Approximation Algorithms for Geometric Packing Problems

Dissertation

zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften
(Dr.-Ing.)
der Technischen Fakultät
der Christian-Albrechts-Universität zu Kiel

Dipl.-Inf. Lars Dennis Prädel

Kiel

2012

1. Gutachter:	Prof. Dr. Klaus Jansen
2. Gutachter:	Prof. Dr. Susanne Albers
3. Gutachter:	Dr. Nikhil Bansal

Datum der mündlichen Prüfung:	21. Dezember 2012
Zum Druck genehmigt:	21. Dezember 2012

ACKNOWLEDGEMENTS

I would like to thank my advisor Klaus Jansen for the support, encouragement and motivation during my studies. I am also thankful to my colleagues and friends Ute Iaquinto and Christina Robenek for many helpful discussions in various issues. Thanks to my coauthors Florian Diedrich, Rolf Harren, Ola Svensson, Ulrich Schwarz and Rob van Stee. I would also like to thank Manja Kürschner for proof-reading and even more for several talks during our morning running. I am also grateful to my parents Maren and Dietmar Prädel for supporting me during my studies and to my closest friends Sönke Thomsen, Jochen Webert and Laura Paduch for always being there. Finally, I would like to thank Laura Garbers for her patient and everlasting support.

ZUSAMMENFASSUNG

In dieser Arbeit stellen wir approximative Algorithmen für zwei-dimensionale, geometrische Packungsprobleme vor. Hierbei haben wir eine Menge von Rechtecken gegeben, die in einem oder mehreren bestimmten Zielbereichen angeordnet werden sollen. Solche Packungsprobleme finden in mehreren industriellen Bereichen Anwendung, so zum Beispiel bei der Anordnung von Schaltelementen auf einem Computerchip oder bei Zuschnittproblemen.

Wir betrachten drei Probleme im Detail. Das sogenannte zwei-dimensionale Strip Packing Problem hat als Eingabe eine Menge von Rechtecken und einen Streifen der Breite 1 und unendlicher Höhe. Das Ziel ist es eine Anordnung der Rechtecke in diesem Streifen zu finden, so dass die Packungshöhe minimiert wird. Dabei müssen die Rechtecke achsenparallel angeordnet werden und dürfen sich nicht überschneiden. Auch eine Rotation der Rechtecke ist nicht erlaubt. Wir präsentieren für dieses Problem einen approximativen Algorithmus mit einer absoluten Güte von $5/3+\varepsilon$, für ein beliebiges $\varepsilon > 0$.

Das zweite Problem, das wir untersuchen, ist das zwei-dimensionale Bin Packing Problem. Bei diesem Problem haben wir ebenfalls eine Menge von Rechtecken gegeben. Das Ziel ist es alle Rechtecke in die kleinstmögliche Anzahl von Quadraten, auch Bins genannt, mit einheitlicher Seitenlänge zu packen. Auch hier ist gefordert, dass die Rechtecke sich nicht überschneiden und achsenparallel angeordnet werden. Wir betrachten hier zwei Varianten des Problems. In der ersten Variante sind Rotationen um 90° erlaubt, d.h. wir können die Breite mit der Höhe eines Rechteckes vertauschen. Bei der anderen Variante sind Rotationen der Rechtecke nicht erlaubt. Für beide Varianten dieses Problems geben wir einen approximativen Algorithmus an mit einer asymptotische Güte von $3/2+\varepsilon$, wobei $\varepsilon > 0$ eine beliebige Zahl ist.

Bei dem dritten Problem ist eine Menge von Rechtecken mit einheitlicher Höhe und beliebiger Breite und ein Streifen einer gegebenen ganzzahligen Höhe und unendlicher Breite gegeben. Das Ziel ist es, eine achsenparallele, sich nicht überschneidende Packung der Rechtecke zu finden, so dass die maximale Packungsbreite minimiert wird. Eine Rotation der Rechtecke ist bei diesem Problem nicht erlaubt. Dieses Problem ist auch als Scheduling Problem bekannt, wobei der Streifen von Maschinen repräsentiert wird und die Rechtecke von Aufträgen. Die Problemdefinition ist wie folgt: Gegeben ist eine Menge von Aufträgen,

die eine bestimmte Ausführungszeit haben und eine Menge von identischen Maschinen. Das Ziel ist es, die Aufträge auf die Maschinen zu verteilen, so dass die gesamte Ausführungszeit minimiert wird. Wir betrachten eine Variante dieses Problems, bei dem die ersten Aufträge bereits auf bestimmten Maschinen zu bestimmten Zeiten vorplatziert sind. Wir geben für dieses Problem einen approximativen Algorithmus mit einer absoluten Güte von $3/2$ an.

ABSTRACT

In this thesis we present approximation algorithms for two-dimensional, geometric packing problems. We have given a set of rectangles that have to be placed in one or several predetermined target regions. Such packing problems can be found in several branches of industry, for example when placing logic elements on a chip, or when cutting stock. We consider three problems in detail. First, the so-called two-dimensional strip packing problem consists of a set of rectangles and a strip of width 1 and infinite height. The objective is to find an axis-parallel and non-overlapping arrangement of the rectangles in this strip in order to minimize the total packing height. Furthermore, it is not allowed to rotate the rectangles. For any $\varepsilon > 0$, we present an approximation algorithm with an absolute approximation ratio of $5/3 + \varepsilon$ for this problem.

The second problem that we study is the two-dimensional bin packing problem. For this problem a set of rectangles is given as input again. The objective is to find an axis-parallel and non-overlapping packing of all rectangles into the minimum number of unit-sized squares, which are also called bins. We consider two versions of this problem. In the first version, we are allowed to rotate the rectangles by 90°, i.e. to exchange the widths and the heights of the rectangles. In the second version it is not allowed to rotate the rectangles at all. For both versions, our result is an approximation algorithm with an asymptotic approximation ratio of $3/2 + \varepsilon$ for an arbitrary value $\varepsilon > 0$.

For the third problem, we have given a set of rectangles of heights 1 and arbitrary widths and a strip of a given integral height and infinite width. The objective is to find an axis-parallel and non-overlapping packing of the rectangles into the strip so that the maximum packing width is minimized. In this setting, it is not allowed to rotate the rectangles. This problem is also known as scheduling problem, with machines representing the strip and jobs representing the rectangles. The definition is as follows: Given a set of jobs with certain processing times and a set of identical machines. The objective is to schedule the jobs on the machines in order to minimize the makespan, i.e. the total length of the schedule. We consider a version of this problem in which the first jobs are already assigned to machines and starting times. We give an approximation algorithm with an absolute approximation ratio of $3/2$ for this problem.

CONTENTS

1 INTRODUCTION

Two-dimensional packing problems arise in several branches of industry. For example in the VLSI chip design, many logic elements have to be placed on one chip. We can treat each logic element as one rectangle and the chip as a square and the problem is to place as many rectangles as possible into this square, or to optimally exploit the space in this square. Another version of a two-dimensional packing problem occurs in stock cutting when we want to cut some items out of some sheets of raw material, for example of metal or wood. We want to minimize the total wasted material. Sometimes, it is not useful to rotate the items that have to be cut out, since we have for example to take care of the grain of the wood. In print layout we have to distribute advertisements within one newspaper. The objective might either be to fit all advertisements into the minimum number of pages or into a fixed number of pages while securing the maximum profit for the newspaper company. If you want to ship packages of the same depth, the three-dimensional packing problem is reduced to a two-dimensional problem. The aim of packing may vary in the logistics since you might sometimes want to minimize the number of containers needed and on other occasions prefer to minimize the size of one container that holds all packages. With regard to scheduling further packing problems occur. Scheduling on identical machines can be modelled as a packing problem while each job correspond to a rectangle of height 1 and the machines correspond to a strip. This strip has an infinite width and a height that correspond to the given number of machines. The objective is to place all rectangles into that strip so that the total packing width is minimized, i.e. the total processing time is minimized. Moreover, we can think of jobs that need several consecutive processors so that the height of the rectangle corresponds to the number of processors that are needed for this job. These scenarios describe two-dimensional packing problems in which we want to place some rectangles into some predetermined regions so that the rectangles are packed in a non-overlapping and axis-parallel way. As the scenarios above occur in several industries, it is important to develop procedures that improve the packing of rectangles. However, many of these problems are strongly NP-hard and there is no hope in finding an efficient algorithm to solve the packing problem optimally, unless P = NP. Therefore, we focus on algorithms that have an improved running time at the expense of an optimal packing.

1.1 Approximation Algorithms

An optimization problem consists of a set of instances, a set of feasible solutions for each instance and a non-negative value for each feasible solution. There are two kinds of optimization problems called maximization and minimization problem. When facing a minimization problem, the objective is to find a feasible solution that has the smallest possible value for each instance. In the case of a maximization problem, the objective is to find a feasible solution that has the largest possible value for each instance. Let Π be an optimization problem and let A be an algorithm for this problem, i.e. A computes a feasible solution for each instance of Π. Furthermore, for each instance I of Π, we denote the optimal value of all feasible solutions of I with $\mathrm{OPT}(I)$ and the value of the feasible solution that is computed by A with $A(I)$. We call A an approximation algorithm for Π with an approximation ratio $\alpha > 0$ if A computes a feasible solution for each instance I in polynomial time of the input length and if there is a constant $\beta \geq 0$ so that

$$A(I) \leq \alpha \mathrm{OPT}(I) + \beta$$

holds in case of a minimization problem and

$$A(I) \geq \alpha \mathrm{OPT}(I) - \beta$$

holds in case of a maximization problem.

The approximation ratio is called absolute approximation ratio, if $\beta = 0$. In all other cases, it is called asymptotic approximation ratio. A family of approximation algorithms $(A_\varepsilon)_{\varepsilon>0}$ is called a polynomial time approximation scheme (PTAS) if A_ε has an absolute approximation ratio of $1+\varepsilon$ in case of a minimization problem and $1-\varepsilon$ in case of a maximization problem. This family is called a fully polynomial time approximation scheme (FPTAS) if it is a PTAS and the running time of A_ε is additionally bounded in a polynomial in $1/\varepsilon$. An efficient polynomial time approximation scheme (EPTAS), is a PTAS whose running time is bounded by $f(1/\varepsilon) \cdot p(n)$ for a function f, a polynomial p and for each instance of length n. Asymptotic (fully) polynomial time approximation schemes (APTAS, AFPTAS) are similarly defined in terms of an asymptotic approximation ratio. For a more detailed introduction concerning the theory of complexity and approximation algorithms, we recommend a reading of the book by Vazirani [53] and by Jansen & Margraf [32].

1.2 Outline of this Thesis

This thesis consists of three self-sufficient chapters. In Chapter 2, we present an approximation algorithm for the two-dimensional strip packing problem, in Chapter 3 we display an approximation algorithm for the two-dimensional bin packing problem and in the Chapter 4 we propose an approximation algorithm for scheduling with fixed jobs.

1.2.1 Two-Dimensional Strip Packing

In the two dimensional strip packing problem, a list of rectangles is given. We are looking for an orthogonal and non-overlapping packing of these rectangles into a strip of unit width and infinite height in order to minimize the total packing height. In this setting it is not allowed to rotate the rectangles. In this chapter, a short introduction will give some background information concerning this problem. Afterwards, our approximation algorithm with an absolute approximation ratio of $5/3+\varepsilon$ for an arbitrary $\varepsilon > 0$ will be presented. This result is published in [23, 24].

1.2.2 Two-Dimensional Bin Packing

In the two-dimensional bin packing problem, a list of rectangles as well as an infinite set of squares of unit side lengths are given. These squares will be called bins in this chapter. The objective is to find a non-overlapping, axis-parallel packing that is apt to put all rectangles into bins in order to minimize the total number of bins used. We consider two versions: one that allows 90° rotations and one that does not. In the Chapter 3, we propose an approximation algorithm with an asymptotic $3/2+\varepsilon$ approximation for an arbitrary $\varepsilon > 0$ with an additive constant of 69 in the version that does not allow rotation and an additive constant of 39 in the version that allows 90° rotation. The main idea of this algorithm is to prove that each solution can be modified so that we are able to round up the rectangles. The steps of modification are explained in Section 3.2 subsequent to the introduction which summarizes the results that have already been published. This modified solution can be processed with the help of our algorithm that is explained in Section 3.3. This result is published in [33].

1.2.3 Scheduling with Fixed Jobs

In this problem, we have given a set of rectangles of heights 1 and arbitrary widths and a strip of a given integral height and infinite width. The objective is to find an axis-parallel and non-overlapping packing of the rectangles into the strip so that the maximum packing width

is minimized. In this setting, it is not allowed to rotate the rectangles. This problem is also known as scheduling problem, with machines representing the strip and jobs representing the rectangles. The definition is as follows: Given a set of jobs with certain processing times and a set of identical machines. The objective is to schedule the jobs on the machines in order to minimize the makespan, i.e. the total length of the schedule. We consider a version of this problem in which the first jobs are already assigned to machines and starting times. In Chapter 4, a short introduction containing old and new results will be followed by our presentation of an approximation algorithm with an absolute $3/2$ approximation ratio for the problem with fixed jobs. This result can also be adopted to a version of scheduling with non-availability. This problem is similar to scheduling with fixed jobs, apart from the fact that the fixed jobs do not count in the total completion time and that we require that a constant fraction of the machines is always available, i.e. without fixed jobs. This result is published in [16, 35].

2 A $(5/3+\varepsilon)$-Approximation for Strip Packing

2.1 Introduction

Two-dimensional packing problems are classical in combinatorial optimization and continue to receive a lot of research interest [4, 5, 6, 25, 34, 36, 40]. One of the most important ones is the strip packing problem also known as the cutting stock problem: given a set of rectangles $I = \{r_1, \dots, r_n\}$ of specified widths w_i and heights h_i, the problem is to find a feasible packing for I (i.e. an orthogonal arrangement where rectangles do not overlap and are not rotated) into a strip of width 1 and minimum height.

The strip packing problem has many practical applications in manufacturing, logistics, and computer science. In many manufacturing settings rectangular pieces need to be cut out of some sheet of raw material, while minimizing the waste. Scheduling independent tasks on a group of processors, each requiring a certain number of contiguous processors or memory allocation during a certain length of time, can also be modeled as a strip packing problem.

Results. The Bottom-Left algorithm by Baker et al. [2] has asymptotic approximation ratio equal to 3 when the rectangles are ordered by decreasing widths. Coffman et al. [13] provided the first algorithms with proven approximation ratios of 3 and 2.7, respectively. The approximation algorithm presented by Sleator [50] generates a packing of height $2\,\mathrm{OPT}(I) + h_{\max}(I)/2$. Since $h_{\max}(I) \leq \mathrm{OPT}(I)$ this implies an absolute approximation ratio of 2.5. This was independently improved by Schiermeyer [49] and Steinberg [51] with algorithms of approximation ratio 2.

In the asymptotic setting we consider instances with large optimal value. Here, the asymptotic performance ratio of the above algorithms was reduced to 4/3 by Golan [20] and then to 5/4 by Baker et al. [1]. An AFPTAS with additive constant of $\mathcal{O}(h_{\max}(I)/\varepsilon^2)$ was given by Kenyon & Rémila [40]. Jansen & Solis-Oba [36] found an APTAS with additive constant of $h_{\max}(I)$.

On the negative side, since strip packing includes the bin packing problem as a special case, there is no algorithm with absolute ratio better than 3/2 unless $\mathscr{P} = \mathscr{NP}$. After the work by Steinberg and Schiermeyer in 1994, there was no improvement on the best known approximation ratio until very recently. Jansen & Thöle [37] presented an approximation algorithm with approximation ratio $3/2 + \varepsilon$ for restricted instances where the widths are of the form i/m for $i \in \{1, \ldots, m\}$ and m is polynomially bounded in the number of items. Notice that the general version that we consider appears to be considerably more difficult. Recently, Harren & van Stee [25] were the first to break the barrier of 2 for the general problem and presented an algorithm with a ratio of 1.9396. Our main result is the following significant improvement.

Theorem 2.1. *For any $\varepsilon > 0$, there is an approximation algorithm A which produces a packing of a list I of n rectangles in a strip of width 1 and height $A(I)$ such that*

$$A(I) \leq \left(\frac{5}{3} + \varepsilon\right) \mathrm{OPT}(I).$$

Although our algorithm uses a PTAS as a subroutine and therefore has very high running time for small values of ε, this result brings us much closer to the lower bound of 3/2 for this problem.

TECHNIQUES. The algorithm approximately guesses the optimal height of a given instance. In the main phase of the algorithm we use a recent result by Bansal et al. [3], a PTAS for the so-called rectangle-packing problem with area maximization (RPA). Given a set I of rectangles, the objective is to find a subset $I' \subseteq I$ of the rectangles and a packing of I' into a unit sized bin while maximizing the total area of I'. For the iteration close to the minimal height, the approximation scheme by Bansal et al. computes a packing of a subset of the rectangles with total area at least $(1 - \delta)$ times the total area of all rectangles in I.

After this step a set of unpacked rectangles with small total area remains. The main idea of our algorithm is to create a *hole* of depth 1/3 and width ε in the packing created by the PTAS, and use this to pack the unpacked tall rectangles (with height possibly very close to 1). (The other unpacked rectangles account for the $+\varepsilon$ in our approximation ratio.) Finding a suitable location for such a hole and repacking the rectangles which we have to move out of the hole account for the largest technical challenges of this chapter. To achieve a packing of the whole input we carefully analyse the structure of the generated packing and use interesting and often intricate rearrangements of parts of the packing.

The techniques of this geometric analysis and the reorganization of the packing could be useful for several other geometric packing problems. Our reoptimization could also be help-

ful for related problems like scheduling parallel tasks (malleable and non-malleable), three-dimensional strip packing and strip packing in multiple strips. To achieve faster heuristics for strip packing, we could apply our techniques on different initial packings rather than using the PTAS from [3].

2.2 Overview of the Algorithm

Let $I = \{r_1, \ldots, r_n\}$ be the set of given rectangles, where $r_i = (w_i, h_i)$ is a rectangle with width w_i and height h_i. For a given packing P we denote the bottom left corner of a rectangle r_i by (x_i, y_i) and its top right corner by (x_i', y_i'), where $x_i' = x_i + w_i$ and $y_i' = y_i + h_i$. So the interior of rectangle r_i covers the area $(x_i, x_i') \times (y_i, y_i')$. It will be clear from the context to which packing P the coordinates refer.

Let $W_\delta = \{r_i \mid w_i > \delta\}$ be the set of so-called *δ-wide* rectangles and let $H_\delta = \{r_i \mid h_i > \delta\}$ be the set of *δ-high* rectangles. To simplify the presentation, we refer to the $1/2$-wide rectangles as *wide* rectangles and to the $1/2$-high rectangles as *high* rectangles. Let $W = W_{1/2}$ and $H = H_{1/2}$ be the sets of wide and high rectangles, respectively.

For a set T of rectangles, let $a(T) = \sum_{i \in T} w_i h_i$ be the total area and let $h(T) = \sum_{r_i \in T} h_i$ and $w(T) = \sum_{r_i \in T} w_i$ be the total height and total width, respectively. Furthermore, let $w_{\max}(T) = \max_{r_i \in T} w_i$ and $h_{\max}(T) = \max_{r_i \in T} h_i$.

We now present two important subroutines of our algorithms, namely Steinberg's algorithm [51] and an algorithm by Bansal et al. [3]. Moreover, we prove the existence of a structured packing of certain sets of wide and high rectangles.

Steinberg's algorithm. Steinberg [51] proved the following theorem for his algorithm that we use as a subroutine multiple times.

Theorem 2.2 (Steinberg's algorithm)**.** *If the following inequalities hold,*

$$w_{\max}(T) \le a, \quad h_{\max}(T) \le b, \text{ and } \quad 2a(T) \le ab - (2w_{\max}(T) - a)_+ (2h_{\max}(T) - b)_+$$

where $x_+ = \max(x, 0)$, then it is possible to pack all rectangles from T into $R = (a, b)$ in time $\mathcal{O}((n \log^2 n)/\log\log n)$.

Area maximization. Bansal, Caprara, Jansen, Prädel & Sviridenko [3] considered the problem of maximizing the total area packed into a unit-sized bin. Using a technical *Structural Lemma* they derived a PTAS for this problem.

Theorem 2.3 (Bansal, Caprara, Jansen, Prädel & Sviridenko)**.** *For any fixed $\delta > 0$, the* PTAS *from [3] returns a packing of $I' \subseteq I$ in a unit-sized bin such that $a(I') \geq (1-\delta)\mathrm{OPT}_{\mathrm{maxarea}}(I)$, where $\mathrm{OPT}_{\mathrm{maxarea}}(T)$ denotes the maximum area of rectangles from T that can be packed into a unit-sized bin.*

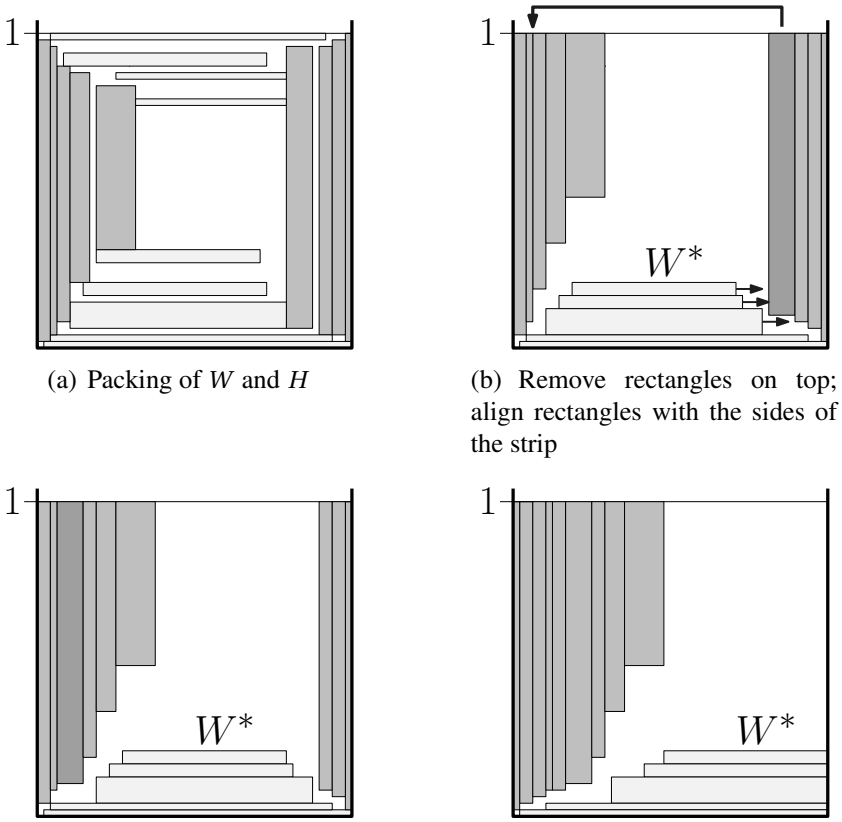
2.2.1 Existence of Structured Packings.

We show that for any set of wide and high rectangles that fits into a strip of height 1, there exists a packing of the high rectangles and of wide rectangles with at least half of their total height with a nice structure, i.e., such that the wide and the high rectangles are packed in stacks in different corners of the strip.

Lemma 2.1. *For sets $H' \subseteq H$ and $W' \subseteq W \setminus H'$ of high and wide rectangles with $\mathrm{OPT}(W \cup H) \leq 1$ there exists a packing of $W^* \cup H'$ with $W^* \subseteq W'$ and $h(W^*) \geq h(W')/2$ such that the high rectangles are stacked in the top left corner of the strip, i.e. sorted by non-increasing heights and packed from the left side with their top edges at height 1 and the rectangles from W^* are stacked in the bottom right corner of the strip, i.e. sorted by non-increasing widths and packed right-aligned on top of each other on the bottom of the strip.*

Proof. See Figure 2.1 for an illustration of the following proof. Consider a packing of high rectangles H' and wide rectangles W' into a strip of height 1. Associate each wide rectangle with the closer boundary of the packing, i.e., either the top or bottom of the strip (a rectangle that has the same distance to both sides of the strip can be associated with an arbitrary side). Assume w.l.o.g. that the total height of the rectangles associated with the bottom is at least as large as the total height of the rectangles associated with the top of the strip. Remove the rectangles that are associated with the top and denote the other wide rectangles by W^*. Push the rectangles of W^* together into a stack that is aligned with the bottom of the strip by moving them purely vertically and move the high rectangles such that they are aligned with the top of the strip and form stacks at the left and right side of the strip. Order the stacks of the high rectangles by non-increasing order of height and the stack of the wide rectangles by non-increasing order of width.

Now apply the following process. Take the shortest rectangle with respect to the height from the right stack of the high rectangles and insert it at the correct position into the left stack, i.e., such that the stack remains in the order of non-increasing heights. Since the total widths of both stacks of high rectangles remains the same, we can move the wide rectangles to the right if this insertion causes an overlap. Obviously this process moves all high rectangles to the left and retains a feasible packing. In the end, all high rectangles form



(a) Packing of W and H

(b) Remove rectangles on top; align rectangles with the sides of the strip

(c) Insert the shortest rectangle from the right in the stack on the left; move W^* horizontally

(d) Final packing with stacks in the bottom right and top left

Figure 2.1: The insertion process of Lemma 2.1.

a stack in the top left corner of the strip. Move the wide rectangles to the right such that they form a stack in the bottom right corner of the strip. □

We utilize the previous existence result with the following Corollary.

Corollary 2.1. *For sets $H' \subseteq H$ and $W' \subseteq W \setminus H'$ of high and wide rectangles with* $\mathrm{OPT}(W \cup H) \leq 1$ *we can derive a packing of $W' \cup H'$ into a strip of height at most $1 + h(W')/2$ such that the wide rectangles are stacked in the bottom right of the strip and the high rectangles are stacked above the wide rectangles at the left side of the strip in time $\mathcal{O}(n \log n)$.*

Proof. Consider a packing of height 1 of $W^* \cup H'$ with $W^* \subseteq W'$ and $h(W^*) \geq h(W')/2$ such that the wide rectangles from W^* are stacked in the bottom right corner of the strip and the high rectangles are stacked above W^* at the top left of the strip. Such a packing exists by Lemma 2.1. Now move up $W^* \cup H'$ by $h(W' \setminus W^*)$, pack $W' \setminus W^*$ below W^* and restore the order in the stack of the wide rectangles. This does not cause a conflict as the original

surface of W^* is not violated. As $h(W' \setminus W^*) = h(W') - h(W^*) \leq h(W')/2$ the height bound of the corollary in satisfied. Since the packing only consists of the ordered stacks of the high and wide rectangles, we can easily derive a packing of at most the same height in time $\mathcal{O}(n \log n)$ by building the stacks and moving down the stack of H' as far as possible. □

2.2.2 Modifying Packings.

Our methods involve modifying existing packings in order to insert some additional rectangles. To describe these modifications or, more specifically, the rectangles involved in these modifications, we introduce the following notations—see Figure 2.2. Let $\mathrm{PointR}(x, y)$ be the rectangle that contains the point (x, y) (in its interior; if no such rectangle exists $\mathrm{PointR}(x, y)$ is empty). We use the notation of *vertical line rectangles* $\mathrm{VLR}(x; y_1, y_2)$ and *horizontal line rectangles* $\mathrm{HLR}(x_1, x_2; y)$ as the rectangles that contain any point of the given vertical or horizontal line in their interiors, respectively. Finally, we introduce two notations for rectangles whose interiors are completely contained in a designated area, namely $\mathrm{AR}(x_1, x_2; y_1, y_2)$ for rectangles completely inside the respective rectangle and $\mathrm{AR}(p)$ for rectangles completely above a given polygonal line p, where p is a staircase-cut on $[0, 1]$.

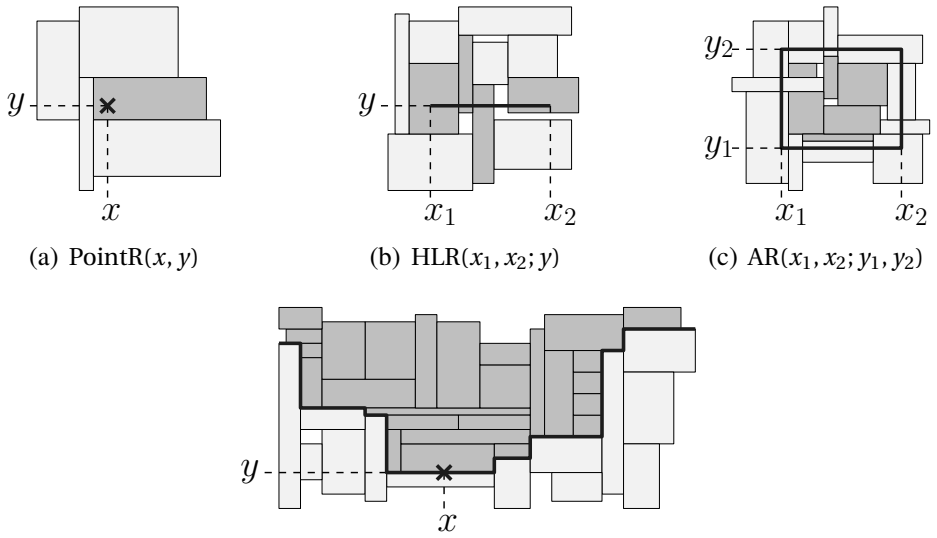
To describe such a polygonal line p we define the *vertical polygonal chain extension* of a point (x, y) inside a given packing P as follows. Start at position (x, y) and move leftwards until hitting a rectangle r_i. Then move upwards to the top of r_i, that is, up to position y'_i. Repeat the previous steps until hitting the left side of the strip. Then do the same thing to the right starting again at (x, y). We denote the polygonal chain that results from this process by $\mathrm{VPCE}(x, y)$. In addition, let $\mathrm{VPCE}_{\mathrm{left}}(x, y)$ and $\mathrm{VPCE}_{\mathrm{right}}(x, y)$ be the left and right parts of this polygonal chain, respectively. Another way to describe a polygonal line is by connecting a given sequence of points, which we denote as $\mathrm{PL}((x_1, y_1), (x_2, y_2), \ldots)$.

2.2.3 Algorithm

We start now with the presentation of our algorithm. Let $\varepsilon < 1/(28 \cdot 151) = 1/4228$ throughout this chapter. With the following lemma we show that we can concentrate on instances I with $\mathrm{OPT}(I) \leq 1$.

Lemma 2.2. *If there exists a polynomial-time algorithm for strip packing that packs any instance I with optimal value at most 1 into a strip of height $h \geq 1$, then there also exists a polynomial-time algorithm for strip packing with absolute approximation ratio at most $h + \varepsilon$.*

Proof. Let ALG be the algorithm that packs any instance I with optimal value at most 1 into a strip of height h and assume that $h \leq 2$ by otherwise applying Steinberg's algorithm. Let ε'



(a) PointR(x, y) (b) HLR$(x_1, x_2; y)$ (c) AR$(x_1, x_2; y_1, y_2)$

(d) AR(VPCE(x, y))

Figure 2.2: Notations

be the maximal value with $\varepsilon' \leq \varepsilon/(2h)$ such that $1/\varepsilon'$ is integer. We guess the optimal value approximately and apply ALG on an appropriately scaled instance. To do this, we first apply Steinberg's algorithm on I to get a packing into height $h' \leq 2\,\mathrm{OPT}(I)$. We split the interval $J = [h'/2, h']$ into $1/\varepsilon'$ subintervals $J_i = [(1+\varepsilon'(i-1))h'/2, (1+\varepsilon' i)h'/2]$ for $i = 1, \ldots, 1/\varepsilon'$. Then we iterate over $i = 1, \ldots, 1/\varepsilon'$, scale the heights of all rectangles by $2/((1+\varepsilon' i)h')$ and apply the algorithm ALG on the scaled instance I'. Convert the packing to a packing of the unscaled instance I and finally output the minimal packing that was derived. We eventually consider $i^* \in \{1, \ldots, 1/\varepsilon'\}$ with $\mathrm{OPT}(I) \in J_{i^*}$. Then we have

$$1-\varepsilon' < 1 - \frac{\varepsilon'}{1+\varepsilon' i^*} = \frac{(1+\varepsilon'(i^*-1))\frac{h'}{2}}{(1+\varepsilon' i^*)\frac{h'}{2}} \leq \mathrm{OPT}(I') \leq \frac{(1+\varepsilon' i^*)\frac{h'}{2}}{(1+\varepsilon' i^*)\frac{h'}{2}} = 1$$

and thus

$$\frac{\mathrm{ALG}(I)}{\mathrm{OPT}(I)} = \frac{\mathrm{ALG}(I')}{\mathrm{OPT}(I')} < \frac{h}{1-\varepsilon'} = h + \frac{\varepsilon' h}{1-\varepsilon'} \leq h + 2\varepsilon' h \leq h + \varepsilon.$$

□

Thus we concentrate on approximating instances that fit into a strip of height 1 and therefore assume $\mathrm{OPT}(I) \leq 1$ for the remainder of this chapter. The overall approach for our algorithm for strip packing is as follows.

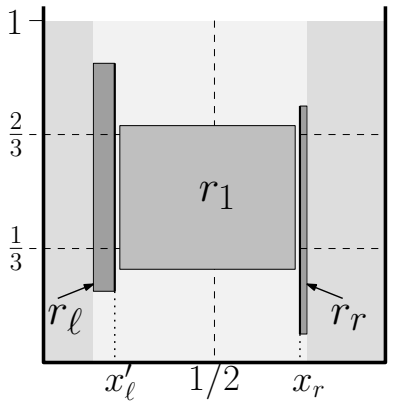
First, we use some direct methods involving Steinberg's algorithm to solve instances I

with $h(W_{1-130\varepsilon}) \geq 1/3$ or $w(H_{2/3}) \geq 27/28$, that is, special cases where many rectangles have a width of almost 1, or almost all of the rectangles are at least $2/3$ high. Having this many high or wide rectangles makes it much easier to pack all rectangles without wasting much space.

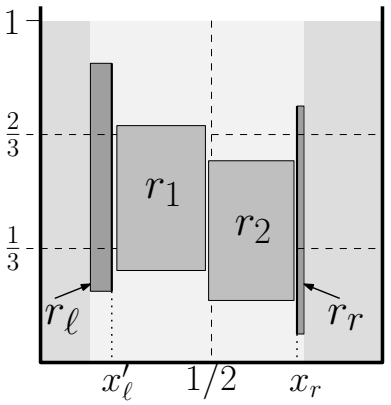
For any instance I that does not satisfy these conditions, we first apply the PTAS from [3] with an accuracy of $\delta = \varepsilon^2/2$ to pack most of the rectangles into a strip of height 1. Denote the resulting packing of $I' \subseteq I$ by P and let $R = I \setminus I'$ be the set of remaining rectangles. By Theorem 2.3 we have $a(R) \leq \varepsilon^2/2 \cdot \mathrm{OPT}_{\mathrm{maxarea}}(I) = \varepsilon^2/2 \cdot a(I) \leq \varepsilon^2/2$. Pack $R \cap H_{\varepsilon/2}$ into a container $C_1 = (\varepsilon, 1)$ (by forming a stack of the rectangles of total width at most $a(R)/(\varepsilon/2) \leq \varepsilon$) and pack $R \setminus H_{\varepsilon/2}$ with Steinberg's algorithm into a container $C_2 = (1, \varepsilon)$ (this is possible by Theorem 2.2 since $h_{\max}(R \setminus H_{\varepsilon/2}) \leq \varepsilon/2$, $w_{\max}(R \setminus H_{\varepsilon/2}) \leq 1$ and $2a(R \setminus H_{\varepsilon/2}) \leq \varepsilon^2 < \varepsilon$).

We will now modify the packing P to free a gap of width ε and height 1 to insert the container C_1 while retaining a total packing height of at most $5/3$. This is the main part of our work. Afterwards, we pack C_2 above the entire packing, achieving a total height of at most $5/3 + \varepsilon$. The entire algorithm to modify the PTAS packing is given in Algorithm 2.1.

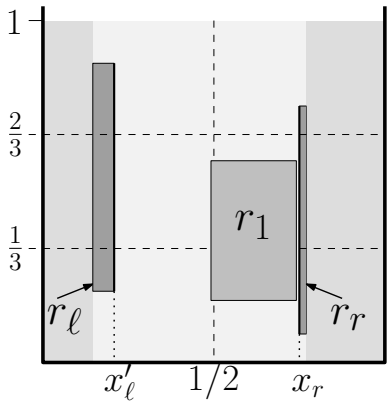
This chapter is organized as follows. Section 2.3.1 and Section 2.3.2 present the direct methods to solve instances I with $h(W_{1-130\varepsilon}) \geq 1/3$ or $w(H_{2/3}) \geq 27/28$. In Section 2.4.1 we consider certain packings with a rectangle of height at least $1/3$ (see Lemma 2.5 and 2.6 and Algorithm 2.2 and 2.3). Section 2.4.2 deals with packings without long rectangles close to the left or right side of the strip (see Lemma 2.7 and Algorithm 2.4). An illustration of the remaining cases, if Algorithm 2.2-2.4 are not applicable, is given in Figure 2.3. The methods for solving these cases are presented in Section 2.4.3, 2.4.4 and 2.4.5. In Section 2.5 we prove that our Algorithm 2.1 covers all the cases.



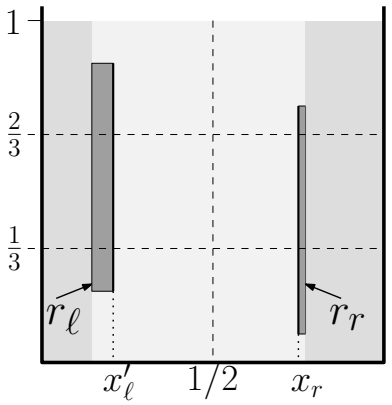
(a) A 1/3-high rectangle almost spans from I_ℓ to I_r: apply Algorithm 2.5



(b) A 1/3-high rectangle spans from I_ℓ to I_M and a 1/3-high rectangle spans from I_M to I_r: apply Algorithm 2.6



(c) A 1/3-high rectangle spans from I_M to I_r but no 1/3-high spans between I_ℓ and I_M: apply Algorithm 2.7



(d) No 1/3-high rectangles span across the intervals: apply Algorithm 2.7

Figure 2.3: Schematic illustration of the main cases if Algorithm 2.2, 2.3 and 2.4 are not applicable. The area to the left of r_ℓ and the area to the right of r_r is almost completely covered by 2/3-high rectangles (and shown in darker shade). I_ℓ, I_r and I_M are horizontal intervals very close to r_ℓ, r_r and the middle of the strip.

Algorithm 2.1 Turn the PTAS packing into a strip packing

Requirement: $\varepsilon < 1/4228$, $h(W_{1-130\varepsilon}) < 1/3$ and $w(H_{2/3}) < 27/28$

Input: packing P produced by the PTAS from [3] with an accuracy of $\delta := \varepsilon^2/2$.

Pack the remaining unpacked rectangles into $C_1 = (\varepsilon, 1)$ and $C_2 = (1, \varepsilon)$.
if there is a rectangle r_1 of height $h_1 > 1/3$ with one side at position $x_1^* \in [\varepsilon, 1/2 - \varepsilon]$, and the total width of 2/3-high rectangles to the left of x_1^* is at most $x_1^* - \varepsilon$ (or if these conditions hold for P mirrored over $x = 1/2$) **then**
    apply Algorithm 2.2 (Lemma 2.5), stop
if there is a rectangle r_1 of height $h_1 \in [1/3, 2/3]$ and width $w_1 \in [\varepsilon, 1 - 2\varepsilon]$ and $y_1 \geq 1/3$ or $y_1' \leq 2/3$ **then**
    apply Algorithm 2.3 (Lemma 2.6), stop
Let r_ℓ be the rightmost 2/3-high rectangle in $\mathrm{AR}(0, 1/2-\varepsilon; 0, 1)$ and let r_r be the leftmost 2/3-high rectangle in $\mathrm{AR}(1/2+\varepsilon, 1; 0, 1)$ (We use dummy rectangles of height 1 and width 0 on the sides of the strip if no such rectangles exist). Redefine r_ℓ and/or r_r if necessary (see Section 2.5).
    \\At this point, all vertical sides of 1/3-high rectangles are to the left of $x = x'_\ell + \varepsilon$,
    \\to the right of $x = x_r - \varepsilon$, or within ε distance of $x = 1/2$.
Let $c_3 = 2$ if $x_\ell^{(<)} 1/2 - 3\varepsilon$ and $x_r > 1/2 + 3\varepsilon$, and $c_3 = 5$ otherwise. Let $c_1 = 5 \cdot c_3$.
if there is no 1/3-high rectangle that intersects $[c_1\varepsilon, (c_1+1)\varepsilon] \times [0, 1]$, and $h(W_{1-5(c_1+1)\varepsilon}) < 1/3$ (or if these conditions hold for P mirrored over $x = 1/2$) **then**
    apply Algorithm 2.4 (Lemma 2.7), stop
Apply Algorithm 2.5, 2.6, or 2.7, depending on which of the cases in Figure 2.3 occurs (see Figure 2.3)

2.3 Direct Methods

In this section, we display direct methods to solve instances I with $h(W_{1-130\varepsilon}) \geq 1/3$ or $w(H_{2/3}) \geq 27/28$ without using the PTAS from [3].

2.3.1 Total Area of Very Wide Rectangles is Large

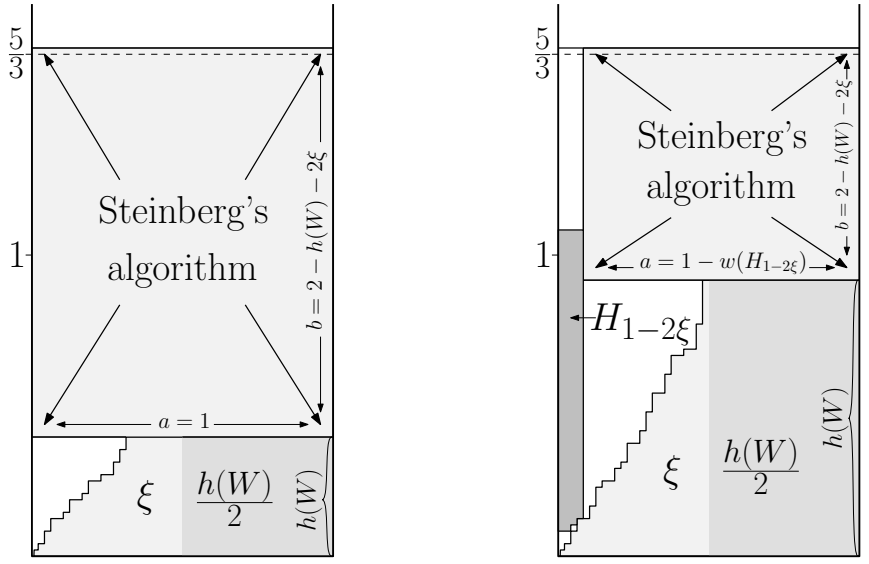
We compute a packing for instances with a large area guarantee of very wide rectangles.

Lemma 2.3. *If $h(W_{1-130\varepsilon}) \geq 1/3$, then we can derive a packing into a strip of height $5/3 + 260\varepsilon/3$ in time $\mathcal{O}((n\log^2 n)/\log\log n)$.*

Proof. Let $W' = W_{1-130\varepsilon}$. The total height of the very wide rectangles in W' gives us a non-trivial additional area guarantee for the wide rectangles as follows.

$$\begin{aligned} A(W) = A(W') + A(W \setminus W') &> h(W') \cdot (1 - 130\varepsilon) + h(W \setminus W') \cdot 1/2 \\ &= h(W') \cdot (1/2 - 130\varepsilon) + h(W)/2 \geq 1/3 \cdot (1/2 - 130\varepsilon) + h(W)/2. \end{aligned}$$

For $\xi = 1/3 \cdot (1/2 - 130\varepsilon)$ we have $A(W) > \xi + h(W)/2$ and $0 < \xi \leq 1/6$. We consider two cases in which we use the additional area guarantee of ξ for the wide rectangles to derive a packing into a strip of height $2 - 2\xi = 5/3 + 260\varepsilon/3$.



(a) Case $1.2 - h(W) - 2\xi \geq 1$ (b) Case $2.2 - h(W) - 2\xi < 1$

Figure 2.4: Packing methods for Lemma 2.3

Case 1. $2 - h(W) - 2\xi \geq 1$.
Stack the wide rectangles in the bottom of the strip and use Steinberg's algorithm to pack $I \setminus W$ above this stack into a rectangle of size (a, b) with $a = 1$ and $b = 2 - h(W) - 2\xi$ (see Figure 2.4(a)). Steinberg's algorithm is applicable since we have $h_{\max}(I \setminus W) \leq 1 \leq b$, $w_{\max}(I \setminus W) \leq 1/2 = a/2$ and

$$2a(I \setminus W) \leq 2 - 2\xi - h(W) = ab = ab - (2w_{\max}(I \setminus W) - a)_+(2h_{\max}(I \setminus W) - b)_+.$$

The total height of the packing is $h(W) + 2 - h(W) - 2\xi = 2 - 2\xi$.

Case 2. $2 - h(W) - 2\xi < 1$.
In this case we cannot apply Steinberg's algorithm to pack $I \setminus W$ into the area of size $(1, 2 - h(W) - 2\xi)$ above the stack of W as $h_{\max}(I \setminus W)$ might be greater than $2 - h(W) - 2\xi$.

We have $1 - 2\xi < h(W) \leq 1$ since 1 is a natural upper bound for the total height of the wide rectangles. Pack the rectangles of W in a stack aligned with the bottom right corner of the strip as before. Pack the rectangles of $H_{1-2\xi} \setminus W$ in a stack aligned with the left side of the strip and move this strip downwards as far as possible (see Figure 2.4(b)). Corollary 2.1 shows that $H_{1-2\xi} \setminus W$ can be moved down such that the total height of the packing so far is at most $1 + h(W)/2 \leq 3/2$. Let $T = I \setminus (W \cup H_{1-2\xi})$ be the set of the remaining rectangles. We have

$$a(T) \leq 1 - a(W) - a(H_{1-2\xi} \setminus W) \leq 1 - \xi - \frac{h(W)}{2} - (1 - 2\xi)w(H_{1-2\xi} \setminus W).$$

Pack T with Steinberg's algorithm in the rectangle of size (a, b) with $a = 1 - w(H_{1-2\xi} \setminus W)$ and $b = 2 - h(W) - 2\xi$ above W and to the right of $H_{1-2\xi} \setminus W$. We have $w(H_{1-2\xi} \setminus W) \leq 1/2$ as otherwise all wide rectangles are either above or below a rectangle from $H_{1-2\xi} \setminus W$ in any optimal packing and thus $h(W) \leq 4\xi$ (which is a contradiction to $2 - h(W) - 2\xi < 1$ for $\xi \leq 1/6$). Thus we have $h_{\max}(T) \leq 1 - 2\xi \leq b$ and $w_{\max}(T) \leq 1/2 \leq a$ and with

$$\begin{aligned}
ab - (2w_{\max}(T) - a)_+(2h_{\max}(T) - b)_+ \geq \\
(1 - w(H_{1-2\xi} \setminus W))(2 - h(W) - 2\xi) - \\
(2 \cdot 1/2 - (1 - w(H_{1-2\xi} \setminus W)))_+ \cdot (2(1 - 2\xi) - (2 - h(W) - 2\xi))_+ \geq \\
2 - h(W) - 2\xi - 2w(H_{1-2\xi} \setminus W) + h(W)w(H_{1-2\xi} \setminus W) + \\
2\xi w(H_{1-2\xi} \setminus W) - (w(H_{1-2\xi} \setminus W))_+(h(W) - 2\xi)_+ = \\
2 - h(W) - 2\xi - 2w(H_{1-2\xi} \setminus W) + 4\xi w(H_{1-2\xi} \setminus W)
\end{aligned}$$

we get $2a(T) \leq ab - (2w_{\max} - a)_+(2h_{\max} - b)_+$. We have $(h(W) - 2\xi)_+ = h(W) - 2\xi$ in the last step of the calculation since $h(W) > 4\xi$. The total height of the packing corresponds to the height of the wide rectangles $h(W)$ plus the height of the target area for Steinberg's algorithm $b = 2 - h(W) - 2\xi$. In total we have a height of $h(W) + 2 - h(W) - 2\xi = 2 - 2\xi$. □

2.3.2 Large Total Width of the 2/3-high Rectangles

In this section we assume that $w(H_{2/3}) \geq 27/28$, i.e., the total width of the 2/3-high rectangles is very large. As in the previous section we can solve this case directly without using the PTAS from [3].

For the ease of presentation, let $\alpha = w(H_{2/3}) \geq 27/28$. Since the total height of the rectangles of $W_{1-\alpha/2} \setminus H_{2/3}$ plays an important role in our method, we introduce the notation $y = h(W_{1-\alpha/2} \setminus H_{2/3})$. Moreover, we use the stronger area guarantee of the 5/6-high rectangles and therefore denote their total width by $\beta = w(H_{5/6})$. Finally, let $\delta = w(H_{1/3} \setminus H)$ be the total width of the rectangles of height within 1/3 and 1/2.

Bounding y and δ. Let $\alpha' < \alpha$ such that $W_{1-\alpha/2} = \{r_i \mid w_i > 1 - \alpha/2\} = \{r_i \mid w_i \geq 1 - \alpha'/2\}$, e.g., set α' such that the shortest rectangle in $W_{1-\alpha/2}$ has width $1 - \alpha'/2$. Note that in any optimal packing, all rectangles from $W_{1-\alpha/2}$ occupy the x-interval $(\alpha'/2, 1-\alpha'/2)$ of width $1 - \alpha'$ completely. On the other hand, there has to be a rectangle from $H_{2/3}$ that intersects this interval since $w(H_{2/3}) = \alpha > \alpha'$. Therefore we have

$$y = h(W_{1-\alpha/2} \setminus H_{2/3}) < 1/3. \tag{2.1}$$

It follows directly that the sets $W_{1-\alpha/2} \setminus H_{2/3}$ and $H_{1/3}$ are disjoint.

Since no rectangle of $H_{1/3}$ fits above a rectangle of $H_{2/3}$, only in a total width of $1 - \alpha$ can rectangles in $H_{1/3} \setminus H_{2/3}$ possibly be packed. It follows that a total width of at most $2(1 - \alpha)$ of rectangles in $H_{1/3}$ can exist, because at most two such rectangles can fit on top of each other. By direct calculation for $\alpha > 4/5$ we get

$$\delta \leq 2(1 - \alpha) < \alpha/2. \tag{2.2}$$

In the following we distinguish three main cases according to y and β. See Figure 2.5(a) for the first two cases and Figure 2.5(b) for the third case.

Case 1. $y \geq \frac{4}{3}\frac{1-\alpha}{1-\alpha/2}$.
We use the methods of Corollary 2.1 for $H \cup W_{1-\alpha/2}$, and need a height of at most $1 + y/2$

which is less than $7/6$ by Inequality (2.1). Above it, we define a container of width δ and height $1/2$ at the left side of the strip where we pack all remaining $1/3$-high rectangles, i.e., $H_{1/3} \setminus H$. Next to it we have an area (a, b) of width $a = 1-\delta$ and height $b = 2/3 - y/2 > 1/2$. In it we pack all remaining rectangles, noted by $T = I \setminus (H_{1/3} \cup W_{1-\alpha/2})$, that have height at most $h_{\max}(T) \le 1/3 < b$, width at most $w_{\max}(T) \le 1 - \alpha/2 < 1 - \delta = a$ by Inequality (2.2), and area at most

$$a(T) \le 1 - a(H_{2/3}) - a(W_{1-\alpha/2} \setminus H_{2/3}) - a(H_{1/3} \setminus H) \le 1 - \frac{2}{3}\alpha - \Big(1 - \frac{\alpha}{2}\Big)\, y - \frac{\delta}{3}.$$

This works according to the Steinberg condition for any $y \ge \frac{4}{3}\frac{1-\alpha}{1-\alpha/2}$ since

$$\begin{aligned}
ab - (2w_{\max}(T) - a)_+(2h_{\max}(T) - b)_+ &= (1-\delta)\Big(\frac{2}{3} - \frac{y}{2}\Big) - (1 - \alpha + \delta)_+\Big(\frac{y}{2}\Big)_+ \\
&= \frac{2}{3} - \frac{2\delta}{3} - y + \frac{\alpha y}{2} \\
&= \frac{2}{3} - \frac{2\delta}{3} - 2y + \alpha y + y(1 - \frac{\alpha}{2}) \\
&\ge \frac{2}{3} - \frac{2\delta}{3} - 2y + \alpha y + \frac{4}{3}\frac{1-\alpha}{1-\alpha/2}(1 - \frac{\alpha}{2}) \\
&= 2(1 - \frac{2}{3}\alpha - (1 - \frac{\alpha}{2})\, y - \frac{\delta}{3}) \\
&\ge 2a(T).
\end{aligned}$$

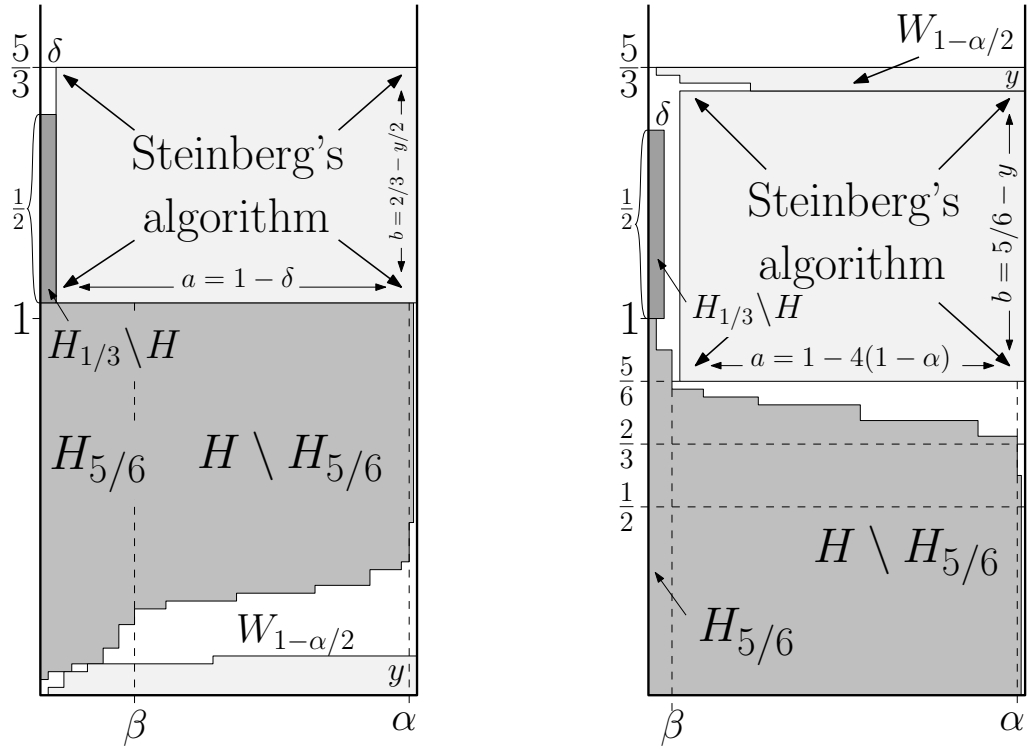
Case 2. $\beta \ge 4(1-\alpha)$.

We use the same packing as in Case 1. The total area of the high rectangles is now at least $\frac{5}{6}\beta + \frac{2}{3}(\alpha - \beta) = \frac{2}{3}\alpha + \frac{1}{6}\beta \ge 2/3$. Therefore, the remaining unpacked rectangles, noted by T, have area at most

$$a(T) \le 1 - a(H_{2/3}) - a(W_{1-\alpha/2} \setminus H_{2/3}) - a(H_{1/3} \setminus H) \le \frac{1}{3} - \Big(1 - \frac{\alpha}{2}\Big) y - \frac{\delta}{3}.$$

Since only the area of T changes compared to Case 1, we only have to verify the third Steinberg condition to pack T with Steinberg's algorithm into the area (a, b).

$$\begin{aligned}
ab - (2w_{\max}(T) - a)_+(2h_{\max}(T) - b)_+ &= \frac{2}{3} - \frac{2\delta}{3} - y + \frac{\alpha y}{2} \\
&= \frac{2}{3} - \frac{2\delta}{3} - (2 - \alpha)\, y + \Big(1 - \frac{\alpha}{2}\Big) y \\
&\ge 2\Big(\frac{1}{3} - \frac{\delta}{3} - \Big(1 - \frac{\alpha}{2}\Big) y\Big) \\
&\ge 2a(T).
\end{aligned}$$



(a) Case 1 or 2.
$y \geq \frac{4}{3}\frac{1-\alpha}{1-\alpha/2}$ or
$\beta \geq 4(1-\alpha)$

(b) Case 3.
$y < \frac{4}{3}\frac{1-\alpha}{1-\alpha/2}$ and
$\beta < 4(1-\alpha)$

Figure 2.5: Packing methods for Lemma 2.4

CASE 3. $y < \frac{4}{3}\frac{1-\alpha}{1-\alpha/2}$ and $\beta < 4(1-\alpha)$.
Note that $y < \frac{4}{3}\frac{1-\alpha}{1-\alpha/2} \leq \frac{4}{3}\frac{1-27/28}{1-27/56} = \frac{56}{609} < \frac{1}{6}$ in this case. We pack the set H of the high rectangles aligned with the bottom of the strip, sorted by non-increasing heights (from left to right). We pack the rectangles of $W_{1-\alpha/2} \setminus H$ stacked in the area $[0,1] \times [5/3-y, 5/3]$ and the rectangles of $H_{1/3} \setminus H$ in the area $[0,\delta] \times [1,3/2]$ (this is possible because $3/2 \leq 5/3-y$). We have $\delta < 4(1-\alpha)$, since by Inequality (2.2) we have $\delta < 2(1-\alpha)$. Furthermore, by assumption we have $\beta < 4(1-\alpha)$. It follows that the area $[4(1-\alpha),1] \times [5/6, 5/3-y]$ of width $a = 1-4(1-\alpha) = 4\alpha-3$ and height $b = 5/3-y-5/6 \geq 2/3$ is still free. We pack all remaining rectangles, noted by T, in this area using Steinberg's algorithm. We have $h_{\max}(T) \leq 1/3 \leq b/2$, $w_{\max}(T) \leq 1-\alpha/2 < 4\alpha-3 = a$, since $\alpha > 8/9$, and area at most

$$a(T) \leq 1 - a(H_{2/3}) - a(W_{1-\alpha/2} \setminus H_{2/3}) \leq 1 - \frac{2}{3}\alpha - \left(1-\frac{\alpha}{2}\right)y.$$

Hence the Steinberg condition is satisfied for $\alpha \geq 27/28 \geq (27-30y)/(28-30y)$ since

$$\begin{aligned} ab - (2w_{\max}(T) - a)_+(2h_{\max}(T) - b)_+ &= ab \\ &= (4\alpha - 3)\left(\frac{5}{6} - y\right) \\ &= -\frac{4\alpha}{3} + y\alpha + \alpha\left(\frac{28-30y}{6}\right) - \frac{5}{2} + 3y \\ &\geq -\frac{4\alpha}{3} + y\alpha + \frac{27-30y}{28-30y} \cdot \frac{28-30y}{6} \\ &\quad -\frac{5}{2} + 3y \\ &= -\frac{4\alpha}{3} + y\alpha + 2 - 2y \\ &\geq 2a(T). \end{aligned}$$

Since the running time of our methods is dominated by the application of Steinberg's algorithm we showed the following lemma.

Lemma 2.4. *If $w(H_{2/3}) \geq 27/28$, then we can derive a packing of I into a strip of height $5/3$ in time $\mathcal{O}((n\log^2 n)/\log\log n)$.*

We have finished the presentation of the methods that we directly apply to the input if $h(W_{1-130\varepsilon}) \geq 1/3$ (Section 2.3.1) or $w(H_{2/3}) \geq 27/28$(Section 2.3.2).

2.4 Modifying a Packing

In the following sections we always assume that we already derived a packing P using the PTAS from [3] and it remains to free a place for the containers C_1 and C_2 of size $(\varepsilon, 1)$ and $(1, \varepsilon)$, respectively.

2.4.1 Rectangle of Height Greater Than $1/3$

Lemma 2.5. *If the following conditions hold for P, namely*

1.1. there is a rectangle r_1 of height $h_1 > 1/3$ with one side at position $x_1^* \in [\varepsilon, 1/2-\varepsilon]$, and

1.2. the total width of $2/3$-high rectangles to the left of x_1^* is at most $x_1^* - \varepsilon$, that is

$$w(\mathrm{AR}(0, x_1^*; 0, 1) \cap H_{2/3}) \leq x_1^* - \varepsilon,$$

then we can derive a packing of I into a strip of height $5/3+\varepsilon$ in additional time $\mathcal{O}(n \log n)$.

Note that Condition 1.1 leaves open whether x_1^* refers to the left or right side of r_1 as our method works for both cases. In particular, r_1 could be one of the $2/3$-high rectangles from Condition 1.2.

Proof. Assume w.l.o.g. $y_1' > 2/3$ by otherwise mirroring the packing P over $y = 1/2$.

We lift up a part of the packing P in order to derive a gap of sufficient height to insert the container C_1. In this case we mirror the part of the packing that we lift up. Algorithm 2.2 gives a compressed version of the following detailed description. See Figure 2.6 for an illustration.

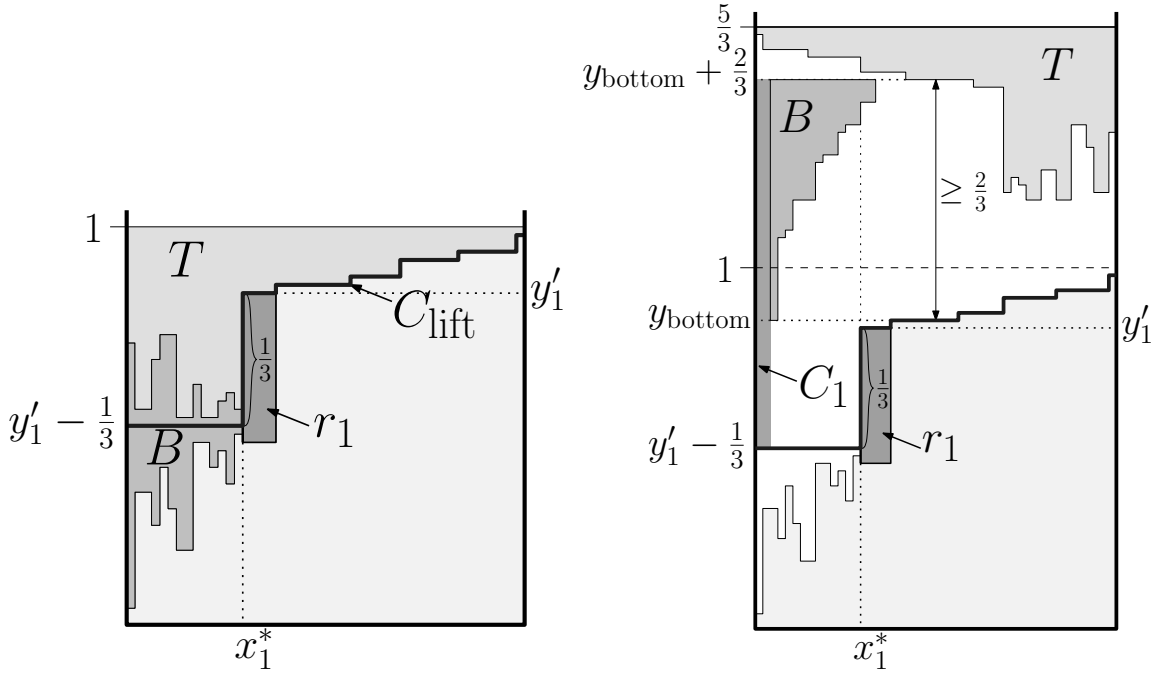
Consider the contour C_{lift} defined by a horizontal line at height $y = y_1' - 1/3$ to the left of x_1^*, a vertical line at width $x = x_1^*$ up to y_1' and a vertical polygonal chain extension to the right starting at the top of r_1. More formally, $C_{\text{lift}} = \text{PL}((0, y_1' - 1/3), (x_1^*, y_1' - 1/3), (x_1^*, y_1')) + \text{VPCE}_{\text{right}}(x_1^*, y_1')$, where the $+$-operator denotes the concatenation of polygonal lines (see thick line in Figure 2.6(a)). Let $T = \text{AR}(C_{\text{lift}})$ be the set of rectangles that are completely above this contour.

Move up T by $2/3$ (and hereby move T completely above the previous packing since $y_1' > 2/3$ and thus $y_1' - 1/3 > 1/3$) and mirror T vertically, i.e., over $x = 1/2$. Let y_{bottom} be the height of C_{lift} at $x = 1/2$ (C_{lift} crosses the point $(1/2, y_{\text{bottom}})$). By definition, C_{lift} is non-decreasing and no rectangle intersects with C_{lift} to the right of x_1^*. Therefore, T is completely packed above $y = y_{\text{bottom}} + 2/3$ on the left side of the strip, i.e., for $x \leq 1/2$, and $P \setminus T$ does not exceed y_{bottom} between $x = x_1^*$ and $x = 1/2$. Thus between $x = x_1^*$ and $x = 1/2$ we have a gap of height at least $2/3$.

Let $B = \text{HLR}(0, x_1^*; y_1' - 1/3)$ be the set of rectangles that intersect height $y = y_1' - 1/3$ to the left of x_1^* (see Figure 2.6(a)). Note that $r_1 \in B$, if x_1^* corresponds to the right side of r_1. Remove B from the packing, order the rectangles by non-increasing order of height and build a top-left-aligned stack at height $y = y_{\text{bottom}} + 2/3$ and distance ε from the left side of the strip. Since we keep a slot of width ε to the left, the stack of B might exceed beyond x_1^*. This overhang does not cause an overlap of rectangles because Condition 1.1 ensures that $x_1^* \leq 1/2 - \varepsilon$ and thus the packing of B does not exceed position $x = 1/2$ and Condition 1.2 ensures that the excessing rectangles have height at most $2/3$ whereas the gap has height at least $2/3$.

Now pack the container C_1 top-aligned at height $y_{\text{bottom}} + 2/3$ directly at the left side of the strip. C_1 fits here since $y_{\text{bottom}} + 2/3 - (y_1' - 1/3) = 1 + y_{\text{bottom}} - y_1' \geq 1$. Finally, pack C_2 above the entire packing at height $y = 5/3$, resulting in a total packing height of $5/3 + \varepsilon$. □

Note that Lemma 2.5 can symmetrically be applied for a 1/3-high rectangle with one side at position $x_1^* \in [1/2+\varepsilon, 1-\varepsilon]$ with $w(\mathrm{AR}(x_1^*, 1; 0, 1) \cap H_{2/3}) \leq 1 - x_1^* - \varepsilon$ by mirroring P over $x = 1/2$.



(a) Definition of C_{lift}, T and B.

(b) Modified packing where C_1 is packed in the shaded slot of height 1 and width ε.

Figure 2.6: Packing methods for Lemma 2.5

Algorithm 2.2 Edge of height greater than 1/3

Requirement: Packing P that satisfies Conditions 1.1 and 1.2 and $y_1' > 2/3$.

1: Move up the rectangles $T = \mathrm{AR}(C_{\text{lift}})$ by $2/3$ and then mirror the packing of these rectangles vertically at position $x = 1/2$.
2: Order the rectangles of $B = \mathrm{HLR}(0, x_1^*; y_1' - 1/3)$ by non-increasing order of height and pack them into a top-aligned stack at position $(\varepsilon, y_{\text{bottom}} + 2/3)$.
3: Pack C_1 top-aligned at position $(0, y_{\text{bottom}} + 2/3)$ and pack C_2 above the entire packing.

Lemma 2.6. *If the following condition holds for P, namely*

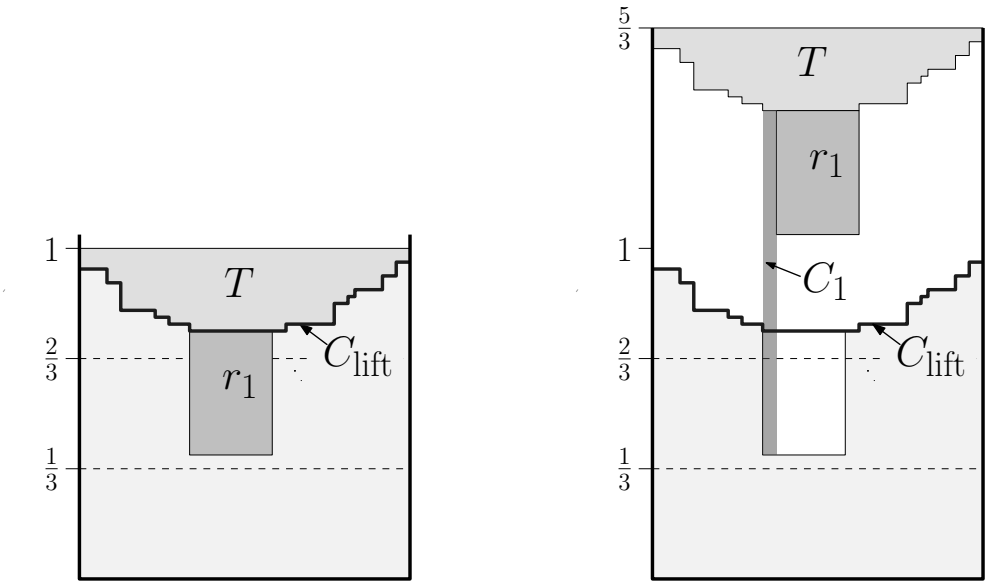
1.3. there is a rectangle r_1 of height $h_1 \in [1/3, 2/3]$ and width $w_1 \in [\varepsilon, 1-2\varepsilon]$, and $y_1 \geq 1/3$ or $y_1' \leq 2/3$,

then we can derive a packing of I into a strip of height $5/3 + \varepsilon$ in additional time $\mathcal{O}(n)$.

Proof. See Figure 2.7 for an illustration of the following proof. W.l.o.g. we assume that $y_1 \geq 1/3$, by otherwise mirroring the packing horizontally, i.e., over $y = 1/2$. Furthermore, we assume that $x_1' \leq 1 - \varepsilon$ since $w_1 \leq 1 - 2\varepsilon$ and otherwise mirror the packing vertically, i.e., over $x = 1/2$.

Define a vertical polygonal chain extension $C_{\text{lift}} = \text{VPCE}(x_1, y_1')$ starting on top of r_1 and let $T = \text{AR}(C_{\text{lift}})$. Move up the rectangles in T and the rectangle r_1 by $2/3$ and hereby move r_1 completely out of the previous packing, since $y_1 \geq 1/3$. Then move r_1 to the right by ε, this is possible, since $x_1' \leq 1 - \varepsilon$.

In the hole vacated by r_1 we have on the left side a free slot of width ε (since $w_1 \geq \varepsilon$ and since we moved r_1 to the right by ε) and height $2/3 + h_1 \geq 1$ (since we moved up T by $2/3$ and since $h_1 \geq 1/3$). Place C_1 in this slot and pack C_2 on top of the packing at height $5/3$. □



(a) Definition of T and C_{lift}.

(b) Modified packing where C_1 is packed in the slot of height 1 and width ε.

Figure 2.7: Packing methods for Lemma 2.6

Algorithm 2.3 Rectangle of height $1/3$

Requirement: Packing P that satisfies Condition 1.3.

1: Define $C_{\text{lift}} := \text{VPCE}(x_1, y_1')$ and move up $T = \text{AR}(C_{\text{lift}})$ by $2/3$.
2: Move up r_1 by $2/3$ and then by ε to the right, i.e., pack r_1 at position $(x_1 + \varepsilon, y_1 + 2/3)$.
3: Pack C_1 into the slot vacated by r_1 and pack C_2 above the entire packing.

2.4.2 No 1/3-high Rectangles Close to the Side of the Bin

Lemma 2.7. *Let $c_1 > 0$ be a constant. If the following conditions hold for P, namely*

2.1. there is no 1/3-high rectangle that intersects $[c_1\varepsilon, (c_1+1)\varepsilon] \times [0,1]$, and

2.2. we have $h(W_{1-5(c_1+1)\varepsilon}) < 1/3$,

then we can derive a packing of I into a strip of height $5/3+\varepsilon$ in additional time $\mathcal{O}(n)$.

Proof. Let $W' = W_{1-5(c_1+1)\varepsilon} \cap \mathrm{VLR}((c_1+1)\varepsilon; 0, 1)$ be the set of rectangles of width larger than $1-5(c_1+1)\varepsilon$ intersecting the vertical line $x = (c_1+1)\varepsilon$. By Condition 2.2 we have $h(W') < 1/3$.

Consider the rectangle $r_1 = \mathrm{PointR}((c_1+1)\varepsilon, 1/2)$ (if no such rectangle exists, we set r_1 as a dummy rectangle of height and width equal to 0). We have to distinguish two cases depending on this rectangle and the set W', or to be more accurate their amount of heights above and below the horizontal line at height $y = 1/2$. Therefore, let $a = 1/2 - y_1$ be the height of r_1 below $y = 1/2$ and $a' = y_1' - 1/2$ the height above $y = 1/2$. Furthermore, let $h = h(W' \cap \mathrm{VLR}((c_1+1)\varepsilon; 0, y_1))$ and $h' = h(W' \cap \mathrm{VLR}((c_1+1)\varepsilon; y_1', 1))$ be the heights of W' above and below $y = 1/2$ excluding r_1 (if $r_1 \in W'$).

Note that by Condition 2.1 the height of r_1 is $h_1 \leq 1/3$, hence it intersects at most one of the horizontal lines at height $y = 1/3$ or $y = 2/3$.

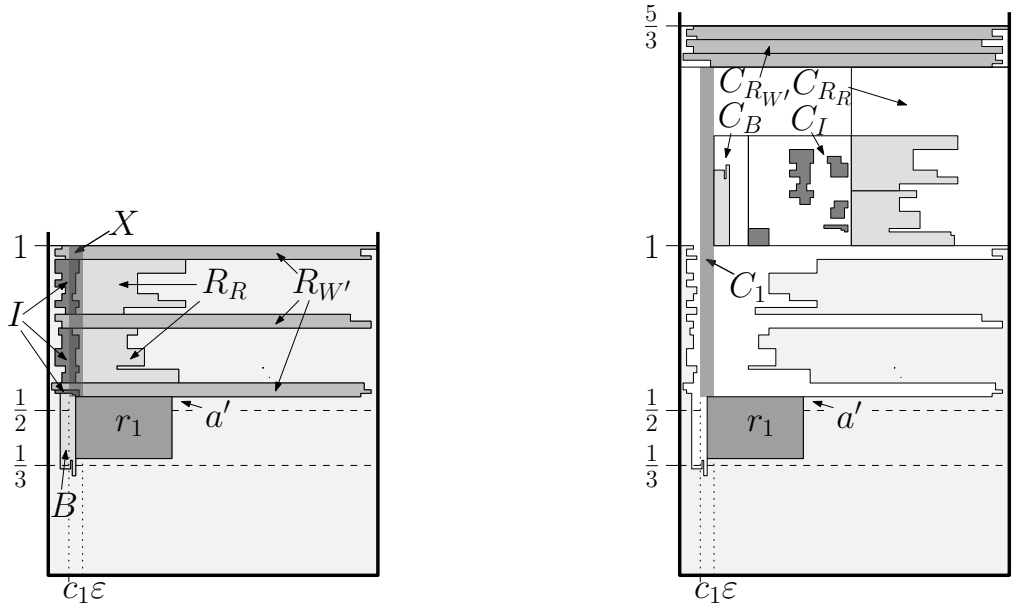
We are going to cut a slot of width ε between $c_1\varepsilon$ and $(c_1+1)\varepsilon$ down to a height y_{cut}. The value y_{cut} depends on the particular packing. So we distinguish between two cases:

1. If $a + h \leq 1/6$ or $a' + h' \leq 1/6$, we will assume w.l.o.g that $a' + h' \leq 1/6$ by otherwise mirroring the packing horizontally over $y = 1/2$. In this case we set $y_{\mathrm{cut}} = y_1'$.

2. If $a + h > 1/6$ and $a' + h' > 1/6$, we will assume w.l.o.g that $y_1 \geq 1/3$ by otherwise mirroring the packing horizontally over $y = 1/2$. Here we set $y_{\mathrm{cut}} = y_1$.

Note, if $r_1 \in W'$ it follows that we are in the first case, since $h + a + a' + h' = h(W') < 1/3$ and so $h + a < 1/6$ or $h' + a' < 1/6$. In both cases we have $y_{\mathrm{cut}} \in [1/3, 2/3]$.

Algorithm. We are going to cut a slot of width ε between $c_1\varepsilon$ and $(c_1+1)\varepsilon$ down to height y_{cut}, which is either y_1' or y_1 (hence $\mathrm{PointR}((c_1+1)\varepsilon, y_{\mathrm{cut}}) = \emptyset$). Let $X = [c_1\varepsilon, (c_1+1)\varepsilon] \times [y_{\mathrm{cut}}, 1]$ be the designated slot that we want to free. To do this we differentiate four sets of rectangles intersecting X. The entire algorithm is given in Algorithm 2.4—see Figure 2.8 for an illustration.



(a) Definition of $R_{W'}$, R_R, B and I (here in Case 1 with $a' + h' \leq 1/6$)

(b) Modified packing where C_1 is packed in the shaded slot of height 1 and width ε.

Figure 2.8: Packing methods for Lemma 2.7 (the x-direction is distorted, i.e., ε is chosen very large, to illustrate the different sets that intersect with X)

- Let $R_{W'} = \text{VLR}((c_1+1)\varepsilon; y_{\text{cut}}, 1) \cap W'$ be the set of rectangles in W' which intersect X by crossing the vertical line at width $x = (c_1+1)\varepsilon$. Notice, that if $r_1 \in W'$, then $y_{\text{cut}} = y'_1$. Therefore, $R_{W'}$ has total height h'. Place the rectangles of $R_{W'}$ into a container $C_{R_{W'}}$ of height $h' < 1/3$ and width 1 and pack it at position $(0, 5/3 - h')$.

- Let $R_R = \text{VLR}((c_1+1)\varepsilon; y_{\text{cut}}, 1) \setminus W'$ be the set of remaining rectangles intersecting X by crossing the vertical line at width $x = (c_1+1)\varepsilon$. Pack these rectangles left-aligned into a container C_{R_R} of width $1 - 5(c_1+1)\varepsilon$ and height at most $1 - y_{\text{cut}} - h'$. This container is placed at position $(5(c_1+1)\varepsilon, 1)$. This does not cause a conflict, since y_{cut} is always greater than $1/3$ and $h(R_R) + h(R_{W'}) \leq 1 - y_{\text{cut}} \leq 2/3$.

- Let $B = \text{HLR}(c_1\varepsilon, (c_1+1)\varepsilon; y_{\text{cut}})$ be the rectangles which intersect X from the bottom. Note, that there is no rectangle at position $((c_1+1)\varepsilon, y_{\text{cut}})$. By Condition 2.1, these rectangles have height at most $1/3$ and fit bottom-aligned into a container C_B of width $(c_1+1)\varepsilon$ and height $1/3$. Place C_B at position $((c_1+1)\varepsilon, 1)$.

- Let $I = \text{AR}(c_1\varepsilon, (c_1+1)\varepsilon; y_{\text{cut}}, 1) \cup \text{VLR}(c_1\varepsilon; y_{\text{cut}}, 1) \setminus (R_{W'} \cup R_R \cup B)$ be the set of remaining rectangles which are completely inside X or intersect X only from the left. This packing has total height $1 - y_{\text{cut}} \in [1/3, 2/3]$.

We want to place I between height 1 and $5/3 - h' \geq 4/3$. Therefore, packing I into a container C_I of height $1/3$ is sufficient. To do this we partition I into three sets. Let $I_1 \subseteq I$ be the subset of rectangles that intersect height $y = 2/3$ (these rectangles fit bottom-aligned into a container of size $((c_1+1)\varepsilon, 1/3)$) and let $I_2 \subseteq I$ and $I_3 \subseteq I$ be the subsets of I that lie completely above and below $y = 2/3$, respectively. By preserving the packing of I_2 and I_3 we can pack each into a container $((c_1+1)\varepsilon, 1/3)$. In total we pack I into $C_I = (3(c_1+1)\varepsilon, 1/3)$. This container is placed at position $(2(c_1+1)\varepsilon, 1)$.

In total the container C_B is placed on the right side of the slot X on height 1. Next to C_B we place the container C_I of width $3(c_1+1)\varepsilon$ at position $(2(c_1+1)\varepsilon, 1)$. Between the container C_I and the right side of the strip we have a space of $1 - 5(c_1+1)\varepsilon$ for the container C_{R_R}. The container C_B and C_I have a height of $1/3$ and C_{R_R} one of $1 - y_{\text{cut}} - h'$. Since the height of $C_{R_{W'}}$ is $h' < 1/3$ it fits on top of these containers so that the top edge of $C_{R_{W'}}$ is on height $5/3$.

Finally, we insert C_1 into the free slot X and pack C_2 above the entire packing. We have to prove that the slot has sufficient depth for C_1. The slot starts at height y_{cut} and goes up to $5/3 - h'$. Therefore, we have to check whether $5/3 - h' - y_{\text{cut}} \geq 1$.

In the first case, we have $h' + a' \leq 1/6$ and $y_{\text{cut}} = y'_1 = a' + 1/2$. Hence,

$$5/3 - h' - y_{\text{cut}} = 5/3 - h' - a' - 1/2 \geq 5/3 - 1/6 - 1/2 = 1.$$

In the second case, we have $h + a > 1/6$ and $y_{\text{cut}} = y_1 = 1/2 - a$. From our discussion above we know that $h + h' < 1/3$. Hence,

$$5/3 - h' - y_{\text{cut}} = 5/3 - h' + a - 1/2 > 5/3 - 1/3 + h + a - 1/2 \geq 5/3 - 1/3 + 1/6 - 1/2 = 1. \quad \square$$

Algorithm 2.4 No 1/3-high rectangles close to the side of the strip

Requirement: Packing P that satisfies Conditions 2.1 and 2.2.

1: Pack $R_{W'} = \text{VLR}((c_1+1)\varepsilon; y_{\text{cut}}, 1) \cap W'$ into a container $C_{R_{W'}} = (1, h')$ at position $(5/3 - h', 0)$.
2: Pack $R_R = \text{VLR}((c_1+1)\varepsilon; y_{\text{cut}}, 1) \setminus W'$ into a container $C_{R_R} = (1 - 5(c_1+1)\varepsilon, 2/3 - h')$ at position $(5(c_1+1)\varepsilon, 1)$.
3: Pack $B = \text{HLR}(c_1\varepsilon, (c_1+1)\varepsilon; y_1) \setminus (R_{W'} \cup R_R)$ into a container $C_B = ((c_1+1)\varepsilon, 1/3)$ at position $((c_1+1)\varepsilon, 1)$.
4: Pack $I = (\text{AR}(c_1\varepsilon, (c_1+1)\varepsilon; y_{\text{cut}}, 1) \cup \text{VLR}(c_1\varepsilon; y_{\text{cut}}, 1)) \setminus (R_{W'} \cup R_R \cup B)$ into a container $C_I = (3(c_1+1)\varepsilon, 1/3)$ at position $(2(c_1+1)\varepsilon, 1)$.
5: Pack C_1 into the slot X at position $(c_1\varepsilon, y_{\text{cut}})$ and pack C_2 above the entire packing.

Obviously, the methods of Lemma 2.7 can similarly be applied if there is no 1/3-high rectangle that intersects $[1 - (c_1+1)\varepsilon, 1 - c_1\varepsilon] \times [0, 1]$ at the right side of P.

2.4.3 One Special Big Rectangle in P

Lemma 2.8. *Let $c_2 > 0$ be a constant. If the following conditions hold for P, namely*

3.1. there is a rectangle r_1 of height $h_1 \in [1/3, 2/3]$ and width $w_1 \in [(4c_2+1)\varepsilon, 1]$ with $y_1 < 1/3$ and with $y_1' > 2/3$, and

3.2. we have $w(H_{2/3}) \geq 1 - w_1 - c_2\varepsilon$,

then we can derive a packing of I into a strip of height $5/3 + \varepsilon$ in additional time $O(n)$.

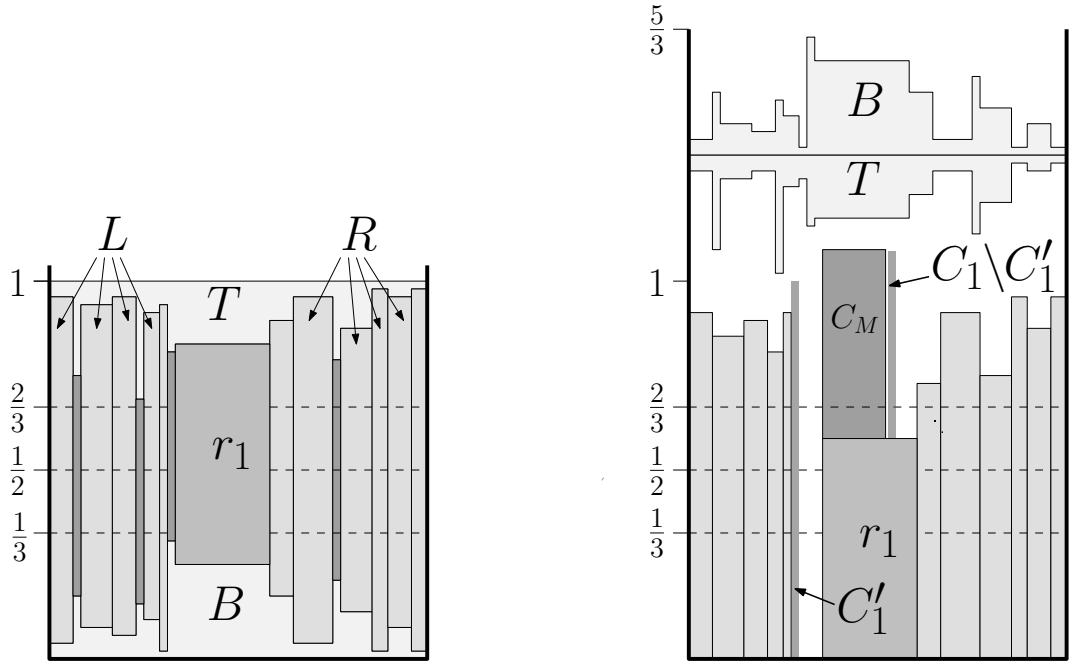
Proof. Since the height of r_1 is $h_1 \leq 2/3$ we can assume w.l.o.g. that r_1 does not intersect $y = 1/6$, i.e., $y_1 \geq 1/6$ (by otherwise mirroring over $y = 1/2$).

We want to line up all rectangles in the instance I of height greater than $h = \max(1/2, 1 - h_1)$ and the rectangle r_1 on the bottom of the strip. These rectangles fit there, since in any optimal solution they have to be placed next to each other (all rectangles of $H_h = \{r_i \mid h_i > h\}$ have to intersect the horizontal line at height $y = 1/2$ and no rectangle of H_h fits above r_1). Since $1 - h_1 \leq 2/3$, $H_{2/3}$ is included in the set H_h. See Figure 2.9 for an illustration of the following algorithm and refer to Algorithm 2.5 for a compressed description.

Let $T = \mathrm{AR}(0, 1; 2/3, 1)$ be the rectangles which lie completely above the horizontal line at height $y = 2/3$. We move up the rectangles in T by $1/3$ into the area $[0,1] \times [1, 4/3]$. Now there is a free space of height at least $1/3$ above r_1.

Let $B = \mathrm{AR}(0, 1; 0, 1/3)$ be the rectangles which lie completely below the horizontal line at height $y = 1/3$. We pack these rectangles into a container $C_B = (1, 1/3)$ by preserving the packing of B and pack C_B at position $(0, 4/3)$, i.e., directly above T. Since by assumption r_1 does not intersect the horizontal line at height $y = 1/6$, there is a free space of height at least $1/6$ below r_1.

The remaining rectangles of height smaller than h except r_1 have to intersect one of the horizontal lines at height $1/3$ or $2/3$ or lie completely between them. We denote these rectangles by $M_1 = \mathrm{HLR}(0, 1; 1/3) \setminus (H_h \cup \{r_1\})$, $M_2 = \mathrm{HLR}(0, 1; 2/3) \setminus (H_h \cup \{r_1\} \cup M_1)$ and $M_3 = \mathrm{AR}(0, 1; 1/3, 2/3)$. Since each rectangle in $H_{2/3}$ and r_1 intersects both of these lines, the rectangles in $M = M_1 \cup M_2 \cup M_3$ lie between them in slots of total width $c_2\varepsilon$. Therefore, we can pack M_1 and M_2 each bottom-aligned into a container $(c_2\varepsilon, h)$. Furthermore, the rectangles in M_3 fit into a container $(c_2\varepsilon, 1/3)$ by pushing the packing of the slots together. In total we pack M into a container $C_M = (3c_2\varepsilon, h)$ and pack it aside for the moment. After these steps we removed all rectangles of height at most h except r_1 out of the previous packing. All remaining rectangles intersect the horizontal line at height $y = 1/2$. We line up the rectangles in $L = \mathrm{HLR}(0, x_1; 1/2)$, i.e., the remaining rectangles on the left of r_1, bottom-aligned from left to right starting at position $(0, 0)$. The rectangles in $R = \mathrm{HLR}(0, x_1'; 1/2)$ (the



(a) Definition of T, B, L, R and M (dark rectangles squeezed between L and R).

(b) Modified packing where C_1' is packed in the left-hand slot of height 1 and $C_1 \setminus C_1'$ is packed in the right-hand shaded slot of height h and width ε.

Figure 2.9: Packing methods for Lemma 2.8

Algorithm 2.5 Single big rectangle of height $1/3$

Requirement: Packing P that satisfies Conditions 3.1 and 3.2.

1: Move up $T = \mathrm{AR}(0,1;2/3,1)$ by $1/3$
2: Pack the rectangles in $B = \mathrm{AR}(0,1;0,1/3)$ into a container $C_B = (1,1/3)$ at position $(0,4/3)$.
3: Pack the rectangles in $M = (\mathrm{AR}(0,1;1/3,2/3) \cup \mathrm{HLR}(0,1;1/3) \cup \mathrm{HLR}(0,1;2/3)) \setminus (H_h \cup \{r_1\})$ into a container $C_M = (3c_2, h)$.
4: Line up the rectangles in $L = \mathrm{HLR}(0, x_1; 1/2)$ on the left of r_1 starting at position $(0,0)$.
5: Line up the rectangles in $R = \mathrm{HLR}(0, x_1'; 1/2)$ on the right of r_1 starting at position $(1,0)$ from right to left.
6: Pack r_1 at position $(1 - w(R) - w_1, 0)$, by moving r_1 to the bottom of the strip and at most $c_2\varepsilon$ to the right.
7: Pack C_M and the resized container C_1 on top of r_1 into the area X'.
8: Pack the rectangles $C_1' \subseteq C_1$ of height greater than h into the slot vacated by r_1 and pack C_2 above the entire packing.

remaining rectangles on the right of r_1) are placed bottom-aligned from right to left starting at position $(1,0)$. Now move r_1 down to the ground, i.e., pack r_1 at position $(x_1,0)$. Above r_1 is a free space of height at least $1/2$, since we moved T up by $1/3$ and r_1 down by at least $1/6$. The free space has also height at least $1-h_1$, since there is no rectangle left above r_1 up to height 1. Hence, in total, this leaves us a free space of width $w_1 \geq (4c_2+1)\varepsilon$ and height h. Denote this area by $X = [x,x'] \times [h_1, h_1+h]$ with $x = x_1$ and $x' = x_1 + w_1$.

Move r_1 to the right by at most $c_2\varepsilon$ until it touches the first rectangle in R, i.e., place r_1 at position $(1-w(R)-w_1,0)$. This reduces the width of the free area on top of r_1 to $X' = [x+c_2\varepsilon, x'] \times [h_1, h_1+h]$. Note, the width of X' is still at least $(3c_2+1)\varepsilon$.

In the next step we reorganize the packing of C_1. Recall, that the rectangles in C_1 are placed bottom-aligned in that container. Let C_1' be the rectangles in C_1 of height larger than h. By removing C_1', we can resize the height of C_1 down to h. The resized container C_1 and the container C_M have both height h and total width at most $(3c_2+1)\varepsilon$. Place them on top of r_1 in the area X'.

Then place the rectangles in C_1' into the free slot on the left side of r_1. They fit there, since in any optimal packing all rectangles of height greater than h in the instance and r_1 have to be placed next to each other (all rectangles of height greater than h have to intersect the horizontal line at height $y = 1/2$ and none of them fits above r_1). Finally, pack C_2 above the entire packing at height $5/3$. □

2.4.4 Two Rectangles of Height Between 1/3 and 2/3

Lemma 2.9. *If the following conditions hold for P, namely*

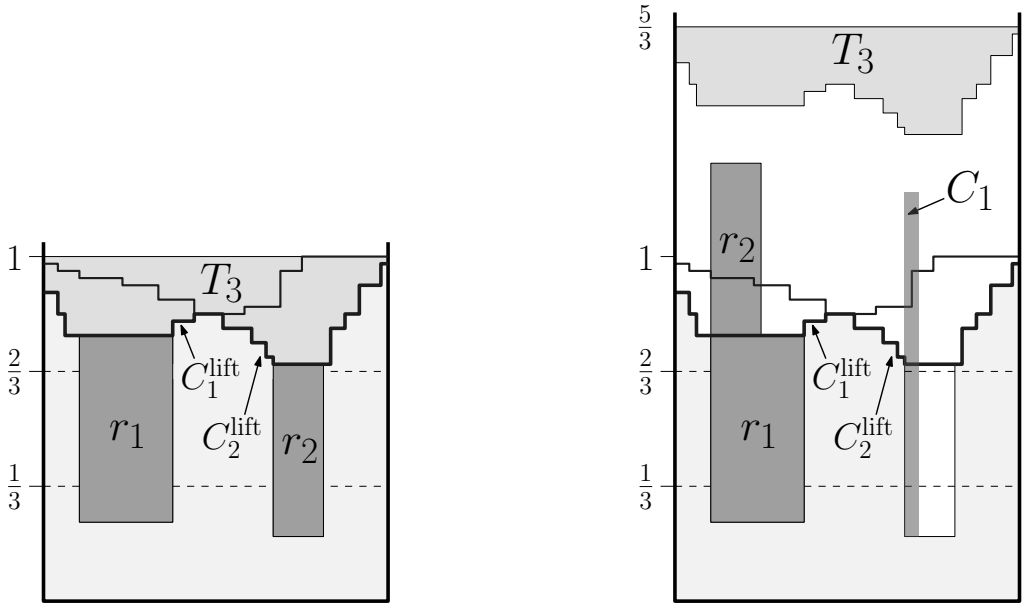
4.1. there are rectangles r_1, r_2 with heights $h_1, h_2 \in [1/3, 2/3]$ and widths $w_1, w_2 \geq \varepsilon$, and

4.2. we have $y_1 < y_2'$ and $y_2 < y_1'$.

then we can derive a packing of I into a strip of height $5/3+\varepsilon$ in additional time $\mathcal{O}(n)$.

Proof. See Figure 2.10 for an illustration of the following algorithm which is given in Algorithm 2.6. W.l.o.g. let r_1 be the wider rectangle ($w_1 \geq w_2$). Let $C_1^{\text{lift}} = \text{VPCE}(x_1', y_1')$ and $C_2^{\text{lift}} = \text{VPCE}(x_2', y_2')$ be the vertical polygonal chain extensions of the top of r_1 and r_2, respectively. Furthermore, let $T_1 = \text{AR}(C_1^{\text{lift}})$ and $T_2 = \text{AR}(C_2^{\text{lift}})$ be the rectangles above these polygons.

Note that $r_1 \notin T_2$ by Condition 4.2, since otherwise we have $y_1 \geq y_2'$. The same argument holds for the statement $r_2 \notin T_1$.



(a) Definition of T_3.

(b) Modified packing where C_1 is packed in the slot of height 1 and width ε.

Figure 2.10: Packing methods for Lemma 2.9

Let $T_3 = T_1 \cup T_2$ be the rectangles above r_1 and r_2. We move up the rectangles in T_3 by $2/3$. This leaves a free area of height $2/3$ above r_1 and r_2. We place r_2 directly above r_1 into that free area. This is possible because $w_1 \geq w_2$ and $h_2 \leq 2/3$. The hole vacated by r_2 has width $w_2 \geq \varepsilon$ and height at least 1, since $h_2 \geq 1/3$ and T_3 was moved up by $2/3$. Finally, we place C_1 into that hole and C_2 on top of the packing at height $5/3$. □

Algorithm 2.6 Two rectangles of height between $1/3$ and $2/3$

Requirement: Packing P that satisfies Conditions 4.1 and 4.2.

1: Define $C_1^{\text{lift}} = \text{VPCE}(x_1, y_1')$, $C_2^{\text{lift}} = \text{VPCE}(x_2, y_2')$, $T_1 = \text{AR}(C_1^{\text{lift}})$ and $T_2 = \text{AR}(C_2^{\text{lift}})$.
2: Move up $T_3 = T_1 \cup T_2$ by $2/3$.
3: Pack r_2 at postion (x_1, y_1').
4: Pack C_1 into the slot vacated by r_2 and pack C_2 above the entire packing.

2.4.5 Gap Between Innermost $2/3$-high Edges

The pre-conditions for this section are quite technical. We first state them formally and present a motivation afterwards. Thus assume that the following conditions on P are satisfied throughout this section for some small constant c_3 (think of $c_3 = 2$ for most cases).

5.1. There are rectangles $r_\ell, r_r \in H_{2/3}$ with x-coordinates $x'_\ell \in [4c_3\varepsilon, 1-4c_3\varepsilon]$ and $x_r \in [x'_\ell + 4c_3\varepsilon, 1-4c_3\varepsilon]$ (note that x'_ℓ refers to the right side of r_ℓ whereas x_r refers to the left side of r_r).

5.2. There is no 1/3-high rectangle that intersects with $[x'_\ell + (c_3-1)\varepsilon, x'_\ell + c_3\varepsilon] \times [0,1]$ and there is no 1/3-high rectangle that intersects with $[x_r - c_3\varepsilon, x_r - (c_3-1)\varepsilon] \times [0,1]$.

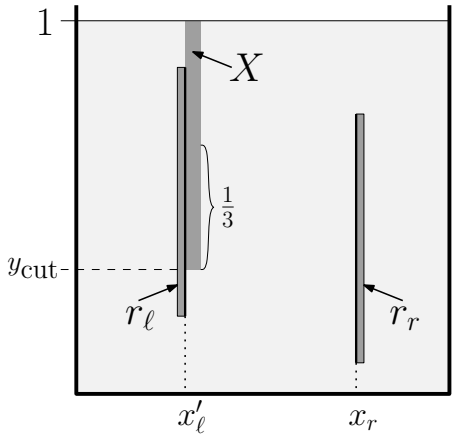
To understand the motivation for these conditions assume that Lemma 2.5 is not applicable, which reads as follows. *If there is a* 1/3*-high edge on the left side of the bin then the space to the left of this edge is almost completely occupied by* 2/3*-high rectangles*. Now we consider r_ℓ and r_r as the *innermost* such 2/3-high rectangles. By Lemma 2.5 we know that there are no further 1/3-high rectangles between x'_ℓ and x_r other than in a very thin slot next to these edges and close to the x-coordinate 1/2 (exceptions are wide 1/3-high rectangles that span across these areas—these cases are handled separately). This property is captured in Condition 5.2. For technical reasons we require $x_r \geq x'_\ell + 4c_3\varepsilon$. If this is not the case (and Lemma 2.5 is not applicable), we have $w(H_{2/3}) \geq 1 - 6c_3\varepsilon$ and can apply Lemma 2.4.

Basic algorithm. We are going to cut out a certain slot of width $c_3\varepsilon$ next to x'_ℓ. The depth of this slot depends on the particular packing P. In a first step we describe our basic algorithm and assume that we cut down to height $y_{\text{cut}} \in [1/3, 2/3]$. In a second step we show how this basic algorithm is used depending on P and prove that it actually returns a valid packing.

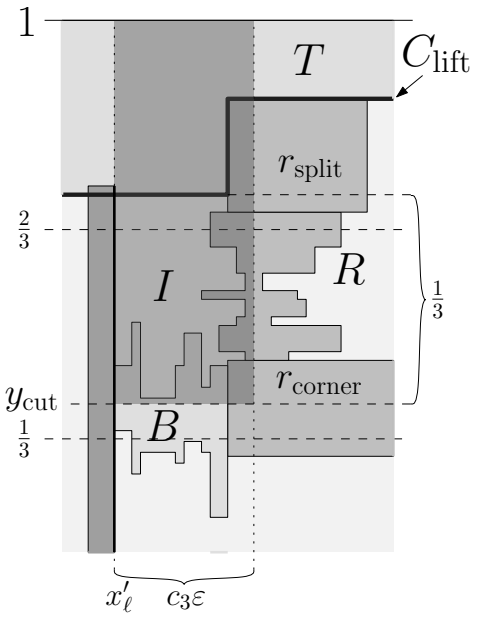
Let $X = [x'_\ell, x'_\ell + c_3\varepsilon] \times [y_{\text{cut}}, 1]$ be the designated slot that we want to free. To do this we differentiate five sets of rectangles that intersect X. The definition of these sets depends on the rectangle $r_{\text{corner}} = \text{PointR}(x'_\ell + c_3\varepsilon, y_{\text{cut}})$ and on the rectangle $r_{\text{split}} = \text{PointR}(x'_\ell + c_3\varepsilon, y_{\text{cut}} + 1/3)$, which are the rectangles that reach into X from the right at height y_{cut} and $y_{\text{cut}} + 1/3$, respectively. If no rectangle contains $(x'_\ell + c_3\varepsilon, y_{\text{cut}})$ or no rectangle contains $(x'_\ell + c_3\varepsilon, y_{\text{cut}} + 1/3)$ in its interior, we introduce dummy rectangles of size $(0,0)$ for r_{corner} and r_{split}, respectively.

One further important ingredient of the basic algorithm (or rather its correctness) is the following *blocking property*. No rectangle that intersects the designated slot X, i.e., that intersects $[x'_\ell, x'_\ell + c_3\varepsilon] \times [y_{\text{cut}}, y_{\text{cut}} + 1/3]$, reaches to the left of x'_ℓ or to the right of $x_r + c_3\varepsilon$, i.e., $\text{VLR}(x'_\ell; y_{\text{cut}}, y_{\text{cut}} + 1/3) = \emptyset$ and $\text{VLR}(x'_\ell + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3) \cap \text{VLR}(x_r + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3) = \emptyset$.

Intuitively, we think of r_ℓ and r_r as the blocking rectangles, i.e., $y_\ell, y_r \leq y_{\text{cut}}$ and $y'_\ell, y'_r \geq y_{\text{cut}} + 1/3$. Hence no rectangle intersecting X can reach beyond r_ℓ and r_r. In one special case, we cannot ensure that r_r is such a blocking rectangle, i.e., $y_{\text{cut}} + 1/3 > y'_r$. In this case we



(a) The rectangles r_ℓ and r_r enforce the blocking property in the bottom of X.



(b) Distorted close-up on all sets except $X_{1/3}$ (which consists of 1/3-high rectangles than can occur between x'_ℓ and $x'_\ell + (c_3 - 1)\varepsilon$) for $y_{\text{cut}} = 1/3$. I consists of all rectangles that lie completely inside X (the shaded region), T consists of all rectangles that lie completely above C_{lift}, rectangles in B reach into X from below, rectangles in R reach into X from the right.

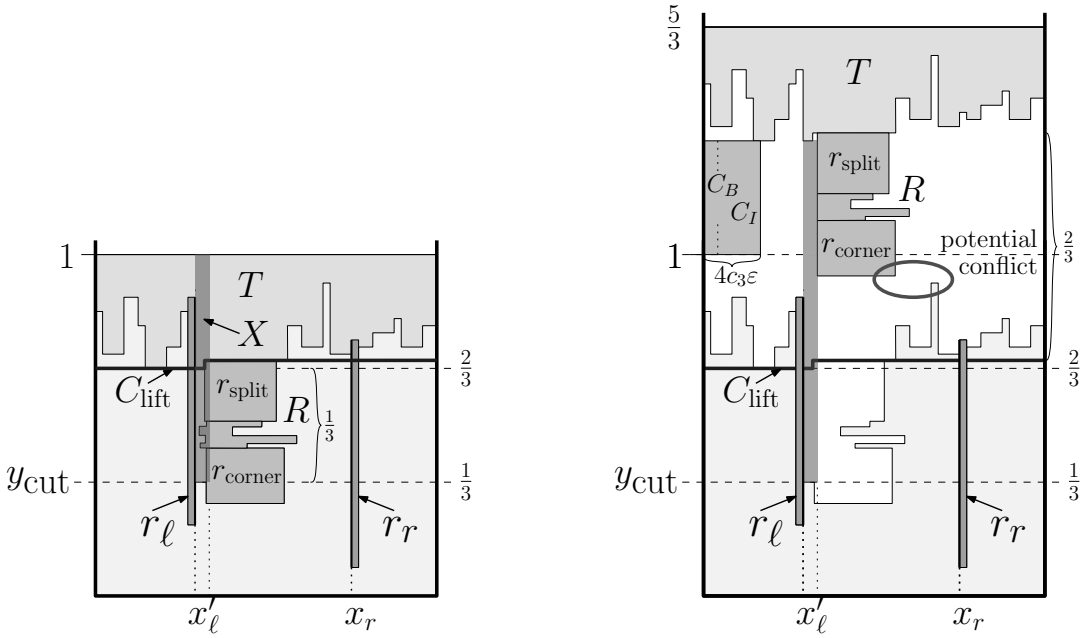
Figure 2.11: Blocking property and definition of sets that intersect X.

need the additional area of width $c_3\varepsilon$ to the right of x_r. See Figure 2.11(a) for an illustration of this property, Figure 2.11(b) for an illustration of the different sets of rectangles that intersect X and Figure 2.12 for an illustration of the following basic algorithm.

For the moment we assume that the blocking property is satisfied for y_{cut} and thus no rectangle reaches into X from the left below $y_{\text{cut}} + 1/3$.

- Let $X_{1/3} = \text{AR}(x'_\ell, x'_\ell + c_3\varepsilon; 0, 1) \cap H_{1/3}$ be the set of 1/3-high rectangles that lie completely within $c_3\varepsilon$ distance to the right of x'_ℓ. By Condition 5.2, the total width that the rectangles of $X_{1/3}$ occupy in the packing is bounded by $(c_3 - 1)\varepsilon$. Therefore, we can remove these rectangles and pack them into a container $C_{X_{1/3}} = ((c_3 - 1)\varepsilon, 1)$ by preserving the packing of the rectangles in $X_{1/3}$. We put $C_{X_{1/3}}$ aside for later insertion into the free slot X together with C_1.

- Let $B = \text{HLR}(x'_\ell, x_{\text{corner}}; y_{\text{cut}}) \setminus (X_{1/3} \cup \{r_{\text{corner}}\})$ be the set of remaining rectangles that intersect the *bottom* of X excluding r_{corner}. Pack B bottom-aligned into a container



(a) Starting situation with $y_{\text{cut}} = 1/3$.

(b) Generated packing where C_1 and $C_{X_{1/3}}$ are packed into the shaded slot of height 1 and width $c_3\varepsilon$. Since r_{corner} reaches below 1 in this case, a conflict might occur between r_{corner} and a rectangle that lies to the right of r_{split} and reaches into T. In the case shown here, r_{corner} is moved to the right of r_{split} to resolve the potential conflict.

Figure 2.12: The basic algorithm

Algorithm 2.7 Basic algorithm

Requirement: Packing P that satisfies Conditions 5.1 and 5.2; $y_{\text{cut}} \in [1/3, 2/3]$ that satisfies the blocking property

1: Remove the rectangles $X_{1/3} = \text{AR}(x'_\ell, x'_\ell + c_3\varepsilon; 0, 1) \cap H_{1/3}$ and pack them into a container $C_{X_{1/3}} = ((c_3 - 1)\varepsilon, 1)$.
2: Pack $B = \text{HLR}(x'_\ell, x_{\text{corner}}; y_{\text{cut}}) \setminus (X_{1/3} \cup \{r_{\text{corner}}\})$ into a container $C_B = (c_3\varepsilon, 1/3)$ at position $(0, 1)$.
3: Pack $I = \text{AR}(X) \setminus X_{1/3}$ into a container $C_I = (3c_3\varepsilon, 1/3)$ at position $(c_3\varepsilon, 1)$.
4: Move up the rectangles $T = \text{AR}(C_{\text{lift}}) \setminus I$ by $2/3$.
5: Move up the rectangles $R = \text{VLR}(x'_\ell + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3)$ by $2/3$ and left-align them with $x'_\ell + c_3\varepsilon$.
6: Resolve potential conflicts.
7: Pack C_1 and $C_{X_{1/3}}$ into the slot X at position $(x'_\ell, y_{\text{cut}})$ and pack C_2 above the entire packing.

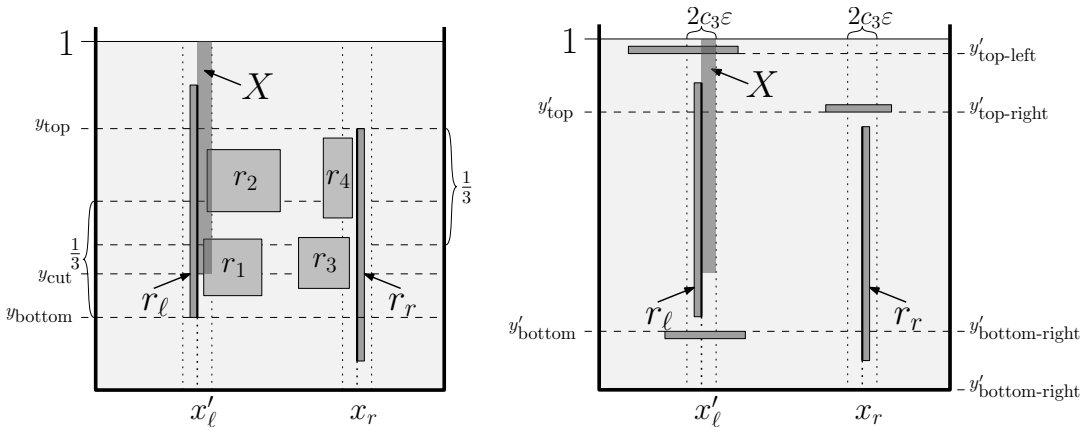
$C_B = (c_3\varepsilon, 1/3)$. This is possible since $x_{\text{corner}} - x'_\ell < c_3\varepsilon$ and the $1/3$-high rectangles have been removed before. Place this container at the left side of the strip above the current packing at position $(0, 1)$.

- Let $I = \text{AR}(X) \setminus X_{1/3}$ be the set of remaining rectangles that lie completely *inside* of X. There are no $1/3$-high rectangles in I due to the removal of $X_{1/3}$ but the packing has a total height of $1 - y_{\text{cut}} \in [1/3, 2/3]$. We use a standard method to repack I into a container of height $1/3$ as follows. Let $I_1 \subseteq I$ be the subset of rectangles that intersect height $y = 2/3$ (these rectangles can be bottom-aligned to fit into $(c_3\varepsilon, 1/3)$) and let $I_2 \subseteq I$ and $I_3 \subseteq I$ be the subsets of I that lie completely above or below $y = 2/3$, respectively. By preserving the packing of I_2 and I_3 we can pack I into $C_I = (3c_3\varepsilon, 1/3)$. Place this container next to C_B at position $(c_3\varepsilon, 1)$. The container does not intersect with the space above the designated slot X since the combined width of C_B and C_I is $4c_3\varepsilon$ and $x'_\ell \geq 4c_3\varepsilon$ by Condition 5.1.

- Consider the contour C_{lift} defined by height $y = y_{\text{cut}} + 1/3$ to the left of r_{split} and by the top of r_{split} to the right. More formally, let $C_{\text{lift}} = \text{PL}((0, y_{\text{cut}} + 1/3), (x_{\text{split}}, y_{\text{cut}} + 1/3), (x_{\text{split}}, y'_{\text{split}}),$
  $(1, y'_{\text{split}}))$. Let $T = \text{AR}(C_{\text{lift}}) \setminus I$ be the set of rectangles that lie completely above this contour but not in I. Move T up by $2/3$. This does not cause an overlap with the containers C_B and C_I since C_{lift} lies completely above $2/3$ (as $y_{\text{cut}} \geq 1/3$) and thus the lowest rectangle in T reaches a final position above $4/3$.

- Let $R = \text{VLR}(x'_\ell + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3)$ be the set of rectangles that intersect with the *right* side of X up to the crucial height of $y_{\text{cut}} + 1/3$. Move R up by $2/3$ and then left-align all rectangles at x-coordinate $x'_\ell + c_3\varepsilon$. This is the sole operation in the basic algorithm that might cause a conflict. This potential conflict only affects r_{corner} and we will later see how to overcome this difficulty. All other rectangles were entirely above y_{cut} in the original packing and are thus moved above height 1. Therefore, they cannot overlap with any rectangle inside the original packing P. Since the blocking property ensures that no rectangle of R has width greater than $x_r + c_3\varepsilon - x'_\ell$ and $x_r \leq 1 - 4c_3\varepsilon$ (by Condition 5.1) we can left-align all rectangles at x-coordinate $x'_\ell + c_3\varepsilon$ without any rectangle intersecting the right side of the strip.

Finally, after resolving the potential conflict from the last step, we insert C_1 and $C_{X_{1/3}}$ into the slot X at position $(x'_\ell, y_{\text{cut}})$ (they fit since $w(C_1) + w(C_{X_{1/3}}) \leq c_3\varepsilon$ and all rectangles in T lie above $y_{\text{cut}} + 1/3 + 2/3 = y_{\text{cut}} + 1$) and pack C_2 above the entire packing as always. See Algorithm 2.7 for the complete basic algorithm.



(a) Definition of y_{top}, y_{bottom}, r_1, r_2, r_3 and r_4 and accentuation of the designated slot X next to r_ℓ. Here r_1 intersects with 1/3 and $w_1 + w_2 \le 1 - x'_\ell - c_3\varepsilon$ and thus $y_{\text{cut}} = 1/3$.

(b) Definition of y'_{top} and y'_{bottom}. No rectangle from X below y'_{top} reaches beyond $x_r + c_3\varepsilon$.

Figure 2.13: Notations

We described the basic algorithm in a way that it always cuts down from the top of the packing next to r_ℓ. But there are four potential cuts since we can also cut next to r_r or from below. To ease the presentation we will stick to cutting next to r_ℓ from above by otherwise mirroring the packing horizontally and/or vertically.

Now let us see how to invoke the basic algorithm such that the blocking property is satisfied for y_{cut} and how to resolve the potential conflict.

Let $y_{\text{top}} = \min(y'_\ell, y'_r) > 2/3$ and let $y_{\text{bottom}} = \max(y_\ell, y_r) < 1/3$. We get the rectangles

$$r_1 = \text{PointR}(x'_\ell + c_3\varepsilon, y_{\text{top}} - 1/3),$$
$$r_2 = \text{PointR}(x'_\ell + c_3\varepsilon, y_{\text{bottom}} + 1/3),$$
$$r_3 = \text{PointR}(x_r - c_3\varepsilon, y_{\text{top}} - 1/3), \text{ and}$$
$$r_4 = \text{PointR}(x_r - c_3\varepsilon, y_{\text{bottom}} + 1/3)$$

as some potential corner pieces of the cut (see Figure 2.13(a)), corresponding to rectangle r_{corner} in the basic algorithm. Note that the rectangles r_1, r_2, r_3 and r_4 do not necessarily have to differ or to exist (while in the latter case we again introduce a dummy rectangle of size $(0,0)$) . By Condition 5.2 we know that $h_1, h_2, h_3, h_4 \le 1/3$. In the following cases we set $y_{\text{cut}} \in [1/3, y_{\text{top}} - 1/3]$. For this value of y_{cut} the blocking property is satisfied since $y_\ell, y_r \le 1/3 \le y_{\text{cut}}$ and $y'_\ell, y'_r \ge y_{\text{top}} \ge y_{\text{cut}} + 1/3$. Thus $\text{VLR}(x'_\ell; y_{\text{cut}}, y_{\text{cut}} + 1/3) \subseteq \text{VLR}(x'_\ell; y_\ell, y'_\ell) = \emptyset$ and $\text{VLR}(x'_\ell + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3) \cap \text{VLR}(x_r + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3) \subseteq \text{VLR}(x'_\ell + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} +$

$1/3) \cap \mathrm{VLR}(x_r; y_r, y'_r) \subseteq \mathrm{VLR}(x_r; y_r, y'_r) = \emptyset$ (no rectangle from $[x'_\ell, x'_\ell + c_3\varepsilon] \times [y_{\mathrm{cut}}, y_{\mathrm{cut}} + 1/3]$ reaches beyond x'_ℓ and x_r).

We now describe the different cases in which we invoke the basic algorithm.

Case 1. $y_1 \geq 1/3$, i.e., r_1 lies above height $1/3$.
In this case we invoke the basic algorithm with $y_{\mathrm{cut}} = y_1 \in [1/3, y_{\mathrm{top}} - 1/3]$ (hence the blocking property is satisfied). We have $r_{\mathrm{corner}} = (0,0)$ and r_1 is the lowest rectangle in R. The rectangle r_1 is moved above height 1 since $y_1 \geq 1/3$. Thus no conflict occurs.

In the following assume conversely that $y_1, y_3 < 1/3$ and $y'_2, y'_4 > 2/3$ (using mirroring). This implies that r_1, r_3 intersect the horizontal line at height $y = 1/3$ and r_2, r_4 intersect the horizontal line at height $y = 2/3$ (by definition of the potential corner pieces and as $y_{\mathrm{top}} - 1/3 > 1/3$ and $y_{\mathrm{bottom}} + 1/3 < 2/3$). Hence, we have $r_1 \neq r_2$ and $r_3 \neq r_4$ since $h_1, h_2, h_3, h_4 \leq 1/3$.

Case 2. $w_1 + w_2 \leq 1 - x'_\ell - c_3\varepsilon$ (and r_1 intersect $y = 1/3$, r_2 intersects $y = 2/3$).
In this case we potentially mirror the packing horizontally, i.e., over $y = 1/2$, to ensure that $h_1 \leq h_2$. We invoke the basic algorithm with $y_{\mathrm{cut}} = 1/3$ (hence the blocking property is satisfied) and thus we have $r_{\mathrm{corner}} = r_1$ and $r_{\mathrm{split}} = r_2$ (as we can assume from the previous case that r_1 intersects $y = 1/3$ and r_2 intersects $y = 2/3$). Since r_1 intersects the horizontal line $y = 1/3$ it is not moved out of the original packing P (and could therefore cause a conflict with the original packing). Since $w_1 + w_2 \leq 1 - x'_\ell - c_3\varepsilon$ and $h_1 \leq h_2$ we can pack r_1 to the right of r_2 which is left-aligned at x-coordinate $x'_\ell + c_3\varepsilon$, i.e., pack r_1 at position $(x'_\ell + c_3\varepsilon + w_2, y_2 + 2/3)$. This handles the potential conflict of the basic algorithm.

In the following we assume that $w_1 + w_2 > 1 - x'_\ell - c_3\varepsilon$ and accordingly $w_3 + w_4 > x_r - c_3\varepsilon$. Thus $\sum_{i=1}^{4} w_i > 1 + x_r - x'_\ell - 2c_3\varepsilon > 2(x_r - x'_\ell)$. This is obviously only possible if $r_1 = r_3$ or $r_2 = r_4$. Let us thus assume $r_2 = r_4$ (by otherwise mirroring the packing over $y = 1/2$).

Case 3. $w_1 \leq x_r - x'_\ell - 2c_3\varepsilon$ (and $r_2 = r_4$ and r_1 intersects $y = 1/3$ and r_2 intersects $y = 2/3$).
Again we invoke the basic algorithm with $y_{\mathrm{cut}} = 1/3$ (hence the blocking property is satisfied) and accordingly we have $r_{\mathrm{corner}} = r_1$ and $r_{\mathrm{split}} = r_2$. All rectangles above r_2 are in T, hence by moving up T by $2/3$ no rectangle intersects the area above r_2, that is, in particular, the area $[x'_\ell + c_3\varepsilon, x_r - c_3\varepsilon] \times [y'_2, 1]$. Furthermore, since r_2 intersects the horizontal line $y = 2/3$, except r_1 no rectangle is placed in $[x'_\ell + c_3\varepsilon, x_r - c_3\varepsilon] \times [2/3, 1]$ after moving up R by $2/3$. The rectangle r_1 has height at most $1/3$ and width at most $x_r - x'_\ell - 2c_3\varepsilon$. Hence by moving

up r_1 by $2/3$ and left-aligning it at x-coordinate $x'_\ell + c_3\varepsilon$ it intersects only with the free area $[x'_\ell + c_3\varepsilon, x_r - c_3\varepsilon] \times [2/3, 1]$ inside the original packing P. Thus no conflict occurs.

On the other hand, if conversely $w_1 > x_r - x'_\ell - 2c_3\varepsilon$ and accordingly $w_3 > x_r - x'_\ell - 2c_3\varepsilon$ we have $r_1 = r_3$ since $x_r \geq x'_\ell + 4c_3\varepsilon$ by Condition 5.1.

Thus for the last case we have $r_1 = r_3$ and $r_2 = r_4$ and r_1 intersects height $y = 1/3$ and r_2 intersects height $y = 2/3$. The challenge in this remaining case is that we cannot move r_1 out of the original packing (since it intersects $y = 1/3$) and thus there might occur a conflict close to r_r. We now show how to resolve this potential conflict close to r_r.

Two wide corner pieces. Let $y'_{\text{top-left}}$ be the height of the bottom of the lowest rectangle above r_ℓ that intersects $x'_\ell - c_3\varepsilon$ and $x'_\ell + c_3\varepsilon$, i.e., $y'_{\text{top-left}} = \min\{y_i \mid r_i \in \text{VLR}(x'_\ell - c_3\varepsilon; y'_\ell, 1) \cap \text{VLR}(x'_\ell + c_3\varepsilon; y'_\ell, 1)\}$. If there is no such rectangle let $y'_{\text{top-left}} = 1$. Let $y'_{\text{top-right}}$, $y'_{\text{bottom-left}}$ and $y'_{\text{bottom-right}}$ be defined accordingly as shown in Figure 2.13(b). Now we define $y'_{\text{top}} = \min(y'_{\text{top-left}}, y'_{\text{top-right}})$ and $y'_{\text{bottom}} = \max(y'_{\text{bottom-left}}, y'_{\text{bottom-right}})$. Let us assume that $y'_{\text{top}} = y'_{\text{top-right}}$ (by otherwise mirroring over $x = 1/2$).

Case 4. $y_1 \geq y'_{\text{top}} - 2/3$ (and $r_1 = r_3$ and $r_2 = r_4$ and r_1 intersects $y = 1/3$ and r_2 intersects $y = 2/3$).
In this case we invoke the basic algorithm with $y_{\text{cut}} = 1/3$ as usual (hence the blocking property is satisfied) and again we have $r_{\text{corner}} = r_1$ and $r_{\text{split}} = r_2$. Let r_5 be the rectangle that defined $y'_{\text{top}} = y'_{\text{top-right}} = y_5$. Then the rectangles r_2 and r_5 intersect the vertical line $x = x_r - c_3\varepsilon$. Since $y'_{\text{top}} \geq \min(y'_\ell, y'_r) > 2/3$ and r_2 intersects the horizontal line $y = 2/3$ and since r_2 and r_5 intersect the same vertical line, it follows that $y_5 = y'_{\text{top}} \geq y'_2$. So r_5 and all rectangles above r_5 are in T and moved up by $2/3$. Therefore, no rectangle intersects with the area $[x'_\ell + c_3\varepsilon, x_r + c_3\varepsilon] \times [y'_{\text{top}}, 1]$. Since $y_1 \geq y'_{\text{top}} - 2/3$, we move up r_1 above y'_{top} and into this area. Thus no conflict occurs. So in the following assume conversely that $y_1 < y'_{\text{top}} - 2/3$ and accordingly $y'_2 > y'_{\text{bottom}} + 2/3$.

Case 5. $y_1 < y'_{\text{top}} - 2/3$ and $y'_2 > y'_{\text{bottom}} + 2/3$.
Now assume that $y_{\text{bottom}} = y_\ell$ (by otherwise mirroring vertically—so $y'_{\text{top}} = y'_{\text{top-right}}$ does not necessarily hold any more). Note that we refer to the original definition of y_{bottom} instead of y'_{bottom} here. We invoke the basic algorithm using $y_{\text{cut}} = y'_1$. So r_1 is left in the original position (and we have $r_{\text{corner}} = (0, 0)$) and since $y'_1 > 1/3$ all rectangles from R are moved out of the original packing. It remains to verify the blocking property for $y_{\text{cut}} = y'_1$, since y'_1 is not necessarily in $[1/3, y_{\text{top}} - 1/3]$.

We have $y'_\ell > y_\ell + 2/3 = y_{\text{bottom}} + 2/3 \geq y'_1 + 1/3 = y_{\text{cut}} + 1/3$ (since $y'_1 \leq y_{\text{bottom}} + 1/3$ as by definition r_2 intersects with $y_{\text{bottom}} + 1/3$ and $r_1 \neq r_2$). So the blocking property is enforced by r_ℓ to the left, i.e., $\text{VLR}(x'_\ell; y_{\text{cut}}, y_{\text{cut}} + 1/3) = \emptyset$. Moreover, we have $y_{\text{cut}} + 1/3 < y'_{\text{top}}$ since $y_{\text{cut}} + 1/3 = y'_1 + 1/3 = y_1 + h_1 + 1/3 \leq y_1 + 2/3 < y'_{\text{top}}$. Thus by definition of y'_{top} no rectangle that intersects $x_r - c_3\varepsilon$ between y'_r and y'_{top} reaches beyond $x_r + c_3\varepsilon$, i.e., $\text{VLR}(x'_\ell + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3) \cap \text{VLR}(x_r + c_3\varepsilon; y_{\text{cut}}, y_{\text{cut}} + 1/3) = \emptyset$. So the blocking property is also satisfied for the right side.

In total we get the following lemma.

Lemma 2.10. *Let $c_3 > 0$ be a constant. If the following conditions hold for P, namely*

5.1. *there are rectangles $r_\ell, r_r \in H_{2/3}$ with x-coordinates $x'_\ell \in [4c_3\varepsilon, 1 - 4c_3\varepsilon]$ and $x_r \in [x'_\ell + 4c_3\varepsilon, 1 - 4c_3\varepsilon]$, and*

5.2. *there is no $1/3$-high rectangle that intersects with $[x'_\ell + (c_3 - 1)\varepsilon, x'_\ell + c_3\varepsilon] \times [0, 1]$ and there is no $1/3$-high rectangle that intersects with $[x_r - c_3\varepsilon, x_r - (c_3 - 1)\varepsilon] \times [0, 1]$,*

then we can derive a packing of I into a strip of height $5/3 + \varepsilon$ in additional time $\mathcal{O}(n \log n)$.

We use the same methods, namely the basic algorithm invoked with $y_{\text{cut}} = 1/3$ and $c_3 = 2$ for another case where we do not have a blocking edge of height $2/3$ on both sides. More specifically, we get the following corollary where the right-hand blocking rectangle r_r is $1/3$-high.

Corollary 2.2. *If the following conditions hold for P, namely*

5.3. *there is a rectangle $r_\ell \in H_{2/3}$ with x-coordinate $x'_\ell \in [8\varepsilon, 1/2 - 9\varepsilon]$, and*

5.4. *there is a rectangle $r_r \in H_{1/3}$ that intersects $y = 1/3$ and $y = 2/3$ with x-coordinate $x_r \in [1/2 - \varepsilon, 1/2 + \varepsilon]$, and*

5.5. *there is no $1/3$-high rectangle that intersects with $[x'_\ell + \varepsilon, x'_\ell + 2\varepsilon] \times [0, 1]$,*

then we can derive a packing of I into a strip of height $5/3 + \varepsilon$ in additional time $\mathcal{O}(n \log n)$.

Proof. As stated above, we invoke the basic algorithm with $y_{\text{cut}} = 1/3$ and $c_3 = 2$. Note that $x_r \geq x'_\ell + 8\varepsilon = x'_\ell + 4c_3\varepsilon$. The blocking property is satisfied, since r_ℓ and r_r intersects with the horizontal lines at height $y = 1/3$ and $y = 2/3$. If $r_{\text{corner}} = (0, 0)$ or $r_{\text{split}} = (0, 0)$ we can use the same methods as in Case 1. Otherwise r_{corner} intersects $y = 1/3$ and r_{split} intersects $y = 2/3$. Since r_r also intersects $y = 1/3$ and $y = 2/3$ we have $w_{\text{corner}} \leq x_r - x'_\ell$ and $w_{\text{split}} \leq x_r - x'_\ell$. Thus $w_{\text{corner}} + w_{\text{split}} \leq 2x_r - 2x'_\ell \leq 1 + 2\varepsilon - 2x'_\ell < 1 - x'_\ell - 4\varepsilon = 1 - x'_\ell - 2c_3\varepsilon$. Hence we can use the same methods as in Case 2. □

2.5 Algorithm Covers All Cases

In this section we prove that our Algorithm 2.1 (stated on page 13) indeed covers all the cases. Recall that $\varepsilon < 1/(28 \cdot 151) = 1/4228$. Suppose (after the inapplicability of Lemma 2.3 and Lemma 2.4),

$$h(W_{1-130\varepsilon}) < 1/3 \qquad \text{and} \tag{2.3}$$

$$w(H_{2/3}) < 27/28. \tag{2.4}$$

Consider the intervals $I_\ell = [0, x'_\ell + \varepsilon]$, $I_M = [1/2 - \varepsilon, 1/2 + \varepsilon]$ and $I_r = [x_r - \varepsilon, 1]$, where x'_ℓ and x_r refer to the rectangles defined in line 6 of the algorithm. From the inapplicability of Algorithm 2.2 (Lemma 2.5) on rectangles r_ℓ and r_r follows that the intervals I_ℓ and I_r are almost occupied with 2/3-high rectangles. To be more precise we have $w(\mathrm{AR}(0, x'_\ell; 0, 1) \cap H_{2/3}) \geq x'_\ell - \varepsilon$ and $w(\mathrm{AR}(x_r, 1; 0, 1) \cap H_{2/3}) \geq 1 - x_r - \varepsilon$. Furthermore, the x-coordinates of the sides of all 1/3-high rectangles are in I_ℓ, I_M or I_r, since otherwise we could apply Algorithm 2.2 (Lemma 2.5) on this rectangle. To put it in another way the rectangles in $H_{1/3}$ are either completely inside one of these intervals or span across one interval to another.

If the algorithm reaches line 6 it is not possible that a 2/3-high rectangle r_1 spans from I_ℓ to I_r, as otherwise we have $w(H_{2/3}) \geq w(\mathrm{AR}(0, x'_\ell; 0, 1) \cap H_{2/3}) + w(\mathrm{AR}(x_r, 1; 0, 1) \cap H_{2/3}) + w_1 \geq x'_\ell - \varepsilon + 1 - x_r - \varepsilon + x_r - x'_\ell - 2\varepsilon \geq 1 - 4\varepsilon > 27/28$ for $\varepsilon < 1/112$. The same holds if there were two 2/3-high rectangles r_1, r_2, that span from I_ℓ to I_M and I_M to I_r, respectively ($w(H_{2/3}) \geq w(\mathrm{AR}(0, x'_\ell; 0, 1) \cap H_{2/3}) + w(\mathrm{AR}(x_r, 1; 0, 1) \cap H_{2/3}) + w_1 + w_2 \geq x'_\ell - \varepsilon + 1 - x_r - \varepsilon + x_r - x'_\ell - 4\varepsilon \geq 1 - 6\varepsilon > 27/28$ for $\varepsilon < 1/168$).

If there is a 2/3-high rectangle r that intersects with $x = x'_\ell + \varepsilon$, i.e., r spans from I_ℓ to I_M, then we redefine r_ℓ as the rightmost 2/3-high rectangle in I_M, or $r_\ell = r$ if there is no 2/3-high rectangle completely in I_M. On the other hand, if there is a rectangle r that intersects with $x = x_r - \varepsilon$, i.e., r spans from I_M to I_r, then we redefine r_r as the leftmost 2/3-high rectangle completely in I_M, or $r_r = r$ if no 2/3-high rectangle is completely in I_M.

P now (after line 6 of the algorithm) has the following properties.

- The areas to the left of r_ℓ and to the right of r_r are almost completely covered by 2/3-high rectangles, i.e., $w(\mathrm{AR}(0, x'_\ell; 0, 1) \cap H_{2/3}) > x'_\ell - 4\varepsilon$ and $w(\mathrm{AR}(x_r, 1; 0, 1) \cap H_{2/3}) > 1 - x_r - 4\varepsilon$.

- The x-coordinates of the sides of all 1/3-high rectangles are in I_ℓ, I_M or I_r.

- We have $x_r - x'_\ell > 143\varepsilon$, since otherwise $w(H_{2/3}) \geq w(\mathrm{AR}(0, x'_\ell; 0, 1) \cap H_{2/3}) + w(\mathrm{AR}(x_r, 1; 0, 1) \cap H_{2/3}) \geq x'_\ell - 4\varepsilon + 1 - x_r - 4\varepsilon \geq 1 - 151\varepsilon \geq 27/28$ for an $\varepsilon < 1/(28 \cdot 151)$.

The first property follows from the inapplicability of Algorithm 2.2 (Lemma 2.5) and the observation that only uncovered area of total width 3ε in $[x'_\ell, x'_\ell + \varepsilon]$ (for the now outdated value of x'_ℓ) and $[1/2 - \varepsilon, 1/2 + \varepsilon]$ can be added if we redefine r_ℓ and/or r_r. Let $c_3 = 2$ if $x'_\ell < 1/2 - 3\varepsilon$ and $x_r > 1/2 + 3\varepsilon$ and $c_3 = 5$ otherwise. The intention of this definition is that $[x'_\ell + (c_3 - 1)\varepsilon, x'_\ell + c_3\varepsilon]$ does not intersect with $I_\ell \cup I_M$ and $[x_r - c_3\varepsilon, x_r - (c_3 - 1)\varepsilon]$ does not intersect with $I_M \cup I_r$ (here we use $x_r - x'_\ell > 143\varepsilon$ as thus if I_ℓ lies close to I_M we have a bigger gap between I_M and I_r and vice versa). Since the x-coordinates of the sides of all $1/3$-high rectangles are in I_ℓ, I_M and I_r we thus get the following property for P.

- If a $1/3$-high rectangle intersects with $[x'_\ell + (c_3 - 1)\varepsilon, x'_\ell + c_3\varepsilon] \times [0, 1]$, then it has to cross the vertical line at $x = x'_\ell + c_3\varepsilon$.
- If a $1/3$-high rectangle intersects with $[x_r - c_3\varepsilon, x_r - (c_3 - 1)\varepsilon] \times [0, 1]$, then it has to cross the vertical line at $x = x_r - c_3\varepsilon$.

Now assume that $x'_\ell \leq 4c_3\varepsilon$ and no $1/3$-high rectangle intersects with $x = x'_\ell + c_3\varepsilon$. Thus no $1/3$-high rectangle spans across I_ℓ and I_M and the precondition of Lemma 2.7 with $c_1 = 5c_3$ is satisfied (we have $h(W_{1-5(c_1+1)\varepsilon}) = h(W_{1-5(5c_3+1)\varepsilon}) \leq h(W_{1-130\varepsilon}) < 1/3$ by Condition 2.3). We use Algorithm 2.4 (Lemma 2.7) to derive a packing into a strip of height $5/3 + \varepsilon$ which we return. For a packing P that is still not processed we get the following property.

- If no $1/3$-high rectangle intersects with $x = x'_\ell + c_3\varepsilon$, then $x'_\ell \geq 4c_3\varepsilon$ and analogously if no $1/3$-high rectangle intersects with $x = x_r - c_3\varepsilon$, then $x_r \leq 1 - 4c_3\varepsilon$.

The specific method that we apply in the next step depends on the existence of $1/3$-high rectangles that span across the intervals I_ℓ, I_M and I_r. See Figure 2.3 for a schematic illustration of the following four cases (by the considerations above, all $1/3$-high rectangles that span across the intervals have height at most $2/3$).

- A $1/3$-high rectangle reaches close to r_ℓ and r_r—see Figure 2.3(a).

  In this case we assume that there is a $1/3$-high rectangle r_1 that intersects with $x = x'_\ell + \varepsilon$ and with $x = x_r - \varepsilon$, i.e., that spans from I_ℓ to I_r. By Inequality (2.3) we have $w_1 \leq 1 - 130\varepsilon$ as $h_1 > 1/3$. Moreover, we have $w_1 \geq x_r - \varepsilon - x'_\ell - \varepsilon \geq 141\varepsilon$ (since $x_r - x'_\ell > 143\varepsilon$). Thus if $y_1 \geq 1/3$ or $y'_1 \leq 2/3$ we can apply Algorithm 2.3 (Lemma 2.6). Otherwise, we can apply Algorithm 2.5 (Lemma 2.8) with $c_2 = 10$ since $w_1 \geq x_r - \varepsilon - x'_\ell - \varepsilon \geq 141\varepsilon > (4c_2 + 1)\varepsilon$ (since $x_r - x'_\ell > 143\varepsilon$) and $w(H_{2/3}) \geq$

$w(\mathrm{AR}(0, x'_\ell; 0, 1) \cap H_{2/3}) + w(\mathrm{AR}(x_r, 1; 0, 1) \cap H_{2/3}) \geq x'_\ell - 4\varepsilon + 1 - x_r - 4\varepsilon \geq 1 - w_1 - 10\varepsilon = 1 - w_1 - c_2\varepsilon$.

In the following we also need to handle the case where r_1 reaches only close to the blocking rectangles r_ℓ and r_r, i.e., r_1 intersects with $x'_\ell + 11\varepsilon$ and $x_r - 11\varepsilon$. Here we can also apply Algorithm 2.3 (Lemma 2.6) or Algorithm 2.5 (Lemma 2.8) with $c_2 = 30$ ($w_1 \geq x_r - 11\varepsilon - x'_\ell - 11\varepsilon \geq 121\varepsilon = (4c_2 + 1)\varepsilon$ and $w(H_{2/3}) \geq w(\mathrm{AR}(0, x'_\ell; 0, 1) \cap H_{2/3}) + w(\mathrm{AR}(x_r, 1; 0, 1) \cap H_{2/3}) \geq x'_\ell - 4\varepsilon + 1 - x_r - 4\varepsilon \geq 1 - w_1 - 30\varepsilon = 1 - w_1 - c_2\varepsilon$).

- *Two* 1/3*-high rectangles lie between* r_ℓ *and* r_r*—see Figure 2.3(b).*

  Assume that there is a 1/3-high rectangle r_1 that intersects with $x = x'_\ell + \varepsilon$ and with $x = 1/2 - \varepsilon$ and there is a 1/3-high rectangle r_2 that intersects with $x = 1/2 + \varepsilon$ and with $x = x_r - \varepsilon$. Note, that if r_1 or r_2 spans from I_ℓ to I_r, then we are in the previous case. Hence we assume that r_1 spans from I_ℓ to I_M and r_2 spans from I_M to I_r. If $x'_\ell \geq 1/2 - 3\varepsilon$ or $x_r \leq 1/2 + 3\varepsilon$ we apply also the method of the previous case, since then r_2 intersects with $x = x'_\ell + 5\varepsilon$ and $x = x_r - \varepsilon$, or r_1 intersects with $x = x'_\ell + \varepsilon$ and $x = x_r - 5\varepsilon$. Otherwise we have $w_1, w_2 \in [\varepsilon, 1/2 + \varepsilon]$. Thus if r_1 or r_2 does not intersect with $y = 1/3$ or with $y = 2/3$, we can apply Algorithm 2.3 (Lemma 2.6). Otherwise, we have $y_1, y_2 < 1/3$ and $y'_1, y'_2 > 2/3$ and thus we can apply Algorithm 2.6 (Lemma 2.9).

The following two cases use Lemma 2.10 and Corollary 2.2. Recall that we have $x'_\ell \geq 4c_3\varepsilon$ if no 1/3-high rectangle intersects with $x = x'_\ell + c_3\varepsilon$ and $x_r \leq 1 - 4c_3\varepsilon$ if no 1/3-high rectangle intersects with $x = x_r - c_3\varepsilon$.

- *A* 1/3*-high rectangle reaches from the middle close to* r_r *but no* 1/3*-high rectangle reaches from* r_ℓ *to the middle—see Figure 2.3(c).*

  In this case we assume that there is a 1/3-high rectangle r_1 that intersects with $x = 1/2 + \varepsilon$ and with $x = x_r - c_3\varepsilon$ but there is no 1/3-high rectangle that intersects with $x = x'_\ell + c_3\varepsilon$. We assume that $x'_\ell \leq 1/2 - 3\varepsilon$ as otherwise we could apply the methods of the first case (as r_1 intersects with $x = x'_\ell + 4\varepsilon \leq x'_\ell + 11\varepsilon$ and $x = x_r - \varepsilon$ in this case). Note that we have $x_r > 1/2 + 3\varepsilon$ as otherwise $c_3 = 5$ and r_1 would intersect with $x = 1/2 - \varepsilon$, i.e., span from I_ℓ to I_r, and the assumption that no 1/3-high intersects with $x = x'_\ell + c_3\varepsilon$ would be violated. Thus we have $c_3 = 2$ (by the definition above) and $x'_\ell \geq 8\varepsilon$.

  Obviously, we have $w_1 \in [\varepsilon, 1 - 2\varepsilon]$ and can thus use Algorithm 2.3 (Lemma 2.6) if $y_1 \geq 1/3$ or $y'_1 \leq 2/3$. Otherwise, the rectangle r_1 intersects $y = 1/3$ and $y = 2/3$. Moreover, we have $x_1 \in [1/2 - \varepsilon, 1/2 + \varepsilon]$ and thus can apply the methods of Corollary 2.2 to

derive a packing into a strip of height $5/3+\varepsilon$. Here we use that no $1/3$-high rectangle intersects with $[x'_\ell + (c_3-1)\varepsilon, x'_\ell + c_3\varepsilon] \times [0,1]$ and that $x'_\ell \leq 1/2 - 9\varepsilon$ since we are otherwise in the first case again (r_1 intersects with $x = x'_\ell + 11\varepsilon$ and $x = x_r - 11\varepsilon$).

The same methods can be applied if the rectangle r_1 reaches from r_ℓ to the middle instead.

- *No $1/3$-high rectangles span across the intervals—see Figure 2.3(d).*

  In this case we assume that no $1/3$-high rectangle intersects with $x = x'_\ell + c_3\varepsilon$ and no $1/3$-high rectangle intersects with $x = x_r - c_3\varepsilon$. Thus we have $x'_\ell, x_r \in [4c_3\varepsilon, 1-4c_3\varepsilon]$ and no $1/3$-high rectangle intersects with $[x'_\ell + (c_3-1)\varepsilon, x'_\ell + c_3\varepsilon] \times [0,1]$ and no $1/3$-high rectangle intersects with $[x_r - c_3\varepsilon, x_r - (c_3-1)\varepsilon] \times [0,1]$. As we have $x_r - x'_\ell > 143\varepsilon > 4c_3\varepsilon$ we can apply the methods of Lemma 2.10.

These four cases cover all possibilities and therefore our algorithm always outputs a packing into a strip of height at most $5/3+260\varepsilon/3$. Thus with Lemma 2.2 we get an approximation ratio for the overall algorithm of $5/3+263\varepsilon/3$. By scaling ε appropriately we proved Theorem 2.1. The running time of the algorithm is $\mathcal{O}(T_{\text{PTAS}} + (n\log^2 n)/\log\log n)$, where T_{PTAS} is the running time of the PTAS from [3].

2.6 Conclusion

We presented an approximation algorithm for the strip packing problem that narrows the approximability gap, which is now between $3/2$ and $5/3+\varepsilon$. This result is an important step to settle the approximability of this problem. We do not see how to adapt our techniques to improve the upper bound even further (for instance, the blocking property in Section 2.4.5 does not hold if we reduce the height of the blocking rectangles and increase the depth of the cut at the same time). So enhancing the upper bound seems to require new techniques. To the best of our knowledge no promising approach to improve the lower bound of $3/2$ is known.

3 New Approximability Results for Two-Dimensional Bin Packing

3.1 Introduction

In the two-dimensional bin packing problem it is desired to pack a list $I = \{r_1, \ldots, r_n\}$ of rectangles with heights h_i and widths w_i into the smallest possible number of unit sized squares, also called bins. The rectangles have to be packed axis-parallel and may not overlap. Our problem consists of two versions; in the first version it is not allowed to rotate the rectangles while in the other it is allowed to rotate the rectangles by 90°, i.e. to exchange the widths and the heights. Two-dimensional packing problems have many real world applications that can be found in the area of scheduling, chip design and logistics. In particular the version of the two-dimensional bin packing problem with rotations can be used for example for stock-cutting, when we want to cut some items out of some sheets of raw material. The version without rotations is for example used for the print and web layout, when we want to place all ads into the minimum number of pages.

Related Work Two-dimensional bin packing is a generalization of its one-dimensional counterpart (where each rectangle has height 1) and is therefore strongly NP-hard. Furthermore Bansal et al. [5] showed that even an APTAS is ruled out. This asymptotic lower bound was further improved by Chlebík & Chlebíková [11] to values $1 + 1/3792$ and $1 + 1/2196$ for the version with and without rotations, respectively. On the positive side there is an asymptotic 2.125-approximation by Chung et al. [12]. The AFPTAS of Kenyon & Rémila [40] and Jansen & van Stee [38] for the related strip packing problem can be used to achieve an asymptotic $2 + \varepsilon$-approximation for the two-dimensional bin packing problem without and with rotations, respectively. Caprara [8] gave the first asymptotic approximation algorithm for the version without rotations that breaks the barrier of 2. The asymptotic approximation

ratio of this algorithm is arbitrary close to the harmonic number $T_\infty = 1.69...$ This result was further improved by Bansal et al. [4] with an asymptotic approximation ratio of arbitrary close to $\ln(T_\infty + 1) = 1.52..$ with and without rotations. The additive constant of this algorithm depends on a precision ε of this algorithm.

In the non-asymptotic setting without rotations there is a 3-approximation by Zhang [54] and by Harren & van Stee [26] with an improved running time. Harren & van Stee [26] also developed a non-asymptotic 2-approximation with rotations. Independently this approximation guarantee is also achieved for the version without rotations by Harren & van Stee [25] and Jansen et al. [34]. These results match the non-asymptotic lower bound of this problem, unless P = NP.

Our Contribution We present the following result for the two-dimensional bin packing problem with and without rotations:

Theorem 3.1. *For any $\varepsilon > 0$, there is an approximation algorithm A which produces a packing of a list I of n rectangles in $A(I)$ bins such that*

$$A(I) \leq (3/2 + \varepsilon) \cdot \mathrm{OPT}(I) + 69.$$

The running time of A is polynomial in n.

This result is an important step in closing the gap between the current asymptotic lower bound and the former best asymptotic approximation ratio. Furthermore, since we have a small additive constant of 69, our algorithm already computes better results for instances with $\mathrm{OPT}(I) \geq 200$ and $\varepsilon \leq 1/8$, or for $\mathrm{OPT}(I) \geq 150$ and $\varepsilon \leq 1/30$ than the non-asymptotic 2-approximations.

In the version that allows rotation we can further improve the additional constant to 39. We obtain the following result.

Theorem' 3.1. *For any $\varepsilon > 0$, there is an approximation algorithm A which produces a packing of a list I of n rectangles that are allowed to be rotated in $A(I)$ bins such that*

$$A(I) \leq (3/2 + \varepsilon) \cdot \mathrm{OPT}(I) + 39.$$

The running time of A is polynomial in n.

Techniques The main idea of our work is to analyse an arbitrary solution of the two-dimensional bin packing problem. Here it does not matter whether the rectangles are rotated

or not. We cut in each bin a small vertical or horizontal strip out of the solution, i.e. we move some rectangles to additional bins, so that a horizontal or vertical strip at one side of the bin is completely free of rectangles. We prove that this is possible for any bin in any possible solution. At these modification steps, we do not rotate the rectangles in order to ensure that it also works for the version where rotations are not allowed. When we have removed a vertical strip of some width ε_c, it is possible to round the widths of all rectangles of width at least ε_c to a multiple of $\varepsilon_c^2/2$ and place them also on an x-coordinate whose value is a multiple of $\varepsilon_c^2/2$. When we have removed a horizontal strip of height ε_c we are able to round the heights of all rectangles of height at least ε_c to a multiple of $\varepsilon_c^2/2$. These rectangles are placed on a y-coordinate whose value is a multiple of $\varepsilon_c^2/2$. It follows that our modified solution consists of two different types of bins. The packing of the bins of the first type satisfy the following property.

Property 3.1. *The width and the x-coordinate of each rectangle in B_i of width at least ε_c is a multiple of $\varepsilon_c^2/2$.*

The packing of the bins of the second type satisfy the analogous property for rounding the heights:

Property 3.2. *The height and the y-coordinate of each rectangle in B_i of height at least ε_c is a multiple of $\varepsilon_c^2/2$.*

We ensure one of these properties also on the additional bins that are used to modify the solution and we obtain the following main result of our work:

Theorem 3.2. *For any value ε_c, with $1/\varepsilon_c$ being a multiple of 24, and for any solution that fits into m bins, we are able to round up the widths and the heights of the rectangles so that they fit into $(3/2 + 5\varepsilon_c) \cdot m + 37$ bins while the packing of each of the bins satisfies either Property 3.1 or Property 3.2.*

After having rounded one side of the rectangles the rounding technique for the unrounded side is fairly standard in the theory of packing algorithms. In general, we employ the rounding technique used in the AFPTAS by Kenyon & Rémila [40]. Small rectangles are packed in containers using some techniques by Jansen & Solis-Oba [36]. Furthermore, we use the algorithm of Steinberg [51], to pack some medium rectangles.

In the non-rotational version, our algorithm initially uses a flow network to assign some big rectangles that have both side lengths at least ε_c to bins of the first and second type. The same flow network is used in the setting that allows rotation to rotate these rectangles. The remaining small, (rotated) long and (rotated) wide rectangles are packed into containers

with a modified version of the algorithm by Kenyon & Rémila [40]. Afterwards we pack the containers and the big rectangles with an integer linear program into the bins. There are only minor differences to improve the additional constant in the version that allows rotation. We state these differences at the end of each section.

3.2 Modifying a Packing

In the following sections, we consider an arbitrary solution, which does not have to be the optimal one, of the rectangles in m bins. We set a coordinate system to each bin, with the origin $(0,0)$ in the lower left corner and with the coordinate $(1,1)$ in the upper right corner. The lower left corner of the rectangle r_j is placed at the position (x_j, y_j) and the upper right corner at the position (x'_j, y'_j). The area of r_j is defined by $a_j := h_j \cdot w_j$. For a set X of rectangles, we have $h(X) := \sum_{r_j \in X} h_j$, $w(X) := \sum_{r_j \in X} w_j$ and $a(X) := \sum_{r_j \in X} h_j \cdot w_j$ for the total height, total width and total area of the rectangles in X. The maximal occurring width and height in X is defined by $w_{\max}(X) := \max_{r_j \in X} w_j$ and $h_{\max} := \max_{r_j \in X} h_j$.

Sometimes, we define a certain rectangular or Γ-shaped region in a bin B_i of our solution. These regions are defined by a closed traverse starting at the lower left corner. A rectangular region, defined by some corner points $(x_1, y_1), (x_2, y_1), (x_2, y_2)$ and (x_1, y_2), is also defined by the Cartesian product $[x_1, x_2] \times [y_1, y_2]$.

Let $\varepsilon_c < 1$ be a value, so that $1/\varepsilon_c$ is a multiple of 24. In order to round the rectangles in our solution, we cut a horizontal strip of height ε_c and width 1 or a vertical strip of width ε_c and height 1 out of each bin B_i. Therefore, we clear always one of the four strips at the sides of the bin, i.e. we remove all rectangles that intersect one of them (except the bins B_i with a very large rectangle that intersects simultaneously all four strips).

Denote the strips of width 1 and height ε_c at the top and at the bottom of the strip by $S_U^{(i)} := [0,1] \times [1-\varepsilon_c, 1]$ and $S_B^{(i)} := [0,1] \times [0, \varepsilon_c]$. The strips of height 1 and width ε_c to the right and left of the bin are called $S_R^{(i)} := [1-\varepsilon_c, 1] \times [0,1]$ and $S_L^{(i)} := [0, \varepsilon_c] \times [0,1]$ (cf. Figure 3.1). There are two kinds of rectangles that intersect these strips. The set of rectangles that lies completely in one of these strips $S_K^{(i)}$, $K \in \{U, B, R, L\}$ is denoted by $C_K^{(i)}$; the set of rectangles that does not lie completely inside a strip but intersects this strip is denoted by $I_K^{(i)}$.

In the following, we want to prove that the union of all sets $C_R^{(i)}, C_L^{(i)}, C_U^{(i)}$ and $C_B^{(i)}$, $i \in \{1, \ldots, m\}$, covers a very small total area and can be moved into few additional bins.

Lemma 3.1. *We move all rectangles in $C_R^{(i)}, C_L^{(i)}, C_U^{(i)}$ and $C_B^{(i)}$ for all $i \in \{1, \ldots, m\}$ into $4\varepsilon_c m + 2$ additional bins. The packing of these bins satisfy either Property 3.1 or Prop-*

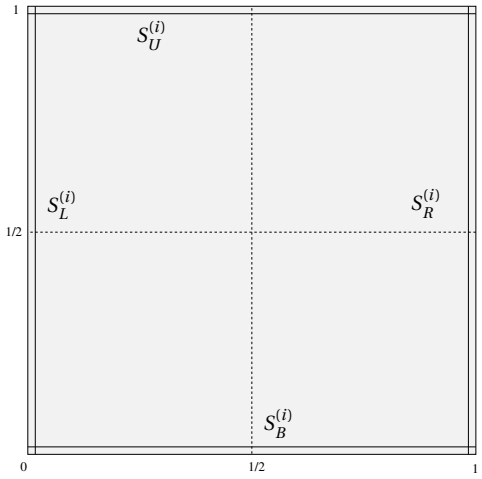


Figure 3.1: Definition of $S_U^{(i)}, S_B^{(i)}, S_L^{(i)}$ and $S_R^{(i)}$

erty 3.2.

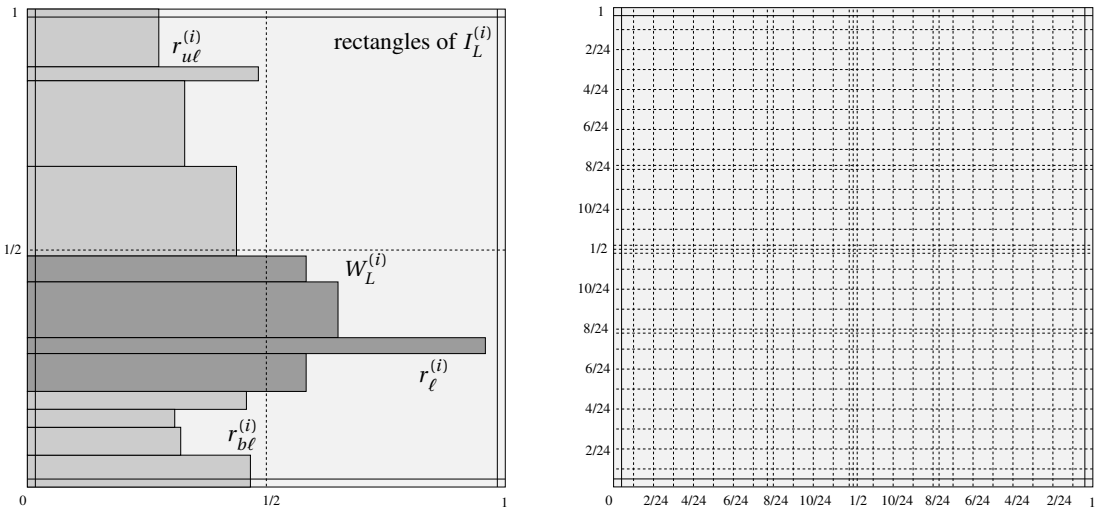
Proof. The rectangles in $C_R^{(i)}$ and $C_L^{(i)}$ are already packed into a strip of height 1 and width ε_c. We pack $1/\varepsilon_c$ of these strips into an additional bin. We have in total $2m$ strips to pack into extra bins. Hence, we need at most $\lceil 2\varepsilon_c \cdot m \rceil \leq 2\varepsilon_c m + 1$ bins. The rectangles of the width ε_c are placed on an x-coordinate whose value is a multiple of ε_c. This value is also a multiple of $\varepsilon_c^2/2$. The remaining rectangles have a width of less than ε_c and hence, this packing satisfies Property 3.1. The analogous packings for the rectangles in the strips S_U and S_R satisfy Property 3.2, and we need in total $4\varepsilon_c m + 2$ bins. □

In the following, we suppose that there is no rectangle completely situated in one of the strips $S_K^{(i)}$, $K \in \{U, B, R, L\}$ and $i \in \{1, \ldots, m\}$, but only rectangles that intersect them. Furthermore, we suppose that the rectangles that intersect $S_K^{(i)}$, i.e. the rectangles of $I_K^{(i)}$, touch the corresponding side of the bin. Therefore, we extend the widths or heights if necessary. Note that this is only for the ease of explanation, the rectangles are rounded later. The rectangles in the corners that intersect a vertical and a horizontal strip are extended in both directions so that they are placed directly in the corners. If there is no such rectangle in one corner, we employ a dummy rectangle of width and height ε_c. We denote the rectangles in the corners by $r_{u\ell}^{(i)} \in I_U^{(i)} \cap I_L^{(i)}; r_{ur}^{(i)} \in I_U^{(i)} \cap I_R^{(i)}; r_{b\ell}^{(i)} \in I_B^{(i)} \cap I_L^{(i)}$ and $r_{br}^{(i)} \in I_B^{(i)} \cap I_R^{(i)}$. Dummy rectangles of the width or the height ε_c are also filled in the remaining gaps. Hence, we suppose that $h(I_L^{(i)}) = 1, h(I_R^{(i)}) = 1, w(I_U^{(i)}) = 1$ and $w(I_B^{(i)}) = 1$. Consequently, the bins that contain a very large rectangle that simultaneously intersects all four strips, contain no further rectangles.

The width and the height of this rectangle is rounded up to 1, and so the packing of these bins satisfies Property 3.1 and Property 3.2.

Lemma 3.2. *If B_i is a bin in our solution that contains a rectangle that simultaneously intersects the strips $S_U^{(i)}, S_B^{(i)}, S_R^{(i)}$ and $S_L^{(i)}$, then we are able to round up this rectangle and the packing satisfies Property 3.1.*

The rectangles intersecting at least one of these strips and having a height or a width larger than $1/2$ play a crucial part in our analysis. Consequently, let $L_U^{(i)} \subseteq I_U^{(i)}$ be the set of rectangles intersecting $S_U^{(i)}$ and having a height larger than $1/2$. $L_B^{(i)} \subseteq I_B^{(i)}$ is the set of rectangles intersecting $S_B^{(i)}$ and having a height larger than $1/2$. Furthermore, let $W_R^{(i)} \subseteq I_R^{(i)}$ and $W_L^{(i)} \subseteq I_L^{(i)}$ be the rectangles of a width larger than $1/2$ and intersecting $S_R^{(i)}$ or $S_L^{(i)}$, respectively. The rectangle of a maximum height in $L_U^{(i)}$ is denoted by $r_u^{(i)}$ and that of $L_B^{(i)}$ is denoted by $r_b^{(i)}$. The rectangle of a maximum width in $W_L^{(i)}$ is denoted by $r_\ell^{(i)}$ and that of $W_R^{(i)}$ is denoted by $r_r^{(i)}$ (cf. Figure 3.2(a)).



(a) Definition of the rectangles intersecting $S_L^{(i)}$ (b) The horizontal and vertical strips

Figure 3.2: Definition of rectangles and strips

We separate each bin B_i into 28 horizontal and vertical strips. Therefore, let

$$\mathrm{IN} = \bigcup_{i=0}^{11} \{i/24\} \cup \{8/24 - \varepsilon_c, 12/24 - \varepsilon_c\},$$

be a set of numbers and let

$$\mathrm{IN}' := \mathrm{IN} \cup \{12/24\}$$

be the extended set. We assume that these sets are sorted according to non-decreasing values. We use two consecutive numbers in_i and in_{i+1} of IN′ as x-coordinates of each vertical strip of the height 1(cf. Figure 3.2(b)). Therefore, we define the 14 vertical strips on the left and right side of the bin by $\mathrm{VL}^{(i)}_{in_i} := [in_i, in_{i+1}] \times [0,1]$ and by $\mathrm{VR}^{(i)}_{in_i} := [1-in_{i+1}, 1-in_i] \times [0,1]$. Analogously, we define the 14 horizontal strips of the width 1 in the lower half and upper half of B_i by $\mathrm{HB}^{(i)}_{in_i} := [0,1] \times [in_i, in_{i+1}]$ and $\mathrm{HU}^{(i)}_{in_i} := [0,1] \times [1-in_{i+1}, 1-in_i]$.

If we completely remove one vertical strip of the width ε_c in one bin, including all rectangles that intersect it, we are able to round up the widths of all rectangles that have a width of at least ε_c.

Lemma 3.3. *If there is a vertical strip of the width ε_c free of rectangles in a bin B_i, then we are able to round up the widths of the rectangles so that the packing of B_i satisfies Property 3.1.*

Proof. W.l.o.g. we assume that $S^{(i)}_R$ is the free strip of rectangles, since we can move all rectangles on the right of one free strip by ε_c to the left. We divide the remaining bin into $2/\varepsilon_c - 2$ vertical strips of the width $\varepsilon_c/2$ by introducing vertical lines at the position $i \cdot \varepsilon_c/2$, for $i \in \{1, \ldots, 2/\varepsilon_c - 2\}$. Each rectangle of a width larger than ε_c intersects at least 3 of these strips and hence crosses at least 2 vertical lines. In a next step we enlarge the strips to a width of $\varepsilon_c/2 + \varepsilon_c^2/2$ by giving some extra space to a rectangle each time it intersects one of these vertical lines by $\varepsilon_c^2/2$. The total width of all strips is $(2/\varepsilon_c - 2) \cdot (\varepsilon_c/2 + \varepsilon_c^2/2) = (1/\varepsilon_c - 1) \cdot (\varepsilon_c + \varepsilon_c^2) = 1 + \varepsilon_c - \varepsilon_c - \varepsilon_c^2 = 1 - \varepsilon_c^2 \leq 1$.

Let r_k be a rectangle that intersects at least 2 vertical lines and that has a width of $w_k \in (i\varepsilon_c^2/2, (i+1)\varepsilon_c^2/2]$ and an x-coordinate $x_k \in (j\varepsilon_c^2/2, (j+1)\varepsilon_c^2/2]$, for some values $i \in \{2/\varepsilon_c, \ldots, 2/\varepsilon_c^2 - 2/\varepsilon_c - 1\}$ and $j \in \{1, \ldots, 2/\varepsilon_c^2 - 2/\varepsilon_c - 1\}$. The extra space of r_k is at least $2 \cdot \varepsilon_c^2/2 = \varepsilon_c^2$ and is large enough for increasing the width from w_k to $\overline{w_k} := (i+1)\varepsilon_c^2/2$ and the x-coordinate from x_k to $\overline{x_k} := (j+1)\varepsilon_c^2/2$. All rectangles of a width larger than ε_c intersect at least 2 vertical lines and thus we round up their widths. It is possible that a rectangle r_k of a width exactly $w_k = \varepsilon_c$ does not intersect 2 vertical lines, because the x-coordinate is already a multiple of ε_c^2. In this case we do not have to change the position and the width of r_k. □

Analogously, we can prove the same result when there is a horizontal strip of the height ε_c free of rectangles.

Lemma 3.4. *If there is a horizontal strip of the height ε_c free of rectangles in a bin B_i, then we are able to round up the heights of the rectangles so that the packing of B_i satisfies Property 3.2.*

PACKING WITH ROTATIONS In the version with rotations, we always clear one of the vertical strips $S_L^{(i)}$ or $S_R^{(i)}$ of the width ε_c. This is possible since we are able to rotate the packing by 90°. This also enables us to use less bins than in the version in which rotations are not allowed. Nevertheless, in the version with rotations we use the same techniques as in the version without rotations. Therefore, we first state the methods without rotations and explain afterwards the minor differences occurring, when we are allowed to rotate the rectangles. In Lemma 3.1, we obtain already an improvement of 1 additional bin, since we are able to rotate the rectangles in the horizontal strips. Thus, we have $4\varepsilon_c \cdot m$ vertical strips to pack into additional bins. The total number of additional bins is therefore $\lceil 4\varepsilon_c \cdot m \rceil \leq 4\varepsilon_c \cdot m + 1$. We obtain the following lemma.

Lemma' 3.1. *We move all rectangles in $C_R^{(i)}, C_L^{(i)}, C_U^{(i)}$ and $C_B^{(i)}$ for all $i \in \{1, \ldots, m\}$ into $4\varepsilon_c m + 1$ additional bins. The packing of these bins satisfies Property 3.1.*

3.2.1 CLASSIFY THE BINS

In the Section 3.2.1 and Section 3.2.2 we explain how to clear a vertical or horizontal strip in each bin of our solution. We start by displaying the following lemma that has some impact on the structure of the packing in the remaining bins.

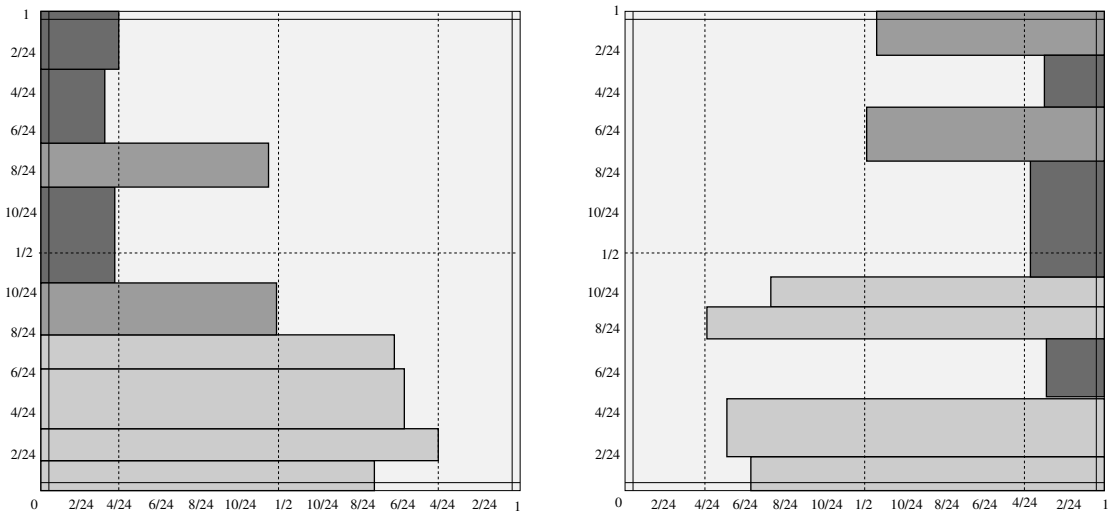
Lemma 3.5. *Let there be two bins B_1, B_2 in our solution, with vertical strips $S_{C_1}^{(1)}, S_{C_2}^{(2)}$ for $C_1 \in \{L, R\}$ and $C_2 \in \{L, R\}$. Furthermore, let there be an $x \in [0, 1/2]$, being a multiple of ε_c, and a value $y \in [0, 1/2]$. If the following conditions hold*

1.1. all rectangles of $W_{C_1}^{(1)}, W_{C_2}^{(2)}$ have a width of at most $1 - x$,

1.2. $h(W_{C_1}^{(1)}) \leq y$ and $h(W_{C_2}^{(2)}) \leq y$,

1.3. there are rectangles in the set $I_{C_1}^{(1)}$ and $I_{C_1}^{(2)}$ that have a width of at most x and a total height of at least y,

then we are able to round up the rectangles and rearrange them into three bins, while the packing of each of the bins satisfies Property 3.1.



(a) Rectangles of $S_L^{(1)}$ in bin B_1

(b) Rectangles of $S_R^{(2)}$ in bin B_2

Figure 3.3: Using Lemma 3.5 with $y = 8/24$ and $x = 4/24$

Proof. We clear the strips $S_{C_1}^{(1)}$ and $S_{C_2}^{(2)}$ in the bins B_1 and B_2 and pack the intersecting sets of rectangles $I_{C_1}^{(1)}$ and $I_{C_2}^{(2)}$ into a new bin B_3 (cf. Figure 3.3). The rectangles of $I_{C_1}^{(1)}$ and $I_{C_2}^{(2)}$ each have a total height of 1 (including the dummy rectangles).

In a first step, we sort these rectangles according to their widths. The rectangles of $I_{C_1}^{(1)}$ are sorted according to non-increasing widths and placed with their x-coordinates at the position 0 in bin B_3. The rectangle with the maximal width is placed at the bottom of the bin and the rectangle with the minimum width is placed at the top. The rectangles of $I_{C_2}^{(2)}$ are sorted according to non-decreasing widths and are placed left aligned with their x'-coordinates at the position 1. Here, the rectangle with the minimum width is at the bottom of the bin and the rectangle with the maximum width is at the top (cf. Figure 3.4). To prove that these two columns of rectangles do not intersect, we look at the three regions between the horizontal lines at height $0, y, 1-y$ and 1. Since $y \leq 1/2$ we have always $y \leq 1-y$.

All rectangles in $W_{C_1}^{(1)}$ have a total height of at most y (cf. Condition 1.2), and are therefore placed in the left column below the horizontal line at height y. They have a width of at most $1-x$ (cf. Condition 1.1). There are rectangles of a total height of at least y that have widths of at most x in $I_{C_1}^{(1)}$ (cf. Condition 1.3). These rectangles are placed in the right column below the horizontal line at height y. Consequently, the rectangles below the horizontal line at height y do not intersect each other. Vice versa, this also holds for the packing above the horizontal line at height $1-y$. The rectangles that are placed between the horizontal lines at height y and $1-y$ have widths of at most $1/2$. Thus, the rectangles in the two columns do

not intersect each other.

After that there is a vertical strip of the width ε_c completely free of rectangles in bin B_1 and B_2. Hence, we are able to round the rectangles according to Lemma 3.3 in order to satisfy Property 3.1. The widths of the rectangles in bin B_3 are also rounded to the next largest multiple of $\varepsilon_c^2/2$, to values of at most $x, 1/2$ and $1-x$, respectively. These values are all multiples of ε_c and therefore also multiples of $\varepsilon_c^2/2$ (for $1/\varepsilon_c = i \cdot 24$ and $x = j\varepsilon_c$ we have $x = j\varepsilon_c = 2j\varepsilon_c^2/(2\varepsilon_c) = (2 \cdot j \cdot i \cdot 24)\varepsilon_c^2/2 = (48 \cdot j \cdot i)\varepsilon_c^2/2$; furthermore, since $x = j\varepsilon_c \leq 1$ it is $1 - x = 1 - j\varepsilon_c = (1/\varepsilon_c - j)\varepsilon_c = (i \cdot 24 - j) \cdot \varepsilon_c = (48 \cdot (i \cdot 24 - j) \cdot i)\varepsilon_c^2/2)$.

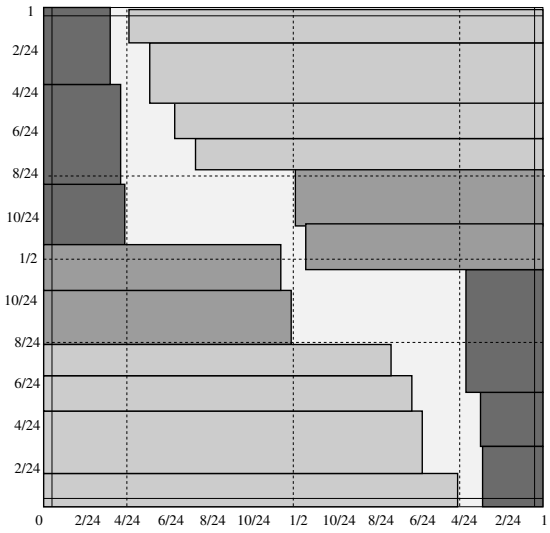


Figure 3.4: Combining the rectangles of $S_L^{(1)}$ and $S_R^{(2)}$

□

The analogous lemma for rounding the heights is as follows. We omit the proof, since it is analogous to the proof of Lemma 3.5.

Lemma 3.6. *Let there be two bins B_1, B_2 in our solution, with horizontal strips $S_{C_1}^{(1)}, S_{C_2}^{(2)}$ for $C_1 \in \{U, B\}$ and $C_2 \in \{U, B\}$. Furthermore, let there be an $x \in [0, 1/2]$, being a multiple of ε_c, and a value $y \in [0, 1/2]$. If the following conditions hold*

1.4. all rectangles of $L_{C_1}^{(1)}, L_{C_2}^{(2)}$ have height at most $1 - x$,

1.5. $w(L_{C_1}^{(1)}) \leq y$ and $w(L_{C_2}^{(2)}) \leq y$,

1.6. there are rectangles in the set $I_{C_1}^{(1)}$ and $I_{C_1}^{(2)}$ that have a height of at most x and a total width of at least y,

then we are able to round up the rectangles and rearrange them into three bins, while the packing of each of the bins satisfies Property 3.2.

We are not able to use this lemma to all values of x and y, since there might be an unbounded number of them. Thus, we use a discretization and employ these lemmas only for all $x \in \mathrm{IN}$ and for $y \in \{0, 1/2\}$

Lemma 3.7. *Let k denote the number of bins B_i in our solution, for which an $x \in \mathrm{IN}$ and a $y \in \{0, 1/2\}$ exists so that one of the following conditions holds:*

1.7. The total height of $W_L^{(i)}$ is at most y, all rectangles of $W_L^{(i)}$ have a width of at most $1-x$ and there are rectangles of a total height of at least y in $I_L^{(i)}$ that have a width of at most x.

1.8. The total height of $W_R^{(i)}$ is at most y, all rectangles of $W_R^{(i)}$ have a width of at most $1-x$ and there are rectangles of a total height of at least y in $I_R^{(i)}$ that have a width of at most x.

1.9. The total width of $L_B^{(i)}$ is at most y, all rectangles of $L_B^{(i)}$ have a height of at most $1-x$ and there are rectangles of a total width of at least y in $I_B^{(i)}$ that have a height of at most x.

1.10. The total width of $L_U^{(i)}$ is at most y, all rectangles of $L_U^{(i)}$ have a height of at most $1-x$ and there are rectangles of a total width of at least y in $I_U^{(i)}$ that have a height of at most x.

We are able to round the rectangles of these k bins and rearrange them into $3/2k+15$ bins, while the packing of each of the bins satisfies either Property 3.1 or Property 3.2.

Proof. We separate these k bins into 30 sets. For each $x \in \mathrm{IN}$ we denote the set of bins, for which either Condition 1.7 or Condition 1.8 holds with $y = 1/2$, by $V_{x,1/2}$. Analogously, we denote the set of the remaining bins, for which either Condition 1.9 or Condition 1.10 holds with $y = 1/2$, by $H_{x,1/2}$. Furthermore, let V_0 denote the set of remaining bins, for which $W_L = \emptyset$ or $W_R = \emptyset$ holds and let H_0 denote the set of remaining bins, for which $L_B = \emptyset$ or $L_U = \emptyset$ holds. These are the bins that satisfy one of the four conditions with $y = 0$.

We employ Lemma 3.5 with each sequence of two bins in each set $V_{x,1/2}$ and V_0 and Lemma 3.6 with each sequence of two bins in each set $H_{x,1/2}$ and H_0. We need one additional bin for each set with an odd cardinality ℓ. This results in a packing of $3/2(\ell-1)+2$ bins. Consequently, we have at most $3/2(k-30)+2\cdot 30 = 3/2k+15$ bins in total, when all 30 sets have an odd cardinality. □

In the following, we present some corollaries following from the lemma above. We prove that the packings in the remaining bins have a certain structure.

Corollary 3.1. *Let B_i be a bin in our solution for which Lemma 3.2 and Lemma 3.7 are not applicable. It follows that the sets $L_U^{(i)}, L_B^{(i)}, W_L^{(i)}$ and $W_R^{(i)}$ are non-empty and disjoint.*

Proof. Suppose by contradiction that $W_L^{(i)} = \emptyset$. It follows that all rectangles intersecting $S_L^{(i)}$ have a width of at most $1/2$. Hence, we have fulfilled Condition 1.7, with $y = 0$, which is a contradiction. The proof for $L_U^{(i)}, L_B^{(i)}$ and $W_R^{(i)}$ is analogous. Thus, the sets $L_U^{(i)}, L_B^{(i)}, W_L^{(i)}$ and $W_R^{(i)}$ are non-empty and as a consequence of Lemma 3.2 there is no rectangle simultaneous in all four sets.

Suppose by contradiction that there is a rectangle $r_1 \in L_U^{(i)} \cap L_B^{(i)}$. The rectangle r_1 has a height of 1. If $r_1 \in W_L^{(i)}$, then $r_1 \notin W_R^{(i)}$ and its x'-coordinate has to be larger than $1/2$. It follows that each rectangle in $W_R^{(i)}$ intersects r_1, which is a contradiction. If $r_1 \notin W_L^{(i)}$, then its x-coordinate has to be larger than $1/2$, since otherwise each rectangle in $W_L^{(i)}$ intersects r_1. However, each rectangle in $W_R^{(i)}$ intersects r_1, which is again a contradiction. Consequently, there is no rectangle in $L_U^{(i)} \cap L_B^{(i)}$ and analogously there is no rectangle in $W_L^{(i)} \cap W_R^{(i)}$.

Suppose by contradiction that there is a rectangle $r_1 \in L_U^{(i)} \cap W_L^{(i)}$. This rectangle has a width and a height of larger than $1/2$. Thus, its y-coordinate is less than $1/2$ and its x'-coordinate is larger than $1/2$. Each rectangle $r_2 \in W_R^{(i)}$ is therefore positioned below r_1 with a y'-coordinate less than $1/2$. If this was the case each rectangle in $L_B^{(i)}$ would intersect either r_1 or r_2, which is a contradiction. The proof for the disjunction of the remaining sets is analogously. □

This result enables us to do a first analysis of the packings in the remaining bins. Let B_i be a bin, in which Lemma 3.2 and Lemma 3.7 are not applicable. Suppose by contradiction that there are two (not necessarily distinct) rectangles $r_1, r_2 \in L_U^{(i)}$ so that r_1 has the x-coordinate $x_1 \leq 1/2$ and r_2 has the x'-coordinate $x_2' \geq 1/2$. The rectangles $r_3 \in W_L^{(i)}$ and $r_4 \in W_R^{(i)}$ have to lie below r_1 and r_2 and their y'-coordinates are less than $1/2$. Hence, each rectangle in $L_B^{(i)}$ intersects either r_3 or r_4, which is a contradiction.

Consequently, all x- and x'-coordinates of the rectangles in $L_U^{(i)}$ are either less than $1/2$ or larger than $1/2$. W.l.o.g. we assume that all x-coordinates are less than $1/2$, since we are able to mirror the packing at the vertical line at the x-coordinate $1/2$. All rectangles of $W_L^{(i)}$ are placed below the rectangles of $L_U^{(i)}$ and their y'-coordinates are less than $1/2$. Consequently, the rectangles of $L_B^{(i)}$ are on the right of the rectangles in $W_L^{(i)}$ and their x-coordinates are larger than $1/2$. The rectangles of $W_R^{(i)}$ are situated above the rectangles in $L_B^{(i)}$ and their y-coordinates are larger than $1/2$ (cf. Figure 3.5). We obtain the following structural theorem.

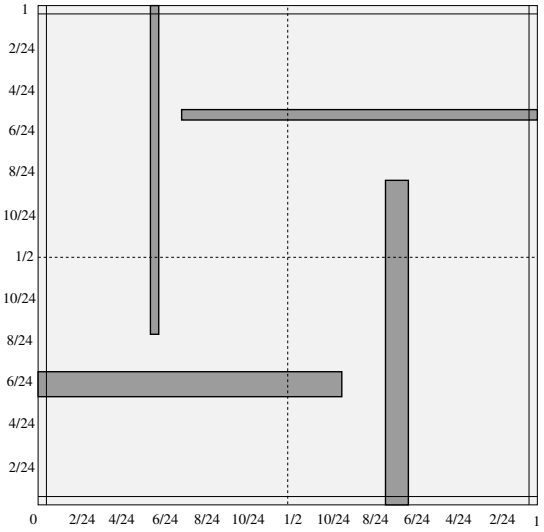


Figure 3.5: A packing as described in Theorem 3.2

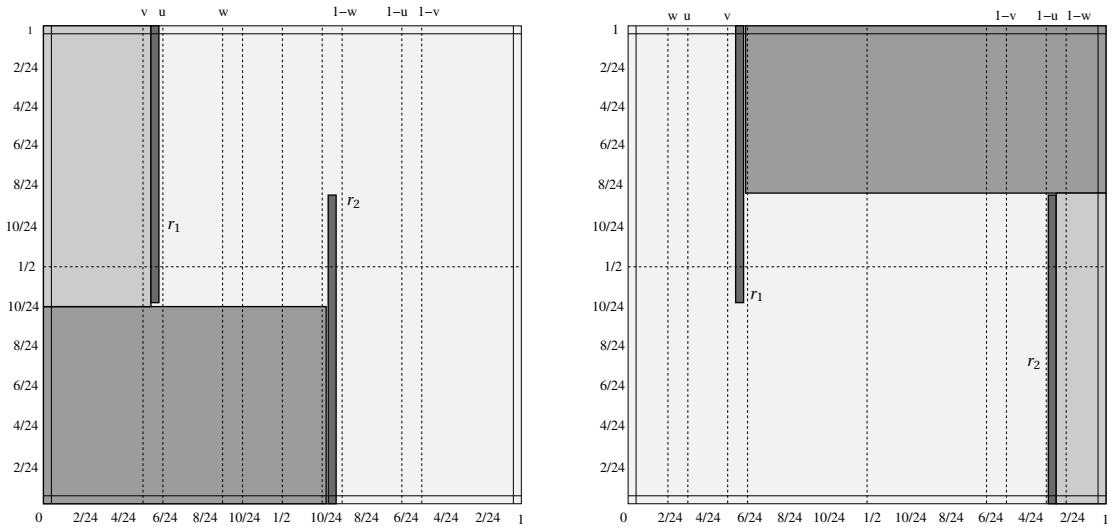
Theorem 3.2. *Consider a bin B_i in our solution, for which Lemma 3.2 and Lemma 3.7 are not applicable. The rectangles of $L_U^{(i)}$ are, w.l.o.g., completely in the left half of these bins (all x'-coordinates are less than $1/2$); the rectangles of $L_B^{(i)}$ are completely in the right half of these bins (all x-coordinates are larger than $1/2$); the rectangles of $W_L^{(i)}$ are completely in the lower half of these bins (all y'-coordinates are less than $1/2$) and the rectangles of $W_R^{(i)}$ are completely in the upper half of these bins (all y-coordinates are larger than $1/2$).*

It is very useful that this structure remains the same when turning the bin by $90°, 180°$ and $270°$ since sometimes we use analogous arguments. Note that by turning the bin by $180°$, we rotate the packing but not the rectangles.

Corollary 3.2. *Let B_i be a bin in our solution for which Lemma 3.2 and Lemma 3.7 are not applicable. Furthermore, let there be a rectangle r_1 in $L_U^{(i)}$ with an x'-coordinate in an interval $\mathrm{VL}_v^{(i)}$, for a $v \in \mathrm{IN}$. It follows that the x-coordinates of all rectangles in $L_B^{(i)}$ are situated in $\mathrm{VR}_v^{(i)}$.*

Proof. Let w be an element of IN. Suppose by contradiction that $w \neq v$ and that there is a rectangle r_2 in $L_B^{(i)}$ with an x-coordinate in $\mathrm{VR}_w^{(i)}$ (cf. Figure 3.6).

Case 1, $w > v$. Let $u \leq w$ be the successor of v in IN, i.e. $\mathrm{VL}_v^{(i)} = [v, u] \times [0, 1]$. The widths of all rectangles in $I_L^{(i)}$ that lie above the horizontal line at the y-coordinate y_1 or that intersect with it are bounded by the rectangle r_1. Hence, their widths are bounded by the value $u \leq w$. The height of r_1 is larger than $1/2$, hence $y_1 < 1/2$. Consequently, there are



(a) Case 1 with $v = 5/24$ and $w = 9/24$

(b) Case 2 with $v = 5/24$ and $w = 2/24$

Figure 3.6: The two cases of Corollary 3.2

rectangles of the total height of at least $1/2$ that have a width of at most w. The remaining rectangles in $I_L^{(i)}$ that lie below the horizontal line at the y-coordinate y_1 are bounded by the rectangle r_2. The x-coordinate of r_2 is within $\mathrm{VL}_w^{(i)}$, it follows that these rectangles have a width of at most $1 - w$. We have fulfilled Condition 1.7 with $y = 1/2$ and $x = w$, which is a contradiction.

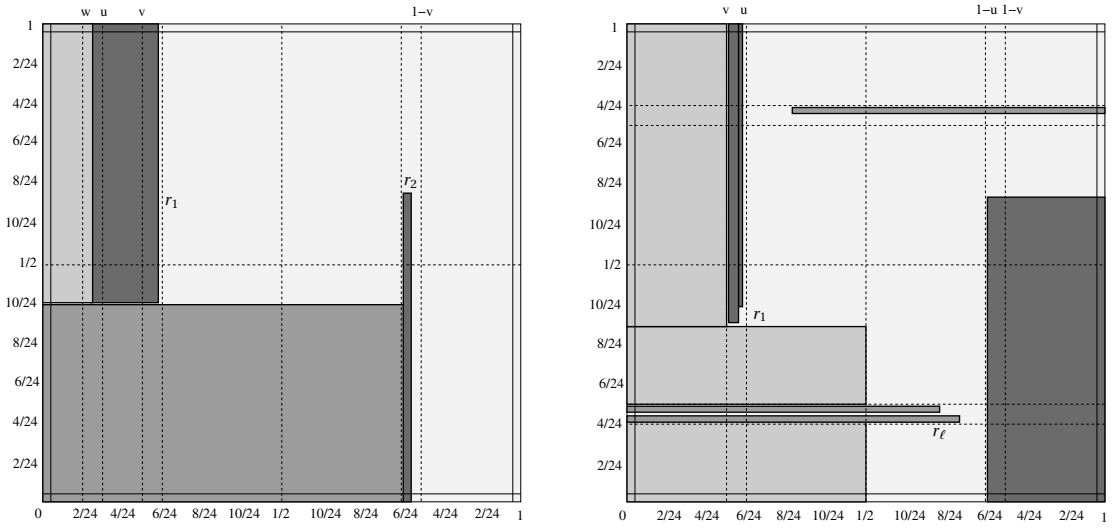
Case 2, $w < v$. Let $u \leq v$ be the successor of w in IN′, i.e. $\mathrm{VR}_w^{(i)} = [1-u, 1-w] \times [0,1]$. We use the same argumentation as in the first case on the strip $S_R^{(i)}$. The widths of the rectangles in $I_R^{(i)}$ that are positioned below the y'-coordinate y_2' are bounded by the rectangle r_2. Their total height is larger than $1/2$ and their widths are at most $u \leq v$. The widths of the rectangles in $I_R^{(i)}$ that lie above the horizontal line at the y'-coordinate y_2' are bounded by r_1. The x'-coordinate of r_1 is in $\mathrm{VL}_v^{(i)}$ and hence the widths are bounded by $1 - v$. It follows that we satisfy Condition 1.8 with $y = 1/2$ and $x = v$, which is a contradiction. □

Consequently, if there is a rectangle in $L_U^{(i)}$ with its x'-coordinate in an interval $\mathrm{VL}_v^{(i)}$, all rectangles of $L_B^{(i)}$ have their x-coordinates in the interval $\mathrm{VR}_v^{(i)}$. Furthermore, we can use this corollary after turning the bin by 180°. We obtain: if there is a rectangle in $L_B^{(i)}$ with its x-coordinate in an interval $\mathrm{VR}_v^{(i)}$, then all rectangles of $L_U^{(i)}$ have their x'-coordinates in the interval $\mathrm{VL}_v^{(i)}$. Hence, the x'-coordinates of all rectangles in $L_U^{(i)}$ are in $\mathrm{VL}_v^{(i)}$ and the x-coordinates of all rectangles in $L_B^{(i)}$ are in $\mathrm{VR}_v^{(i)}$. The same holds for the wide rectangles intersecting $S_R^{(i)}$ and $S_L^{(i)}$ by employing this corollary on the bin turned by 90°. Thus, the

y'-coordinates of all rectangles in $W_L^{(i)}$ are in $\mathrm{HB}_h^{(i)}$ and the y-coordinates of all rectangles in $W_R^{(i)}$ are in $\mathrm{HU}_h^{(i)}$, for some $h \in \mathrm{IN}$.

Corollary 3.3. *Let B_i be a bin in our solution for which Lemma 3.2 and Lemma 3.7 are not applicable. Furthermore, let there be a rectangle $r_1 \neq r_{u\ell}^{(i)}$ in $L_U^{(i)}$ with x'-coordinate in an interval $\mathrm{VL}_v^{(i)}$ for $v \in \mathrm{IN}$ ($r_{u\ell}^{(i)}$ is the rectangle in the upper-left corner). It follows that the x-coordinate of r_1 is also situated in $\mathrm{VL}_v^{(i)}$.*

Proof. Suppose by contradiction that there is a member $w < v$ of IN and the x-coordinate of r_1 is in $\mathrm{VL}_w^{(i)}$. It holds that $r_1 \neq r_{u\ell}^{(i)}$ and hence r_1 does not intersect $S_L^{(i)}$ (cf. Figure 3.7(a)).



(a) Corollary 3.3; $v = 5/24$, $w = 2/24$.

(b) Corollary 3.4; $v = 5/24$.

Figure 3.7: The situation in the Corollary 3.3 and Corollary 3.4

Let $u \leq v$ be the successor of w in IN′, i.e. $\mathrm{VL}_w^{(i)} = [w, u] \times [0, 1]$. The widths of the rectangles in $I_L^{(i)}$ that lie above the horizontal line at y-coordinate y_1 or that intersect this line are bounded by r_1. It follows that their total height is at least $1 - y_1 > 1/2$ and their widths are at most $u \leq v$. Furthermore, as a consequence of Corollary 3.1 and Corollary 3.2 there exists a rectangle r_2 of $L_B^{(i)}$ that has its x-coordinate x_2 within $\mathrm{VR}_v^{(i)}$. Thus, all remaining rectangles in $I_L^{(i)}$ that are below the horizontal line at y-coordinate y_1 have a bounded width of at most $1 - v$. Consequently, we satisfy Condition 1.7 with $y = 1/2$ and $x = v$, which is a contradiction. □

As a consequence of Corollary 3.2 and Corollary 3.3, all rectangles of $L_U^{(i)}$ except $r_{u\ell}^{(i)}$, if $r_{u\ell}^{(i)} \in L_U^{(i)}$, are completely situated in an interval $\mathrm{VL}_v^{(i)}$. Again, we employ this corollary on

each side of the bin and as a consequence, we achieve that all rectangles of $L_B^{(i)} \setminus \{r_{br}^{(i)}\}$ are completely in $\mathrm{VR}_v^{(i)}$. All rectangles of $W_L^{(i)} \setminus \{r_{br}^{(i)}\}$ are completely in an interval $\mathrm{HB}_h^{(i)}$ and all rectangles of $W_R^{(i)} \setminus \{r_{ur}^{(i)}\}$ are completely in $\mathrm{HU}_h^{(i)}$.

Corollary 3.4. *Let B_i be a bin in our solution, for which Lemma 3.2 and Lemma 3.7 are not applicable and let there be a rectangle $r_1 \in L_U^{(i)}$ with x'-coordinate in an interval $\mathrm{VL}_v^{(i)}$. It follows that the x'-coordinate $x_\ell'^{(i)}$ is situated within the interval $\mathrm{VR}_v^{(i)}$ ($r_\ell^{(i)}$ is the rectangle of maximum width in $W_L^{(i)}$).*

Proof. Suppose by contradiction that $r_\ell^{(i)}$, does not intersect $\mathrm{VR}_v^{(i)}$, i.e. the x'-coordinate $x_\ell'^{(i)}$ is not within the interval $\mathrm{VR}_v^{(i)}$ (cf. Figure 3.7(b)). Let u be the successor of v in IN', i.e. $\mathrm{VL}_v^{(i)} = [v, u] \times [0, 1]$ and $\mathrm{VR}_v^{(i)} = [1 - u, 1 - v] \times [0, 1]$.

Since $r_\ell^{(i)}$ has the maximum width among the rectangles in $W_L^{(i)}$, no rectangle of $W_L^{(i)}$ intersects $\mathrm{VR}_v^{(i)}$. Thus all rectangles of $W_L^{(i)}$ have a bounded width of at most $1 - u$. The rectangles of $I_L^{(i)}$ that lie above the horizontal line at height y_1 or that intersect this line are bounded by rectangle r_1. Consequently, their total height is at least $1 - y_1 > 1/2$ and their widths are at most u. Thus, we satisfy Condition 1.7 with $y = 1/2$ and $x = u$, which is a contradiction

□

We also adopt this corollary to the bins turned by 90°, 180° and 270° and obtain the following structural theorem of the packing in the remaining bins in our solution (cf. Figure 3.8).

Theorem 3.3. *Let there be a packing in one bin B_i of our solution for which Lemma 3.2 and Lemma 3.7 are not applicable. It follows that values $h^{(i)}, v^{(i)} \in \mathrm{IN}$ with the following conditions exist:*

1.11. The sets $L_U^{(i)}, L_B^{(i)}, W_L^{(i)}$ and $W_R^{(i)}$ are non-empty and disjoint.

1.12. All rectangles in $L_U^{(i)} \setminus r_{u\ell}^{(i)}$ and $L_B^{(i)} \setminus r_{br}^{(i)}$ are completely situated within $\mathrm{VL}_{v^{(i)}}^{(i)}$ and $\mathrm{VR}_{v^{(i)}}^{(i)}$, respectively.

1.13. All rectangles in $W_L^{(i)} \setminus r_{b\ell}^{(i)}$ and $W_R^{(i)} \setminus r_{u\ell}^{(i)}$ are completely situated within $\mathrm{HB}_{h^{(i)}}^{(i)}$ and $\mathrm{HU}_{h^{(i)}}^{(i)}$, respectively.

1.14. If $r_{u\ell}^{(i)} \in L_U^{(i)}$, then the x'-coordinate $x_{u\ell}'^{(i)}$ is situated within $\mathrm{VL}_{v^{(i)}}^{(i)}$;
if $r_{br}^{(i)} \in L_B^{(i)}$, then the x-coordinate $x_{br}^{(i)}$ is situated within $\mathrm{VR}_{v^{(i)}}^{(i)}$.

1.15. If $r_{b\ell}^{(i)} \in W_L^{(i)}$, then the y'-coordinate $y_{b\ell}'^{(i)}$ is situated within $\mathrm{HB}_{h^{(i)}}^{(i)}$;
if $r_{ur}^{(i)} \in W_R^{(i)}$, then the y-coordinate $y_{ur}^{(i)}$ is situated within $\mathrm{HU}_{h^{(i)}}^{(i)}$.

1.16. the y-coordinates $y_u^{(i)}$ and $y_b'^{(i)}$ are situated within $\mathrm{HB}_{h^{(i)}}^{(i)}$ and $\mathrm{HU}_{h^{(i)}}^{(i)}$, respectively; the x-coordinates $x_r^{(i)}$ and $x_\ell'^{(i)}$ are situated within $\mathrm{VL}_{v^{(i)}}^{(i)}$ and $\mathrm{VR}_{v^{(i)}}^{(i)}$, respectively.

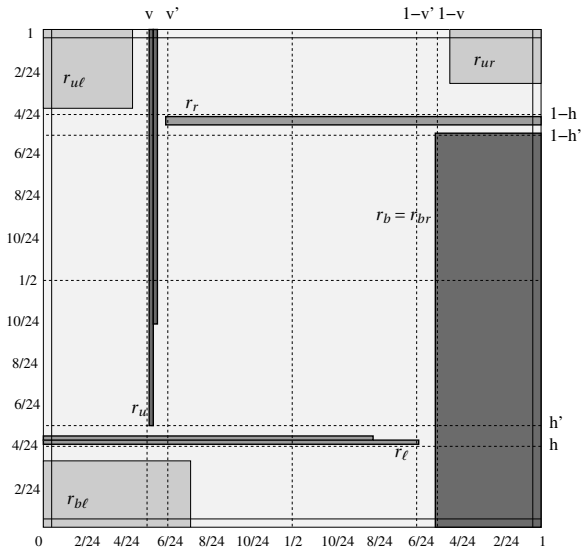


Figure 3.8: A packing as described in Theorem 3.3, with $v = 5/24$, $h = 4/24$ and $r_{br}^{(i)} \in L_B^{(i)}$.

In the following, we classify the remaining bins B_i, for which Lemma 3.2 and Lemma 3.7 are not applicable according to the values $v \in \mathrm{IN}$ and $h \in \mathrm{IN}$. Therefore, denote the values for which Theorem 3.3 for bin B_i holds by $h^{(i)} \in \mathrm{IN}$ and $v^{(i)} \in \mathrm{IN}$. Furthermore, let $h'^{(i)} \in \mathrm{IN}'$ and $v'^{(i)} \in \mathrm{IN}'$ be the successor of $h^{(i)}$ and $v^{(i)}$, respectively.

The packing that is described in Theorem 3.3 consists of almost five different regions. The region at the left of the rectangle $r_u^{(i)}$, the region below $r_\ell^{(i)}$, the region to the right of $r_b^{(i)}$, the region on top of $r_r^{(i)}$ and the region in the middle of the bin. There are only few rectangles that intersect two of these regions, since they have to lie completely inside the horizontal strips $\mathrm{HB}_{h^{(i)}}^{(i)}$ or $\mathrm{HU}_{h^{(i)}}^{(i)}$ or inside the vertical strips $\mathrm{VL}_{v^{(i)}}^{(i)}$ or $\mathrm{VR}_{v^{(i)}}^{(i)}$. We make use of this structure in the following section and remove the rectangles from two of the regions in order to remove a horizontal or vertical strip.

Classify the Bins with Rotations Before we continue with our analysis, we first state the differences in the version with rotations. In the proof of Lemma 3.7 we use 30 sets of bins $V_{x,1/2}, H_{x,1/2}, V_0$ and H_0, for $x \in \mathrm{IN}$. We rotate the packing of one bin B_i that is in a set $H_{x,1/2}$ or in H_0. The packing of this bin satisfies either Condition 1.7 or Condition 1.8. Therefore, it belongs to the set $V_{x,1/2}$ or V_0, respectively. Thus, when we are allowed to

rotate, we have only 15 different sets of bins. We use Lemma 3.5 on each sequence of two bins in each set. If all 15 sets have an odd cardinality, we pack k bins, each satisfying at least one of the conditions Condition 1.7-1.10, into $3/2(k-15)+2\cdot 15 = 3/2k+15/2 < 3/2k+8$ bins.

Lemma' 3.7. *Let k denote the number of bins B_i in our solution for which an $x \in \mathrm{IN}$ and a $y \in \{0, 1/2\}$ exist so that one of the Conditions 1.7-1.10 holds. We are able to round up the rectangles of these k bins and rearrange them into $3/2k+8$ bins, while the packing of each of the bins satisfies Property 3.1.*

3.2.2 Case Analysis

In the following, we suppose that for every bin there are values $v^{(i)} \in \mathrm{IN}$ and $h^{(i)} \in \mathrm{IN}$ so that Theorem 3.3 holds. We do a case analysis for the values $h^{(i)}$ and $v^{(i)}$.

Lemma 3.8. *Let $B_1, \ldots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \ldots, k\}$, has a packing with the following conditions:*

2.1. $r_{b\ell}^{(i)} \notin W_L^{(i)}$ *or* $r_{ur}^{(i)} \notin W_R^{(i)}$,

2.2. $h^{(i)} \in \mathrm{IN}$,

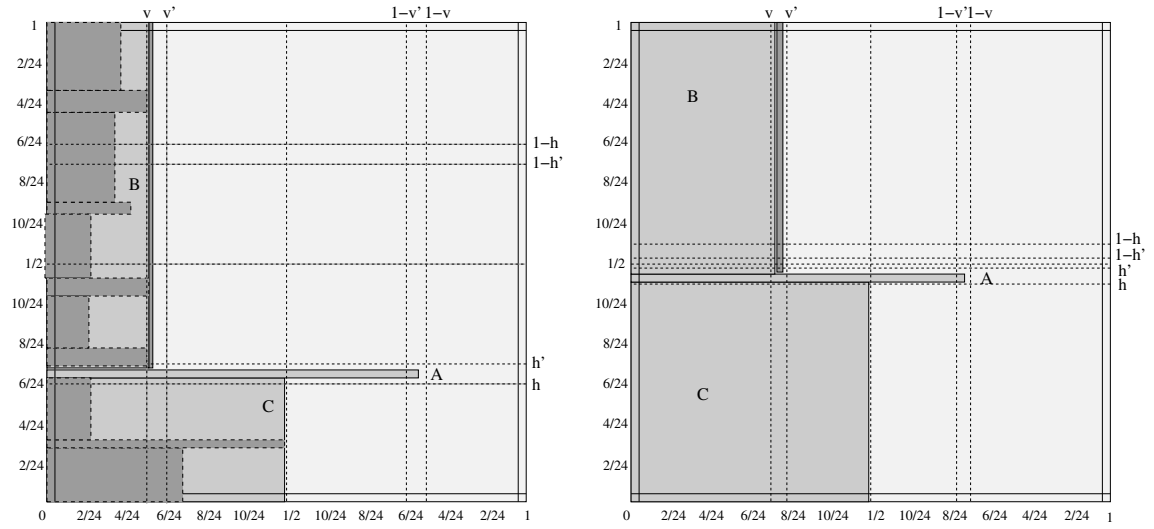
2.3. $v^{(i)} \in \{0/24, \ldots, 7/24, 8/24 - \varepsilon_c\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2k+2$ bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.

Proof. Let $i \in \{1, \ldots, k\}$. W.l.o.g. we assume that $r_{b\ell}^{(i)} \notin W_L^{(i)}$ since we are able to turn the bin by 180°. We clear the strip $S_L^{(i)}$ in each of the k bins by using the property that the rectangles $W_L^{(i)}$ of width larger than $1/2$ are completely situated within $\mathrm{HB}_{h^{(i)}}^{(i)}$ and thus have a total height of at most $1/24$ (cf. Figure 3.9).

We divide the rectangles in $I_L^{(i)}$ into three sets. Let $A^{(i)} = W_L^{(i)}$ be the set of rectangles that have a width of larger than $1/2$. Let $B^{(i)} \subset I_L^{(i)}$ denote the set of rectangles that have a width of at most $1/3$. Finally, let $C^{(i)}$ denote the set of the remaining rectangles in $I_L^{(i)}$ that have a width within $(1/3, 1/2]$. As mentioned above, the total height of the rectangles in $A^{(i)}$ is at most $1/24$, since $r_{b\ell}^{(i)} \notin W_L^{(i)}$ (Condition 2.1). We pack these rectangles on top of each other into a container of height $h(A^{(i)}) \leq 1/24$ and width $w_{\max}(A^{(i)}) \leq 1$. We treat this container as a rectangle $r_A^{(i)}$ of width $w_A^{(i)} = 1$ and height $h_A^{(i)} = h(A^{(i)})$.

The rectangles in $L_U^{(i)}$ have their x'-coordinates within $\mathrm{VL}_{v^{(i)}}^{(i)}$. Thus, the rectangles of $I_L^{(i)}$ that lie on the left of the rectangles in $L_U^{(i)}$, including $r_{u\ell}^{(i)}$ if $r_{u\ell}^{(i)} \in L_U^{(i)}$, have a bounded



(a) Bin with $h = 7/24$ and $v = 5/24$

(b) Bin with $h = 11/24$ and $v = 7/24$

Figure 3.9: Two possible initial situations of Lemma 3.8.

width of $v'^{(i)} \leq 8/24 = 1/3$ (Condition 2.3). Remember that $v'^{(i)}$ is the successor of $v^{(i)}$ in IN′. Therefore, these rectangles belong to the set $B^{(i)}$. Consequently, the total height of the rectangles in the set $B^{(i)}$ is at least $h(B^{(i)}) = h_u^{(i)} > 1/2$. The rectangles are also packed on top of each other into the container/rectangle $r_B^{(i)}$ of a width $w_B^{(i)} = w_{\max}(B^{(i)}) \leq 1/3$ and a height $h_B^{(i)} = h(B^{(i)}) > 1/2$. Moreover, the rectangles in $C^{(i)}$ are packed on top of each other into a container/rectangle $r_C^{(i)}$ of a width $w_C^{(i)} = w_{\max}(C^{(i)}) \leq 1/2$ and a height $h_C^{(i)} = h(C^{(i)}) \leq 1 - h(B^{(i)}) = 1 - h_B^{(i)}$.

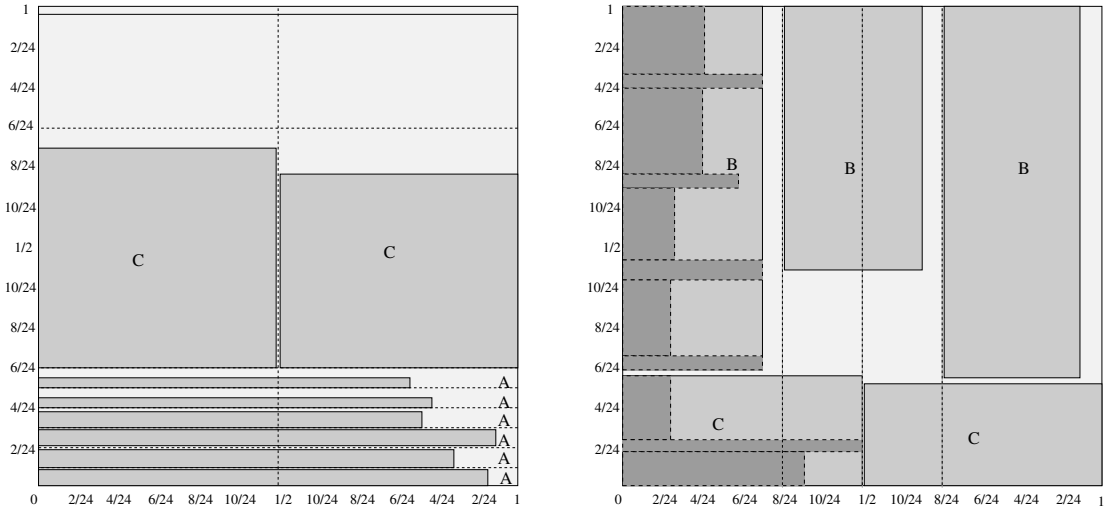
We clear the strips $S_L^{(i)}$ of each bin by packing the rectangles of each sequence of 6 bins into 3 additional bins C_1, C_2, C_3. Let $B_1, \ldots, B_6$ be 6 bins among the k bins. W.l.o.g. we assume that these bins are sorted by non-decreasing heights of $h_B^{(i)}$.

The rectangles $r_A^{(1)}, \ldots, r_A^{(6)}$ have a total height of at most $6 \cdot 1/24 = 1/4$. We pack them on top of each other at the bottom of bin C_1 with their x-coordinates positioned on the value 0. On top of these rectangles we pack $r_C^{(1)}$ and $r_C^{(2)}$. They have both a width and a height of at most $1/2$ and fit next to each other on the positions $(0, 1/4)$ and $(1/2, 1/4)$. The uppermost horizontal strip of the height $1/4 \geq \varepsilon_c$ is still free of rectangles (cf. Figure 3.10(a)). We employ Lemma 3.4 on this bin in order to round up the heights.

The rectangles $r_B^{(i)}$ and $r_C^{(i)}$ always fit on top of each other since $h_C^{(i)} < 1 - h_B^{(i)}$. This allows us to place $r_C^{(3)}$ and $r_B^{(3)}$ on top of each other in bin C_2 with their x-coordinates positioned on the value 0. The rectangles $r_C^{(4)}$ and $r_B^{(4)}$ are also placed on top of each other, where

$r_C^{(4)}$ is placed on position $(1/2, 0)$ and $r_B^{(4)}$ is placed on top of $r_C^{(4)}$ on the x-coordinate $2/3$. Between the rectangles $r_B^{(3)}$ and $r_B^{(4)}$ there is a free space of width $1/3$ and height at least $\min\{h_B^{(3)}, h_B^{(4)}\} = h_B^{(3)}$. Since $h_B^{(1)} \leq h_B^{(3)}$ and $w_B^{(1)} \leq 1/3$ this space is sufficient to place $r_B^{(1)}$ on top of $r_C^{(3)}$ and $r_C^{(4)}$ on the x-coordinate $1/3$ (cf. Figure 3.10(b)).

A horizontal or vertical strip free of rectangles does not necessarily have to exists in this bin. However, we are able to round up the widths of the rectangles $r_B^{(1)}, r_B^{(3)}$ and $r_B^{(4)}$ to the next largest multiple of $\varepsilon_c^2/2$ that is at most $1/3$. The widths of the rectangles $r_C^{(3)}$ and $r_C^{(4)}$ are also rounded to the next largest multiple of $\varepsilon_c^2/2$ that is at most $1/2$. The rectangles that are inside the rectangles $r_B^{(i)}$ and $r_C^{(i)}$ are packed on top of each other. This enables us also to round up their widths to the next largest multiple of $\varepsilon_c^2/2$, which is at most $1/3$ or $1/2$, respectively. Furthermore, their x-coordinates are either $0, 1/3, 1/2, 2/3$ and hence multiples of $\varepsilon_c^2/2$. Consequently, this packing satisfies Property 3.1. The packing of bin C_3



(a) Structure of the packing in bin C_1

(b) Structure of the packing in bin C_2 and C_3

Figure 3.10: The structure of the additional bins of Lemma 3.8

is analogous to the packing of C_2 with rectangles $r_C^{(5)}, r_C^{(6)}, r_B^{(2)}, r_B^{(5)}$ and $r_B^{(6)}$.

We modify each sequence of 6 of the k bins. If $\ell \leq 4$ bins remain, we pack the rectangles that intersect S_L into ℓ additional bins. In this case, we need in total $3/2(k-\ell)+2\ell = 3/2k+\ell/2 \leq 3/2k+2$. If $\ell = 5$ bins remain, we adopt the same packing as described above without the rectangles $r_A^{(6)}, r_B^{(6)}$ and $r_C^{(6)}$. We need in total $3/2(k-\ell)+\ell+3 = 3/2k-(3/2\cdot 5)+8 \leq 3/2k+1$. The case analysis in this paragraph is also used in some of the following lemmas, we do not repeat it there. □

Analogously, we achieve the same result for the following corollary. By turning the bin by 90°, the proof is exactly the same. Consequently, we do not need this lemma, if we are allowed to rotate the rectangles.

Lemma 3.9. *Let $B_1, \dots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \dots, k\}$, has a packing with the following conditions:*

2.4. $r_{u\ell}^{(i)} \notin L_U^{(i)}$ *or* $r_{br}^{(i)} \notin L_B^{(i)}$,

2.5. $h^{(i)} \in \{0/24, \dots, 7/24, 8/24 - \varepsilon_c\}$,

2.6. $v^{(i)} \in \text{IN}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2k+2$ *bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.*

The following lemma covers the case that there is a bin B_i with $r_{u\ell}^{(i)} \in L_U^{(i)}, r_{br}^{(i)} \in L_B^{(i)}, r_{ur}^{(i)} \in W_R^{(i)}$ and $r_{b\ell}^{(i)} \in W_L^{(i)}$.

Lemma 3.10. *Let $B_1, \dots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \dots, k\}$, has a packing with the following conditions:*

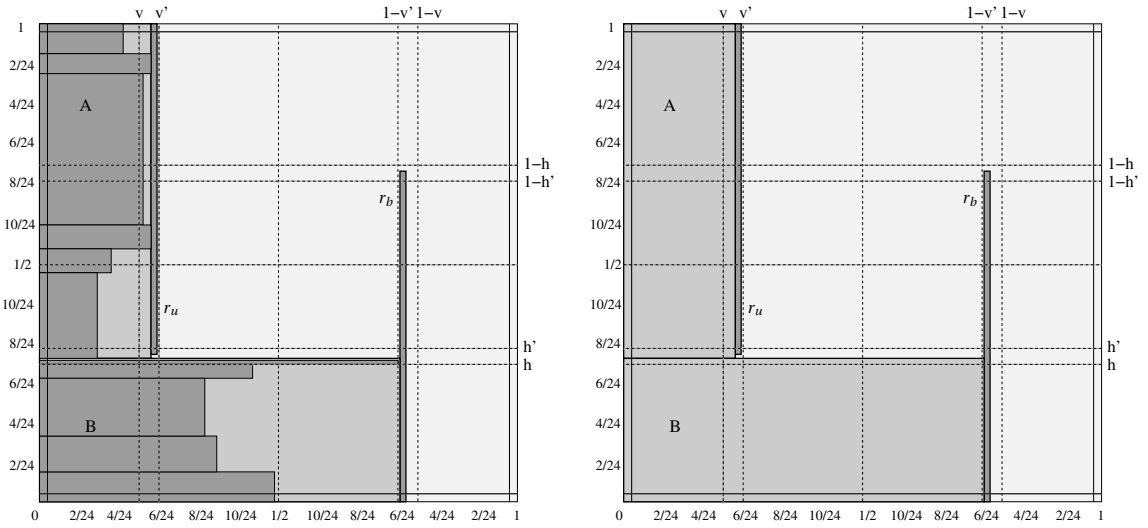
2.7. $h^{(i)} \in \{0/24, \dots, 7/24\}$,

2.8. $v^{(i)} \in \{0/24, \dots, 7/24, 8/24 - \varepsilon_c\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2k+1$ *bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.*

Proof. Let $i \in \{1, \dots, k\}$. W.l.o.g. we assume that $x_u'^{(i)} \leq 1 - x_b^{(i)}$ since we are able to turn the bin by 180°. We want to remove the rectangles intersecting the strip $S_L^{(i)}$ in each bin.

Similar to the proof of Lemma 3.8 we use containers/rectangles for grouping the rectangles in $I_L^{(i)}$. If $r_{u\ell}^{(i)} \in L_U^{(i)}$ and $r_{u\ell}^{(i)} = r_u^{(i)}$, i.e. the rectangle in the upper left corner is the largest rectangle in $L_U^{(i)}$, the rectangle $r_A^{(i)}$ is $r_{u\ell}^{(i)}$. In any other case, let $A^{(i)} \subset I_L^{(i)}$ be the set of rectangles that are at the left of $r_u^{(i)}$, i.e. the rectangles in $I_L^{(i)}$ that lie above the horizontal line at height $y_u^{(i)}$ and that intersect with it. In this case, we define $r_A^{(i)}$ as a rectangle of height $h_A^{(i)} = h(A^{(i)})$ and width $w_A^{(i)} = w_{\max}(A^{(i)})$. The width of $r_A^{(i)}$ is limited in both cases to at most $w_A^{(i)} \leq x_u'^{(i)} \leq v'^{(i)} \leq 8/24 = 1/3$ (Condition 2.8). Since $r_u^{(i)}$ intersects the horizontal interval $\text{HB}_{h^{(i)}}^{(i)}$, the height of $r_A^{(i)}$ is at least $h_A^{(i)} \geq 1 - h'^{(i)} \geq 1 - (8/24 - \varepsilon_c) = 16/24 + \varepsilon_c = 2/3 + \varepsilon_c$ (Condition 2.7).



(a) Rectangles intersecting $S_L^{(i)}$

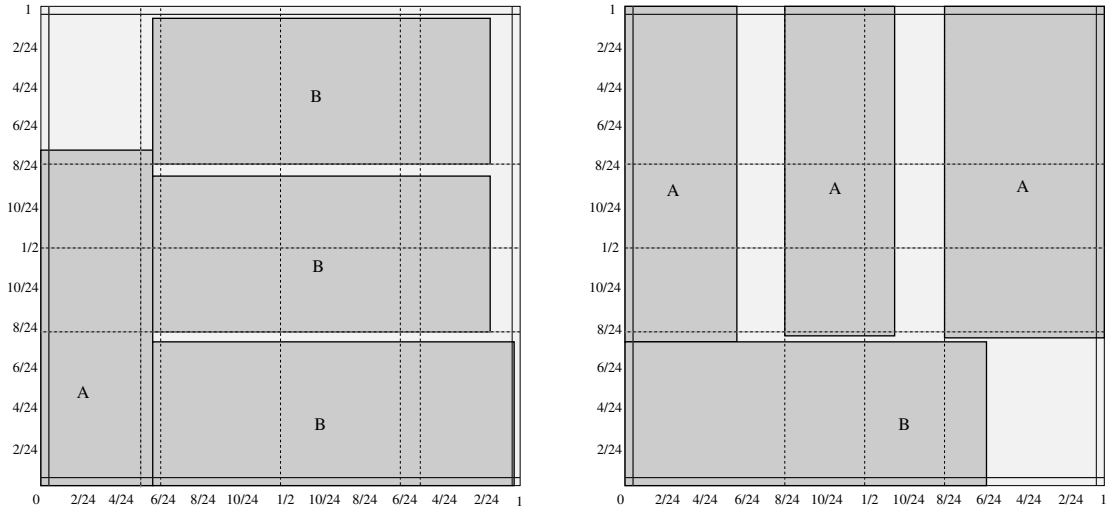
(b) Construction of rectangles $r_A^{(i)}$ and $r_B^{(i)}$

Figure 3.11: A possible initial situation of Lemma 3.10

The widths of the rectangles in $W_L^{(i)}$ are bounded by the leftmost rectangle of $L_B^{(i)}$. Hence, their widths are at most $x_b^{(i)} \leq 1 - x_u'^{(i)}$. We use a set $B^{(i)} = I_L^{(i)} \setminus A^{(i)}$ of the remaining rectangles that intersect $S_L^{(i)}$. Analogously, we define a rectangle $r_B^{(i)}$ for these rectangles of width $w_B^{(i)} = w_{\max}(B^{(i)}) \leq x_b^{(i)} \leq 1 - x_u'^{(i)}$ and height $h_B^{(i)} = 1 - h_A^{(i)} \leq 1/3 - \varepsilon_c$. Note that $w_A^{(i)} + w_B^{(i)} \leq x_u'^{(i)} + (1 - x_u'^{(i)}) = 1$ and therefore the rectangles $r_A^{(i)}$ and $r_B^{(i)}$ fit next to each other in one bin (cf. Figure 3.11).

In order to employ Lemma 3.3, we pack $r_A^{(i)}$ and $r_B^{(i)}$ and with them all rectangles that intersect with $S_L^{(i)}$ into an additional bin. To this end, we pack the rectangles of each sequence of 4 bins $B_1, \ldots, B_4$ of the k bins into 2 additional bins C_1 and C_2. W.l.o.g. we assume that $r_A^{(1)}$ is the rectangle with the minimum width among the rectangles $r_A^{(1)}, \ldots, r_A^{(4)}$ and $r_B^{(2)}$ is the rectangle with the minimum height among the rectangles $r_B^{(2)}, r_B^{(3)}, r_B^{(4)}$.

We pack $r_A^{(1)}$ in the lower left corner of C_1 on the position $(0,0)$. Since $w_A^{(1)} \leq w_A^{(3)} \leq 1 - w_B^{(3)}$ and $w_A^{(1)} \leq w_A^{(4)} \leq 1 - w_B^{(4)}$, we are able to pack the rectangles $r_B^{(1)}, r_B^{(3)}$ and $r_B^{(4)}$ on the right side of $r_A^{(1)}$. These three rectangles each have a height of at most $1/3 - \varepsilon_c$ and thus we are able to place them on top of each other on the positions $(w_A^{(1)}, 0), (w_A^{(1)}, 1/3)$ and $(w_A^{(1)}, 2/3)$. On top of $r_B^{(4)}$ there is still a free space of height ε_c. The rectangle $r_A^{(1)}$ also has a height of at most $1 - \varepsilon_c$, as there would otherwise be no rectangle in $W_L^{(1)}$. Consequently, the uppermost strip of height ε_c is free and we are able to employ Lemma 3.3 on C_1 (cf. Figure 3.12(a))



(a) Structure of the packing in bin C_1

(b) Structure of the packing in bin C_2

Figure 3.12: The structure of the additional bins of Lemma 3.10

The remaining rectangles $r_A^{(2)}, r_A^{(3)}, r_A^{(4)}$ and $r_B^{(2)}$ have to be packed into bin C_2. It holds that $h_A^{(i)} + h_B^{(i)} = 1$ and therefore $h_B^{(2)} \leq h_B^{(3)} = 1 - h_A^{(3)}$ and $h_B^{(2)} \leq h_B^{(4)} = 1 - h_A^{(4)}$. Thus, the rectangles $r_A^{(2)}, r_A^{(3)}$ and $r_A^{(4)}$ each fit above $r_B^{(2)}$. The widths of $r_A^{(2)}, r_A^{(3)}$ and $r_A^{(4)}$ are at most $1/3$. Hence, we are able to place $r_B^{(2)}$ on the position $(0,0)$, $r_A^{(2)}$ on the position $(0, h_B^{(2)})$, $r_A^{(3)}$ on the position $(1/3, h_B^{(2)})$ and $r_A^{(4)}$ on the position $(2/3, h_B^{(2)})$(cf. Figure 3.12(b)). A horizontal or vertical strip free of rectangles does not necessarily have to exist in this bin. However, there is no rectangle on the right of $r_B^{(2)}$ and therefore we are able to round up the rectangle $r_B^{(2)}$ and the rectangles inside it to the next largest multiple of $\varepsilon_c^2/2$. The rectangles $r_A^{(2)}, r_A^{(3)}$ and $r_A^{(4)}$ are positioned on x-coordinates $0, 1/3$ and $2/3$ that are multiples of $\varepsilon_c^2/2$. We are able to round up the widths of these rectangles to the next largest multiple of $\varepsilon_c^2/2$ to at most $1/3$. If $r_A^{(i)}$ is a container, the rectangles within it are therefore also positioned on a multiple of $\varepsilon_c^2/2$ and can be rounded up to the next largest multiple of $\varepsilon_c^2/2$. Consequently, this packing satisfies Property 3.1.

We do this for each sequence of 4 of the k bins. Let $\ell \leq 3$ denote the number of remaining bins. If $\ell \leq 2$ we employ an additional bin for each of the ℓ bins and we pack the rectangles of the strip $S_L^{(i)}$ into it. We have $3/2(k-\ell)+2\ell = 3/2 \cdot k + \ell/2 \leq 3/2 \cdot k + 1$ bins in total. If $\ell = 3$ we pack them according to the method described above without the rectangles $r_A^{(4)}$ and $r_B^{(4)}$ and use 2 additional bins. We obtain in this case $3/2(k-\ell)+\ell+2 = 3/2 \cdot k - \ell/2 + 2 \leq 3/2 \cdot k + 1$ bins. □

The lemma described above does not work for $h^{(i)} = 8/24 - \varepsilon_c$ since the rectangles $r_B^{(i)}$ might have a height close to $1/3$ and hence the uppermost strip in bin C_1 is not free. Therefore, we have to use a slight modification.

Lemma 3.11. *Let $B_1, \ldots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \ldots, k\}$, has a packing with the following conditions:*

2.9. $r_{ur}^{(i)} \in W_R^{(i)}$ *and* $r_{b\ell}^{(i)} \in W_L^{(i)}$,

2.10. $h^{(i)} = 8/24 - \varepsilon_c$,

2.11. $v^{(i)} \in \{0/24, \ldots, 7/24, 8/24 - \varepsilon_c\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $(3/2 + \varepsilon_c) \cdot k + 2$ bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.

Proof. Let $i \in \{1, \ldots, k\}$. We use the same packing as in the proof of Lemma 3.10. The

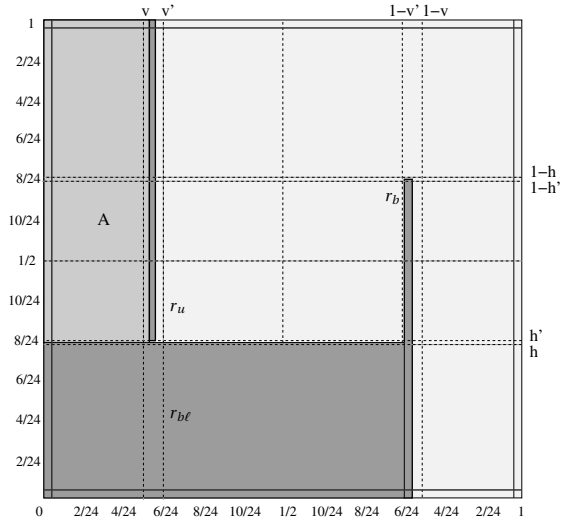


Figure 3.13: Initial packing of Lemma 3.11

difference is that, after possibly turning the bin by 180°, we have $r_{b\ell}^{(i)} \in W_L^{(i)}$ and hence the y'-coordinate of $r_{b\ell}^{(i)}$ is within $\mathrm{HB}_{8/24-\varepsilon_c}^{(i)}$ (Condition 2.10). Furthermore, the rectangle $r_u^{(i)}$ also intersects $\mathrm{HB}_{8/24-\varepsilon_c}^{(i)}$ and has its y-coordinate within it. All rectangles in $I_L^{(i)}$ that are above the horizontal line at height $y_u^{(i)}$ and that intersect it, belong to $A^{(i)}$. Consequently, the rectangles in $B^{(i)}$ consist of $r_{b\ell}^{(i)}$ and rectangles of a total height of at most ε_c. We move

these rectangles of the total height at most ε_c into additional bins by packing them on top of each other at the x-coordinate 0. For all k bins, we need at most $\lceil \varepsilon_c \cdot k \rceil \leq \varepsilon_c \cdot k + 1$ additional bins. We are able to round up the widths to the next largest multiple of $\varepsilon_c^2/2$ and satisfy Property 3.1.

At this moment, we have $B^{(i)} = \{r_{b\ell}^{(i)}\}$ and thus $r_B^{(i)} = r_{b\ell}^{(i)}$. We adopt the same packing as in the proof of Lemma 3.10. We round the heights of the rectangles $r_{b\ell}^{(1)}, r_{b\ell}^{(3)}$ and $r_{b\ell}^{(4)}$ in the bin C_1 to the next largest multiple of $\varepsilon_c^2/2$ which is at most $1/3$. The rounding of the remaining rectangles is the same as in the proof of Lemma 3.10. In total, we have $(3/2 + \varepsilon_c) \cdot k + 2$ bins. □

This finishes the case analysis for the values $v^{(i)} < 8/24 = 1/3$ and $h^{(i)} < 8/24 = 1/3$. What is left is the case in that the rectangles of a height larger than $1/2$ or the rectangles of a width larger than $1/2$ are situated close to the middle of the bin.

Intervals in the Middle

If we have $h^{(i)} \in \{0/24, \ldots, 7/24, 8/24 - \varepsilon_c\}$ and $v^{(i)} \in \{0/24, \ldots, 7/24, 8/24 - \varepsilon_c\}$, we are able to use one of the lemmas above. Hence, the packing in the remaining bins has at least one of the two values $v^{(i)}$ and $h^{(i)}$ in the set $\{8/24, \ldots, 11/24, 12/24 - \varepsilon_c\}$.

Lemma 3.12. *Let $B_1, \ldots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \ldots, k\}$, has a packing with the following conditions:*

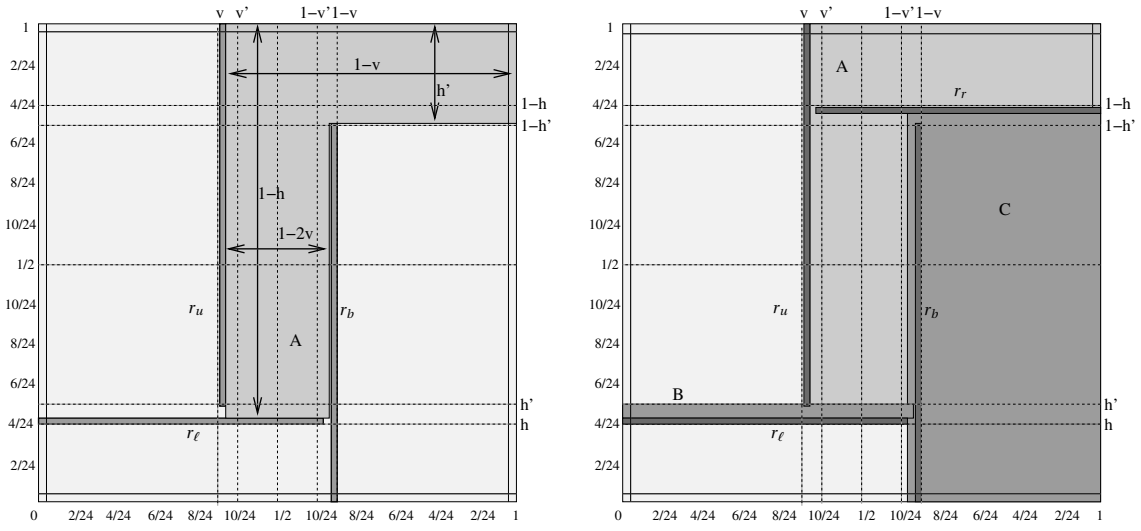
2.12. $v^{(i)} \in \{8/24, 9/24, 10/24, 11/24\}$,

2.13. $h^{(i)} \in \{2/24, \ldots, 9/24\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2k+1$ bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.

Proof. Let $i \in \{1, \ldots, k\}$. We want to move the rectangles in the middle of the bin B_i to an additional bin, in order to free $S_R^{(i)}$. Let w.l.o.g. $y_\ell'^{(i)} \leq 1 - y_r^{(i)}$ since we are able to turn the bin by 180°.

Define a non-rectangular, Γ-shaped region $A^{(i)}$, consisting of the rectangles in the middle and upper right side of B_i. We define this region along the rectangles $r_u^{(i)}, r_\ell^{(i)}, r_b^{(i)}$ and the right and top of the bin, in order to ensure that there are only few rectangles that intersect this region from the sides. The region $A^{(i)}$ is defined by the coordinates $(x_u'^{(i)}, y_\ell'^{(i)}), (x_b^{(i)}, y_\ell'^{(i)}), (x_b^{(i)}, y_b'^{(i)}), (1, y_b'^{(i)}), (1,1)$ and $(x_u'^{(i)}, 1)$ (cf. Figure 3.14(a)). We treat this region with all rectangles that are completely situated inside it as one object $o_A^{(i)}$ and move this object to an additional bin. The heights and widths are bounded as follows:



(a) Definition of region $A^{(i)}$

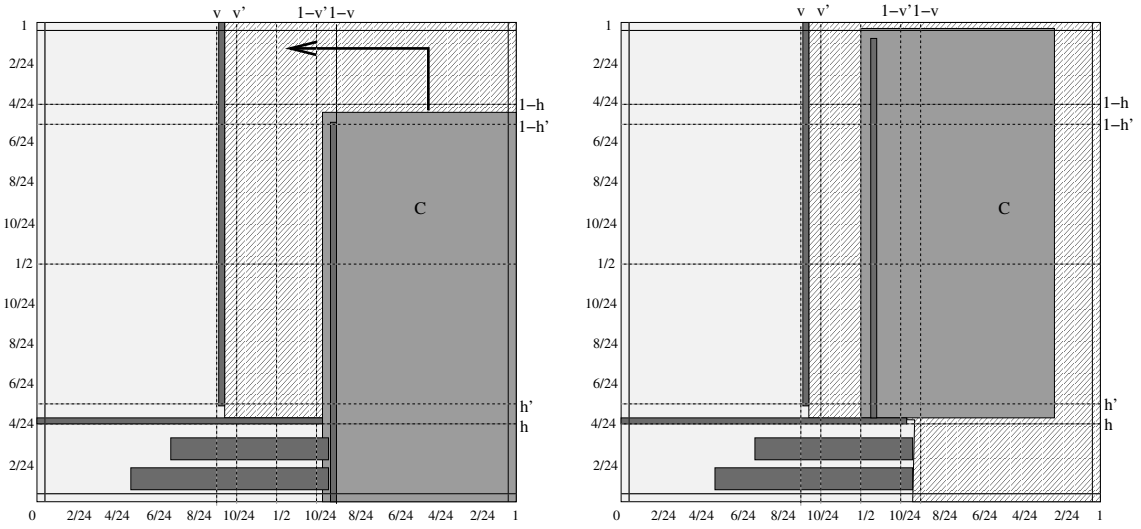
(b) Definition of region $B^{(i)}$ and $C^{(i)}$

Figure 3.14: The definition of the regions of Lemma 3.12

The longer, left side has a height of $1 - y_\ell'^{(i)} \leq 1 - h^{(i)} \leq 1 - 2/24 = 22/24$, the shorter, right side a height of at most $1 - y_b'^{(i)} \leq 1 - (1 - h'^{(i)}) = h'^{(i)} \leq 10/24$ (Condition 2.12). The lower part of this object has a width of at most $x_b^{(i)} - x_u'^{(i)} \leq (1 - v^{(i)}) - v^{(i)} = 1 - 2v^{(i)} \leq 8/24 = 1/3$, the upper part one of $1 - x_u'^{(i)} \leq 1 - v^{(i)} \leq 1 - 8/24 = 16/24 = 2/3$ (Condition 2.13).

It is possible that there are rectangles that are not completely located inside this region, but intersect it from below or from the left. The rectangles that intersect it from the left have to be situated between the rectangles $r_\ell^{(i)}$ and $r_u^{(i)}$ since we defined the region $A^{(i)}$ along the right side of rectangle $r_u^{(i)}$. The rectangles $r_\ell^{(i)}$ and $r_u^{(i)}$ intersect both with $\mathrm{HB}_{h^{(i)}}^{(i)}$, hence the total height of these rectangles is bounded by $1/24$. We call the set of these rectangles $B^{(i)}$ and pack them into a container/rectangle $r_B^{(i)}$ of height $h_B^{(i)} = h(B^{(i)}) \leq 1/24$ and width $w_B^{(i)} = w_{\max}(B^{(i)}) \leq x_b^{(i)} \leq 1 - v^{(i)} \leq 1 - 8/24 = 16/24 = 2/3$ (cf. Figure 3.14(b)).

The rectangles that intersect $A^{(i)}$ from below are situated on the right of the x'-coordinate $x_\ell'^{(i)}$ since $A^{(i)}$ is defined along $r_\ell^{(i)}$. Furthermore, these rectangles are all bounded by the rectangle $r_r^{(i)}$ that is completely situated inside $A^{(i)}$. Therefore, we define a region $C^{(i)}$ that contains all rectangles that intersect $A^{(i)}$ from below by the coordinates $(x_\ell'^{(i)}, 0), (1, 0), (1, y_r^{(i)})$ and $(x_\ell'^{(i)}, y_r^{(i)})$. There is no rectangle that intersects $C^{(i)}$ from above since we defined it along the lower edge of $r_r^{(i)}$. The rectangles that intersect region $C^{(i)}$ from the left above the rectangle $r_\ell^{(i)}$ are completely situated inside $A^{(i)}$. The rectangles that intersect $C^{(i)}$ from the left below the rectangle $r_\ell^{(i)}$ are bounded by the rectangle $r_b^{(i)}$. Hence, they do not intersect



(a) Moving $r_C^{(i)}$ on top of $r_\ell^{(i)}$

(b) The strip $S_R^{(i)}$ is completely free

Figure 3.15: Moving the region $C^{(i)}$ in the proof of Lemma 3.12

$S_R^{(i)}$ and we do not move these rectangles. We treat the region $C^{(i)}$ as one container/rectangle $r_C^{(i)}$ of height $h_C^{(i)} = y_r^{(i)} \leq 1 - y_\ell'^{(i)}$ and width $w_C^{(i)} = 1 - x_\ell'^{(i)} \leq 1 - (1 - v'^{(i)}) \leq 12/24 - \varepsilon_c$ that contains all rectangles completely situated inside this region.

We move the objects $o_A^{(i)}$ and $r_B^{(i)}$ into an additional bin, while $r_C^{(i)}$ is moved inside B_i. Without these three objects the region $(x_u'^{(i)}, y_\ell'^{(i)}), (1, y_\ell'^{(i)}), (1,1)$ and $(x_u'^{(i)}, 1)$ at the right side of $r_u^{(i)}$ is completely free of rectangles. It is $h_C^{(i)} \leq 1 - y_\ell'^{(i)}$ and $w_C^{(i)} \leq 12/24 - \varepsilon_c$. Thus, we can place $r_C^{(i)}$ on the position $(1/2, y_\ell'^{(i)})$ leaving the left strip $S_R^{(i)}$ completely free of rectangles (cf. Figure 3.15). We employ Lemma 3.3 on this bin in order to round up the rectangles and to satisfy Property 3.1.

Let there be two bins B_1 and B_2 of the k bins and let C_1 be an additional empty bin. We pack the objects $o_A^{(1)}, o_A^{(2)}, r_B^{(1)}$ and $r_B^{(2)}$ into C_1.

We place $r_B^{(1)}$ at the bottom of the bin C_1 on the position $(1/3, 0)$. The object $o_A^{(1)}$ is turned by 180° so that the long edge is at the bottom. We pack this object on top of $r_B^{(1)}$ at position $(1/3, 1/24)$. Both objects occupy the region$(1/3, 0), (1, 0), (1, 23/24), (2/3, 23/24), (2/3, 11/24)$ and $(1/3, 11/24)$.

The object $o_A^{(2)}$ is placed on top of $o_A^{(1)}$ with the top edge at height $22/24$. It occupies the region $(0,0), (1/3, 0)$, $(1/3, 12/24), (2/3, 12/24), (2/3, 22/24)$ and $(0, 22/24)$ (cf. Figure 3.16). These regions do not overlap. On top of $o_A^{(2)}$, there is still a free space of width $2/3$ and height $2/24$. In this space we place $r_B^{(2)}$ on position $(0, 22/24)$. It follows that there is a strip

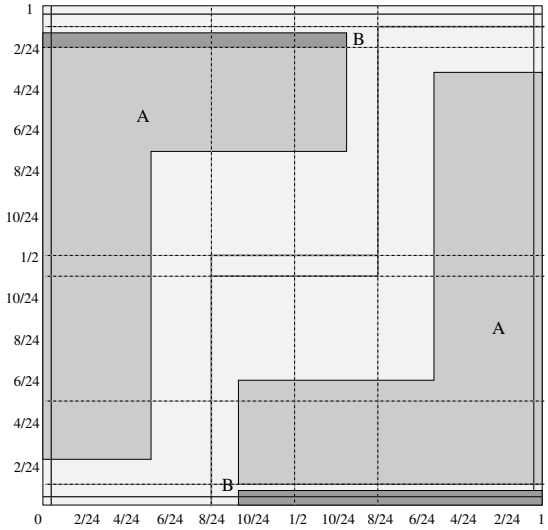


Figure 3.16: Packing of the additional bins in the proof of Lemma 3.12

of the height $1/24$ free of rectangles including the strip S_U. This allows us to use Lemma 3.4 in order to round up the rectangles and to satisfy Property 3.2.

We repeat this step with each sequence of 2 of the k bins and achieve a packing of $3/2k+1$ bins in total, when k is odd. □

The analogous lemma by exchanging the values $h^{(i)}$ and $v^{(i)}$ is as follows. However, in the version that allows rotation we do not need this lemma, turn the packing by 90° to adopt Lemma 3.12 instead.

Lemma 3.13. *Let $B_1,\dots,B_k$ be k bins so that each bin B_i, for $i \in \{1,\dots,k\}$, has a packing with the following conditions:*

2.14. $h^{(i)} \in \{8/24, 9/24, 10/24, 11/24\}$,

2.15. $v^{(i)} \in \{2/24,\dots,9/24\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2k+1$ bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.

In the following lemma we use the same technique as in the lemma above. We also move the region in the middle and in the upper right corner into an additional bin. The region in the lower right corner is shifted in the same way as in the Lemma 3.12 above. Furthermore, we use the fact that the values $v^{(i)}$ and $h^{(i)}$ are in the same range.

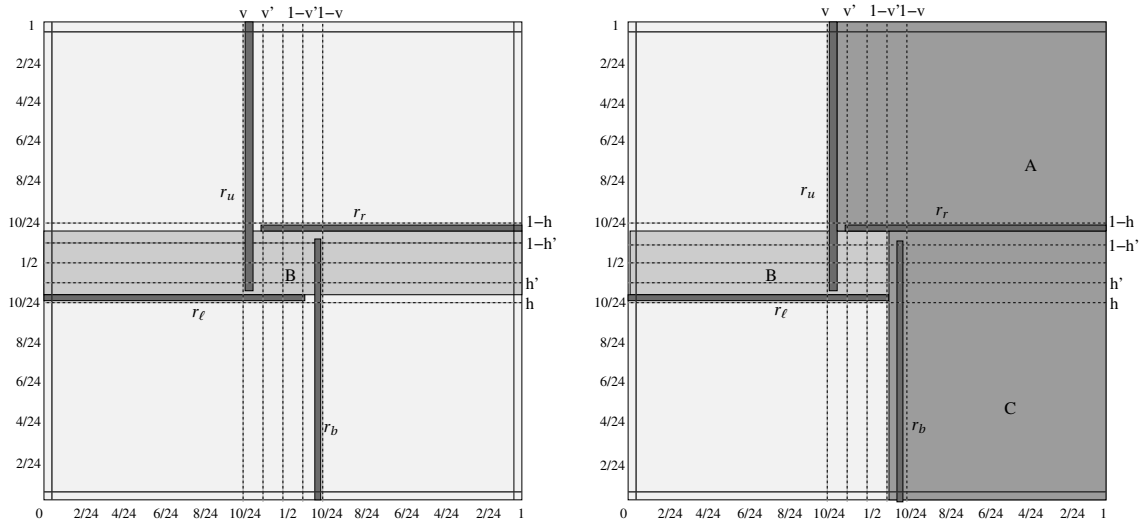
Lemma 3.14. *Let $B_1, \dots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \dots, k\}$, has a packing with the following conditions:*

2.16. $h^{(i)} \in \{10/24, 11/24\}$,

2.17. $v^{(i)} \in \{10/24, 11/24\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2k+1$ *bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.*

Proof. Let $i \in \{1, \dots, \lfloor k/2 \rfloor\}$ and $j \in \{\lfloor k/2 \rfloor + 1, \dots, k\}$ and let w.l.o.g. $y_\ell'^{(i)} \leq 1 - y_r^{(i)}$ and $x_b^{(j)} \leq 1 - x_u'^{(j)}$ since we are able to turn the bin B_i or B_j by 180°. We use a similar region definition as in Lemma 3.12.

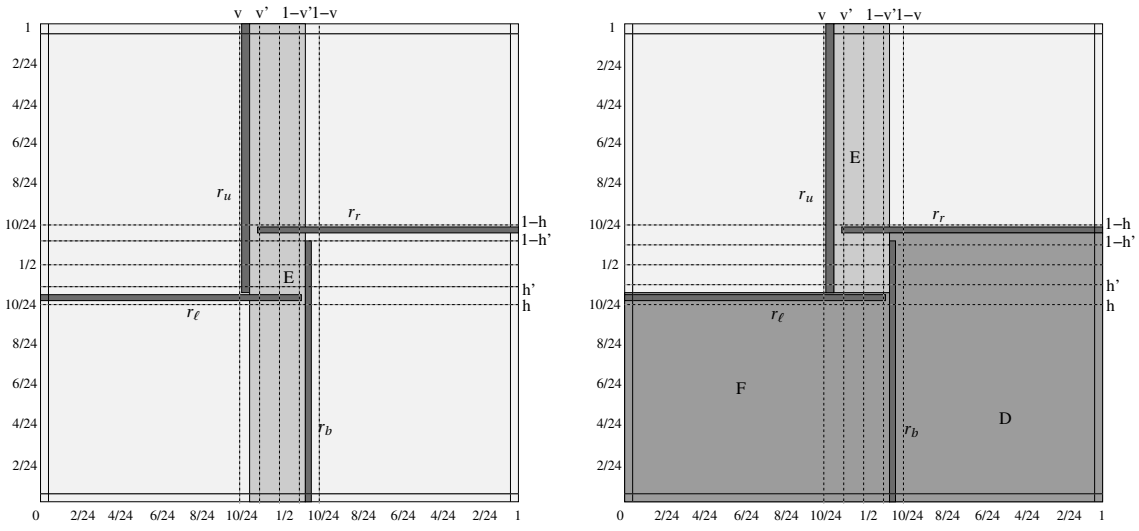


(a) Definition of the region $B^{(i)}$

(b) Definition of region $A^{(i)}$ and $C^{(i)}$

Figure 3.17: Definitions of the regions of the first $\lfloor k/2 \rfloor$ bins in the proof of Lemma 3.14

Let $A^{(i)}$ be the region defined by $(x_u'^{(i)}, y_r^{(i)}), (1, y_r^{(i)}), (1,1)$ and $(x_u'^{(i)}, 1)$. The region $B^{(i)}$ in the middle of bin B_i is defined by $(0, y_\ell'^{(i)}), (1, y_\ell'^{(i)}), (1, y_r^{(i)})$ and $(0, y_r^{(i)})$. The region $C^{(i)}$ is defined by $(x_\ell'^{(i)}, 0), (1,0), (1, y_r^{(i)})$ and $(x_\ell'^{(i)}, y_r^{(i)})$ (cf. Figure 3.17). Again, we treat these regions as containers/rectangles. $r_A^{(i)}$ has height $h_A^{(i)} \leq 1 - y_r^{(i)} \leq 1 - (1 - h'^{(i)}) = h'^{(i)} \leq 12/24 - \varepsilon_c$ (Condition 2.17) and width $w_A^{(i)} \leq 1 - x_u'^{(i)} \leq 1 - v^{(i)} \leq 1 - 10/24 = 14/24$ (Condition 2.16). The width of $r_B^{(i)}$ is $w_B^{(i)} = 1$ and the height is $h_B^{(i)} = y_r^{(i)} - y_\ell'^{(i)} \leq (1 - h^{(i)}) - h^{(i)} = 1 - 2h^{(i)} \leq 1 - 20/24 = 4/24$ (Condition 2.16). The rectangle $r_C^{(i)}$ has width $w_C^{(i)} = 1 - x_\ell'^{(i)} \leq 1 - (1 - h'^{(i)}) = h'^{(i)} \leq 12/24 - \varepsilon_c$ (Condition 2.16) and height $h_C^{(i)} = y_r^{(i)} \leq 1 - y_\ell'^{(i)}$. All rectangles that



(a) Definition of region $E^{(i)}$

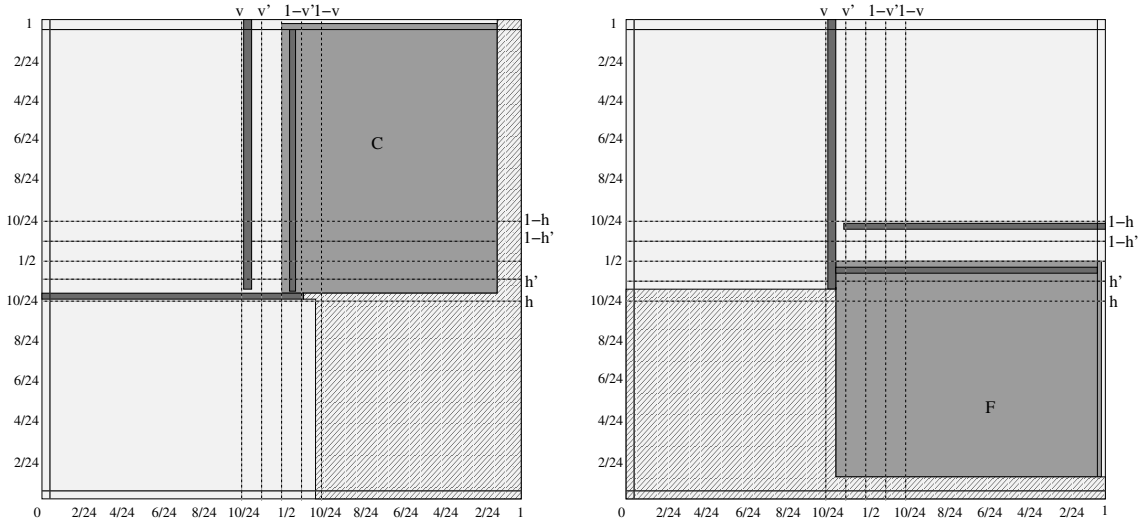
(b) Definition of region $D^{(i)}$ and $F^{(i)}$

Figure 3.18: Definitions of the regions of the last $\lceil k/2 \rceil$ bins in the proof of Lemma 3.14

intersect the region $B^{(i)}$ from below are situated at the right of the rectangle $r_\ell^{(i)}$. Hence, these rectangles are all completely situated inside the region $C^{(i)}$. The rectangles that intersect $B^{(i)}$ from above have to be on the left of rectangle $r_r^{(i)}$. Thus, they are completely situated at the left of the vertical line at x-coordinate $1/2$. These rectangles are not moved. This holds especially for the rectangles that intersect the region $A^{(i)}$ from below between the rectangles $r_u^{(i)}$ and $r_r^{(i)}$. There are no further rectangles that intersect $A^{(i)}$ while they are not completely situated inside this region. The rectangles that intersect region $C^{(i)}$ are either completely situated inside region $B^{(i)}$ or are situated below $r_\ell^{(i)}$ and thus bounded by rectangle $r_b^{(i)}$. The values of the x'-coordinates of these rectangles is at most $x_b^{(i)}$. We do not move these rectangles.

We define similar regions in the bin B_j. The only difference is that they are rotated by 90° (cf. Figure 3.18). Let $D^{(j)}$ be the region defined by the coordinates $(x_b^{(j)},0),(1,0),(1,y_r^{(j)})$ and $(x_b^{(j)},y_r^{(j)})$, $E^{(j)}$ is defined by the coordinates $(x_u'^{(j)},0),(x_b^{(j)},0),(x_b^{(j)},1)$ and $(x_u'^{(j)},1)$. The region $F^{(j)}$ is defined by the coordinates $(0,0),(x_b^{(j)},0),(x_b^{(j)},y_u^{(j)})$ and $(0,y_u^{(j)})$. The values for the heights and the widths of the corresponding rectangles $r_D^{(j)}, r_E^{(j)}$ and $r_F^{(j)}$ are the same as the values for $r_A^{(i)}, r_B^{(i)}$ and $r_C^{(i)}$ by exchanging the widths and the heights. Consequently, $w_D^{(j)} \leq 12/24 - \varepsilon_c, h_D^{(j)} \leq 14/24, w_E^{(j)} \leq 4/24, h_E^{(j)} = 1, w_F^{(j)} = x_b^{(j)} \leq 1 - x_u'^{(j)}$ and $h_F^{(j)} \leq 12/24 - \varepsilon_c$.

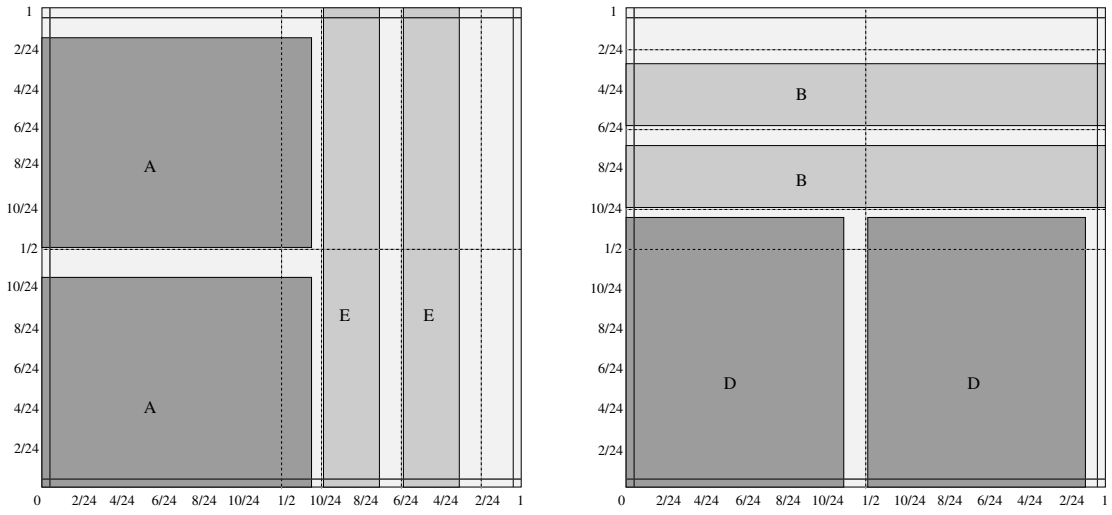
In a first step, we move the regions out of the bins. Analogously to Lemma 3.12, we place



(a) Packing in the first $\lfloor k/2 \rfloor$ bins

(b) Packing in the last $\lceil k/2 \rceil$ bins

Figure 3.19: Packing of the bins in the proof of Lemma 3.14



(a) Packing in the additional bin C_1

(b) Packing in the additional bin C_2

Figure 3.20: Packing of the additional bins in the proof of Lemma 3.14

$r_C^{(i)}$ on the position $(1/2, y_\ell'^{(i)})$ and $r_F^{(j)}$ on the position $(x_u'^{(j)}, 1/2)$ (cf. Figure 3.19). Let there be two bins B_1, B_2 of the first $\lfloor k/2 \rfloor$ bins and two bins B_3, B_4 of the remaining bins. We move the rectangles $r_A^{(1)}, r_A^{(2)}$ and $r_E^{(3)}, r_E^{(4)}$ into an additional bin C_1 and the rectangles $r_D^{(3)}, r_D^{(4)}$ and $r_B^{(1)}, r_B^{(2)}$ into an additional bin C_2 (cf. Figure 3.20).

We pack $r_A^{(1)}$ in the lower left corner of bin C_1 on the position $(0,0)$ and $r_A^{(2)}$ on top of it on the position $(0, 1/2)$. The widths of both rectangles are at most $14/24$. On the right side there is enough space to place $r_E^{(3)}$ and $r_E^{(4)}$. These rectangles have a total width of $8/24$. On the right side there is still a free space of width $2/24$ and hence the strip S_R is completely free of rectangles.

The packing in C_2 is analogous. $r_D^{(3)}$ and $r_D^{(4)}$ are placed next to each other at the bottom of bin C_2 on the positions $(0,0)$ and $(1/2, 0)$. On top of them there is a free space of at least $10/24$. We place $r_B^{(1)}$ and $r_B^{(2)}$ on top of them, leaving the strip S_U free of rectangles. We repeat this method with each sequence of 4 of the k bins. Having observed the same results as in the last paragraph of the proof of Lemma 3.10, we need $3/2k + 1$ bins in total, when k is not a multiple of 4. In each bin there is either a horizontal or a vertical strip free of rectangles, hence we are able to employ Lemma 3.3 or Lemma 3.4 to satisfy either Property 3.1 or Property 3.2. □

The next lemma considers the case that the horizontal intervals are close to the bottom and to the top of the bin while the vertical intervals are close to the middle.

Lemma 3.15. *Let $B_1, \ldots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \ldots, k\}$, has a packing with the following conditions:*

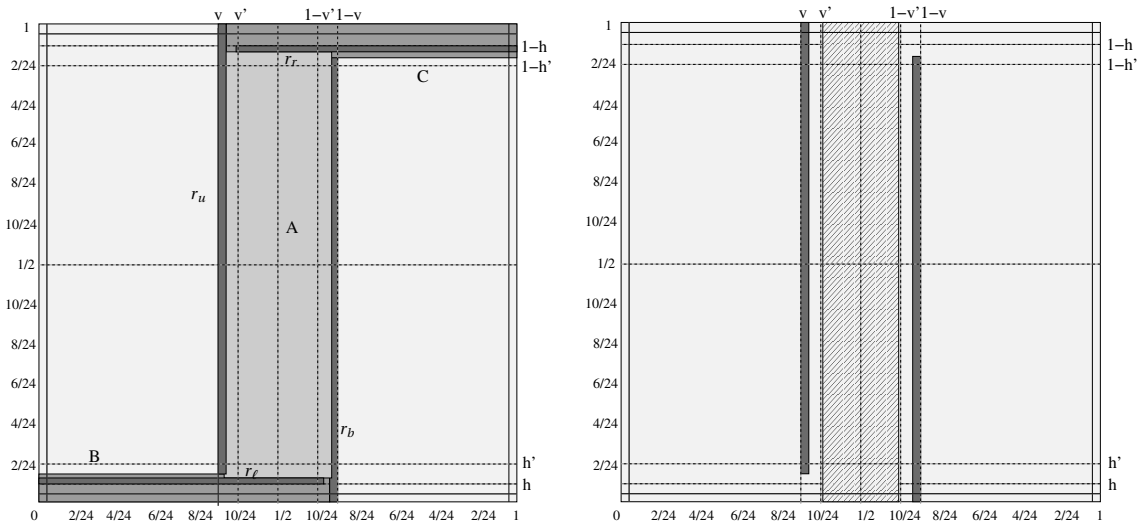
2.18. $v^{(i)} \in \{8/24, 9/24, 10/24, 11/24\}$,

2.19. $h^{(i)} \in \{0/24, 1/24\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2 \cdot k + 2$ bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.

Proof. Let $i \in \{1, \ldots, k\}$. We use a similar region definition as in the lemmas before. The height of the region in the middle is very large, but the regions at the top and at the bottom are small.

Let $A^{(i)}$ be the rectangular region defined by the points $(x_u'^{(i)}, y_\ell'^{(i)}), (x_b^{(i)}, y_\ell'^{(i)}), (x_b^{(i)}, y_r^{(i)})$ and $(x_u'^{(i)}, y_r^{(i)})$. The corresponding rectangle $r_A^{(i)}$ of this region has a width of $w_A^{(i)} = x_b^{(i)} - x'^{(i)}_u \leq 1 - v^{(i)} - v^{(i)} \leq 16/24 - 8/24 = 1/3$ (Condition 2.18). The rectangles $r_\ell^{(i)}$ and $r_r^{(i)}$ are not completely situated inside the strips $S_B^{(i)}$ and $S_U^{(i)}$, as they would have been removed in Lemma 3.1. It follows that the height of $r_A^{(i)}$ is bounded by $h_A^{(i)} = y_r^{(i)} - y_\ell'^{(i)} \leq 1 - 2\varepsilon_c \leq 1 - \varepsilon_c$.



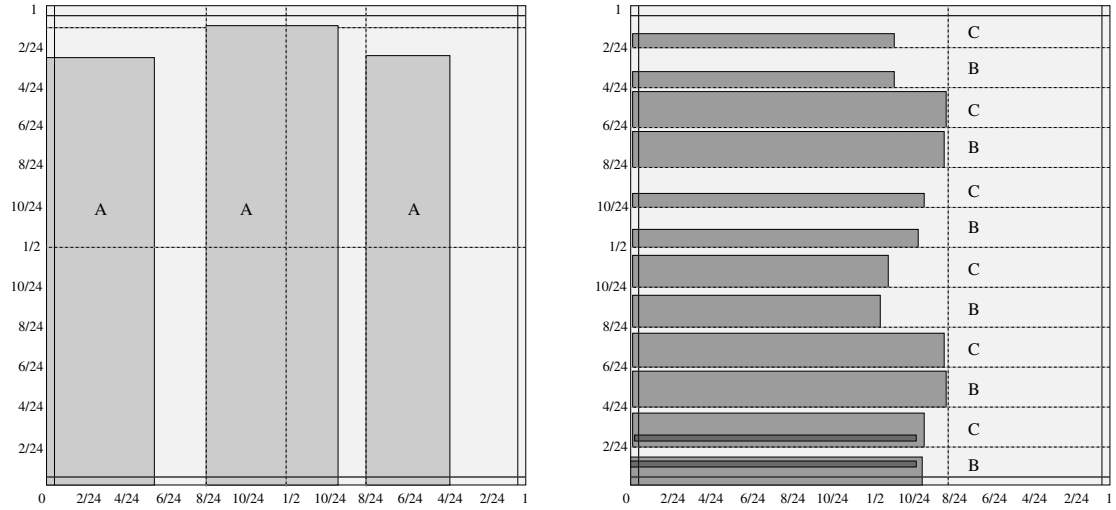
(a) Definition of the regions $A^{(i)}$,$B^{(i)}$ and $C^{(i)}$ (b) The whole strip in the middle is free of rectangles

Figure 3.21: Definitions of the regions in the proof of Lemma 3.15

In the next step, we define the region $B^{(i)}$ by the coordinates $(0,0),(x_b^{(i)},0),(x_b^{(i)},y_u^{(i)}),(0,y_u^{(i)})$ and the region $C^{(i)}$ by the coordinates $(x_u'^{(i)},y_b'^{(i)}),(1,y_b'^{(i)}),(1,1),(x_u'^{(i)},1)$. The heights of the corresponding containers/rectangles $r_B^{(i)}$ and $r_C^{(i)}$ of these regions are bounded by $h_B^{(i)} = y_u^{(i)} - 0 \leq h'^{(i)} \leq 2/24$ and $h_C^{(i)} = 1 - y_b'^{(i)} \leq 1 - (1 - h'^{(i)}) = h'^{(i)} \leq 2/24$ (Condition 2.19). The widths are bounded by $w_B^{(i)} = x_b^{(i)} - 0 \leq 1 - v^{(i)} \leq 1 - 8/24 = 16/24 = 2/3$ and $w_C^{(i)} = 1 - x_u'^{(i)} \leq 1 - v^{(i)} \leq 1 - 8/24 = 16/24 = 2/3$ (Condition 2.18). We move these rectangles and with them all rectangles that are completely situated inside these regions into additional bins. There are some rectangles left that intersect the region $A^{(i)}$ and that are not completely situated in one of these regions. These rectangles have to intersect $A^{(i)}$ from above or below since all rectangles that intersect $A^{(i)}$ from the left are below $r_u^{(i)}$ and hence located inside $B^{(i)}$ and the rectangles that intersect from the right have to lie above $r_b^{(i)}$ and are hence situated inside $C^{(i)}$. The rectangles that intersect $A^{(i)}$ from below have to be located between $r_\ell^{(i)}$ and $r_b^{(i)}$; the rectangles that intersect from above have to be situated between $r_u^{(i)}$ and $r_r^{(i)}$. It follows that, after moving the rectangles $r_A^{(i)}, r_B^{(i)}$ and $r_C^{(i)}$ into additional bins, the complete vertical strip between $x_r^{(i)}$ and $x_\ell'^{(i)}$ is completely free of rectangles (cf. Figure 3.21). Since $x_\ell'^{(i)} \geq 1 - v'^{(i)} \geq 1 - (12/24 - \varepsilon_c) = 1/2 + \varepsilon_c$ and $x_r^{(i)} \leq v'^{(i)} \leq 12/24 - \varepsilon_c = 1/2 - \varepsilon_c$, the strip has a width of at least $2\varepsilon_c$. Thus, we can move all rectangles on the right of the x'-coordinate $x_\ell'^{(i)}$ by ε_c to the left and secure that the strip $S_R^{(i)}$ is completely free of rectangles.

We move the rectangles of each sequence of 6 bins $B_1,\ldots,B_6$ of the k bins into 3 addi-

tional bins C_1, C_2, C_3. The rectangles $r_A^{(1)}, r_A^{(2)}$ and $r_A^{(3)}$ are moved into bin C_1, the rectangles $r_A^{(4)}, r_A^{(5)}, r_A^{(6)}$ are moved into bin C_2 and the remaining rectangles into bin C_3. Each rectangle $r_A^{(i)}$ has a width of at most $1/3$. Thus, we place them on the positions $(0,0), (1/3,0)$ and $(2/3,0)$ into bins C_1 and C_2. The uppermost horizontal strip S_U of height ε_c is still free of rectangles. The total height of the rectangles $r_B^{(1)}, \ldots, r_B^{(6)}$ and the rectangles $r_C^{(1)}, \ldots, r_C^{(6)}$ is $6 \cdot (2/24 + 2/24) = 1$. We place them on top of each other into bin C_3 with their y-coordinates situated on position 0. Since they have a width of at most $2/3$ the strip S_R is still free (cf. Figure 3.22).



(a) Packing in the bin C_1 and C_2

(b) Packing in the bin C_3

Figure 3.22: Packing in the additional bins in the proof of Lemma 3.15

Accordingly to the discussion in the last paragraph of the proof of Lemma 3.8, we need $3/2k + 2$ bins in total, if k is not a multiple of 6. In each bin there is either a vertical strip of the width ε_c or a horizontal strip of the height ε_c free of rectangles. This allows us to employ Lemma 3.3 or Lemma 3.4 in order to round up the rectangles and to satisfy either Property 3.1 or Property 3.2. □

The analogous lemma is stated as follows. In the version that allows rotation, we only use Lemma 3.15.

Lemma 3.16. *Let $B_1, \ldots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \ldots, k\}$, has a packing with the following conditions:*

2.20. $h^{(i)} \in \{8/24, 9/24, 10/24, 11/24\}$,

2.21. $v^{(i)} \in \{0/24, 1/24\}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2 \cdot k+2$ *bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2.*

Before concluding our case analysis, there is one remaining case left in which either $h^{(i)} = 12/24 - \varepsilon_c$ or $v^{(i)} = 12/24 - \varepsilon_c$.

LAST REMAINING CASE

We distinguish, if the rectangles in the corners intersect the intervals that are very close to the middle of the bin, or if they do not. If $v^{(i)} = 12/24 - \varepsilon_c$ and $r_{u\ell}^{(i)} \in L_U^{(i)}$, then the width of $r_{u\ell}^{(i)}$ is close to $1/2$ and the height is larger than $1/2$. If $r_{u\ell}^{(i)} \notin L_U^{(i)}$, then there are only rectangles of a total width of ε_c in $I_L^{(i)}$. In the first lemma, we have $v^{(i)} = 12/24 - \varepsilon_c$ and $r_{u\ell}^{(i)} \in L_U^{(i)}$ and $r_{br}^{(i)} \in L_B^{(i)}$. Hence there are two very large rectangles in the packing.

Lemma 3.17. *Let* $B_1, \dots, B_k$ *be* k *bins so that each bin* B_i*, for* $i \in \{1, \dots, k\}$*, has a packing with the following conditions:*

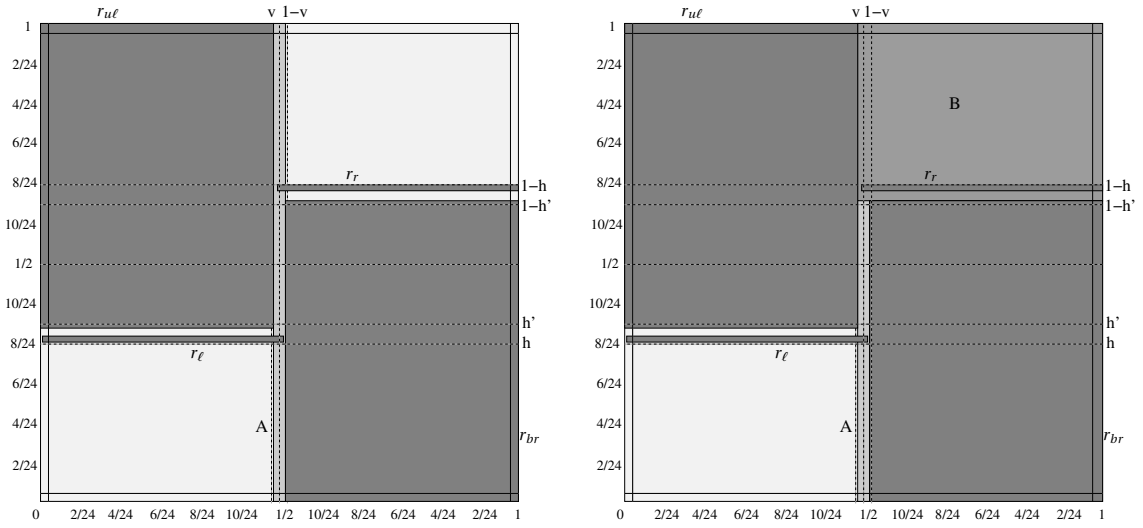
2.22. $v^{(i)} = 12/24 - \varepsilon_c$,

2.23. $h^{(i)} \in \mathrm{IN}$,

2.24. $r_{u\ell}^{(i)} \in L_U^{(i)}$ *and* $r_{br}^{(i)} \in L_B^{(i)}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2 \cdot k + 1$ *bins, while the packing of each of them satisfies Property 3.1.*

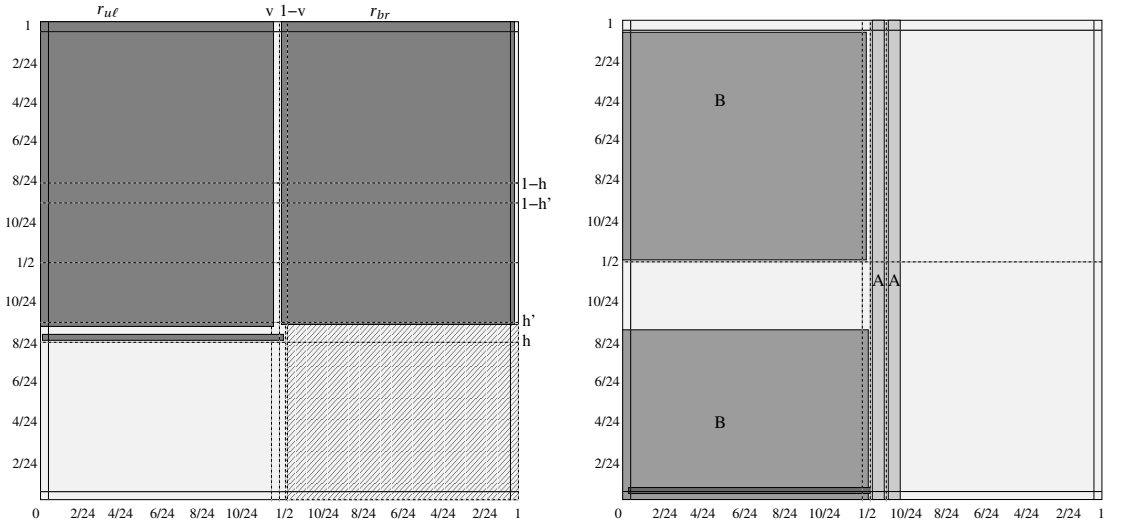
Proof. Let $i \in \{1, \dots, k\}$. The rectangles $r_{u\ell}^{(i)}$ and $r_{br}^{(i)}$ have a width of at least $1/2 - \varepsilon_c$, since they have to intersect the interval $\mathrm{VL}_{12/24-\varepsilon_c}^{(i)}$ and $\mathrm{VR}_{12/24-\varepsilon_c}^{(i)}$. Therefore, the space between the rectangles is very small. W.l.o.g. let $h_{u\ell}^{(i)} \geq h_{br}^{(i)}$ since we are able to turn the bin by 180°. We define the region $A^{(i)}$ between these rectangles by the coordinates $(x_{u\ell}^{\prime(i)}, 0)$, $(x_{br}^{(i)}, 0)$, $(x_{br}^{(i)}, 1)$ and $(x_{u\ell}^{\prime(i)}, 1)$. We treat this region as a rectangle $r_A^{(i)}$ of height $h_A^{(i)} = 1$ and width $w_A^{(i)} = x_{br}^{(i)} - x_{u\ell}^{\prime(i)} \leq (1 - v^{(i)}) - v^{(i)} = 1 - (12/24 - \varepsilon_c) - (12/24 - \varepsilon_c) = 2\varepsilon_c$ (Condition 2.22). Moreover, we define a region $B^{(i)}$ by the coordinates $(x_{u\ell}^{\prime(i)}, y_{br}^{\prime(i)}), (1, y_{br}^{\prime(i)}), (1, 1)$ and $(x_{u\ell}^{\prime(i)}, 1)$. The corresponding rectangle $r_B^{(i)}$ has a width of at most $w_B^{(i)} = 1 - x_{u\ell}^{\prime(i)} \leq 1 - v^{(i)} = 1 - (12/24 - \varepsilon_c) = 12/24 + \varepsilon_c = 1/2 + \varepsilon_c$ and a height of at most $h_B^{(i)} = 1 - y_{br}^{\prime(i)} \leq 1 - (1 - h^{\prime(i)}) \leq 1/2$ (cf. Figure 3.23). We move the rectangles $r_A^{(i)}$ and $r_B^{(i)}$ of each sequence of 2 bins into one additional bin. We move $r_{br}^{(i)}$ out of bin B_i for a moment. There is no rectangle on the right of $r_{u\ell}^{(i)}$, since all rectangles that were on top of $r_{br}^{(i)}$ are situated in $r_B^{(i)}$ and all rectangles



(a) Definition of the region $A^{(i)}$

(b) Definition of the region $B^{(i)}$

Figure 3.23: Definitions of the regions in the proof of Lemma 3.17



(a) Packing in the bin B_i

(b) Packing in the bin C_1

Figure 3.24: Packing in the proof of Lemma 3.17

that are located between $r_{u\ell}^{(i)}$ and $r_{br}^{(i)}$ are now to be found in $r_A^{(i)}$. At this moment, we use Lemma 3.3 on this bin since $S_R^{(i)}$ is completely free of rectangles. The width of the rectangle $r_{u\ell}^{(i)}$ is rounded up to the next largest multiple of $\varepsilon_c^2/2$ which is at most $1/2$. Thus, the new x'-coordinate is $x_{u\ell}'^{(i)} \leq 1/2$. We reinsert the rectangle $r_{br}^{(i)}$ into the bin after rounding up its width to the next largest multiple of $\varepsilon_c^2/2$ to at most $1/2$. Since $h_{u\ell}^{(i)} \geq h_{br}^{(i)}$, we can place $r_{br}^{(i)}$ on the right side of $r_{u\ell}^{(i)}$ on the x-coordinate $1/2$ (cf. Figure 3.24).

Let there be two bins B_1, B_2 of the k bins and let C_1 be an additional bin. We pack $r_B^{(1)}$ and $r_B^{(2)}$ on top of each other on the positions $(0,0)$ and $(0,1/2)$ into C_1. On the right side there is still a free space of at least $12/24 - \varepsilon_c$. Hence, there is enough space to place $r_A^{(1)}$ on the position $(12/24 + \varepsilon_c, 0)$ and $r_A^{(2)}$ on the position $(12/24 + 3\varepsilon_c, 0)$. There is still a space of at least $12/24 - 5\varepsilon_c$ free of rectangles including the strip S_R. Thus, we are able to employ Lemma 3.3 on bin C_1. In total, we have $3/2k + 1$ bins when k is odd while the packing of each bin satisfies Property 3.1. □

By exchanging the values $v^{(i)}$ and $h^{(i)}$, we obtain the following lemma. Since it is the analogous lemma, by turning the bin by 90°, we do not need it in the version that allows rotation.

Lemma 3.18. *Let $B_1, \ldots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \ldots, k\}$, has a packing with the following conditions:*

2.25. $v^{(i)} \in \mathrm{IN}$,

2.26. $h^{(i)} = 12/24 - \varepsilon_c$,

2.27. $r_{ur}^{(i)} \in W_R^{(i)}$ *and* $r_{b\ell}^{(i)} \in W_L^{(i)}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $3/2 \cdot k + 1$ bins, while the packing of each of them satisfies Property 3.2.

If there are no such big rectangles in the packing of bin B_i, we cannot adopt the method described above. However, when $v^{(i)} = 12/24 - \varepsilon_c$, the total width of either $L_U^{(i)}$ or $L_B^{(i)}$ is very small.

Lemma 3.19. *Let $B_1, \ldots, B_k$ be k bins so that each bin B_i, for $i \in \{1, \ldots, k\}$, has a packing with the following conditions:*

2.28. $v^{(i)} = 12/24 - \varepsilon_c$,

2.29. $h^{(i)} \in \mathrm{IN}$,

2.30. $r_{u\ell}^{(i)} \notin L_U^{(i)}$ *or* $r_{br}^{(i)} \notin L_B^{(i)}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $(3/2+\varepsilon_c)\cdot k+2$ *bins, while the packing of each of them satisfies Property 3.2.*

Proof. Let $i \in \{1,\dots,k\}$. W.l.o.g. we suppose that $r_{u\ell}^{(i)} \notin L_U^{(i)}$ since we are able to turn the bin by 180°. The rectangles in $L_U^{(i)}$ have a total width of at most ε_c since they have to be completely situated in the interval $\mathrm{VL}_{12/24-\varepsilon_c}^{(i)}$. We move these rectangles in $L_U^{(i)}$ into an additional bin. We are able to pack $1/\varepsilon_c$ sets next to each other at the bottom. Therefore, we need at most $\lceil \varepsilon_c \cdot k \rceil < \varepsilon_c \cdot k + 1$ additional bins for all k bins. We are able to round up the heights of these rectangles to the next multiple of $\varepsilon_c^2/2$ to at most 1 and satisfy Property 3.2.

No rectangle of a height larger than $1/2$ remains to intersect $S_U^{(i)}$. Thus, we are able to employ Lemma 3.6 with $y = 0$ and $x = 1/2$, and achieve a packing into $3/2 \cdot k + 1$ bins, while each packing satisfies Property 3.2. In total, we have a packing of $(3/2+\varepsilon_c)\cdot k+2$ bins. □

The last remaining case is analogous to Lemma 3.19. We do not need it in the version that allows rotation.

Lemma 3.20. *Let* $B_1,\dots,B_k$ *be* k *bins so that each bin* B_i*, for* $i \in \{1,\dots,k\}$*, has a packing with the following conditions:*

2.31. $v^{(i)} \in \mathrm{IN}$,

2.32. $h^{(i)} = 12/24 - \varepsilon_c$,

2.33. $r_{ur}^{(i)} \notin W_R^{(i)}$ *or* $r_{b\ell}^{(i)} \notin W_L^{(i)}$.

It follows that we are able to round up the rectangles in these bins and rearrange them into $(3/2+\varepsilon_c)\cdot k+2$ *bins, while the packing of each of them satisfies Property 3.1.*

Résumé of the Case Analysis

In conclusion we obtain the following theorem:

Theorem 3.4. *For any value* ε_c*, with* $1/\varepsilon_c$ *being a multiple of* 24*, and for any solution that fits into* m *bins, we are able to round up the widths and the heights of the rectangles so that they fit into* $(3/2+5\varepsilon_c)\cdot m+37$ *bins while the packing of each of the bins satisfies either Property 3.1 or Property 3.2.*

Proof. Let B_i be a bin in our solution. We prove that no matter how the packing looks like, one of the previous lemmas can be employed. A packing that does not satisfy the properties of Theorem 3.3 can be solved with Lemma 3.2 or Lemma 3.7. Thus, we suppose that the packing has the properties of Theorem 3.3 with certain values $h^{(i)} \in \mathbb{N}$ and $v^{(i)} \in \mathbb{N}$. Depending on these values and whether the rectangles in the corner have width or height larger than $1/2$, we use different lemmas (cf. Figure 3.25).

Case 1: $h^{(i)}$ and $v^{(i)}$ are both in the set $\{0/24, \ldots, 8/24 - \varepsilon_c\}$.
If $r_{b\ell}^{(i)} \notin W_L^{(i)}$ or $r_{ur}^{(i)} \notin W_R^{(i)}$, we employ Lemma 3.8, if $r_{u\ell}^{(i)} \notin L_U^{(i)}$ or $r_{br}^{(i)} \notin L_B^{(i)}$, we employ Lemma 3.9. Consequently, we conclude $r_{b\ell}^{(i)} \in W_L^{(i)}$, $r_{ur}^{(i)} \in W_R^{(i)}$, $r_{u\ell}^{(i)} \in L_U^{(i)}$ and $r_{br}^{(i)} \in L_B^{(i)}$. We solve this case either according to Lemma 3.10 or, if $h^{(i)} = 8/24 - \varepsilon_c$, according to Lemma 3.11.

Case 2: $v^{(i)} = 12/24 - \varepsilon_c$ or $h^{(i)} = 12/24 - \varepsilon_c$.
If $v^{(i)} = 12/24 - \varepsilon_c$, we employ either Lemma 3.17, if $r_{u\ell}^{(i)} \in L_U^{(i)}$ and $r_{br}^{(i)} \in L_B^{(i)}$ and else Lemma 3.19. If $h^{(i)} = 12/24 - \varepsilon_c$ and if $r_{ur}^{(i)} \in W_R^{(i)}$ and $r_{b\ell}^{(i)} \in W_L^{(i)}$ we adopt Lemma 3.18 and else Lemma 3.20.

Case 3: $v^{(i)} \in \{8/24, \ldots, 11/24\}$ and $h^{(i)} \in \{0/24, \ldots, 9/24\}$.
If $h^{(i)} \in \{0/24, 1/24\}$ we make use of Lemma 3.15. Otherwise, if $h^{(i)} \in \{2/24, \ldots, 9/24\}$, we employ Lemma 3.12.

Case 4: $h^{(i)} \in \{8/24, \ldots, 11/24\}$ and $v^{(i)} \in \{0/24, \ldots, 9/24\}$.
Analogous to the third case we use Lemma 3.16 if $v^{(i)} \in \{0/24, 1/24\}$ and Lemma 3.13 if $v^{(i)} \in \{2/24, \ldots, 9/24\}$.

Case 5: $h^{(i)}$ and $v^{(i)}$ are both in the set $\{10/24, 11/24\}$.
In this case we employ Lemma 3.14.

In each lemma mentioned here we modify a packing of k bins into a packing with at most $(3/2 + \varepsilon_c)k$ bins and a constant number of additional bins. Furthermore, we remove all rectangles that are completely in the strips at the sides of the bin and need therefore $4\varepsilon_c \cdot m + 2$ additional bins in Lemma 3.1. It follows that we need $(3/2 + 5\varepsilon_c) \cdot m$ bins plus a constant number of bins in total.

In Lemma 3.7, the constant number of additional bins is 15. Two additional bins are needed in the Lemma 3.8, Lemma 3.9,Lemma 3.11,Lemma 3.15,Lemma 3.16, Lemma 3.19 and Lemma 3.20. We need one additional bin in the Lemma 3.10, Lemma 3.12, Lemma 3.13, Lemma 3.14, Lemma 3.17 and Lemma 3.18. By summing up these constant numbers, we obtain a value of $2 + 15 + 2 \cdot 7 + 6 = 2 + 15 + 20 = 37$.

□

$v^{(i)} \backslash h^{(i)}$	0/24	1/24	2/24	3/24	4/24	5/24	6/24	7/24	$8/24-\varepsilon_c$	8/24	9/24	10/24	11/24	$12/24-\varepsilon_c$
0/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.16	3.16	3.16	3.16	3.18,3.20
1/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.16	3.16	3.16	3.16	3.18,3.20
2/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.13	3.13	3.13	3.13	3.18,3.20
3/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.13	3.13	3.13	3.13	3.18,3.20
4/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.13	3.13	3.13	3.13	3.18,3.20
5/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.13	3.13	3.13	3.13	3.18,3.20
6/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.13	3.13	3.13	3.13	3.18,3.20
7/24	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.13	3.13	3.13	3.13	3.18,3.20
$8/24-\varepsilon_c$	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.10	3.8,3.9,3.11	3.13	3.13	3.13	3.13	3.18,3.20
8/24	3.15	3.15	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.12,3.13	3.12,3.13	3.13	3.13	3.18,3.20
9/24	3.15	3.15	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.12,3.13	3.12,3.13	3.13	3.13	3.18,3.20
10/24	3.15	3.15	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.14	3.14	3.18,3.20
11/24	3.15	3.15	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.12	3.14	3.14	3.18,3.20
$12/24-\varepsilon_c$	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.19	3.17,3.18,3.19,3.20

Figure 3.25: Overview of the lemmas that are applied for different v and h

RÉSUMÉ OF THE CASE ANALYSIS WITH ROTATIONS The additive constant changes in the version that allows rotation. We employ Lemma' 3.1 with an additive constant of 1 and Lemma' 3.7 with an additive constant of 8 instead of Lemma 3.1 with an additive constant of 2 and Lemma 3.7 with an additive constant of 15, respectively. Furthermore, we do not need Lemma 3.9, Lemma 3.13, Lemma 3.16, Lemma 3.18 and Lemma 3.20. This reduces the additive constant to the value $1+8+2\cdot 4+4=21$.

Theorem' 3.4. *For any value ε_c, with $1/\varepsilon_c$ being a multiple of 24, and for any solution that fits into m bins, we are able to rotate and to round up the widths of the rectangles so that they fit into $(3/2+5\varepsilon_c)\cdot m+21$ bins while the packing of each of the bins satisfies Property 3.1.*

3.2.3 ROUNDING THE OTHER SIDE

In this section, we round the remaining unrounded side of the rectangles after employing Theorem 3.4 on an optimal solution in OPT bins. However, before we adopt Theorem 3.4 we divide the instance into big, wide, long, small and medium rectangles. Let $\varepsilon' \leq \min\{\varepsilon/39, 1/48\}$, so that $1/\varepsilon'$ is a multiple of 24. Similar as in [36], we find a value δ, so that the rectangles with at least one side length between δ and δ^4 have a small total area.

Lemma 3.21. *We find a value $\delta \leq \varepsilon'$, so that $1/\delta$ is a multiple of 24 and all rectangles r_i of the width $w_i \in [\delta^4, \delta)$ or the height $h_i \in [\delta^4, \delta]$ have a total area of at most $\varepsilon' \cdot \mathrm{OPT}$.*

Proof. Define a sequence $\sigma_1, \ldots, \sigma_{2/\varepsilon'+1}$, whereas σ_1 is the largest value with $\sigma_1 = \varepsilon'$ and $\sigma_{k+1} = \sigma_k^4$, for $k \in \{1, \ldots, 2/\varepsilon'\}$. Each reciprocal of the members in the sequence is a multiple of 24. This is a consequence of an inductive argument: $1/\varepsilon'$ is a multiple of 24; let $1/\sigma_k = i \cdot 24$ for one integer i. It follows that, $1/\sigma_{k+1} = 1/\sigma_k^4 = (1/\sigma_k)^4 = (i \cdot 24)^4 = (i^4 \cdot 24^3) \cdot 24$. Hence, $1/\sigma_{k+1}$ is a multiple of 24.

Let M_{σ_k} be the set of rectangles r_i with $w_i \in [\sigma_{k+1}, \sigma_k)$ or $h_i \in [\sigma_{k+1}, \sigma_k)$. Each rectangle r_i in the instance belongs to at most two sets M_{σ_k}. Since the area is a lower bound for OPT we have $\sum_{k=1}^{2/\varepsilon'} a(M_{\sigma_k}) \leq 2\mathrm{OPT}$. Suppose that all sets have an area of $a(M_{\sigma_k}) > \varepsilon' \cdot \mathrm{OPT}$, for

all $k \in \{1,\dots,2/\varepsilon\}$. We obtain $\sum_{k=1}^{2/\varepsilon'} a(M_{\sigma_k}) > 2/\varepsilon' \cdot \varepsilon' \mathrm{OPT} = 2 \cdot \mathrm{OPT}$, which is a contradiction. Therefore, there exists at least one set with $a(M_{\sigma_k}) \leq \varepsilon' \cdot \mathrm{OPT}$. We set $\delta := \sigma_k$, so that k is the smallest value with $a(M_{\sigma_k}) \leq \varepsilon' \cdot \mathrm{OPT}$. □

The rectangles are separated into *big* rectangles of width and height at least δ, *wide* rectangles of width at least δ and height smaller than δ^4, *long* rectangles of width smaller than δ^4 and height at least δ, *small* rectangles of width and height less than δ^4 and *medium* rectangles of width or height in $[\delta^4, \delta)$.

Since $1/\delta$ is a multiple of 24 we are able to adopt Theorem 3.4 with an optimal solution consisting of OPT bins and with $\varepsilon_c := \delta$. The resulting solution consists of $k \leq (3/2 + 5\delta)\mathrm{OPT} + 37$ bins which satisfy Property 3.1 or Property 3.2. A bin is of Type 1, if its packing satisfies Property 3.1 and otherwise it is of Type 2. Therefore, let $B_1,\dots,B_{p_1}$, be the bins of Type 1 and $B_{p_1+1},\dots,B_k$ be the bins of Type 2.

The widths of the big and wide rectangles in the bins $B_1,\dots,B_{p_1}$ and the heights of the big and long rectangles in the bins $B_{p_1+1},\dots,B_k$ are therefore multiples of $\delta^2/2$. We denote the set of big and wide rectangles that are packed into the bins $B_1,\dots,B_{p_1}$ and that have a width of $i\delta^2/2$, for $i \in \{2/\delta,\dots,2/\delta^2\}$ by B_i^w and W_i^w, respectively. The set of big and long rectangles that are packed in the bins $B_{p_1+1},\dots,B_k$ and that have a height of $i\delta^2/2$ are denoted by B_i^h and L_i^h, respectively. The wide rectangles that are packed in the bins $B_{p_1+1},\dots,B_k$ are denoted by W^h and the long rectangles that are packed in the bins $B_1,\dots,B_{p_1}$ are denoted by L^w. The set of small rectangles is denoted by S. The set of medium rectangles that have a width within $[\delta^4,\delta)$ are denoted by $M_{w\delta}$ and the remaining rectangles of height within $[\delta^4,\delta)$ are denoted by $M_{h\delta}$.

The medium rectangles have a total area of at most $\varepsilon' \cdot \mathrm{OPT}$. Therefore, we are able to move them into few additional bins with Steinberg's Algorithm [51] (cf. also Section 2.2)

Theorem 3.5 (Steinberg's algorithm)**.** *If the following inequalities hold,*

$$w_{\max}(T) \leq a,\ \ h_{\max}(T) \leq b, \text{ and } \quad 2a(T) \leq ab - (2w_{\max}(T) - a)_+ (2h_{\max}(T) - b)_+$$

where $x_+ = \max(x, 0)$*, then it is possible to pack all items from* T *into* $R = (a, b)$ *in time* $\mathcal{O}((n \log^2 n)/\log\log n)$.

Therefore, we only have to partition the set of medium rectangles into subsets of a total area of at most $1/2$.

Lemma 3.22. *We pack the medium rectangles into at most* $3\varepsilon'\mathrm{OPT} + 2$ *additional bins*

Proof. We split the sets $M_{h\delta}$ and $M_{w\delta}$ into sets of a total area of at most $1/2$. Each medium rectangle has an area of at most δ. Therefore, we are able to greedily divide $M_{h\delta}$ and $M_{w\delta}$ into sets of rectangles of a total area within $(1/2-\delta, 1/2]$ and two additional sets of rectangles with a bounded total area of at most $1/2-\delta$. It holds that $3\varepsilon'\mathrm{OPT}\cdot(1/2-\delta) > 3\varepsilon'\mathrm{OPT}\cdot(1/2-1/6) = \varepsilon'\mathrm{OPT}$, since $\delta \leq \varepsilon' \leq 1/48 < 1/6$. Hence, the total number of sets is bounded by $\lceil 3\varepsilon'\mathrm{OPT}\rceil \leq 3\varepsilon'\mathrm{OPT}+2$. The rectangles in each set have either a maximum width of at most $\delta < 1/2$ or a maximum height of at most $\delta < 1/2$. This enables us to pack each set into one bin using Steinberg's Theorem 3.5. □

In the next step, we round up the heights of the big and long rectangles in the bins $B_1,\dots,B_{p_1}$ and the widths of the big and wide rectangles in the bins $B_{p_1+1},\dots,B_k$. Furthermore, we pack the wide and long rectangles fractionally into wide and long containers.

Packing Medium Rectangles with Rotations We use the same construction in the version that allows rotation. The difference is that we rotate the rectangles in the set $M_{h\delta}$. Consequently, these rectangles belong to the set $M_{w\delta}$. We partition this set analogous as in the proof of Lemma 3.22 into $3\varepsilon'\mathrm{OPT}+1$ subsets of total area at most $1/2$. These sets are packed with Steinberg's Theorem 3.5. We obtain the following lemma for the version that allows rotation.

Lemma' 3.22. *We rotate and pack the medium rectangles into at most* $3\varepsilon'\mathrm{OPT}+1$ *additional bins*

The next steps are explained for the bins $B_1,\dots,B_{p_1}$, the rounding for the remaining bins is analogous.

Rounding Big and Long Rectangles

We round the heights of the big and long rectangles in the sets B_i^w and L_i^w, for each $i \in \{2/\delta,\dots,2/\delta^2\}$.

To do this, we adopt a similar rounding technique as in the algorithm by Kenyon & Rémila [40] and in the algorithm by Fernandez de la Vega & Lueker [19]. We focus on one set B_i^w of big rectangles, for $i \in \{2/\delta,\dots,2/\delta^2\}$. We sort the rectangles in this set according to non-decreasing heights. Let k_i be the number of rectangles in B_i^w, denoted by $r_{i,1},\dots,r_{i,k_i}$. The rectangle $r_{i,1}$ has the largest and r_{i,k_i} the smallest height. We define at most $1/\delta^2$ subsets $B_{i,j}^w$, which consist of $\lfloor\delta^2\cdot k_i\rfloor$ rectangles except the last subset with possibly less items. This is done by assigning $\lfloor\delta^2\cdot k_i\rfloor$ rectangles into one subset, then we leave one rectangle out and assign the next $\lfloor\delta^2\cdot k_i\rfloor$ rectangles into the next subset. Thus, the first

rectangle $r_{i,1}$ is not assigned to a subset and is called the first cut-rectangle. The rectangles $r_{i,2},\ldots,r_{i,\lfloor\delta^2\cdot k_i\rfloor+1}$ are assigned to subset $B^w_{i,1}$. The rectangle $r_{i,\lfloor\delta^2\cdot k_i\rfloor+2}$ is called the second cut-rectangle. The rectangles $r_{i,\lfloor\delta^2\cdot k_i\rfloor+3},\ldots,r_{i,2\cdot\lfloor\delta^2\cdot k_i\rfloor+2}$ are assigned to subset $B^w_{i,2}$ and so on. Hence, the jth cut-rectangle is $r_{i,(j-1)\cdot\lfloor\delta^2\cdot k_i\rfloor+j}$ and the subset $B^w_{i,j}$ contains the rectangles $r_{i,(j-1)\cdot\lfloor\delta^2\cdot k_i\rfloor+(j+1)},\ldots,r_{i,j\cdot\lfloor\delta^2\cdot k_i\rfloor+j}$. We have at most $1/\delta^2$ cut-rectangles and subsets, since we have at most $1/\delta^2\cdot(1+\lfloor\delta^2\cdot k_i\rfloor)\geq 1/\delta^2\cdot\delta^2\cdot k_i = k_i$ rectangles.

We round the heights of the rectangles in each subset $B^w_{i,j}$ to the height of the jth cut-rectangle. Afterwards, we move the rectangles of the first subset $B^w_{i,1}$ into additional bins. Note that the total width of subset $B^w_{i,1}$, denoted by $w(B^w_{i,1})$, is at most $\lfloor\delta^2\cdot k_i\rfloor\cdot i\delta^2/2\leq\delta^2\cdot k_i\cdot i\delta^2/2=\delta^2\cdot w(B^w_i)$, with $w(B^w_i)$ being the total width of all rectangles in B^w_i. Each rectangle in a remaining subset $B^w_{i,j}$ is placed on a position of one rectangle in subset $B^w_{i,j-1}$. This is done by placing the ℓth rectangle of subset $B^w_{i,j}$ on the position of the ℓth rectangle of subset $B^w_{i,j-1}$. This is possible since all rectangles have the same width, all subsets, except the last, have the same cardinality and the height of the ℓth rectangle in subset $B^w_{i,j-1}$ is larger than or equal to the height of the jth cut rectangle. The cut-rectangles are reinserted at their origin positions.

This step is done for all sets B^w_i. The rounding method for the long rectangles in L^w is almost the same. However, we do not have the property that the rectangles in L^w have all the same width. Therefore, we have to slice the long rectangles vertically. We also sort the rectangles of set L^w according to non-decreasing heights. $w(L^w)$ denotes the total width of the rectangles in set L^w. We divide the set L^w into subsets $L^w_1,\ldots,L^w_{1/\delta^2}$ of the same total width, by splitting rectangles vertically if necessary. The subset L^w_1 contains the largest and the subset L^w_{1/δ^2} the shortest rectangles. The rectangles in each subset have a total width $w(L^w_i)$ of $\delta^2\cdot w(L^w)$. We round up the heights of the rectangles in each subset to the height of the largest rectangle in it. The rectangles in subset L^w_1 are packed later into additional bins. The rectangles of the remaining subsets are packed on the positions where the rectangles of the previous subset have been. Again, we split the rectangles vertically if necessary.

It is left to pack the rectangles in the subsets $L^w_1, B^w_{2/\delta,1},\ldots,B^w_{2/\delta^2,1}$ into additional bins. The total width of all rectangles in all sets $w(L^w\cup B^w_{2/\delta}\cup\ldots\cup B^w_{2/\delta^2})$ is at most $1/\delta\cdot p_1$, since each rectangle has a height of at least δ and they would not fit into p_1 bins otherwise. Hence,

$$w(L^w_1)+\sum_{i=2/\delta}^{2\delta^2} w(B^w_{i,1})\leq\delta^2\cdot w(L^w)+\sum_{i=2/\delta}^{2\delta^2}\delta^2\cdot w(B^w_i)=$$
$$\delta^2\cdot(w(L^w)+\sum_{i=2/\delta}^{2/\delta^2} w(B^w_i))\leq\delta^2\cdot 1/\delta\cdot p_1=\delta\cdot p_1.$$

We pack these rectangles greedily on the floor of additional bins. We start by packing each big rectangle of width larger than $1/2$ into one additional bin. This bin is closed and we do not pack further rectangles into it. The remaining big rectangles have a width of at most $1/2$. We pack these rectangles greedily on the floor of the bin, until the next big rectangle does not fit. Then a new bin is opened and we continue with the packing. Afterwards, we pack the long rectangles with the same method. We secure, that the rectangles in each but the last bin have a total width of at least $1/2$. In total, we need at most $\lceil 2(\delta \cdot p_1) \rceil \leq 2\delta \cdot p_1 + 1$ bins while each packing satisfies Property 3.1, since each big rectangle is placed on a multiple of $\delta^2/2$.

The same steps are done analogously for rounding the widths of the big and wide rectangles that are packed in the bins $B_{p_1+1}, \ldots, B_k$. Therefore, we need $2\delta \cdot (1 - p_1) + 1$ additional bins. In doing so, we use similar subsets $B^h_{i,j}$ and W^h_j for $i \in \{2/\delta, \ldots, 2/\delta^2\}$ and $j \in \{1, \ldots, 1/\delta^2\}$.

With the above discussion we obtain the following result.

Lemma 3.23. *We round up the heights of the long and big rectangles in each set L^w and B^w_i, and the widths of the wide and big rectangles in each set W^h and B^h_i for $i \in \{2/\delta, \ldots, 2/\delta^2\}$ to at most $1/\delta^2$ values. Therefore, we need $2\delta \cdot k + 2$ additional bins.*

Rounding Big and Long Rectangles with Rotations In the version that allows rotation, we only have bins with a packing that satisfies Property 3.1. In this version, we employ Theorem' 3.4 and obtain a solution in $k' \leq (3/2+\delta)\cdot\mathrm{OPT}+21$ bins. Consequently, we have the following lemma.

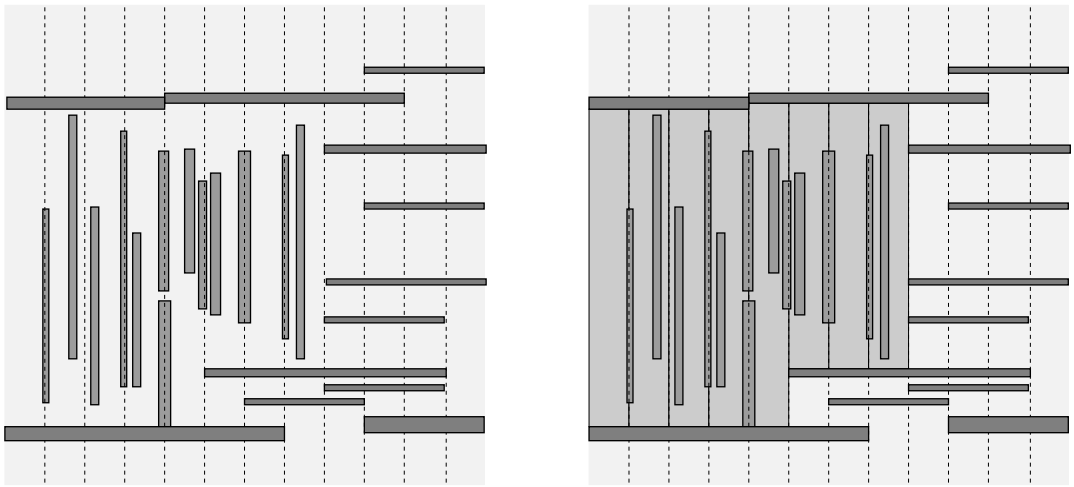
Lemma' 3.23. *We round up the heights of the long and big rectangles in each set L^w and B^w_i for $i \in \{2/\delta, \ldots, 2/\delta^2\}$ to at most $1/\delta^2$ values and need therefore $2\delta \cdot k' + 1$ additional bins.*

Containers for the Wide and Long Rectangles

In this section, we construct rectangular containers for the wide and long rectangles. To do this, we employ some techniques of Jansen & Solis-Oba [36]. We explain these steps only for the bins $B_1, \ldots, B_{p_1}$ of Type 1 as it works analogously for the remaining bins. We define a set of long containers C^w_L for the long rectangles and a set of wide containers C^w_W for the wide rectangles. We focus on one bin B_i, for $i \in \{1, \ldots, p_1\}$.

We define slots of width $\delta^2/2$ by drawing vertical lines on each multiple of $\delta^2/2$ in the bin B_i for the long containers. A small or long rectangle that intersects one of these vertical lines is vertically cut by it. Each wide and big rectangle intersects either a slot completely or it

does not intersect it at all. Hence, a long container is a part of a slot which is bounded at the top and at the bottom by a wide or a big rectangle or the top or the floor of the bin. We only consider containers with at least one long rectangle, i.e. the height of each long container is at least δ. The width of each long container is $\delta^2/2$ (cf. Figure 3.26). As an upper bound there are at most $1/\delta - 1$ long containers in each slot separated by wide rectangles of small height. Hence, in bin B_i there are at most $(1/\delta - 1) \cdot 2/\delta^2 = 2/\delta^3 - 2/\delta^2$ long containers. This is an upper bound when there are no big rectangles situated in this bin. Each big rectangle that is situated in this bin decreases the number of long containers by $2/\delta$ since it intersects $2/\delta$ slots and it has a height of at least δ.



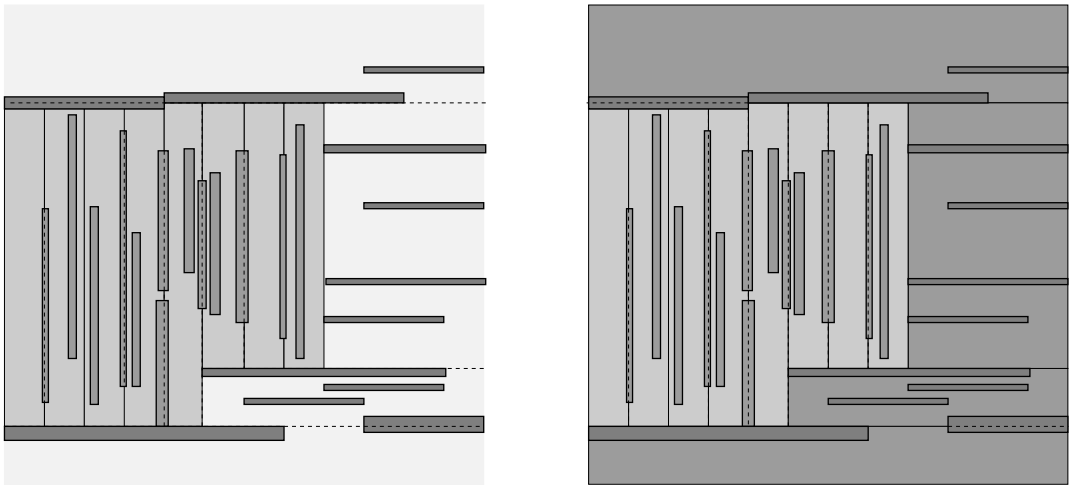
(a) Part of a packing in one bin

(b) Construction of long containers

Figure 3.26: Construction of long containers; long and small rectangles are sliced vertically

Afterwards, we construct horizontal lines by extending the upper and lower edge of each big rectangle and each long container in both directions until it hits another big rectangle, a long container or the sides of the bin. Wide and small rectangles are horizontally cut by these lines. The horizontal lines are the upper and lower edges of the wide containers. Since they are bounded at the sides by big rectangles, long containers or the sides of the bin, the wide containers always have a width of a multiple of $\delta^2/2$. There are at most $2/\delta^3 - 2/\delta^2$ big rectangles and long containers in the solution, hence we extend at most $2/\delta^3 - 2/\delta^2$ upper and lower edges. Furthermore, we have two additional lines with the bottom and the top of the bin. Each extended upper edge is a possible lower edge of one wide container. Each extended lower edge may be the lower edges of two wide containers, one on the left and one

on the right side. Furthermore, the bottom of the bin is a possible lower edge of one wide container. It follows that there are at most $3 \cdot (2/\delta^3 - 2/\delta^2) + 1 = 6/\delta^3 - 6/\delta^2 + 1$wide containers in bin B_i. At this moment, the complete region of the bin is filled with big rectangles, long containers and wide containers. There is no empty space left, hence all small rectangles are fractionally in the long and wide containers (cf. Figure 3.27). This construction is done for all bins $B_1, \ldots, B_{p_1}$.



(a) Drawing horizontal lines

(b) Construction of wide containers

Figure 3.27: Construction of wide containers; wide and small rectangles are sliced horizontally

Rounding the Wide and Long Containers

Let c_w be one of the wide containers in C_W^w. The height of c_w is now reduced so that it has a height of a multiple of δ^4. We cut the uppermost wide and small rectangles horizontally by the new height. The (fractions of the) wide and small rectangles of height less than δ^4 that do not fit in the reduced container are placed next to each other into an additional bin. We do this with all wide containers in the bins $B_1, \ldots, B_{p_1}$. Since all rectangles have a height of less than δ^4 we are able to place the rectangles of $1/\delta^4$ containers on top of each other into one bin. In total, we have less than $6/\delta^3 \cdot p_1$ wide containers and hence we need $\lceil \delta^4 \cdot 6/\delta^3 \cdot p_1 \rceil \leq 6\delta \cdot p_1 + 1$ additional bins. These bins contain only wide and small rectangles, whereas the wide rectangles are placed on x-coordinates of a multiple of $\delta^2/2$. Hence, the

packing of these bins satisfy Property 3.1. We treat each additional bin as one wide container of height and width 1.

After the construction of the wide containers we round down the heights of the long containers. We focus one long container c_ℓ in C_L^w in the bins $B_1, \dots, B_{p_1}$. We remove all short rectangles in this container and shift all long rectangles vertically down, so that they all touch either the bottom of the container or another long rectangle. The total used height in this container is a combination of the heights of the long rectangles. Since the heights of the long rectangles are rounded, the possible number of heights is bounded by a polynomial in the length of the input. If the remaining space on top of the uppermost long rectangle in the container c_ℓ has a height of at least δ^4, then we are able to round the height of the container down to the next multiple of δ^4. If the remaining space is less than δ^4 we round the height of c_ℓ down to the height of the top edge of the uppermost long rectangle in c_ℓ. Hence, the height of c_ℓ is either a combination of the rounded heights of the long rectangles or a multiple of δ^4.

It is not possible to reinsert all small rectangles. Hence, we pack them fractionally into the reduced container until the free space is exceeded. The remaining rectangles are packed fractionally into additional bins. The total area loss for each long container is at most $\delta^4 \cdot \delta^2/2 = \delta^6/2$. There are at most $2/\delta^3$ long containers in each bin, hence the total area is at most $\delta^6/2 \cdot 2/\delta^3 \cdot p_1 = \delta^3 \cdot p_1$. Thus, we need at most $\lceil \delta^3 \cdot p_1 \rceil \leq \delta^3 \cdot p_1 + 1$ additional bins. These additional bins contain only small rectangles and the packing satisfies Property 3.1. We treat each bin as one wide container of height and width 1, in order to ensure that all small rectangles are packed into containers. By this construction, the total number of wide containers is at most $(6/\delta^3 - 6/\delta^2 + 1) \cdot p_1 + 6\delta \cdot p_1 + 1 + \delta^3 \cdot p_1 + 1 = (6/\delta^3 - 6/\delta^2 + 6\delta + \delta^3 + 1) \cdot p_1 + 2 < 6/\delta^3 \cdot p_1 + 2$.

We also construct long containers in the $2\delta \cdot p_1 + 1$ additional bins that are needed to round the heights of the long rectangles (cf. Lemma 3.23). The long rectangles are placed on the bottom of these bins and there are no small rectangles in it. Furthermore, we packed them after we packed the big rectangles so that there is at most one bin that contains big and long rectangles. The other bins contain either only big rectangles or only long rectangles. We draw also vertical lines in the bins that contain long rectangles at each multiple of $\delta^2/2$ and cut the long rectangles intersecting these lines. These lines form already the long containers in these bins, since there are no rectangles on top of the long rectangles. Therefore, we have in these bins at most $(2\delta \cdot p_1 + 1) \cdot 2/\delta^2 = 4/\delta \cdot p_1 + 2/\delta^2$ long containers of width $\delta^2/2$ and height 1.

There is still a huge number of different types of the long containers, since we have $1/\delta^2$ different heights of the long rectangles and hence at least $(1/\delta^2)^{1/\delta}$ possibilities for

the heights. In order to reduce this number to $1/\delta^2$ different heights, we use the same rounding technique as for the big rectangles.

Let $k_\ell \leq (2/\delta^3 - 2/\delta^2) \cdot p_1 + 4/\delta \cdot p_1 + 2/\delta^2 \leq 2/\delta^3 \cdot p_1 + 2/\delta^2$ be the total number of long containers in C_L^w, i.e. in the bins of Type 1. We sort all k_ℓ containers according to non-decreasing heights and denote the sorted containers by $c_1, \ldots, c_{k_\ell}$. We partition the long containers into at most $1/\delta^2$ subsets of $\lfloor \delta^2 k_\ell \rfloor$ containers. The construction is analogous to the rounding of the big rectangles by calling c_1 the first cut-container and assigning the next $\lfloor \delta^2 k_\ell \rfloor$ containers to the first subset and so on. The heights in each subset are rounded up to the height of the previous cut-container. The containers in the first subset are moved into additional bins, whereas we are able to pack $2/\delta^2$ containers next to each other at the bottom in one bin. Hence, we need at most $\lceil \lfloor \delta^2 k_\ell \rfloor \cdot \delta^2/2 \rceil \leq \lceil \delta^2 \cdot (2/\delta^3 \cdot p_1 + 2/\delta^2) \cdot \delta^2/2 \rceil \leq \delta \cdot p_1 + \delta^2 + 1$ additional bins. The long containers in the remaining subsets are packed on the position of a long container in the previous subset and the cut-container are placed at their origin position.

We do the analogous steps for the bins $B_{p_1+1}, \ldots, B_k$ and achieve the set of wide containers C_W^h and the set of long containers C_L^h with analogous bounds by replacing p_1 with $k - p_1$. This leads us to the following result:

Lemma 3.24. *Suppose we have a packing without medium rectangles in k bins, while the packing of each of them satisfies either Property 3.1 or Property 3.2. If the long and wide rectangles are rounded according to Lemma 3.23, then we are able to pack the wide, long and small rectangles fractionally into containers with at most $8\delta \cdot k + 2\delta^2 + 6$ additional bins. The rectangles of W^w and W^h are sliced horizontally and packed into wide containers of C_W^w and C_W^h, respectively. The long rectangles of L^w and L^h are sliced vertically and packed into long containers of C_L^w and C_L^h. All small rectangles are packed fractionally (vertically and horizontally sliced) into these containers. The containers have the following properties:*

3.1. there are at most $6/\delta^3 \cdot p_1 + 2$ wide containers in C_W^w, that have a width of a multiple of $\delta^2/2$ and a height of a multiple of δ^4.

3.2. there are at most $2/\delta^3 \cdot (k - p_1) + 2/\delta^2$ wide containers in C_W^h, of at most $1/\delta^2$ different widths of either a multiple of δ^4 or a combination of the rounded widths of the wide rectangles in W^h and of a height $\delta^2/2$

3.3. there are at most $6/\delta^3 \cdot (k - p_1) + 2$ long containers in C_L^h, that have a height of a multiple of $\delta^2/2$ and a width of a multiple of δ^4.

3.4. there are at most $2/\delta^3 \cdot p_1 + 2/\delta^2$ long containers in C_L^w, of at most $1/\delta^2$ different heights of either a multiple of δ^4 or a combination of the rounded heights of the long rectangles in L^w and of a width $\delta^2/2$

Proof. By rounding the heights of the wide containers in C_W^w in the bins of Type 1 and the long containers in C_L^h in the bins of Type 2 to a multiple of δ^4 we need $6\delta \cdot p_1 + 1 + 6\delta \cdot (k - p_1) + 1 = 6\delta \cdot k + 2$ additional bins. The heights of the long containers in C_L^w in the bins of Type 1 are rounded to at most $1/\delta^2$ values of either a multiple of δ^4 or a combination of the rounded heights of the long rectangles. Therefore, we need $\delta^3 \cdot p_1 + 1 + \delta \cdot p_1 + \delta^2 + 1 = (\delta + \delta^3) \cdot p_1 + \delta^2 + 2$ additional bins. The analogous steps for rounding the widths of the wide containers in C_W^h in the bins of Type 2 need $(\delta + \delta^3) \cdot (k - p_1) + \delta^2 + 2$ additional bins and hence together $(\delta + \delta^3) \cdot k + 2\delta^2 + 4$ additional bins. The total number of additional bins is thus $(6\delta + \delta + \delta^3) \cdot k + 2\delta^2 + 6 \leq 8\delta \cdot k + 2\delta^2 + 6$. □

To conclude, we obtain the following result:

Theorem 3.6. *Given an optimal solution of an instance I into* OPT *bins. We are able to round up the widths and heights of the rectangles and to modify the solution so that it fits into at most $(3/2 + 25\varepsilon') \cdot \mathrm{OPT} + 55$ bins, while all medium rectangles are packed into $3\varepsilon' \cdot \mathrm{OPT} + 2$ bins and $3/2\mathrm{OPT} + 22\delta\mathrm{OPT} + 53$ bins have a packing that satisfies either Property 3.1 or Property 3.2. Furthermore, the heights of the long and big rectangles in each set L^w and B_i^w, and the widths of the wide and big rectangles in each set W^h and B_i^h for $i \in \{2/\delta, \ldots, 2/\delta^2\}$ are rounded up to at most $1/\delta^2$ values. The wide and long rectangles are sliced horizontally and vertically, respectively. They are packed into wide and long containers with the Properties 3.1-3.4. The small rectangles are packed fractionally into the wide and long containers.*

Proof. In the first step we employ Theorem 3.4 and need in total $k \leq (3/2 + 5\delta)\mathrm{OPT} + 37$ bins. Afterwards, we pack all medium rectangles into $3\varepsilon' \cdot \mathrm{OPT} + 2$ additional bins with Lemma 3.22. We round up the big, long and wide rectangles with Lemma 3.23 and need $2\delta \cdot k + 2$ additional bins. The wide and long rectangles are packed into long and wide containers with Lemma 3.24. Therefore, we need $8\delta \cdot k + 2\delta^2 + 6$ additional bins. It holds $\delta \leq 1/48$, hence in total we need at most

$$
\begin{aligned}
(1 + 10\delta) \cdot ((3/2 + 5\delta) \cdot \mathrm{OPT} + 37) + 2\delta^2 + 8 = \\
3/2 \cdot \mathrm{OPT} + 5\delta \cdot \mathrm{OPT} + 15\delta \cdot \mathrm{OPT} + 50\delta^2 \cdot \mathrm{OPT} + 37 + 370\delta + 2\delta^2 + 8 \leq \\
3/2 \cdot \mathrm{OPT} + 20\delta \cdot \mathrm{OPT} + 50\delta/48 \cdot \mathrm{OPT} + 45 + 370/48 + 2/48^2 \leq \\
3/2 \cdot \mathrm{OPT} + 22\delta \cdot \mathrm{OPT} + 53
\end{aligned}
$$

bins that have a packing that satisfies either Property 3.1 or Property 3.2. Since $\delta \leq \varepsilon'$, we have at most $(3/2+25\varepsilon')\cdot \mathrm{OPT}+55$ bins (including the bins for the medium rectangles). □

This finishes our analysis and modification of an optimal solution. It enables us to construct an algorithm that computes a packing that almost matches the modified solution.

Rounding the Wide and Long Containers with Rotations The steps are analogous for the versions with rotations.

Lemma' 3.24. *Suppose we have a packing without medium rectangles in k' bins, while the packing of each of them satisfies Property 3.1. If the long and wide rectangles are rounded according to Lemma 3.23, then we are able to pack the wide, long and small rectangles fractionally into containers with at most $8\delta \cdot k' + \delta^2 + 3$ additional bins. The rectangles of W^w are sliced horizontally and packed into wide containers of C_W^w. The long rectangles of L^w are sliced vertically and packed into long containers of C_L^w. All small rectangles are packed fractionally (vertically and horizontally sliced) into these containers. The containers have the following properties:*

3.5. *there are at most $6/\delta^3 \cdot k' + 2$ wide containers in C_W^w, that have a width of a multiple of $\delta^2/2$ and a height of a multiple of δ^4.*

3.6. *there are at most $2/\delta^3 \cdot k' + 2/\delta^2$ long containers in C_L^w, of at most $1/\delta^2$ different heights of either a multiple of δ^4 or a combination of the rounded heights of the long rectangles in L^w and of a width of $\delta^2/2$*

To conclude, we obtain a better additional constant for modifying the packing.

Theorem' 3.6. *Given an optimal solution of an instance I into* OPT *bins. We are able to rotate and round up the widths and heights of the rectangles and to modify the solution so that it fits into at most $(3/2+25\varepsilon')\cdot \mathrm{OPT}+31$ bins, while all medium rectangles are packed into $3\varepsilon'\cdot \mathrm{OPT}+1$ bins and $3/2\cdot \mathrm{OPT}+22\delta\cdot \mathrm{OPT}+30$ bins have a packing that satisfies Property 3.1. Furthermore, the heights of the long and big rectangles in the sets L^w and B_i^w for $i \in \{2/\delta,\ldots,2/\delta^2\}$ are rounded to at most $1/\delta^2$ values. The wide and long rectangles are sliced horizontally and vertically, respectively. They are packed into wide and long containers with the Property 3.5 and Property 3.6. The small rectangles are packed fractionally into the wide and long containers.*

Proof. We adopt Theorem' 3.4 with an optimal solution and obtain $k' \leq (3/2+5\delta)\mathrm{OPT}+21$ bins. We round up the big and long rectangles according to Lemma' 3.23 and need $2\delta\cdot k'+1$

additional bins. For rounding the containers with Lemma' 3.24, we need $8\delta \cdot k' + \delta^2 + 3$ additional bins. The additional constant is therefore, $21 + 10\delta \cdot 21 + \delta^2 + 1 + 3 \leq 25 + 210/48 + 1/48^2 \leq 25 + 5 = 30$. Together with the $3\varepsilon' \cdot \mathrm{OPT} + 1$ additional bins that are used to pack the medium rectangles in Lemma' 3.22, we obtain $(3/2 + 25\varepsilon') \cdot \mathrm{OPT} + 31$ bins. □

3.3 Algorithm

In the last sections we modified an optimal solution in order to achieve a simpler structure. In this section we describe our algorithm. The algorithm works in two parts. The first part is to transform an instance I of n rectangles into the rounded instance, the second part is to pack the rounded rectangles into the bins.

3.3.1 Transform an Instance I

For dual approximation we use binary search to find the optimum $\mathrm{OPT} = \mathrm{OPT}(I)$ of I. In each iteration with a candidate OPT' for OPT we either find a solution with at most $(3/2 + 22\delta) \cdot \mathrm{OPT}' + 69$ bins or conclude that $\mathrm{OPT}' < \mathrm{OPT}$. In the first case we decrease OPT' in order to try if there is a solution with less bins and in the second case we increase OPT'. The upper bound for OPT is the number of rectangles in the instance I, the lower bound is the total area of the rectangles. In the following, we assume that we found an $\mathrm{OPT}' \leq \mathrm{OPT}$ so that our algorithm is able to compute a solution. In the next step we are able to set δ according to Lemma 3.21 and divide the instance into big, long, wide, small and medium rectangles. We pack all medium rectangles with Steinberg's Theorem 3.5 into $3\varepsilon'\mathrm{OPT} + 2$ additional bins (cf. Lemma 3.22). Afterwards, we have to distinguish whether the width or the height of each big rectangle is rounded up to a multiple of $\delta^2/2$. In other words, we have to distinguish, whether a rectangle belongs to a bin of Type 1, i.e. it is in a set B_i^w, or to a bin of Type 2, i.e. it is in a set B_i^h, for one $i \in \{2/\delta, \ldots, 2/\delta^2\}$.

In the version that allows rotation, we only have bins of Type 1. However, we do not know which side we have to round up to the next largest multiple of $\delta^2/2$. Therefore, we have to solve a similar problem.

Transform Big Rectangles

Let $i \in \{2/\delta, \ldots, 2/\delta^2\}$ and $j \in \{1, \ldots, 1/\delta^2\}$. We guess the number of big rectangles that are rounded to each width of $i\delta^2/2$ and to each height of $i\delta^2/2$. In other words, we guess the cardinality of the sets B_i^w and B_i^h. This can be done by choosing less than $2 \cdot 2/\delta^2 = 4/\delta^2$

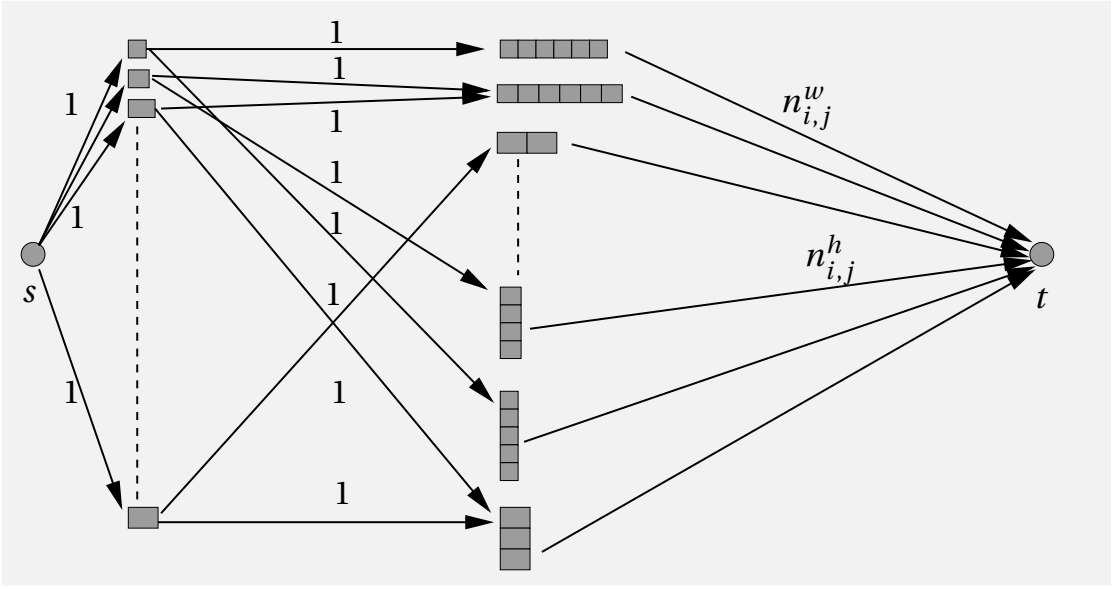


Figure 3.28: The flow-network

values out of n. With the guessed cardinality we compute the number of the at most $1/\delta^2$ subsets $B^w_{i,j}$ and $B^h_{i,j}$ and also the number of the at most $1/\delta^2$ cut-rectangles. We guess the cut-rectangles by choosing $2 \cdot 2/\delta^2 \cdot 1/\delta^2 = 4/\delta^4$ rectangles out of n possible rectangles. We denote the number of rectangles that are rounded up to the width $i \cdot \delta^2/2$ and to the height of the jth cut-rectangle in B^w_i by $n^w_{i,j}$ and we denote the number of rectangles that are rounded up to the height $i \cdot \delta^2/2$ and to the width of the jth cut-rectangle in B^h_i by $n^h_{i,j}$.

These values give us the structure of the subsets of the Section 3.2.3, since we know the rounded heights and widths of the big rectangles and the number of rectangles with these side lengths. To assign big rectangles to these subsets we set up a flow network $G = (V, E)$ with the set V of vertices and the set E of edges. Each big rectangle of I has a corresponding node in this network and is connected with an edge of capacity 1 to the source s. Either the width or the height of a big rectangle is rounded up to the next multiple of $\delta^2/2$. If this is decided, we know also the corresponding subset where the rectangle belongs to since it has to be between the heights or the widths of two cut-rectangles. However, when the height or the width is exactly that one of a cut-rectangle it could belong to possibly more subsets. For each subset $B^w_{i,j}$ and $B^h_{i,j}$ we have one node in the network and connect it to each rectangle that may belong to it with an edge of capacity 1. There are at most $2 \cdot 2/\delta^2 \cdot 1/\delta^2 = 4/\delta^4$ subset-nodes and each big-rectangle-node is connected to at most $2 \cdot 1/\delta^2 = 2/\delta^2$ of them, when all cut-rectangles have the same height and width, respectively. Each subset-node

$B^w_{i,j}$ and $B^h_{i,j}$ is connected with an edge to the sink t of capacity $n^w_{i,j}$ and $n^h_{i,j}$, respectively (cf. Figure 3.28). The total number of vertices is in total $|V| \leq 2 + n + 4/\delta^4$. The number of edges is $n + n \cdot 2/\delta^2 + 4/\delta^4$. We find a flow with the algorithm of Dinic [17] in time $\mathcal{O}(|E| \cdot |V|^2) = \mathcal{O}((n + (2n)/\delta^2 + 4/\delta^4) \cdot (2 + n + 4/\delta^4)^2) = \mathcal{O}(n^3/\delta^2 + n^2/\delta^6 + n/\delta^{10} + 1/\delta^{14})$. If there is a possible assignment of big rectangles to the subsets then there is a flow with the same value as the number of big rectangles, hence each edge at the source s is satisfied. If there is no flow with this value at all, there exists no assignment of big rectangles to the subsets and we have to try another guess.

Transform Big Rectangles with Rotations In the version that allows rotation, we only have subsets $B^w_{i,j}$. In this setting we connect each big rectangle with the corresponding subset-nodes before and after rotating it by 90°. When we find a flow that satisfies each edge from the source we have assigned all big rectangles to the corresponding subsets and have decided whether we have to rotate a big rectangle, or not.

Transform Wide and Long Rectangles

We explain these steps for the wide rectangles, the transformation for the long rectangles is analogous. We have to decide whether a wide rectangle belongs to a bin of Type 1, i.e. to one set $W^w_{2/\delta}, \ldots, W^w_{2/\delta^2}$, or to a bin of Type 2, i.e. to one set $W^h_1, \ldots, W^h_{1/\delta^2}$. To do this, we guess the $1/\delta^2$ widths $w_{c_1}, \ldots, w_{c_{1/\delta^2}}$ that are used to round up the rectangles in the sets $W^h_1, \ldots, W^h_{1/\delta^2}$. This is done by choosing $1/\delta^2$ rectangles $r_{c_1}, \ldots, r_{c_{1/\delta^2}}$ out of n rectangles. Remember that $w_{c_1} \geq w_{c_2} \geq \ldots \geq w_{c_{1/\delta^2}}$. The total heights of the sets are identical, therefore we guess the total height of the whole set W^h approximately and divide it by $1/\delta^2$ to obtain the total height of each subset. The total height of all wide rectangles is bounded by $\delta^4 \cdot n$ since each wide rectangle has a height of at most δ^4. We choose 1 integral value i_0 out of $1/\delta^4 \cdot \delta^4 \cdot n = n$, so that $i_0 \cdot \delta^4 \leq h(W^h) < (i_0 + 1) \cdot \delta^4$ holds. This leads to an approximately guessed structure of the sets $W^h_1, \ldots, W^h_{1/\delta^2}$, since we know the widths and the heights.

The widths of the wide rectangles that are in the sets $W^w_{2/\delta}, \ldots, W^w_{2/\delta^2}$ are rounded up to the next multiple of $\delta^2/2$, hence the width of each rectangle in the set W^w_j is rounded up to $j\delta^2/2$. We guess approximately the total height of the rectangles in each set. Therefore, we choose $2/\delta^2 - 2/\delta + 1$ integral values $i_{2/\delta}, \ldots, i_{2/\delta^2}$ out of $1/\delta^4 \cdot \delta^4 \cdot n = n$, so that $i_j \cdot \delta^4 \leq h(W^w_j) < (i_j + 1)\delta^4$ holds for all $j \in \{2/\delta, \ldots, 2/\delta^2\}$. Consequently, we have the structure of all sets of wide rectangles and we have to assign the wide rectangles into them.

Note that $(i_0 + 1)\delta^4 + \sum_{z_1=2/\delta}^{2/\delta^2} (i_{z_1} + 1)\delta^4$ is larger than the total height of all wide rectangles in the instance.

ASSIGNING THE RECTANGLES We sort the wide rectangles according to non-increasing widths. Let r_w be a wide rectangle. r_w is a candidate for the set W_j^w, if r_w has a width of $w_w \in ((j-1)\delta^2/2, j\delta^2/2]$, for $j \in \{2/\delta+1,\dots,2/\delta^2\}$ and a candidate for $W_{2/\delta}^w$ if $w_w = \delta$. Furthermore, this rectangle is a candidate for W_j^h, if $w_w \in [w_{c_j}, w_{c_{j+1}}]$, for $j \in \{1,\dots,1/\delta^2-1\}$ or for W_{1/δ^2}^h if $\delta \le w_w \le w_{c_{1/\delta^2}}$.

First, we assign the wide rectangles greedily to the set W_{2/δ^2}^w. Therefore, we take the widest candidates for this set until the total height exceeds $(i_{2/\delta^2}+1)\cdot\delta^4$, i.e. the total height of these rectangles is at most $(i_{2/\delta^2}+1)\cdot\delta^4+\delta^4$. If we run out of candidates before we reach this height, we take the widest candidates for W_{2/δ^2-1}^w and so on. This is repeated for all sets $W_{2/\delta^2-1}^w,\dots,W_{2/\delta}^w$. We selected the widest candidates for these sets (cf. Figure 3.29). The remaining wide rectangles are greedily assigned in the same way into the sets $W_1^h,\dots,W_{1/\delta^2}^h$.

Lemma 3.25. *For the right guess of the values $i_0, i_{2/\delta},\dots,i_{2/\delta^2}$ and for the right guess of the rectangles $r_{c_1},\dots,r_{c_{1/\delta^2}}$ we assign each wide rectangle to one of the sets $W_{2/\delta}^w,\dots,W_{2/\delta^2}^w$ and $W_1^h,\dots,W_{1/\delta^2}^h$.*

Proof. Suppose by contradiction that there are rectangles that can not be assigned to one set for the right guess of the values $i_0, i_{2/\delta},\dots,i_{2/\delta^2}$ and for the right guess of the rectangles $r_{c_1},\dots,r_{c_{1/\delta^2}}$. Let the rectangle r_w be the widest rectangle among them. Suppose that r_w is a candidate for W_i^w and a candidate for W_j^h, for the largest possible j if r_w is a candidate for several sets. All rectangles that have a width of at least w_w, including r_w, have to fit fractionally into the sets $W_{2/\delta^2}^w,\dots,W_i^w$ and the sets $W_1^h,\dots,W_j^h$. Let X denote the set of wide rectangles that have a width of at least w_w, including r_w. Consequently, we have

$$h(X) \le \sum_{z_1=i}^{2/\delta^2} h(W_{z_1}^w) + \sum_{z_2=1}^{j} h(W_{z_2}^h) \le \sum_{z_1=i}^{2/\delta^2} (i_{z_1}+1)\delta^4 + \sum_{z_2=1}^{j} (i_0+1)\delta^4\cdot\delta^2$$

On the other hand, the sets have to be completely full, since we are not able to pack r_w into one of these sets, i.e. the total height of the rectangles of $W_{z_1}^w$ is larger than $(i_{z_1}+1)\delta^4$ and the total height of the rectangles in $W_{z_2}^h$ is larger than $(i_0+1)\delta^4\cdot\delta^2$, for all $z_1 \in \{i,\dots,2/\delta^2\}$ and $z_2 \in \{1,\dots,j\}$. It follows that $h(X) > \sum_{z_1=i}^{2/\delta^2}(i_{z_1}+1)\delta^4 + \sum_{z_2=1}^{j}(i_0+1)\delta^4\cdot\delta^2$ which is a contradiction. □

Consequently, all rectangles have to fit into these sets. Afterwards, we remove the shortest rectangles in each set $W_{2/\delta}^w,\dots,W_{2/\delta^2}^w$ and $W_1^h,\dots,W_{1/\delta^2}^h$, in order to secure that the total height is at most $i_j\delta^4$ and $i_0\delta^4\cdot\delta^2$, respectively. Therefore, we have to remove wide rectangles of a total height of at most $3\delta^4$ for each set since we have to reduce the total height by at most $2\delta^4$ and we have to remove the wide rectangle of height δ^4 that is cut by the new

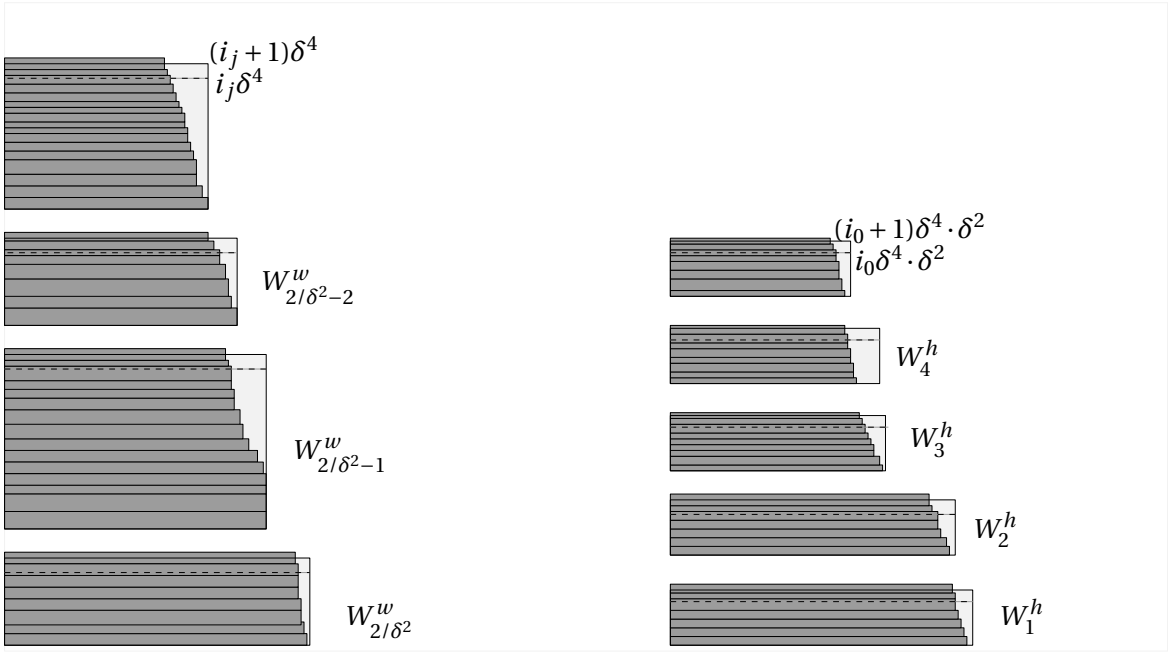


Figure 3.29: A greedy assignment of wide rectangles; sort the rectangles by their widths, pack them into the sets $W^w_{2/\delta^2},\dots,W^w_{2/\delta}$ until the last rectangle exceeds $(i_j+1)\delta^4$; afterwards pack the remaining rectangles into the sets $W^h_1,\dots,W^h_{1/\delta^2}$.

height. In total, we have wide rectangles of a total height of $(2/\delta^2-2\delta+1+1/\delta^2)\cdot 3\delta^4 \leq 9\delta^2$. These rectangles fit into one additional bin by packing them on top of each other. The same steps are done to assign the long rectangles to the sets $L^w_1,\dots,L^w_{1/\delta^2}$ and $L^h_{2/\delta},\dots,L^h_{2/\delta^2}$ and we need one additional bin.

To conclude, we guess $2\cdot(1/\delta^2+1+2/\delta^2-2/\delta+1) \leq 6/\delta^2$ rectangles and values out of n values and need two additional bins to transform the wide and long rectangles.

Transform Long and Wide Rectangles with Rotations In the version that allows rotation, we rotate each long rectangle in order to have only wide rectangles. We assign them to the sets $W^w_{2/\delta},\dots,W^w_{2/\delta^2}$ and $L^w_1,\dots,L^w_{1/\delta^2}$. As above, we guess approximately the total width of L^w and the total height of the sets $W^w_{2/\delta},\dots,W^w_{2/\delta^2}$. Furthermore, we guess the heights of the cut-rectangles, in order to get the structure of the sets $L^w_1,\dots,L^w_{1/\delta^2}$. Therefore, we choose $1/\delta^2$ rectangles and take the widths of the selected rectangles as the heights of the cut-rectangles, i.e. we rotate the selected rectangles. Afterwards, we greedily assign the wide rectangles to the sets $W^w_{2/\delta^2},\dots,W^w_{2/\delta}$. The remaining rectangles are rotated and they are assigned to the sets $L^w_1,\dots,L^w_{1/\delta^2}$. The rectangles that have to be removed fit into one additional bin. These are the only algorithmic differences between the version with and

without rotations and the remaining steps work for both versions. However, in the following we continue to state the minor differences to improve the additive constant.

CONSTRUCT THE CONTAINERS

We do the following steps for the long and wide containers that are packed in the bins of Type 1, the construction of the containers in the bins of Type 2 is analogous. Each wide container of C_W^w has a height of a multiple of δ^4 and a width of a multiple of $\delta^2/2$ (cf. Lemma 3.24). Hence, there are at most $1/\delta^4 \cdot 2/\delta^2 = 2/\delta^6$ different types of wide containers in the solution. We guess the number $n_{i,j}^w$ of the wide containers of each width $i\delta^2/2$ and each height $j\delta^4$ by choosing $2/\delta^6$ values out of n (we suppose, that each wide container contains at least one wide or small rectangle).

There are at most $1/\delta^2$ different types of long containers in C_L^w since all long containers have the same width and we rounded their heights to at most $1/\delta^2$ values. Each height is either a combination of the rounded heights of the long rectangles or a multiple of δ^4 (cf. Lemma 3.24). There are at most $(1/\delta^2+1)^{1/\delta}$ possibilities for the combinations of the rounded heights, since we have to choose at most $1/\delta$ values out of $1/\delta^2$ different heights. The additional 1 represents a dummy rectangle to choose less than $1/\delta$ values. We guess the heights of the long containers by choosing $1/\delta^2$ values out of $1/\delta^4 + (1/\delta^2+1)^{1/\delta}$. In a next step we guess the number n_i^ℓ of long containers of the ℓth height by choosing $1/\delta^2$ values out of n.

This is also done for the long and wide containers that are packed in the bins of Type 2. In total, we have to guess $2 \cdot (2/\delta^6 + 1/\delta^2) = 4/\delta^6 + 2/\delta^2$ values out of n and $2/\delta^2$ values out of $1/\delta^4 + (1/\delta^2)^{1/\delta}$. Note that each wide container has a width of at least $\delta^2/2$, no matter if it is packed in a bin of Type 1 or in a bin of Type 2. Furthermore, we can assume that $((3/2+22\delta)\cdot \mathrm{OPT} + 53) \leq n$, since otherwise we can pack each rectangle in a separate bin. Hence, the total height of the wide containers is bounded by $2/\delta^2 \cdot k \leq 2/\delta^2 \cdot ((3/2+22\delta)\cdot \mathrm{OPT}+53) \leq 2/\delta^2 \cdot n$. The same holds for the total width of all long containers.

3.3.2 PACKING THE RECTANGLES

We assigned the rectangles to their corresponding sets. It remains to pack the small, wide and long rectangles into the containers and to pack the containers and the big rectangles into the bins.

Packing Wide and Long Rectangles into the Containers

For packing the wide and long rectangles into the containers we use four similar linear programs. We explain the next steps for packing the wide rectangles into the wide containers, the packing for the long rectangles is analogous. A similar linear program formulation can be found in the AFPTAS by Kenyon & Rémila [40]. First, we focus on the wide containers of C_W^w that are packed in the bins of Type 1. Remember that all wide rectangles of W^w fit fractionally into the wide containers of C_W^w. There are at most $t := 2/\delta^2$ different widths of wide containers. We pack all containers of the same width $\ell \cdot \delta^2/2$, for $\ell \in \{1, \dots, t\}$, on top of each other and treat them as one target region T_ℓ of height $h(T_\ell) = h(C_{W_\ell}^w)$ and width $w(T_\ell) = \ell \cdot \delta^2/2$. The linear program will divide the target regions into slots of a certain width in which we will pack wide rectangles of the same width. Therefore, let $m := 2/\delta^2 - 2/\delta + 1$ be the number of different slots that have a width of $(i-1) \cdot \delta^2/2 + \delta$, for $i \in \{1, \dots, m\}$. For each target region T_ℓ we define a set of configurations $\mathfrak{C}_j^{(\ell)}$. A configuration in $\mathfrak{C}_j^{(\ell)}$ consists of a set of at most $1/\delta$ slots that have a total width of at most $w(T_\ell) = \ell \cdot \delta^2/2$. The total number of possibilities to select at most $1/\delta$ slots out of m different types is $(m+1)^{1/\delta}$. Therefore, the total number of configurations $q^{(\ell)}$ for target region T_ℓ is bounded by the number $(m+1)^{1/\delta}$. The value $a(i, \mathfrak{C}_j^{(\ell)})$ gives the number of slots of width $(i-1) \cdot \delta^2/2 + \delta$ in configuration $\mathfrak{C}_j^{(\ell)}$. We solve the following feasibility linear program, where the variable $x_j^{(\ell)}$ describes the height of configuration $\mathfrak{C}_j^{(\ell)}$.

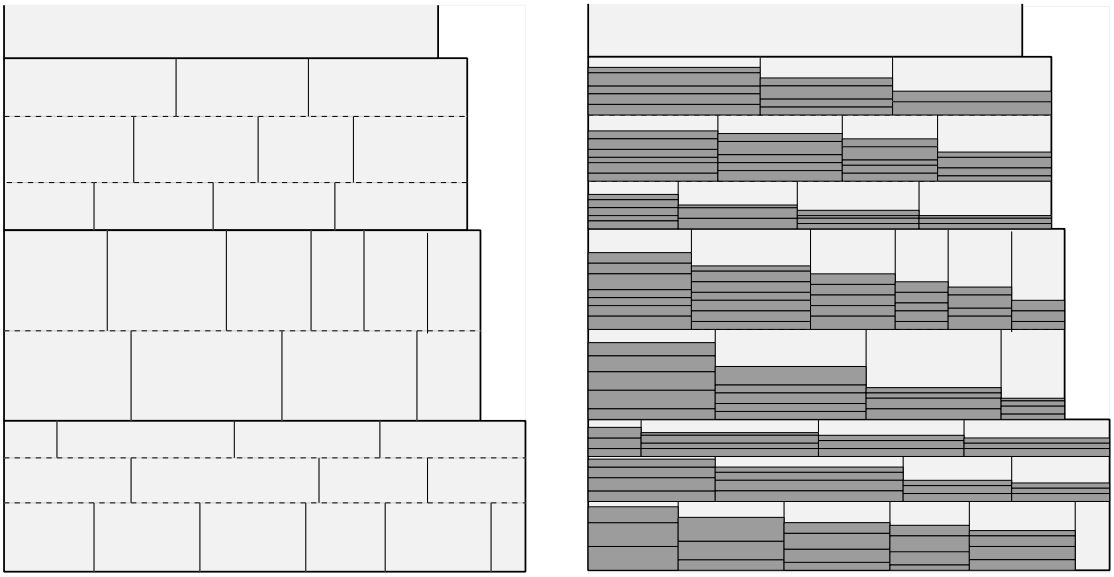
$$\sum_{j=1}^{q^{(\ell)}} x_j^{(\ell)} = h(T_\ell) \qquad \ell \in \{1, \dots, t\}$$

$$\sum_{\ell=1}^{t} \sum_{j=1}^{q^{(\ell)}} a(i, \mathfrak{C}_j^{(\ell)}) \cdot x_j^{(\ell)} \geq h(W_i^w) \qquad i \in \{1, \dots, m\}$$

$$x_j^{(\ell)} \geq 0 \qquad j \in \{1, \dots, q^\ell\}, \ell \in \{1, \dots, t\}$$

The first t constraints ensure that the total heights of the configurations do not exceed the total height of the target regions. We pack the wide rectangles of width $(i-1) \cdot \delta^2/2 + \delta$ into the slots of the same width. To this end, the following m constraints ensure that the total height of the slots is large enough to occupy the wide rectangles (cf. Figure 3.30). For the right guess of the values above, this linear program computes a feasible solution. The corresponding matrix of the linear program has $t + m \leq 4/\delta^2$ rows and $q := \sum_{\ell=1}^{t} q^{(\ell)} \leq t \cdot (m+1)^{1/\delta} \leq (2/\delta^2)^{2/\delta+1}$ columns. Each entry of the matrix is bounded by $1/\delta$. Since the total height of all containers is bounded by $2/\delta^2 \cdot n$ the entries on the right side are bounded by $2/\delta^2 \cdot n$. It follows that the encoding length of the input is bounded by $L :=$

$(t+m)\cdot(q+1)\cdot\log(2/\delta^2\cdot n) \leq 4/\delta^2\cdot((2/\delta^2)^{2/\delta+1}+1)\cdot\log(2/\delta^2\cdot n) \leq 4\cdot(2/\delta^2)^{2/\delta+2}\cdot\log(2/\delta^2\cdot n)$. We can solve this linear program with a result of Vaidya [52], that computes a feasible basic solution in time $\mathcal{O}((((t+m)+q)q^2+((t+m)+q)^{3/2}q)L) = \mathcal{O}(q^3\cdot L) = \mathcal{O}((2/\delta^2)^{6/\delta+3}\cdot (2/\delta^2)^{2/\delta+2}\cdot\log(2/\delta^2\cdot n)) = \mathcal{O}((2/\delta^2)^{8/\delta+5}\cdot\log(2/\delta^2\cdot n))$



(a) The slots in the configurations computed by the linear program

(b) Packing the rectangles into the slots

Figure 3.30: Packing the wide rectangles into the containers

The rank of the matrix is bounded by the number of constraints which is at most $m+t = 2/\delta^2 - 2/\delta + 1 + 2/\delta^2 < 4/\delta^2$. It follows that the feasible basic solution contains less than $4/\delta^2$ non-zero variables x_j^ℓ and thus we have less than $4/\delta^2$ configurations. We pack the wide rectangles of $W^w_{2/\delta},\ldots,W^w_{2/\delta^2}$ greedily into the configurations, by packing them on top of each other into a slot of the same width until the last rectangle exceeds the height of the configuration. Since the rectangles fit fractionally into the slots, there is no rectangle unpacked. Afterwards, we remove the uppermost rectangles that exceed the height of the configuration. Therefore we pack the removed rectangles of one configuration next to each other into an additional bin. Thus, we need a total height of $\delta^4\cdot 4/\delta^2 = 4\delta^2$ to pack all rectangles into one additional bin.

We sort the slots in each configuration by non-increasing packing heights, i.e. the leftmost slot occupies rectangles of the largest total height and the rightmost slot occupies rectangles of the smallest total height. The free space on the right of the configurations is separated into rectangular regions in order to pack small rectangles into it. This is done by drawing

a horizontal line on the topmost rectangle in each slot. We have at most $1/\delta$ slots in each configuration and hence at most $1/\delta + 1$ different rectangular regions. In total, there are less than $(4/\delta^2) \cdot (1/\delta + 1) = 4/\delta^3 + 4/\delta^2 \leq 5/\delta^3$ different rectangular regions (cf. Figure 3.31).

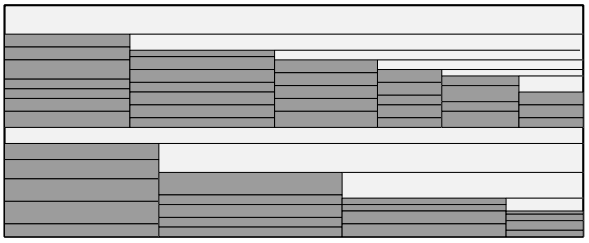


Figure 3.31: Rectangular regions for the small rectangles

We use the same linear program for the wide rectangles that are packed in wide containers in the bins of Type 2. The difference is that there are only $t = 1/\delta^2$ different target regions and that there are only $m = 1/\delta^2$ different widths of the rectangles and slots. Hence, we have only $m + t = 2/\delta^2$ constraints and therefore only $2/\delta^2$ different configurations. We pack the wide rectangles of $W_1^h, \ldots, W_{2/\delta^2}^h$ greedily into the configurations and we remove again the topmost rectangles. The removed rectangles are packed on top of the wide rectangles in the additional bin above. They need an additional space of height $\delta^4 \cdot 2/\delta^2 = 2\delta^2$. This results into a total packing height of at most $4\delta^2 + 2\delta^2 = 6\delta^2$. On the right side of these configurations we have at most $2/\delta^2 \cdot (1/\delta + 1) \leq 3/\delta^3$ rectangular regions.

We do the same steps for packing the long rectangles into the long containers by packing some remaining long rectangles into a second additional bin. Consequently, we have at most $2 \cdot (5/\delta^3 + 3/\delta^3) = 2 \cdot 8/\delta^3$ free rectangular regions for small rectangles. In the version that allows rotations we only use one additional bin for occupying the wide and the rotated long rectangles.

Solving the Linear Programs Approximately We give now a short description how to solve the linear programs approximately in order to reduce the running time. However, for the sake of readability we assume in the following sections that we have solved the linear programs exactly, as mentioned above. We solve each linear program approximately with an algorithm for the max-min resource sharing problem [22, 29] as explained in [7]. For a precision of δ^4 the algorithm stops after $\mathcal{O}(m(1/\delta^8 + \ln m)) = \mathcal{O}(1/\delta^{10})$ iterations. In each iteration a block problem has to be solved approximately with precision $\delta^4/6$. In our case the block problem consists of t knapsack problems with m unbounded variables. The t knapsack problems can be solved in time $\mathcal{O}(t \cdot (m \log(1/\delta^4) + 1/\delta^{16})) = \mathcal{O}(1/\delta^{18})$ [41]. The

total running time is therefore $\mathcal{O}(1/\delta^{28})$. We obtain variables $\overline{x}_j^{(\ell)}$ that satisfies

$$\sum_{\ell=1}^{t}\sum_{j=1}^{q^{(\ell)}} a(i,\mathfrak{C}_j^{(\ell)})\cdot\overline{x}_j^{(\ell)} \geq (1-\delta^4)h(W_i^w) \qquad i\in\{1,\ldots,m\}$$

$$\sum_{j=1}^{q^{(\ell)}} \overline{x}_j^{(\ell)} = h(T_\ell) \qquad \ell\in\{1,\ldots,t\}$$

$$\overline{x}_j^{(\ell)} \geq 0 \qquad j\in\{1,\ldots,q^\ell\},\ell\in\{1,\ldots,t\}.$$

The number of configurations is bounded by the number of iterations multiplied with t by $\mathcal{O}(t\cdot 1/\delta^{10}) = \mathcal{O}(1/\delta^{12})$. We reduce the number of configurations to $t+m$ by solving $\mathcal{O}(1/\delta^{12})$ systems of $t+m$ linear equalities with $t+m+1$ variables in time $\mathcal{O}((t+m)^3\cdot 1/\delta^{10}) = \mathcal{O}(1/\delta^{16})$ as explained in [29].

In order to secure the covering constraints we extend each configuration by setting $x_j^{(\ell)} := (1+2\delta^4)\overline{x}_j^{(\ell)}$. Since $\delta^4 \leq 1/2$ we have for each $i\in\{1,\ldots,m\}$

$$\sum_{\ell=1}^{t}\sum_{j=1}^{q^{(\ell)}} a(i,\mathfrak{C}_j^{(\ell)})\cdot x_j^{(\ell)} \geq (1+2\delta^4)(1-\delta^4)h(W_i^w) = (1+\delta^4-2\delta^8)h(W_i^w) \geq h(W_i^w).$$

The heights of the configurations are extended for each target region T_ℓ to $\sum_{j=1}^{q^{(\ell)}} x_j^{(\ell)} = (1+2\delta^4)h(T_\ell)$.

We approximately solve also the linear program for packing the wide rectangles into the bins of Type 2 and pack all wide rectangles as described above. Afterwards we have to remove the rectangles in the uppermost strips of each target region. There are $2/\delta^2 + 1/\delta^2 = 3/\delta^2$ target regions for the wide rectangles that are packed in bins of both types. We remove strips of wide rectangles of the total height at most $2\delta^4 h(T_\ell) + \delta^4$ from each target region. The total height of all target regions is bounded by $2/\delta^2\cdot((3/2+22\delta)\cdot\mathrm{OPT}+53)$ (each target region has a width of at least $2/\delta^2$). Therefore, we remove strips of wide rectangles of the total height

$$\begin{aligned}
2\delta^4\cdot 2/\delta^2\cdot((3/2+22\delta)\cdot\mathrm{OPT}+53) + \delta^4\cdot 3/\delta^2 = \\
4\delta^2\cdot((3/2+22\delta)\cdot\mathrm{OPT}+53) + 3\delta^2 \leq \\
(6\delta^2+88\delta^3)\mathrm{OPT} + 215\delta^2 \leq \\
(6\delta^2+88\delta^2/48)\mathrm{OPT} + 215/48^2 \leq \\
8\delta^2\mathrm{OPT}+1
\end{aligned}$$

We pack these strips on top of each other and cut the packing on each integral height in order to pack the rectangles in $\lceil 8\delta^2\mathrm{OPT}+1\rceil$ additional bins. We remove the rectangles that are split by these cutting lines and pack them separately. We are able to pack rectangles of $1/\delta^4$ cutting lines on top of each other into one bin. Therefore, we need $\lceil \delta^4\cdot(8\delta^2\mathrm{OPT}+1)\rceil \leq \lceil \delta^2\mathrm{OPT}+1\rceil$ additional bins. In total we need less than $9\delta^2\mathrm{OPT}+4$ additional bins. The same holds for packing the long rectangles into the target regions.

Packing Small Rectangles into Rectangular Regions

We pack the small rectangles into the rectangular regions defined above. Remember that they fit fractionally into these regions. We have at most $2\cdot 8/\delta^3$ different rectangular regions for the small rectangles. We use Next Fit Decreasing Height by Coffman et al. [13] to pack them into these regions. Since these rectangles are small we are able to cover almost the whole region with small rectangles.

We give a short description of this algorithm for the sake of completeness. Next Fit Decreasing Height sorts the rectangles according to non-increasing heights. In this order the algorithm packs the rectangles left-justified on a level, until there is insufficient space at the right to accommodate the next rectangle. This level is closed and the algorithm packs no further rectangle on it. If this is the first level, the algorithm packs the level on the ground of the target region, in any other case, we place this level on top of the first rectangle of the previous level. Then the algorithm proceeds packing on the next level, until it runs out of rectangles, or the next level does not fit into the target region.

We obtain the following Theorem by Coffman et al. [13]:

Lemma 3.26. *Let A be a rectangular region of width w_A and height h_A. We are able to pack small rectangles into A with a total area of at least $w_A\cdot h_A-(w_A+h_A)\cdot\delta^4$.*

Proof. Let t be the number of levels $L_1,\dots,L_t$ that are packed with Next Fit Decreasing Height into region A. We suppose that we have enough small rectangles and the algorithm stops, because the next level does not fit into A. Let r_i be the first rectangle on the level L_i and let $r_{i'}$ be the last rectangle on the level L_i. The height of the level L_i is the height h_i of the first rectangle on this level. Additionally, let $h_{t+1}=h_A-\sum_{i=1}^{t}h_i$ be the free space on top of the level L_t. Consequently, we have $\sum_{i=1}^{t+1}h_i=h_A$. It holds $h_{i'}\geq h_{i+1}$ since the rectangles are sorted according to non-increasing heights. Furthermore, let $w(L_i)$ be the total width of the rectangles on level L_i. We have $w(L_i)>w_A-\delta^4$ since the next small rectangle of width at most δ^4 does not fit on this level. The total area of the rectangles on level L_i is hence $a(L_i)\geq h_{i'}\cdot w(L_i)\geq h_{i+1}\cdot w(L_i)>h_{i+1}\cdot(w_A-\delta^4)$. Thus, the total area of the packed

rectangles is at least

$$\sum_{i=1}^{t} a(L_i) > \sum_{i=1}^{t} h_{i+1} \cdot (w_A - \delta^4) = (h_A - h_1) \cdot (w_A - \delta^4) >$$
$$(h_A - \delta^4) \cdot (w_A - \delta^4) > w_A \cdot h_A - (w_A + h_A) \cdot \delta^4.$$

□

The total height of the wide containers in the bins of Type 1 and Type 2 is bounded by $2/\delta^2 \cdot ((3/2+22\delta) \cdot \mathrm{OPT}+53)$ (each wide container has a width of at least $\delta^2/2$). Thus, the sum of the heights of the rectangular regions on the right side of the configurations with the wide rectangles is bounded by $2/\delta^2 \cdot ((3/2+22\delta) \cdot \mathrm{OPT}+53)$. The width of each wide container is at most 1 and hence the sum of the widths of the at most $8/\delta^3$ rectangular regions is bounded by $8/\delta^3$. Therefore, in all target regions A for the wide rectangles there is only a free total area of at most

$$\begin{aligned}
\sum_A (w_A + h_A) \cdot \delta^4 &= \delta^4 \cdot (\sum_A w_A + \sum_A h_A) \\
&\leq \delta^4 \cdot (8/\delta^3 \cdot 1 + 2/\delta^2 \cdot ((3/2+22\delta) \cdot \mathrm{OPT}+53)) \\
&= 8\delta + 2\delta^2 \cdot ((3/2+22\delta) \cdot \mathrm{OPT}+53) \\
&= 3\delta^2 \cdot \mathrm{OPT} + 44\delta^3 \cdot \mathrm{OPT} + 8\delta + 106\delta^2 \\
&\leq 3\delta^2 \cdot \mathrm{OPT} + 44\delta^2/48 \cdot \mathrm{OPT} + 8\delta + 106\delta/48 \\
&< 4\delta^2 \cdot \mathrm{OPT} + 11\delta
\end{aligned}$$

left. The same bound holds also for the target areas for the long rectangles and we obtain a total free area of at most $2 \cdot (4\delta^2 \cdot \mathrm{OPT} + 11\delta) \leq 8\delta^2 \cdot \mathrm{OPT} + 22\delta$. Since all small rectangles fit fractionally into the containers, it follows that the total area of the unpacked small rectangles is bounded by this value.

These small rectangles fit with Lemma 3.26 into $\delta\mathrm{OPT} + 1$ additional bins, since we are able to pack small rectangles of a total area at least $1 - (1+1) \cdot \delta^4 = 1 - 2\delta^4$ into one bin and we have

$$(1 - 2\delta^4) \cdot (\delta\mathrm{OPT} + 1) = \delta\mathrm{OPT} - 2\delta^5\mathrm{OPT} + 1 - 2\delta^4$$
$$\geq 48\delta^2\mathrm{OPT} - 2\delta^5\mathrm{OPT} + 48\delta - 2\delta^4 \geq 8\delta^2\mathrm{OPT} + 22\delta.$$

CUTTING OUT CONTAINERS

We treated all wide containers of the same width and all long containers of the same height as one target region. The total number of containers is bounded in Lemma 3.24 by $6/\delta^3 \cdot p_1 + 2 + 2/\delta^3 \cdot (k - p_1) + 2/\delta^2$ wide containers and $6/\delta^3 \cdot (k - p_1) + 2 + 2/\delta^3 \cdot p_1 + 2/\delta^2$ long containers. It is left to cut the containers out of the target regions. We cut hereby wide and small rectangles of height δ^4 or long and small rectangles of width δ^4. Hence, we are able to pack the cut rectangles of $1/\delta^4$ horizontal or vertical cut lines into one additional bin. Consequently, we need

$$\begin{aligned}
&\lceil \delta^4 \cdot (6/\delta^3 \cdot p_1 + 2 + 2/\delta^3 \cdot (k - p_1) + 2/\delta^2) \rceil + \\
&\lceil \delta^4 \cdot (6/\delta^3 \cdot (k - p_1) + 2 + 2/\delta^3 \cdot p_1 + 2/\delta^2) \rceil \leq \\
&\delta^4 \cdot (8/\delta^3 \cdot k + 4/\delta^2 + 4) + 2 \leq \\
&\delta^4 \cdot (8/\delta^3 \cdot ((3/2 + 5\delta)\mathrm{OPT} + 37) + 4/\delta^2 + 4) + 2 = \\
&8\delta \cdot ((3/2 + 5\delta)\mathrm{OPT} + 37) + 4\delta^2 + 4\delta^4 + 2 = \\
&8\delta \cdot (3/2 + 5\delta)\mathrm{OPT} + 8\delta \cdot 37 + 4\delta^2 + 4\delta^4 + 2 = \\
&(12\delta + 40\delta^2)\mathrm{OPT} + 296\delta + 4\delta^2 + 4\delta^4 + 2 \leq \\
&(12\delta + 40\delta/48)\mathrm{OPT} + 296/48 + 4/48^2 + 4/48^4 + 2 \leq \\
&13\delta\mathrm{OPT} + 7 + 2 = \\
&13\delta\mathrm{OPT} + 9
\end{aligned}$$

additional bins.

CUTTING OUT CONTAINERS WITH ROTATIONS The number of wide containers and long containers in the version that allows rotation is bounded in Lemma' 3.24 by $6/\delta^3 \cdot k' + 2$ and $2/\delta^3 \cdot k' + 2/\delta^2$, respectively. Therefore, we have $6/\delta^3 \cdot k' + 2 + 2/\delta^3 \cdot k' + 2/\delta^2 = 8/\delta^3 \cdot k' + 2/\delta^2 + 2$ cutting lines. We rotate the rectangles that are cut by the construction of the long containers. Thus, we have only horizontal cutting lines and thus cut rectangles of height δ^4.

Consequently, we need $\lceil \delta^4 \cdot (8/\delta^3 \cdot k' + 2/\delta^2 + 2) \rceil \leq \delta^4 \cdot (8/\delta^3 \cdot k' + 2/\delta^2 + 2) + 1$ bins. We obtain

$$\begin{aligned}
\delta^4 \cdot (8/\delta^3 \cdot k' + 2/\delta^2 + 2) + 1 \leq \\
\delta^4 \cdot (8/\delta^3 \cdot ((3/2 + 5\delta)\mathrm{OPT} + 21) + 2/\delta^2 + 2) + 1 \leq \\
8\delta \cdot (3/2 + 5\delta)\mathrm{OPT} + 8\delta \cdot 21 + 2\delta^2 + 2\delta^4 + 1 \leq \\
13\delta\mathrm{OPT} + 168\delta + 2\delta^2 + 2\delta^4 + 1 \leq \\
13\delta\mathrm{OPT} + 168/48 + 2/48^2 + 2/48^4 + 1 \leq \\
13\delta\mathrm{OPT} + 4 + 1 = \\
13\delta\mathrm{OPT} + 5
\end{aligned}$$

additional bins.

Packing Big Rectangles and Containers

The last remaining step is to pack the big rectangles and the long and wide containers into the bins. Therefore, we use almost the same linear program as above. Again, we explain the following steps for the bins of Type 1. One configuration C_j, for $j \in \{1, \ldots, q\}$, consists of a packing into one bin. There are at most $2/\delta^2 \cdot 1/\delta^2 = 2/\delta^4$ different types of big rectangles, $2/\delta^6$ different types of wide containers and $1/\delta^2$ different types of long container. In each bin/configuration there are at most $1/\delta^2$ big rectangles, $6/\delta^3$ wide containers and $2/\delta^3$ long containers (see description above Lemma 3.24). Therefore, there are at most $(2/\delta^4 + 1)^{1/\delta^2}$ possibilities to select at most $1/\delta^2$ big rectangles out of $2/\delta^4$ different types. The additional 1 represents a dummy rectangle and is needed for selecting less than k big rectangles. There are at most $(2/\delta^6 + 1)^{6/\delta^3}$ possibilities to select at most $6/\delta^3$ wide containers out of $2/\delta^6$ different types and $(1/\delta^2 + 1)^{2/\delta^3}$ possibilities to select at most $2/\delta^3$ long containers out of $1/\delta^2$ different types. All together, the number q of different configurations is therefore bounded by

$$\begin{aligned}
q &\leq (2/\delta^4 + 1)^{1/\delta^2} \cdot (2/\delta^6 + 1)^{6/\delta^3} \cdot (1/\delta^2 + 1)^{2/\delta^3} \\
&< (4/\delta^4)^{1/\delta^2} \cdot (4/\delta^6)^{6/\delta^3} \cdot (2/\delta^2)^{2/\delta^3} \\
&= (1/\delta)^{4/\delta^2} \cdot (1/\delta)^{36/\delta^3} \cdot (1/\delta)^{4/\delta^3} \cdot 2^{2/\delta^2} \cdot 2^{12/\delta^3} \cdot 2^{2/\delta^3} \\
&\leq (1/\delta)^{41/\delta^3} \cdot 2^{15/\delta^3}.
\end{aligned}$$

We have to verify, if a candidate for a configuration fits into a bin. Each wide and long container and each big rectangle has a width of a multiple of $\delta^2/2$. Therefore, we are able

to pack them with its x-coordinate on a multiple of $\delta^2/2$. For each candidate we guess the x-coordinates of all containers and big rectangles by choosing $1/\delta^2 + 6/\delta^3 + 2/\delta^3 \leq 9/\delta^3$ values out of $2/\delta^2$. Consequently, we have for each multiple of $\delta^2/2$ one set of big rectangles and containers that starts on the corresponding x-coordinate or intersect this x-coordinate completely. It remains to find an order of these containers and big rectangles to find a packing. Since there are at most $1/\delta^4$ objects in each set, there are at most $1/\delta^4!$ possible permutations. In total we have to try $2/\delta^2 \cdot 1/\delta^4! \leq 2/\delta^2 \cdot (1/\delta^4)^{1/\delta^4}$ permutations to find a packing of this configuration. If we do not find a packing at all, then there exists no packing of this configuration and we delete it.

Afterwards, we select the configurations that have to be packed into the bins. We need for each configuration one bin. To select these configurations we employ an integer linear program. Therefore, denote with $b(i,j,C_k)$ the number of big rectangles in the set $B^w_{i,j}$ in configuration C_k. With $w(i,j,C_k)$, we denote the number of wide containers of width $i\delta^2/2$ and height $j\delta^4$ and with $\ell(i,C_k)$ we denote the number of long containers of the ith height in configuration C_k. The total number of big rectangles in the set $B^w_{i,j}$ is denoted by $n^b_{i,j}$, the total number of wide containers of the width $i\delta^2/2$ and of the height $j\delta^4$ is denoted by $n^w_{i,j}$ and the number of long containers of the ith height by n^ℓ_i.

The integer linear program is defined as follows:

$$
\begin{aligned}
&\min \sum_{k=1}^{q} x_k && \\
s.t. &\sum_{k=1}^{q} b(i,j,C_k) \cdot x_k \geq n^b_{i,j} && i \in \{2/\delta, \dots, 2/\delta^2\}, j \in \{1, \dots, 1/\delta^2\} \\
&\sum_{k=1}^{q} w(i,j,C_k) \cdot x_k \geq n^w_{i,j} && i \in \{2/\delta, \dots, 2/\delta^2\}, j \in \{1, \dots, 1/\delta^4\} \\
&\sum_{k=1}^{q} \ell(i,C_k) \cdot x_k \geq n^\ell_i && i \in \{1, \dots, 1/\delta^2\} \\
&x_k \in \mathbb{N} && k \in \{1, \dots, q\}
\end{aligned}
$$

This integer linear program can be solved with the algorithm of Kannan [39] in time $q^{\mathcal{O}(q)} \cdot s$, while s is the input size. We have $(2/\delta^2 - 2/\delta) \cdot 1/\delta^2 + (2/\delta^2 - 2/\delta) \cdot 1/\delta^4 + 1/\delta^2 \leq 3/\delta^6$ constraints. Each coefficient in the matrix is bounded by $6/\delta^3$ and the values $n^b_{i,j}, n^w_{i,j}$ and n^ℓ_i are bounded by n. It follows, that $s \leq (q+1) \cdot 3/\delta^6 \cdot \log(n)$. Thus, the the total running time, including the construction of the configurations, is bounded by $\mathcal{O}(\log(n) \cdot q^{\mathcal{O}(q)})$. We can improve the running-time with a result of Eisenbrand & Shmonin [18]:

Theorem 3.7. *Let $X \subset \mathbb{Z}^d$ be a finite set of integer vectors and let $b \in \{\sum_{i=1}^{t} \lambda_i x_i | t \geq$*

$0; x_1, \ldots, x_t \in X; \lambda_1, \ldots \lambda_t \in \mathbb{Z}_{\geq 0}\}$. *Then there exists a subset* $\tilde{X} \subseteq X$ *such that* $b \in \{\sum_{i=1}^{t} \lambda_i x_i | t \geq 0; x_1, \ldots, x_t \in \tilde{X}; \lambda_1, \ldots \lambda_t \in \mathbb{Z}_{\geq 0}\}$ *and* $|\tilde{X}| \leq 2d \log(4dM)$ *with* $M = \max_{x \in X} \|x\|_\infty$.

In our case, the set X belongs to the configurations. We have at most $3/\delta^6$ constraints, thus $d \leq 3/\delta^6$. The coefficients of the matrix are bounded by $M = 6/\delta^3$. Theorem 3.7 states that there are at most $q' := 2d \log(4dM) \leq 2 \cdot 3/\delta^6 \log(4 \cdot 3/\delta^6 \cdot 6/\delta^3) \leq 6/\delta^6 \log(62/\delta^9)$ non-zero variables in any solution b of our integer linear program. We enumerate all configurations of cardinality at most q' and have at most $(q+1)^{q'}$ possibilities. For each set of at most q' configurations we solve the reduced integer linear program with the algorithm of Kannan [39] in time $q'^{\mathcal{O}(q')} \cdot s'$ while s' is the input size of the reduced integer linear program. We bound s' by $(q'+1) \cdot d \cdot \log(n)$. Hence, the total running time is bounded by $\mathcal{O}((q+1)^{q'} \cdot \log(n) \cdot q'^{\mathcal{O}(q')})$.

The same integer linear program is solved for the bins of Type 2. Since we know that there is a packing into $(3/2 + 24\delta) \cdot \mathrm{OPT} + 53$, these integer linear programs compute for the right guess of all above described values a solution with at most $(3/2 + 24\delta) \cdot \mathrm{OPT} + 53$ bins.

3.3.3 Résumé of the Algorithm

A compressed version of our algorithm is given in Algorithm 3.1.

Algorithm 3.1 Algorithm for Two-Dimensional Bin Packing

```
1: set ε' := min{ε/39, 1/48}
2: Find OPT' ≤ OPT with binary search so that algorithm computes feasible solution for each guess do
3:     Compute δ and pack medium rectangles with Steinberg's Theorem 3.5
4:     Find structure of the set of big, long and wide rectangles and of the set of wide and long containers for each guess do
5:         Solve flow network with the algorithm of Dinic [17]
6:         Greedy assignment of long and wide rectangles into groups
7:         Pack the long and wide rectangles into containers with linear programs that are solved by the algorithm of Vaidya [52]
8:         Pack the small rectangles with Next Fit Decreasing Height by Coffman et al. [13]
9:         Pack containers and big rectangles with integer linear programs that are solved by the algorithm of Kannan [39]
```

The running time of the steps are given as follows. The binary search takes $\mathcal{O}(\log n)$ time. To find δ, we have to compute $2/\varepsilon'$ sets and check whether their value is at most $\varepsilon' \cdot \mathrm{OPT}$. This takes time $\mathcal{O}(n \cdot 2/\varepsilon') = \mathcal{O}(n/\varepsilon)$.

We pack the $3\varepsilon'\mathrm{OPT} + 2$ sets with the algorithm of Steinberg that has a running time of $\mathcal{O}((n \log^2 n)/\log\log n)$. Since $\mathrm{OPT} \leq n$ and $\varepsilon' < 1$, we obtain a total running time for this step

of $\mathcal{O}((n^2 \log^2 n)/\log\log n)$. The value of δ is at least $\delta \geq \varepsilon'^{4 \cdot 2/\varepsilon'}$. For the structure of the sets for the big rectangles we have to guess $4/\delta^2 + 4/\delta^4$ values out of n. We obtain the structure of the sets of the wide and long rectangles by guessing at most $6/\delta^3$ values out of n. We compute the structure of the wide and long containers by guessing $4/\delta^6 + 2/\delta^2$ values out of n and $2/\delta^2$ values out of $1/\delta^4 + (1/\delta^2)^{1/\delta}$. In total we have to choose less than $5/\delta^6$ values out of n and $2/\delta^2$ values out of $1/\delta^4 + (1/\delta^2)^{1/\delta}$. This takes time $\mathcal{O}(n^{5/\delta^6} \cdot (1/\delta)^{2/\delta})$. To solve the flow network, we need time $\mathcal{O}(n^3/\delta^2 + n^2/\delta^6 + n/\delta^{10} + 1/\delta^{14})$. The assignment of the long and wide rectangles into the groups is done in linear time. We solve the four linear programs to pack the wide and long rectangles into the containers in time $\mathcal{O}((2/\delta^2)^{8/\delta+5} \cdot \log(2/\delta^2 \cdot n))$. The packing of the small rectangles and to cut out the containers afterwards is done in less than $\mathcal{O}(n \log n/\delta^3)$ time. The integer linear programs are solved in time $\mathcal{O}((q+1)^{q'} \cdot \log(n) \cdot q'^{\mathcal{O}(q')})$, with $q \leq (1/\delta)^{41/\delta^3} \cdot 2^{15/\delta^3}$ and $q' \leq 6/\delta^6 \log(62/\delta^9)$.

To conclude, the running time is bounded by $\mathcal{O}(n^{f(1/\varepsilon)} \cdot g(1/\varepsilon))$ for some functions f and g. We obtain the following result for the two-dimensional bin packing problem with and without rotations:

Theorem 3.1. *For any $\varepsilon > 0$, there is an approximation algorithm A which produces a packing of a list I of n rectangles in $A(I)$ bins such that*

$$A(I) \leq (3/2 + \varepsilon) \cdot \mathrm{OPT}(I) + 69.$$

The running time of A is polynomial in n.

Proof. The integer linear programs packs the big rectangles and the containers in at most $(3/2 + 22\delta) \cdot \mathrm{OPT} + 53$ bins. The medium rectangles are packed into $3\varepsilon'\mathrm{OPT} + 2$ bins. To transform the wide and long rectangles we need 2 additional bins. We pack the wide and long rectangles with the linear programs into the target regions. Therefore, we need also 2 additional bins. The small rectangles are packed into the target regions and into $\delta\mathrm{OPT} + 1$ additional bins. Afterwards, we cut the containers out of the target regions and need $13\delta\mathrm{OPT} + 9$ additional bins. It follows that we need

$$(3/2 + 22\delta) \cdot \mathrm{OPT} + 53 + 3\varepsilon'\mathrm{OPT} + 2 + 2 + 2 + \delta\mathrm{OPT} + 1 + 13\delta\mathrm{OPT} + 9 \leq (3/2 + 39\varepsilon') \cdot \mathrm{OPT} + 69$$

bins. Since $\varepsilon' \leq \varepsilon/39$ we obtain $(3/2 + \varepsilon)\mathrm{OPT} + 69$ bins in total. □

RÉSUMÉ OF THE ALGORITHM WITH ROTATIONS In the version that allows rotation, the integer linear program packs the rectangles into $k' \leq (3/2 + 22\delta) \cdot \mathrm{OPT} + 30$ bins. The medium rectangles are packed into $3\varepsilon' \cdot \mathrm{OPT} + 1$ additional bins. To assign the long and wide rectangles to the groups and to pack them into the target regions we need in total 2 additional bins. To pack the small rectangles we need $\delta\mathrm{OPT}+1$ additional bins and to cut the containers out of the target regions we need $13\delta\mathrm{OPT}+5$ additional bins. Consequently, the total number of used bins is at most

$$(3/2 + 22\delta) \cdot \mathrm{OPT} + 30 + 3\varepsilon' \cdot \mathrm{OPT} + 1 + 2 + \delta\mathrm{OPT} + 1 + 13\delta\mathrm{OPT} + 5 \leq$$
$$(3/2 + 39\varepsilon') \cdot \mathrm{OPT} + 39 \leq (3/2 + \varepsilon) \cdot \mathrm{OPT} + 39$$

bins. We obtain the following result:

Theorem' 3.1. *For any $\varepsilon > 0$, there is an approximation algorithm A which produces a packing of a list I of n rectangles that are allowed to be rotated in $A(I)$ bins such that*

$$A(I) \leq (3/2 + \varepsilon) \cdot \mathrm{OPT}(I) + 39.$$

The running time of A is polynomial in n.

3.4 CONCLUSION

We presented a technique that allows us to modify any solution of the two-dimensional bin packing problem into a solution that consists of a simpler structure. This enables our algorithm to compute a solution into $(3/2 + \varepsilon) \cdot \mathrm{OPT} + 69$ bins and an improved solution of $(3/2 + \varepsilon) \cdot \mathrm{OPT} + 39$ in the version that allows rotation for any fixed $\varepsilon > 0$ and any instance that fits optimally into OPT bins. The current lower bound is given by Chlebík & Chlebíková [11] with values $1 + 1/3792$ and $1 + 1/2196$ for the version with and without rotations, respectively. An open question is to close the gap between the current lower bounds and our presented asymptotic approximation ratios. Therefore, it is of interest to find an approximation algorithm with an asymptotic approximation ratio of $4/3$, if there is any. Maybe there is a way to adopt our techniques by modifying an optimal solution so that the rectangles are rounded up. However, there would be only one additional bin for each sequence of three bins, instead of one additional bin for each sequence of two bins. Therefore. it would be necessary to do a much more sensitive and complex case analysis.

4 Faster Approximation Algorithms for Scheduling with Fixed Jobs

4.1 Introduction

We consider the problem of parallel machine scheduling where either some jobs are already fixed in the system or there are intervals of non-availability of some machines. These problems are already studied in [14, 15, 28, 42, 43, 44, 47] and relevant for turnaround scheduling [45] and distributed computing where machines are donated on a volunteer basis.

Formally, the problem can be defined as follows: an instance consists of m, the number of machines, considered part of the input and n jobs with non-negative processing times $p_1, \ldots, p_n \in \mathbb{N}$. The first k jobs are fixed via a list $(m_1, s_1), \ldots, (m_k, s_k)$ giving a machine index and starting time for the respective job. We assume that these fixed jobs do not overlap. A schedule is a non-preemptive assignment of the jobs to machines and starting times such that the first k jobs are assigned as encoded in the instance and that the jobs do not intersect.

For the problem with *fixed jobs*, the objective is to minimize the makespan $C_{\max}$ of all jobs, including the fixed ones. In the setting with *non-availability*, the fixed jobs are not included when finding the makespan.

Without loss of generality, we may assume $m < n$: if $m \geq n$, there are at least $m-k \geq n-k$ machines without fixed jobs on them, which can execute the $n-k$ unfixed jobs optimally in a trivial way.

Both problems generalize the well-known problem $P||C_{\max}$(scheduling jobs on parallel identical machines to minimize makespan) [27] and hence are strongly NP-hard.

Related work. Scheduling with fixed jobs was studied by Scharbrodt [46] and Scharbrodt et al. [47]. They mainly studied the problem for constant m; for this strongly NP-hard formulation (which consequently does not admit an FPTAS) they present a PTAS. They also

found approximation algorithms for general m with ratios 3 [46] and $2+\epsilon$ [47]; since the finishing time of the last fixed job is a lower bound for the optimal makespan OPT, we can simply use a PTAS for the well-known problem $P||C_{\max}$ [27] to schedule the remaining $n-k$ jobs after the fixed job which finishes last. Scharbrodt et al. [47] also proved that for scheduling with fixed jobs there is no approximation algorithm with ratio $3/2-\epsilon$, unless P = NP, for any $\epsilon \in (0, 1/2]$. In [14], Diedrich and Jansen present a $3/2+\epsilon$-approximation for arbitrary $\epsilon > 0$ for both settings, however, it relies on large enumeration steps and involves up to $m^{1/\epsilon^{1/\epsilon^2}}$ calls to a subroutine to approximately solve a difficult maximization subproblem, the multiple subset sum problem (MSSP; see Section 4.2), with accuracy ϵ. We denote by $T_{MSSP}(n,\epsilon)$ the time complexity of this subroutine.

RESULTS. We present improved algorithms for scheduling with fixed jobs and scheduling with non-availability. These algorithms on the one hand achieve exactly the bound of $3/2$ and, on the other hand, are both faster and conceptually simpler than the previous algorithms in [14]. Formally stated, our results are the following with $p_{\max} = \max_{j\in\{1,\dots,m\}} p_j$:

Theorem 4.1. *Scheduling with fixed jobs admits an approximation algorithm with ratio* $3/2$ *and running time* $\mathcal{O}(n\log n + \log(np_{\max})(n + T_{MSSP}(n, 1/8)))$.

For scheduling with non-availability, the result is slightly weaker for technical reasons. In [16, 35] the following inapproximability result is displayed: Scheduling with non- availability, even if at any time, there is only at most one unavailable machine, does not admit a polynomial time algorithm with a constant approximation ratio unless P = NP.

Theorem 4.2. *Scheduling with non-availability, as long as a constant fraction* $\rho \geq 1/m$ *of machines is always available, admits a* $3/2$*-approximation with running time* $\mathcal{O}(n\log n + \log(np_{\max}/(\rho m))(n + T_{MSSP}(n, \rho/8)))$.

The remainder of this chapter is structured as follows: in Sections 4.2, we describe the algorithm and prove its correctness for the case of fixed jobs, i.e. Theorem 4.1. In Section 4.3, we show the minor changes that are needed to use our algorithm for the case of non-availability.

4.2 SCHEDULING WITH FIXED JOBS

Our approach, as well as the one presented in [14], relies heavily on algorithms for the multiple subset sum problem(MSSP). In its optimization variant, this problem is defined as follows:

Definition 4.1. *Given n items with sizes $w_1, \ldots, w_n$ and $m \leq n$ target capacities $C_1, \ldots, C_m$, possibly not all equal, we are asked to find a partition of the items into $m+1$ sets $S_1, \ldots, S_{m+1}$ so that $\sum_{j \in S_i} w_j \leq C_i$ for all $i \in \{1, \ldots, m\}$ and $\sum_{i=1}^{m} \sum_{j \in S_i} w_j$ is maximized. The set S_{m+1} collects items that remain unpacked.*

This problem in itself is strongly NP-hard, as shown by Caprara et al. [9]. The problem that is more commonly considered is the multiple knapsack problem(MKP), where items have profits that may be different from their size and the overall packed profit has to be maximized. Chekuri and Khanna were the first to present a PTAS for this problem [10]. The best currently known algorithm for both MKP and MSSP is an EPTAS due to Jansen [30] with a running time of $T_{MSSP}(n, \epsilon) = 2^{\mathcal{O}(\log(1/\epsilon) \cdot 1/\epsilon^5)} + \text{poly}(n)$ for n items and $m \leq n$ target capacities, which was subsequently improved to $T_{MSSP}(n, \epsilon) = 2^{\mathcal{O}(\log(1/\epsilon)^4 \cdot 1/\epsilon)} + \text{poly}(n)$ [31] and, if the *Modified Integer Roundup Conjecture* (MIRUP) of Scheithauer and Terno [48] holds, this even reduces to $T_{MSSP}(n, \epsilon) = 2^{\mathcal{O}(\log(1/\epsilon)^2 \cdot 1/\epsilon)} + \text{poly}(n)$ [31].

The connection of MSSP to our scheduling problem is the following: guessing a makespan T for the scheduling problem will induce, along with the prepositioned jobs, bins of different sizes into which the remaining jobs have to be placed, so solving MSSP exactly is equivalent to solving the decision version of the scheduling problem. Since MSSP is hard, we can only solve it approximately, but with arbitrary precision ϵ. The major problem now is that even though the total size of jobs not assigned by an MSSP algorithm (which will have to be scheduled after the guessed optimal makespan T) will only be a small fraction of the total length of all jobs, some of these rejected jobs may still be long, which results in a bad approximation ratio.

Much of the running time of the algorithm in [14] is spent in solving network flow problems for a very large number of candidate solutions which have to be enumerated in order to avoid this problem by placing large jobs in advance. In contrast, our new algorithm uses a single post-processing step and does not need any enumerative steps, nor network flow solvers, beyond those in the MSSP subroutine.

The outline of our algorithm is given in Algorithm 4.1. We give a relaxed decision algorithm that generates a schedule of length at most $3/2 \cdot T$ provided there exists some schedule that packs all jobs in the interval $[0, T)$. This algorithm is combined with a binary search. The number of iterations of the binary search is polynomially bounded in the input length by the following easy insight:

Remark 4.1. *For the shortest possible makespan,* OPT*, we have*

$$\max_{j \in \{1,\ldots,k\}} (s_j + p_j) \leq \text{OPT} \leq \max_{j \in \{1,\ldots,k\}} (s_j + p_j) + n \max_{j \in \{k+1,\ldots,n\}} p_j$$

Algorithm 4.1 Outline of the approximation algorithm for scheduling with fixed jobs.

Set $LB := \max_{j\in\{1,\dots,k\}}(s_j + p_j)$, $UB := LB + \max_{j\in\{k+1,\dots,n\}} p_j + \frac{1}{m}\sum_{j=k+1}^{n} p_j$
Let σ_{best} a schedule of makespan at most UB.
Sort the jobs by non-increasing length.
Generate and sort the gaps $G(T)$ as described in Section 4.2.1.
while $UB - LB \geq 1$ **do**
    Set $T := \lceil (UB + LB)/2 \rceil$.
    Update the gaps and partition into large and small jobs $J_L(T) \dot\cup J_S(T)$ as described in Section 4.2.1. Keep the gaps sorted by non-increasing sizes.
    for $j = 1, \dots, m$ **do**
        if the jth largest job is large and bigger than the jth largest gap **then**
            reject T.
    Run a 7/8-approximation for MSSP on $G(T)$ and the jobs, as described in Section 4.2.2.
    if more than $mT/8$ is unpacked **then**
        reject T.
    if T was rejected **then**
        $LB := T$
    else
        set σ_{best} to the generated packing and $UB := T$.
Modify the packing to include all items from $J_L(T)$ as described in Section 4.2.2.
Partition the remaining jobs into two groups and use a Next-Fit heuristic to schedule the remaining jobs in the interval $[UB, 3/2\,UB]$ as described in Section 4.2.3.

i.e. the range to be searched over has length $\max_{j\in\{1,\dots,k\}}(s_j + p_j) + n\max_{j\in\{k+1,\dots,n\}} p_j - \max_{j\in\{1,\dots,k\}}(s_j + p_j) \leq np_{\max}$*, for* $p_{\max} = \max_{j\in\{1,\dots,n\}} p_j$ *which is pseudopolynomial in the instance size. In particular, we have a polynomial number* $\mathcal{O}(\log(np_{\max}))$ *of binary search steps.*

Proof. The lower bound follows from the fact that the "latest" fixed job counts towards the makespan; a schedule proving the upper bound is easily obtained in linear time by scheduling all unfixed jobs on a single machine. □

We note that by using Graham's List Scheduling algorithm [21] after all the fixed jobs have finished, we can reduce the size of the search region to $\sum_{j\in\{k+1,\dots,n\}} p_j/m + P_{\max} \leq (1 + n/m)p_{\max}$, which would be preferable for a practical implementation.

Hence, we will concentrate on one iteration of the binary search in the following. In each iteration, we first apply a low-complexity check, described more closely in Section 4.2.1, that correctly rejects some infeasible guesses of T. We then pack almost all jobs into the schedule as described in Section 4.2.2. A novel postprocessing step ensures that the unpacked jobs have suitable properties to pack them later. Finally, we pack these jobs into an extra timeframe of length $T/2$ as described in Section 4.2.3.

4.2.1 Quickly Discarding Too-Small T

For a given target makespan T we generate all intervals of availability of machines, in the following called *gaps*, within the planning horizon $[0, T)$ from the k encoded fixed jobs.

Let $q(T) \in \mathbb{N}^*$ denote the number of gaps and let $G(T) := \{G_1, \dots, G_{q(T)}\}$ denote the set of gaps. For each $i \in \{1, \dots, q(T)\}$ we also use G_i to denote the size of gap G_i. Without loss of generality, we assume $G_1 \geq \dots \geq G_{q(t)}$. Note that $q(T) \leq k + m \leq 2n$ since at most k fixed jobs induce a gap "left" to them and there are at most m gaps whose "right" limit is not created by a fixed job but by the limit of the planning horizon. In total, $q(T)$ is polynomially bounded in the instance size. These gaps can easily be processed by sorting the fixed jobs on each machine by their execution times and assigning the gaps between them. This is done in time $\mathcal{O}(n \log n)$. Furthermore, we need that in each iteration of the while-loop all gaps are sorted. The gaps that are not limited by the planning horizon will not changed in the algorithm. Hence we can sort them by their sizes in line 4 before the while-loop is processed. The sizes of the other gaps are modified in each loop by the same amount. Hence we can sort them also before the while-loop is processed and update only the lengths of the gaps in line 7. Afterward we do a merge step of the two sets of gaps to have a sorting of all gaps by their sizes. In total we need $\mathcal{O}(n \log n)$ to compute and sort the two sets of gaps,

before the while-loop in line 5 is processed. In each iteration we need time $\mathcal{O}(n)$ to update the lengths of the gaps and merge them. We define

$$J_L(T) := \{j \in \{k+1, \ldots, n\} | p_j \in (T/2, T]\},$$
$$J_S(T) := \{j \in \{k+1, \ldots, n\} | p_j \in (0, T/2]\}$$

to partition the set of non-fixed jobs into *large* and *small* jobs. If any unplaced job is longer than T, we can obviously immediately reject the guessed makespan of T as too small. We can partition the jobs and sort the large jobs in each iteration in linear time, by taking the sorting of all jobs before the while-loop is processed.

The procedure in step 8 of the algorithm quickly checks a necessary condition for feasible solutions: bear in mind that a large job has length $p_j > T/2$, so there are at most m large jobs, one per machine, and no gap, being of size at most T by definition, can be large enough to accommodate two large jobs at once.

Lemma 4.1. *If a guessed makespan T is rejected in step 10 of the algorithm, then no feasible schedule of length T exists.*

Proof. Assume j minimal such that the jth largest job is large and does not fit into the jth biggest gap. For convenience, denote the lengths of the j largest jobs $p_1 \geq \ldots \geq p_j$ and the size of the gaps $G_1 \geq \ldots \geq G_j$, adding "dummy gaps" of size 0 if needed.

By definition, we have $p_1 \geq \ldots \geq p_j > G_j$, hence a feasible schedule of length T would have to assign these j jobs to at most $j-1$ gaps $G_1, \ldots, G_{j-1}$. This is a contradiction, since

$$p_1 \geq \ldots \geq p_j > T/2 \geq G_1/2 \geq \ldots \geq G_{j-1}/2,$$

so no two large jobs fit into one single gap. □

4.2.2 Packing Almost All Jobs

In this section, we will show that if $T \geq \mathrm{OPT}$, we can generate a schedule of length T that assigns "almost all" jobs. To ensure an approximation guarantee of $3/2 \cdot T$ it is critical that all jobs are scheduled within the time window $[0, T]$. We proceed in two steps. First, we show:

Lemma 4.2. *If a feasible schedule of length T exists, then step 11 generates a packing such that the total length of unpacked jobs is bounded by $mT/8$.*

To do this, we create an instance of MSSP as follows: each gap G_i corresponds to a knapsack of capacity G_i and each job of length p_i, $i = k+1 \dots n$, corresponds to an item of size p_i. We run the EPTAS of [30, 31] on this instance with accuracy $1/8$.

Observe that if (and only if) our current guessed makespan T is at least the optimal makespan OPT, it is possible to pack all items into the gaps, so the EPTAS will leave items of total area at most $(\sum_{i=k+1}^{n} p_i)/8 \leq mT/8$ unpacked. Here, we use that mT is a natural upper bound on $\sum_{i=1}^{n} p_i$ if $T \geq \text{OPT}$. Hence, if more than total length $mT/8$ is not packed, we can also immediately reject our guessed T.

At this stage, the unpacked jobs may still include up to $\lfloor (mT/8)/(T/2) \rfloor = \lfloor m/4 \rfloor$ large jobs. Obviously, if large jobs, which have length $> T/2$, are not packed in the gaps in the period $[0, T)$, we cannot hope to find an overall schedule of length at most $\frac{3}{2}T$. Hence, we modify the packing to include all large jobs, at the cost of increasing the total area of unpacked jobs by a constant factor, using the following construction:

Lemma 4.3. *Given a packing of some jobs into the gaps such that jobs of total length δ are unpacked, amongst them a large job j_1 of length $p_{j_1} > T/2$, we can either find in polynomial time a modified packing such that the total length of unpacked jobs is at most $\delta + p_{j_1}$ and the additional large job j_1 is packed as well as all previously packed large jobs or else prove that no packing of all jobs into the gaps exists at all.*

Proof. Let t_1 be the largest index such that $G_{t_1} \geq p_{j_1}$. (Recall that $T \geq G_1 \geq \dots \geq G_{q(T)}$.) Clearly, p_{j_1} can only possibly be scheduled in one of the gaps $G_1, \dots, G_{t_1}$, so if each one of these already contains a job at least as large as p_{j_1}, no packing can exist at all. This condition is already tested for in step 8 of the algorithm, before the EPTAS is called. Hence, we select one gap G_{j_1} among the gaps $G_1, \dots, G_{t_1}$ that contains a large job of minimal size. For this purpose, a gap without large job contains a 'dummy large job' of size 0. Denote this job j_2, of size $p_{j_2} < p_{j_1}$. We temporarily unpack j_2, and permanently unpack all small jobs that might have been in its gap as well, which have total length $\ell_1 \leq G_{j_1} - p_{j_2} \leq T - p_{j_2}$. (See also Figure 4.1 for this construction.)

If $p_{j_2} = 0$, we have now scheduled one more large job. Otherwise, we need to re-schedule j_2. As for j_1, let $t_2 \geq t_1$ be the largest index so that $G_{t_2} \geq p_{j_2}$. Furthermore, we already know that gaps $G_1, \dots, G_{t_1}$ all carry large jobs at least as large as j_2, since j_2 was chosen to be of minimal size amongst the large jobs there. Hence, we can restrict our attention to the gaps $G_{t_1+1}, \dots, G_{t_2}$. Again, if all these gaps already contain jobs at least as large as j_2, no feasible packing exists for this choice of T at all. Otherwise, we select a gap G_{j_2} with a large job j_3 of minimal size p_{j_3} (possibly 0) and iterate as above, discarding small jobs of total size $\ell_2 \leq G_{j_2} - p_{j_3} \leq G_{t_1+1} - p_{j_3} \leq p_{j_1} - p_{j_3}$.

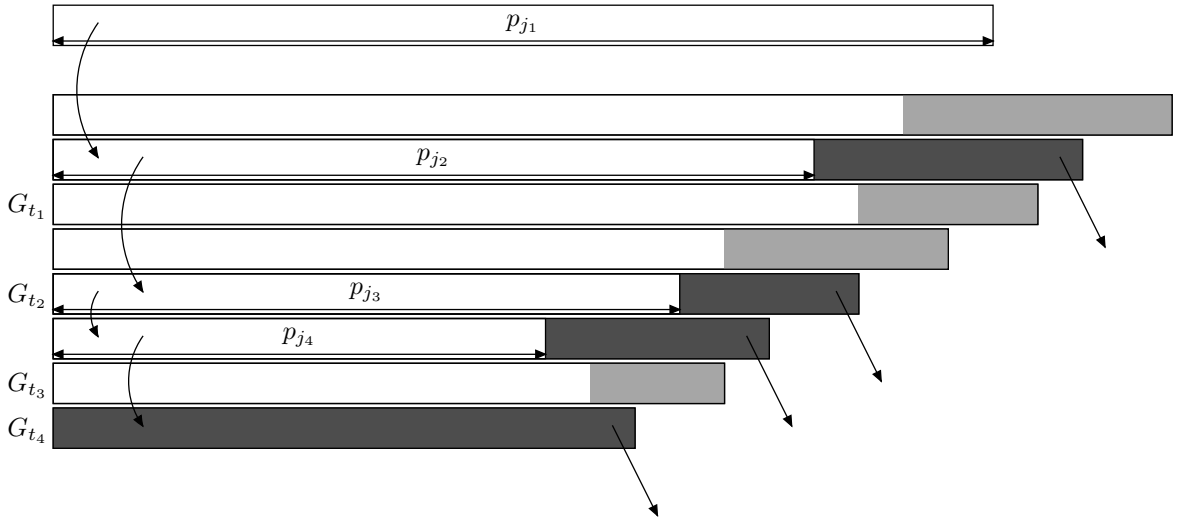


Figure 4.1: Choice of gaps $G_{j_1}, \ldots$ in the proof of Lemma 4.3. Shaded areas indicate possible small jobs; the darker areas are actually unpacked.

After some number $r \le m$ of iterations, we have $p_{j_{r+1}} = 0$, i.e. we did not need to unpack another large job, and the number of packed large jobs has increased by one. (Otherwise, step 10 would have rejected T already because there are more large jobs than large gaps.) Finally, the total size of small jobs that were unpacked can be bounded by

$$\sum_{i=1}^{r} \ell_i \le (T - p_{j_2}) + (p_{j_1} - p_{j_3}) + \cdots + (p_{j_{r-1}} - p_{j_{r+1}})$$
$$= T + \sum_{i=1}^{r-1} p_{j_i} - \sum_{i=2}^{r+1} p_{j_i} = T + p_{j_1} - p_{j_r} \le 2p_{j_1}.$$

The final inequality holds since we know that j_r is a large job, so $T - p_{j_r} < T/2 < p_{j_1}$. Since we have now additionally packed p_{j_1}, the net loss incurred is bounded by p_{j_1}. □

With a slight modification we can schedule all large unpacked jobs of total size $mT/8$ in time $\mathcal{O}(n \log n)$. Note that a naive approach would require n^2 steps to assign all large jobs.

Lemma 4.4. *Given a packing of some jobs into the gaps such that jobs of total length $mT/8$ are unpacked, we obtain a packing that includes all large jobs and has unpacked jobs with total size at most $2mT/8 = mT/4$.*

The running time of the procedure is bounded by $\mathcal{O}(n \log n)$.

Proof. We schedule the unpacked large jobs with the Algorithm 4.2. Here we assume that the large unpacked jobs are initially sorted by non-increasing lengths and the gaps by non-increasing sizes.

Algorithm 4.2 Outline of the algorithm for scheduling the unpacked large jobs.

1: Build the heap JH of all unpacked jobs, sorted by non-increasing lengths.
2: Let GH be an empty heap of the gaps.
3: Let $t = 1$.
4: **while** JH is not empty **do**
5: extract root j_h of JH, i.e. the unscheduled job of maximal processing time.
6: **while** $G_t \geq p_{j_h}$ **do**
7: add G_t to GH sorted by non-decreasing sizes of large jobs containing in the gaps.
8: $t = t + 1$.
9: extract root G_h of GH, i.e. the gap that contains a job j_{G_h} of minimal size, or one gap with a dummy large job.
10: **if** j_{G_h} has processing time larger 0 **then**
11: add j_{G_h} to JH.
12: unpack all jobs in G_h.
13: schedule j_h on G_h.

For the data structure we use two heaps, one heap stores jobs, denoted by JH, and one heap stores the gaps, denoted by GH. The heap JH is sorted by non-increasing processing lengths of the unscheduled jobs, i.e. the root of this heap is a large job of maximal processing time. The heap GH is initially empty and we add the gaps one after another. This heap is sorted by non-decreasing lengths of a large job inside the gap (note that there is at most one large job in each gap). The gap that contains a large job of minimal size, possibly a 'dummy large job' of length 0, is the root of the heap. By using heaps it allows us to extract the root (and rebuild the heap) and insert an element in logarithmic time in the number of elements inside the heap.

For the analysis we use the same sequences of job replacements as in Lemma 4.3. It does not matter for the analysis whether we schedule one sequence after another, or we schedule the jobs sorted by their lengths. Note that after we scheduled a job into a gap, the gap will never be considered again, since afterwards we schedule only jobs of smaller sizes. We use the analysis of Lemma 4.3 for all initially large unscheduled jobs of total size $mT/8$. Here consider the sequence of jobs generated by our algorithm to schedule the unpacked large jobs. Using the analysis in Lemma 4.3, each large job of length p_{j_1} is inserted by removing small jobs of total length $2p_{j_1}$. Therefore, we have removed (or unpacked) small jobs of total size at most $2mT/8 = mT/4$ after applying the algorithm.

The running time of the algorithm given in Algorithm 4.2 is as follows. The algorithm re-schedules at most n jobs, hence the while loop in step 4 has at most $z \leq n$ iterations. In each iteration i we extract one job of the heap JH, which will not re-scheduled again. To

extract and delete one job and rebuild the heap we need time at most $\mathcal{O}(\log\lfloor m/4\rfloor)$, since there are at most $\lfloor m/4\rfloor$ many jobs in the heap. We possibly add one job to the heap JH, which needs the same amount of time as extracting one.

While we are in the ith iteration of the outer loop, let v_i denote the number of iterations of the inner while loop in step 6. Since we add at most $q(T)$ gaps to the heap GH we have $\sum_{i=1}^{z} v_i \leq q(T)$. The time for extracting or adding one gap needs time at most $\mathcal{O}(\log q(T))$. In total, we have in each iteration i one extraction of a large job, v_i additions of gaps, one extraction of a gap, and possibly one addition of a job. Since $m, q(T) \leq n$ we have $\sum_{i=1}^{z} 2\mathcal{O}(\log\lfloor m/4\rfloor) + (v_i+1)\mathcal{O}(\log q(T)) \leq \sum_{i=1}^{n}(v_i+3)\mathcal{O}(\log(n)) = 3n\mathcal{O}(\log n) + \sum_{i=1}^{n} v_i\mathcal{O}(\log(n)) = \mathcal{O}(n\log n)$. □

4.2.3 Packing Remaining Jobs

After the construction of the previous section, we are left with the minimal value T such that we first have successfully packed almost all jobs, (all but total processing time $mT/8$), which we have modified by Lemma 4.3 to a packing of all but total processing time $2mT/8 = mT/4$. Since the construction is valid for makespan $T = \text{OPT}$, we know that the final $T \leq \text{OPT}$.

We will now schedule the remaining jobs in the interval $[T, \frac{3}{2}T]$ using a Next-Fit heuristic as follows: for convenience, denote these jobs $j_1 \ldots, j_{n'}$. Partition the jobs into

$$J'_S(T) := \{i \in \{1,\ldots,n'\} | p_{j_i} \in (T/4, T/2]\},$$
$$J''_S(T) := \{i \in \{1,\ldots,n'\} | p_{j_i} \in (0, T/4].\}$$

Then schedule each of the jobs in the set $J'_S(T)$ on one machine. These machines will not considered again. For every remaining machine, we greedily assign jobs in $J''_S(T)$ to it until its extra load would exceed $\frac{1}{2}T$ or we run out of jobs. Clearly, the running time of this procedure is $\mathcal{O}(n)$.

Lemma 4.5. *If the total size of jobs to be scheduled in this way is at most $mT/4$, all jobs can be assigned in the interval $[T, \frac{3}{2}T]$.*

Proof. Each machine is not closed unless its load is larger than $T/4$, either from a job in $J'_S(T)$ or because a job of size at most $T/4$ could not be assigned to it. Hence, assuming we run out of machines, the total length would be strictly larger than $mT/4$. □

In total, this proves the correctness of the algorithm. As to the running time, note that for each iteration of the binary search, the running time is essentially $\mathcal{O}(n) + T_{MSSP}(n, 1/8)$. By

Lemma 4.1, the number of iterations is bounded by $\mathcal{O}(\log(np_{\max}))$, so the overall running time is bounded by $\mathcal{O}(\log(np_{\max})(n + T_{MSSP}(n, 1/8)) + n\log n)$.

In total, we obtain:

Theorem 4.1. *Scheduling with fixed jobs admits an approximation algorithm with ratio* $3/2$ *and running time* $\mathcal{O}(n\log n + \log(np_{\max})(n + T_{MSSP}(n, 1/8)))$.

4.3 Scheduling with Non-Availability

In this section, we briefly discuss how the algorithm given above can be adapted to scheduling with *non-availability*. This setting is very closely related to scheduling with fixed jobs, the main difference is that where for fixed jobs, the makespan is given as

$$C_{\max} = \max_{j\in\{1,\dots,n\}} s_j + p_j\,,$$

it is

$$C_{\max} = \max_{j\in\{k+1,\dots,n\}} s_j + p_j$$

here, i.e. the "fixed jobs" are not proper jobs, but, for example, downtime needed for maintenance reasons.

This difference makes the problem slightly harder, as a reservation late in the schedule does not increase lower bounds on the optimal value. In [14], it is shown that this can be exploited to prevent any constant approximation ratio (unless P = NP), as long as reservations can occur on all the machines. In [16, 35] there is a slightly stronger result presented: Scheduling with non-availability, even if at any time, there is only at most one unavailable machine, does not admit a polynomial time algorithm with a constant approximation ratio unless P = NP.

If we parametrize the problem by the fraction $\rho \in (0, 1)$ of machines that is guaranteed not to have any non-availability at all, Diedrich and Jansen [14] again give a $3/2+\epsilon$ approximation (with running time doubly exponential in $1/\epsilon$ and $1/\rho$) and show that an approximation ratio of $3/2-\epsilon$ is not possible unless P = NP.

Again, we can apply the algorithm described above for fixed jobs also to the case of unavailability. The relaxed decision procedure is virtually the same: first, "almost all" of the jobs are scheduled in the interval $[0, T)$ using the multiple subset sum problem as a subroutine. The remaining jobs are scheduled in the interval $[T, (3T/2)]$, but only on the ρm machines which are not affected by reservations. To make this possible, the notion of "almost all" needs to be slightly stronger, i.e. the total size of unscheduled jobs must be

bounded by $\rho/8$ instead of $1/8$, which simply means we call the MSSP EPTAS subroutine with a higher accuracy $\rho/8$. After applying the exchange step of Lemma 4.3, unpacked non-large jobs of total area at most $\rho T/4$ remain. Then, Lemma 4.5 can be applied on the ρm machines which are guaranteed to be available after time T.

The only other consideration that needs to be made is the range over which the binary search is to be conducted: since the "fixed jobs" do not count towards the makespan, our bounds are different. The optimal makespan OPT certainly satisfies $\text{OPT} \geq p_{\max}$; on the other hand, there exists a schedule of size $1/(\rho m)\sum_{j=1}^{n} p_j + p_{\max} \leq (1 + n/(\rho m))p_{\max}$ by using Graham's List Scheduling Algorithm [21] on the permanently-available machines. Hence, we can use the lower bound $p_{\max}$ and the upper bound $(1+n/(\rho m))p_{\max}$. Again, the range is pseudopolynomial in the instance size, so the number of binary search steps needed in the outer loop of the algorithm is polynomial in the input length. Hence, we obtain:

Theorem 4.2. *Scheduling with non-availability, as long as a constant fraction $\rho \geq 1/m$ of machines is always available, admits a $3/2$-approximation with running time $\mathcal{O}(n\log n + \log(np_{\max}/(\rho m))(n + T_{MSSP}(n, \rho/8)))$.*

4.4 Conclusion

We have studied non-preemptive scheduling with fixed jobs where the objective is to minimize the makespan. For this problem, we obtain a polynomial time algorithm with ratio $3/2$, which is tight unless P = NP holds. These techniques can also be used for the closely related setting of scheduling with non-availability; there, one needs to additionally assume that a constant percentage of the machines is permanently available.

In total, our approach yields a tight approximation result. However, our algorithm uses a very general MKP EPTAS for a fixed value of $\epsilon = 1/8$. It is an interesting open question if the more restricted problem MSSP admits a faster EPTAS or a faster combinatorial $7/8$-approximation that can be used to speed up our algorithm. So far, the best known non-PTAS result is a $3/4$-approximation due to Caprara et al. [9] for the case of identical bin capacities.

5 Concluding Remarks

In this thesis, we presented approximation algorithms for the strip packing problem in Chapter 2, the two-dimensional bin packing problem in Chapter 3 and for scheduling with fixed jobs in Chapter 4.

For the strip packing problem we displayed an absolute $5/3+\varepsilon$-approximation for any arbitrary $\varepsilon > 0$. This result is an important step to settle the approximability of this problem. However, it is still open, whether there exists an approximation algorithm that matches the current lower bound with an absolute approximation ratio of $3/2$.

For the two-dimensional bin packing problem, we proposed for any $\varepsilon > 0$ an asymptotic $3/2+\varepsilon$-approximation with an additive constant of 69 and an improved additive constant of 39 for the version that allows rotation. There is no asymptotic approximation algorithm with an approximation ratio of less than $1+1/3792$ for the version with rotations and $1+1/2196$ for the version without rotations, unless P = NP [11]. An open question is to close the gap between this lower bound and the current best asymptotic approximation ratio of our algorithm. Therefore, it is of interest to find an approximation algorithm with an asymptotic approximation ratio of $4/3$, if there exists any. Maybe there is a way to adapt our techniques by modifying an optimal packing so that the rectangles of each sequence of 3 bins are rounded by using only 1 additional bin.

We settled the approximability for scheduling with fixed jobs by developing an approximation algorithm whose absolute approximation ratio matches the lower bound of $3/2$. For an improved running time it would be of interest to find a fast $7/8$-approximation for the multiple subset sum problem.

List of Figures

BIBLIOGRAPHY

[1] B. S. Baker, D. J. Brown, and H. P. Katseff. A 5/4 algorithm for two-dimensional packing. *Journal of Algorithms*, 2(4):348–368, 1981.

[2] B. S. Baker, E. G. C. Jr., and R. L. Rivest. Orthogonal packings in two dimensions. *SIAM Journal on Computing*, 9(4):846–855, 1980.

[3] N. Bansal, A. Caprara, K. Jansen, L. Prädel, and M. Sviridenko. A structural lemma in 2-dimensional packing, and its implications on approximability. In *Proceedings of the 20th International Symposium on Algorithms and Computation (ISAAC 2009), LNCS 5878*, pages 77–86, 2009.

[4] N. Bansal, A. Caprara, and M. Sviridenko. A new approximation method for set covering problems, with applications to multidimensional bin packing. *SIAM Journal on Computing*, 39(4):1256–1278, 2009.

[5] N. Bansal, J. R. Correa, C. Kenyon, and M. Sviridenko. Bin packing in multiple dimensions: Inapproximability results and approximation schemes. *Mathematics of Operations Research.*, 31(1):31–49, 2006.

[6] N. Bansal and M. Sviridenko. New approximability and inapproximability results for 2-dimensional bin packing. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2004)*, pages 196–203, 2004.

[7] M. Bougeret, P.-F. Dutot, K. Jansen, C. Robenek, and D. Trystram. Approximation algorithms for multiple strip packing and scheduling parallel jobs in platforms. *Discrete Mathematics, Algorithms and Applications*, 3(4):553–586, 2011.

[8] A. Caprara. Packing d-dimensional bins in d stages. *Mathematics of Operations Research*, 33:203–215, 2008.

[9] A. Caprara, H. Kellerer, and U. Pferschy. A 3/4-approximation algorithm for multiple subset sum. *Journal of Heuristics*, 9(2):99–111, 2003.

[10] C. Chekuri and S. Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 35(3):713–728, 2005.

[11] M. Chlebík and J. Chlebíková. Inapproximability results for orthogonal rectangle packing problems with rotations. In *Proceedings of the 6th Conference on Algorithms and Complexity (CIAC 2006), LNCS 3998*, pages 199–210, 2006.

[12] F. R. K. Chung, M. R. Garey, and D. S. Johnson. On packing two-dimensional bins. *SIAM Journal of Algebraic Discrete Methods*, 3:66–76, 1982.

[13] E. G. Coffman Jr., M. R. Garey, D. S. Johnson, and R. E. Tarjan. Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM Journal on Computing*, 9(4):808–826, 1980.

[14] F. Diedrich and K. Jansen. Improved approximation algorithms for scheduling with fixed jobs. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2009)*, pages 675–684, 2009.

[15] F. Diedrich, K. Jansen, F. Pascual, and D. Trystram. Approximation algorithms for scheduling with reservations. *Algorithmica*, 58(2):391–404, 2010.

[16] F. Diedrich, K. Jansen, L. Prädel, U. M. Schwarz, and O. Svensson. Tight approximation algorithms for scheduling with fixed jobs and non-availability. *Transactions on Algorithms*, 8(3):27–27:15, 2012.

[17] E. Dinic. An algorithm for solution of a problem of maximum flow in a network with power estimation. *Soviet Mathematics Doklady*, 11(5):1277–1280, 1970.

[18] F. Eisenbrand and G. Shmonin. Carathéodory bounds for integer cones. *Operations Research Letters*, 34(5):564–568, 2006.

[19] W. Fernandez de la Vega and G. S. Lueker. Bin packing can be solved within $1+\epsilon$ in linear time. *Combinatorica*, 1(4):349–355, 1981.

[20] I. Golan. Performance bounds for orthogonal oriented two-dimensional packing algorithms. *SIAM Journal on Computing*, 10(3):571–582, 1981.

[21] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.

[22] M. D. Grigoriadis, L. G. Khachiyan, L. Porkolab, and J. Villavicencio. Approximate max-min resource sharing for structured concave optimization. *SIAM Journal on Optimization*, 11(4):1081–1091, 2001.

[23] R. Harren. *Two-Dimensional Packing Problems*. PhD thesis, Saarland University, 2010.

[24] R. Harren, K. Jansen, L. Prädel, and R. van Stee. A $(5/3+\epsilon)$-approximation for strip packing. In *Proceedings of the 12th International Symposium on Algorithms and Data Structures (WADS 2011), LNCS 6844*, pages 475–499, 2010.

[25] R. Harren and R. van Stee. Improved absolute approximation ratios for two-dimensional packing problems. In *Proceedings of the 12th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2009), LNCS 5687*, pages 177–189, 2009.

[26] R. Harren and R. van Stee. Absolute approximation ratios for packing rectangles into bins. *Journal of Scheduling*, 15(1):63–75, 2012.

[27] D. S. Hochbaum and D. B. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. *SIAM Journal on Computing*, 17(3):539–551, 1988.

[28] H.-C. Hwang, K. Lee, and S. Y. Chang. The effect of machine availability on the worst-case performance of LPT. *Discrete Applied Mathematics*, 148(1):49–61, 2005.

[29] K. Jansen. *Efficient Approximation and Online Algorithms*, chapter Approximation algorithms for min-max and max-min resource sharing problems and applications, LNCS 3484, pages 156–202. Springer, 2006.

[30] K. Jansen. Parameterized approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 39(4):1392–1412, 2009.

[31] K. Jansen. A fast approximation scheme for the multiple knapsack problem. In *Proceedings of the 38th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2012), LNCS 7147*, pages 313–324, 2012.

[32] K. Jansen and M. Margraf. *Approximative Algorithmen und Nichtapproximierbarkeit*. de Gruyter, 2008.

[33] K. Jansen and L. Prädel. New approximability results for two-dimensional bin packing. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2013)*, pages 919–936, 2013.

[34] K. Jansen, L. Prädel, and U. M. Schwarz. Two for one: Tight approximation of 2d bin packing. In *Proceedings of the 11th International Symposium on Algorithms and Data Structures (WADS 2009), LNCS 5664*, pages 399–410, 2009.

[35] K. Jansen, L. Prädel, U. M. Schwarz, and O. Svensson. Faster approximation algorithms for scheduling with fixed jobs. In *Proceedings of the 17th Computing: The Australasian Theory Symposium (CATS 2011), CRPIT 119*, pages 3–10, 2011.

[36] K. Jansen and R. Solis-Oba. Rectangle packing with one-dimensional resource augmentation. *Discrete Optimization*, 6(3):310–323, 2009.

[37] K. Jansen and R. Thöle. Approximation algorithms for scheduling parallel jobs. *SIAM Journal on Computing*, 39(8):3571–3615, 2010.

[38] K. Jansen and R. van Stee. On strip packing with rotations. In *Proceedings of the 37th annual ACM symposium on Theory of computing (STOC 2005)*, pages 755–761, 2005.

[39] R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.

[40] C. Kenyon and E. Rémila. A near-optimal solution to a two-dimensional cutting stock problem. *Mathematics of Operations Research*, 25(4):645–656, 2000.

[41] E. L. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4:339–356, 1979.

[42] C.-Y. Lee. Machine scheduling with an availability constraint. *Journal of Global Optimization, Special Issue on Optimization of Scheduling Applications*, 9:363–384, 1996.

[43] J. Y.-T. Leung. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. Chapman and Hall/CRC, 2004.

[44] C.-J. Liao, D.-L. Shyur, and C.-H. Lin. Makespan minimization for two parallel machines with an availability constraint. *European Journal of Operational Research*, 160:445–456, 2003.

[45] N. Megow, R. H. Möhring, and J. Schulz. Decision support and optimization in shutdown and turnaround scheduling. *INFORMS Journal on Computing*, 23(2):189–204, 2011.

[46] M. Scharbrodt. *Produktionsplanung in der Prozessindustrie: Modelle, effiziente Algorithmen und Umsetzung*. PhD thesis, Technical University of Munich, 2000.

[47] M. Scharbrodt, A. Steger, and H. Weisser. Approximability of scheduling with fixed jobs. *Journal of Scheduling*, 2(6):267–284, 1999.

[48] G. Scheithauer and J. Terno. The modified integer round-up property of the one-dimensional cutting stock problem. *European Journal of Operational Research*, 84(3):562 – 571, 1995.

[49] I. Schiermeyer. Reverse-fit: A 2-optimal algorithm for packing rectangles. In *Proceedings of the 2nd European Symposium on Algorithms (ESA 1994), LNCS 855*, pages 290–299, 1994.

[50] D. D. Sleator. A 2.5 times optimal algorithm for packing in two dimensions. *Information Processing Letters*, 10(1):37–40, 1980.

[51] A. Steinberg. A strip-packing algorithm with absolute performance bound 2. *SIAM Journal on Computing*, 26(2):401–409, 1997.

[52] P. M. Vaidya. An algorithm for linear programming which requires $\mathcal{O}(((m+n)n^2 + (m+n)^{1.5}n)L)$ arithmetic operations. *Mathematical Programming*, 47:175–201, 1990.

[53] V. V. Vazirani. *Approximation algorithms*. Springer-Verlag New York, Inc., 2001.

[54] G. Zhang. A 3-approximation algorithm for two-dimensional bin packing. *Operations Research Letters*, 33(2):121–126, 2005.