:10.4230/LIPIcs.FSTTCS.2016.9.

12 Igal Golan. Performance Bounds for Orthogonal Oriented Two-Dimensional Packing Algorithms. *SIAM Journal on Computing*, 10(3):571–582, 1981. doi:10.1137/0210042.

13 Fabrizio Grandoni, Tobias Mömke, Andreas Wiese, and Hang Zhou. To Augment or Not to Augment: Solving Unsplittable Flow on a Path by Creating Slack. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2411–2422, 2017. doi:10.1137/1.9781611974782.159.

14 Fabrizio Grandoni, Tobias Mömke, Andreas Wiese, and Hang Zhou. A $(5/3 + \epsilon)$-approximation for unsplittable flow on a path: placing small tasks into boxes. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 607–619, 2018. doi:10.1145/3188745.3188894.

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

16 Rolf Harren and Rob van Stee. Improved Absolute Approximation Ratios for Two-Dimensional Packing Problems. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques,*, volume 5687 of *Lecture Notes in Computer Science*, pages 177–189. Springer, 2009. doi:10.1007/978-3-642-03685-9_14.

17 Sören Henning, Klaus Jansen, Malin Rau, and Lars Schmarje. Complexity and Inapproximability Results for Parallel Task Scheduling and Strip Packing. In *Computer Science - Theory and Applications - 13th International Computer Science Symposium in Russia, CSR 2018, Moscow, Russia, June 6-10, 2018, Proceedings*, pages 169–180, 2018. doi:10.1007/978-3-319-90530-3_15.

18 Sandy Heydrich and Andreas Wiese. Faster approximation schemes for the two-dimensional knapsack problem. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 79–98, 2017. doi:10.1137/1.9781611974782.6.

19 Klaus Jansen, Kim-Manuel Klein, and José Verschae. Closing the Gap for Makespan Scheduling via Sparsification Techniques. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 72:1–72:13, 2016. doi:10.4230/LIPIcs.ICALP.2016.72.

20 Klaus Jansen and Felix Land. Scheduling Monotone Moldable Jobs in Linear Time. In *2018 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2018, Vancouver, BC, Canada, May 21-25, 2018*, pages 172–181, 2018. doi:10.1109/IPDPS.2018.00027.

21 Klaus Jansen and Malin Rau. Improved approximation for two dimensional strip packing with polynomial bounded width. In *WALCOM: Algorithms and Computation*, volume 10167 of LNCS, pages 409–420, 2017. doi:10.1007/978-3-319-53925-6_32.

22 Klaus Jansen and Roberto Solis-Oba. Rectangle packing with one-dimensional resource augmentation. *Discrete Optimization*, 6(3):310–323, 2009. doi:10.1016/j.disopt.2009.04.001.

23 Klaus Jansen and Ralf Thöle. Approximation Algorithms for Scheduling Parallel Jobs. *SIAM Journal on Computing*, 39(8):3571–3615, 2010. doi:10.1137/080736491.

24 Claire Kenyon and Eric Rémila. A Near-Optimal Solution to a Two-Dimensional Cutting Stock Problem. *Mathematics of Operations Research*, 25(4):645–656, 2000. doi:10.1287/moor.25.4.645.12118.

25 Walter Ludwig and Prasoon Tiwari. Scheduling Malleable and Nonmalleable Parallel Tasks. In *5th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 167–176, 1994.

26 Gregory Mounie, Christophe Rapine, and Denis Trystram. A 3/2-Approximation Algorithm for Scheduling Independent Monotonic Malleable Tasks. *SIAM J. Comput.*, 37(2):401–412, 2007. doi:10.1137/S0097539701385995.

27 Giorgi Nadiradze and Andreas Wiese. On approximating strip packing with a better ratio than 3/2. In *27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1491–1510, 2016. doi:10.1137/1.9781611974331.ch102.

28 Ingo Schiermeyer. Reverse-Fit: A 2-Optimal Algorithm for Packing Rectangles. In *2nd Annual European Symposium on Algorithms (ESA) - Algorithms*, pages 290–299, 1994. doi:10.1007/BFb0049416.

29 Daniel Dominic Sleator. A 2.5 Times Optimal Algorithm for Packing in Two Dimensions. *Information Processing Letters*, 10(1):37–40, 1980. doi:10.1016/0020-0190(80)90121-0.

30 A. Steinberg. A Strip-Packing Algorithm with Absolute Performance Bound 2. *SIAM Journal on Computing*, 26(2):401–409, 1997. doi:10.1137/S0097539793255801.

31 Maxim Sviridenko. A note on the Kenyon-Remila strip-packing algorithm. *Information Processing Letters*, 112(1-2):10–12, 2012. doi:10.1016/j.ipl.2011.10.003.

32 John Turek, Joel L. Wolf, and Philip S. Yu. Approximate Algorithms Scheduling Parallelizable Tasks. In *4th annual ACM symposium on Parallel algorithms and architectures (SPAA)*, pages 323–332, 1992. doi:10.1145/140901.141909.

33 Andreas Wiese. A (1+epsilon)-Approximation for Unsplittable Flow on a Path in Fixed-Parameter Running Time. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 67:1–67:13, 2017. doi:10.4230/LIPIcs.ICALP.2017.67.