Improved Approximation Algorithms for Three-Dimensional Bin Packing

Debajyoti Kar ✉ ⬤
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Arindam Khan ✉ ⬤
Department of Computer Science and Automation, Indian Institute of Science, Bengaluru, India

Malin Rau ✉ ⬤
Department of Mathematical Sciences, Chalmers University of Technology and University of Gothenburg, SE-412 96 Göteborg Sweden

───── **Abstract** ─────

We study three fundamental three-dimensional (3D) geometric packing problems: 3D (Geometric) Bin Packing (3D-BP), 3D Strip Packing (3D-SP), and Minimum Volume Bounding Box (3D-MVBB), where given a set of 3D (rectangular) cuboids, the goal is to find an axis-aligned nonoverlapping packing of all cuboids. In 3D-BP, we need to pack the given cuboids into the minimum number of unit cube bins. In 3D-SP, we need to pack them into a 3D cuboid with a unit square base and minimum height. Finally, in 3D-MVBB, the goal is to pack into a cuboid box of minimum volume.

It is NP-hard to even decide whether a set of rectangles can be packed into a unit square bin – giving an (absolute) approximation hardness of 2 for 3D-BP and 3D-SP. The previous best (absolute) approximation for all three problems is by Li and Cheng (SICOMP, 1990), who gave algorithms with approximation ratios of 13, 46/7, and 46/7 + ε, respectively, for 3D-BP, 3D-SP, and 3D-MVBB. We provide improved approximation ratios of 6, 6, and 3 + ε, respectively, for the three problems, for any constant $\varepsilon > 0$.

For 3D-BP, in the asymptotic regime, Bansal, Correa, Kenyon, and Sviridenko (Math. Oper. Res., 2006) showed that there is no asymptotic polynomial-time approximation scheme (APTAS) even when all items have the same height. Caprara (Math. Oper. Res., 2008) gave an asymptotic approximation ratio of $T_\infty^2 + \varepsilon \approx 2.86$, where T_∞ is the well-known Harmonic constant in Bin Packing. We provide an algorithm with an improved asymptotic approximation ratio of $3T_\infty/2 + \varepsilon \approx 2.54$. Further, we show that unlike 3D-BP (and 3D-SP), 3D-MVBB admits an APTAS.

1 Introduction

Three-dimensional (3D) packing problems are used to model several practical settings in production and transportation planning – ranging from cargo management, manufacturing, 3D printing and prototyping, to cutting and loading applications. In the 1960s, Gilmore and Gomory [28] introduced 3D packing in the context of the cutting stock problem in operations research, where given a stock material (3D cuboid), the goal is to cut out a set of required items (smaller 3D cuboids) by a sequence of end-to-end cuts. Around the same time, Meir and Moser [51] asked a combinatorial question: given a set of cubes, when can we pack them in a given cuboid? Since then, due to its inherent mathematical aesthetics, computational

nature, and practical relevance, the study of 3D packing has led to the development of several techniques in mathematics, computer science, and operations research.

In this paper, we consider three classical 3D packing problems. In all of these problems, the input is a collection of (rectangular) cuboids (items), each specified by their height, width, and depth. In the 3D Bin Packing (3D-BP) problem, the goal is to output a packing of all the items using the minimum number of bins, where each bin is a unit cube. In the 3D Strip Packing (3D-SP) problem, we are given a three-dimensional strip having a 1×1 square base and unbounded height, and we have to pack all items minimizing the height of the strip. Finally, in the Minimum Volume Bounding Box (3D-MVBB) problem, we seek to obtain a cuboidal box of minimum volume that can accommodate all input items. In all these problems, the items cannot be rotated about any axis, and they must be packed non-overlappingly. Further, we assume that all items and bins/boxes are axis-aligned.

With the recent exponential growth in transportation and shipping, specially with the advent of e-commerce and UAVs, these problems are receiving increasingly more attention. For instance, in container ship loading, it is crucial to optimize the placement of cargo to maximize space usage while minimizing the number of containers needed. In pallet loading, manufacturers strive to stack goods on pallets in a way that maximizes storage capacity and ensures secure transport. Further, in supply chain management, it is crucial to optimize the arrangement of goods in storage to fit within the smallest possible space, reducing storage costs and enhancing inventory accessibility. The survey by Ali, Ramos, Carravilla, and Oliveira [3] provides a comprehensive overview of 3D packing, with more than two hundred research articles. We refer readers to [23, 33, 46, 17, 54, 16, 39, 45, 49, 48] and [27, 11, 4, 21, 58, 57] for important empirical procedures and heuristics to 3D-BP and 3D-SP, respectively. There are also many practical programming competitions for these problems, e.g., OPTIL 3D Bin Packing Challenge [2] and ICRA VMAC Palletization Competition [1].

In contrast to the above, the theoretical exploration of 3D packing has been significantly limited due to its inherently complicated nature. All three considered problems are NP-hard. In fact, 3D-BP and 3D-SP generalize several classical strongly NP-hard problems in scheduling and packing, including (1D) bin packing, multiprocessor scheduling [44], packing squares into squares [24], and packing cubes into cubes [47]. In this paper, we study the absolute and asymptotic approximation algorithms for these problems. Given an algorithm $\mathcal{A}$ for a minimization problem Π, the *absolute approximation ratio* of $\mathcal{A}$ is defined as $\max_{I \in \mathcal{I}}\{\mathcal{A}(I)/\text{OPT}(I)\}$, where $\mathcal{I}$ is the set of all input instances for Π, and $\mathcal{A}(I), \text{OPT}(I)$ are the values of the solution provided by $\mathcal{A}$ and the optimal solution for an input instance I, respectively. The *asymptotic approximation ratio* (AAR) is defined as: $\limsup\limits_{m \to \infty} \max_{I \in \mathcal{I}} \left\{ \frac{\mathcal{A}(I)}{\text{OPT}(I)} \mid \text{OPT}(I) = m \right\}$. A problem is said to admit an asymptotic polynomial-time approximation scheme (APTAS) if, for any $\varepsilon > 0$, there exists a polynomial-time algorithm $\mathcal{A}_\varepsilon$ with AAR of $(1 + \varepsilon)$. 3D-BP and 3D-SP generalize 2D Bin Packing. Thus they do not admit an APTAS, as 2D Bin Packing has an asymptotic approximation hardness of $1 + 1/2196$ [13]. Furthermore, even for squares, it is NP-hard to decide if a set of squares can be packed in a single square bin or not [24] – thus giving an absolute approximation hardness of 2 for 3D-BP and 3D-SP.

Two-dimensional variants of these problems have been extensively studied. For 2D-BP, Harren, Jansen, Prädel, Schwarz, and van Stee [31] gave a *tight* absolute 2-approximation, and a line of work [12, 7, 36] culminated in an asymptotic 1.406-approximation due to Bansal and Khan [10]. For 2D-SP, the asymptotic approximation regime is settled by the AFPTAS due to Kenyon and Rémila [40]. However, the best-known absolute approximation ratio for 2D-SP stands at $(5/3 + \varepsilon)$ [30] and there is a 3/2-hardness. In pseudopolynomial-time,

there is an almost tight (absolute) $(5/4 + \varepsilon)$-approximation algorithm [37, 32]. Finally, for 2D-MVBB, Bansal, Correa, Kenyon, and Sviridenko [8] gave a PTAS.

For 3D-BP, Csirik and van Vliet [18] gave an asymptotic approximation ratio of $T_\infty^3 + \varepsilon \approx 4.836$, where $T_\infty \approx 1.691$ is the omnipresent Harmonic constant [41] in Bin Packing, and the same ratio was achieved by Epstein and van Stee [22] by an online algorithm using bounded space. This was later improved to $T_\infty^2 + \varepsilon \approx 2.86$ by Caprara [12], which stands as the currently best-known asymptotic approximation ratio for 3D-BP.

For 3D-SP, Li and Cheng [44] demonstrated that simple heuristics such as NFDH or FFDH for 2D packing [15] have unbounded AARs. Then they provided an algorithm that returns a packing into a strip of height at most $(13/4)\text{OPT}_{\text{3D-SP}} + 8h_{\max}$, where $\text{OPT}_{\text{3D-SP}}$ denotes the optimal Strip Packing height, and $h_{\max}$ is the maximum height of an input item. Afterwards, there has been a long line of work [44, 43, 53, 52, 38, 9] on the asymptotic approximability of 3D-SP, culminating in a $(3/2 + \varepsilon)$-approximation by Jansen and Prädel [35]. However, all these improved asymptotic approximation algorithms incur huge additive loss (more than 100).

The authors in [20] obtained an absolute approximation ratio of $(29/4 + \varepsilon)$ for 3D-SP, and claimed it to be the best-known ratio for the problem. However, Li and Cheng [44] had also designed an algorithm that returns a packing into a strip of height at most $(32/7)\text{OPT}_{\text{3D-SP}} + 2h_{\max}$. This already gives a better absolute approximation ratio of $46/7$ for 3D-SP. Since $\text{OPT}_{\text{3D-SP}}$ is a lower bound on the minimum number of unit (cube) bins needed to pack all items, an absolute α-approximation for 3D-SP directly implies an absolute $(2\lfloor \alpha \rfloor + 1)$-approximation for 3D-BP – one can obtain bins by cutting a 3D-SP solution at integral heights, followed by packing the items intersected by the cutting planes into additional separate bins. Thus, the $46/7$-approximation for 3D-SP implies an absolute 13-approximation for 3D-BP, which we believe to be the best-known approximation ratio for 3D-BP.

Finally, it is easy to obtain an absolute $\alpha(1 + \varepsilon)$-approximation for 3D-MVBB from an absolute α-approximation for 3D-SP. Applying this strategy to the claimed $(29/4 + \varepsilon)$-approximation algorithm for 3D-SP [20], Alt and Scharf [6] obtained a $(29/4 + \varepsilon)$-absolute approximation for 3D-MVBB. However, as mentioned before, the result of Li and Cheng [44] can also be extended to a $(46/7 + \varepsilon)$-approximation for 3D-MVBB.

There have been some improvements for special cases. For example, Bansal, Correa, Kenyon, and Sviridenko [8] provided an APTAS for d-dimensional bin packing with d-dimensional cubes. Harren [29] gave APTAS for d-dimensional strip packing with d-dimensional cubes when the base of the strip has a bounded aspect ratio. Jansen, Khan, Lira, and Sreenivas [34] extended the APTAS to more general bases (not necessarily rectangular).

However, for general cuboids, there has been no progress on the absolute approximation ratios for any of the three problems since 1990 [44] and for asymptotic approximation ratio of 3D-BP since 2008 [12]. In [8], the authors mention the inherent difficulty in extending results from 2D packing to 3D packing, due to the more complicated nature of interactions between different types of items in three dimension. In fact, improved approximability of d-dimensional geometric Bin Packing and Strip Packing, for $d > 2$, was listed as *one of the ten major open problems* in the survey on multidimensional packing [14].

1.1 Our contribution

We present improved absolute approximation algorithms for 3D-BP, 3D-SP, and 3D-MVBB. Further, we obtain improved asymptotic approximation algorithms for 3D-BP and 3D-MVBB.

First, we discuss our results on the absolute approximation algorithms. We show how a packing in k bins can be transformed into $6k$ *structured* bins; following which, for constant k,

it is possible to find such a structured packing efficiently using a variant of the Generalized Assignment Problem – giving us an absolute approximation ratio of 6. One interesting idea is that we use an asymptotic approximation algorithm to obtain improved absolute approximation. One of our key ingredients is the asymptotic approximation algorithm for 3D-SP by Jansen and Prädel [38], which provides a packing into height at most $(3/2 + \varepsilon)\text{OPT}_{3\text{D-SP}} + \varepsilon + O_\varepsilon(1)h_{\max}$. [1] Thus, for a sufficiently small appropriate constant μ, if $h_{\max} \leq \mu$, then we can actually pack all items into $\lfloor 3k/2 \rfloor + 1$ bins, assuming there exists a packing into k bins. So, we partition the items into four classes: L (large items: all dimensions are greater than μ), I_w, I_d, I_h (width, depth, height less than μ, resp.). If an item belongs to multiple classes, we assign them to anyone arbitrarily. Now *large* items can be packed in k bins by brute-force enumeration in polynomial-time (for constant k, μ). Each of the remaining three classes can be packed into $\lfloor 3k/2 \rfloor + 1$ bins. In total, we get $3(\lfloor 3k/2 \rfloor + 1) + k \leq 7k$ bins.

To improve further, we pack *large* items together with some items from one of the three classes. First, we observe that one of these classes has a volume less than $k/3$. W.l.o.g. let us assume it to be I_h. Now, first, we use a volume-based argument and use an algorithm from [44] to show that we can pack all items in I_h whose width or depth is less than $1/2$. The remaining items in I_h have both width and depth exceeding $1/2$. Next, we show that we can guess the packing of *large* items and almost all items in I_h, except a set of items with small volume. However, with a refined and technical analysis, we finally show that even these remaining items can be packed in the free regions of the six bins. For 3D-BP, this yields an improvement over the previous bound of 13 [44].

▶ **Theorem 1.** *There exists a polynomial-time 6-approximation algorithm for 3D-BP.*

This directly implies an absolute $(6 + \varepsilon)$-approximation for 3D-SP – guess the optimal Strip Packing height within a $(1 + \varepsilon)$-factor, then use appropriate scaling to apply the above 3D-BP result, and finally stack the obtained six bins. With a more careful analysis, we can show there is some extra empty space in the strip, and the resulting height is strictly below 6.

▶ **Theorem 2.** *There exists a small absolute constant $\rho > 0$, such that for any $\varepsilon > 0$, there is a polynomial-time $(6 - \rho + \varepsilon)$-approximation algorithm for 3D-SP.*

Another implication of our result is a $(6 + \varepsilon)$-approximation for the 3D-MVBB problem, using the connection between 3D-SP and 3D-MVBB [6]. However, we then use the power of resource augmentation in 2D-BP to obtain an APTAS for 3D-SP when we are allowed to use resource augmentation. With additional technical adaptations, we obtain a $(3 + \varepsilon)$-approximation for 3D-MVBB.

Furthermore, surprisingly, unlike 3D-BP and 3D-SP, we show that 3D-MVBB admits an APTAS – settling the asymptotic approximability for the problem.

▶ **Theorem 3.** *For any $\varepsilon > 0$, there exists a polynomial-time $(3 + \varepsilon)$-approximation algorithm and an asymptotic polynomial-time approximation scheme for 3D-MVBB.*

Finally, we turn our attention to the asymptotic approximability of 3D-BP. Towards this, we exploit connections between 3D-SP and 3D-BP. Let $\text{OPT}_{3\text{D-SP}}(I), \text{OPT}_{3\text{D-BP}}(I)$ be the values of the optimal solution for Strip Packing and Bin Packing, for input I, respectively.

---

[1] The notation $O_\varepsilon(f(n))$ means that the implicit constant hidden in big-O notation can depend on ε.

Then $\mathrm{OPT}_{\text{3D-SP}}(I) \leq \mathrm{OPT}_{\text{3D-BP}}(I)$, as the bins can be stacked on top of each other to provide a feasible solution for Strip Packing. Thus, one can trivially obtain a $(3 + \varepsilon)$-approximation algorithm as follows. First, we obtain a packing in height $(\frac{3}{2} + \varepsilon)\mathrm{OPT}_{\text{3D-SP}}(I) + O_\varepsilon(1)$ using [35]. We then can cut the strip into unit cube bins by cutting it at integral heights. All items that are completely contained within heights $[i, i+1)$ are packed in the $(2i+1)$-th bin. Remaining items that are cut by the x-y axis-aligned plane at height i (these items form one layer of items where each item has height at most $h_{\max} \leq 1$) are packed in $(2i)$-th bin. This would give us a packing into $(3 + \varepsilon)\mathrm{OPT}_{\text{3D-SP}}(I) + O_\varepsilon(1)$ bins.

To improve beyond T_∞^2, our approach will be to find a packing such that the items that are cut do not have large heights. Towards this, we use *harmonic rounding* [41], where the function f_k rounds up $\alpha \in (1/k, 1]$ to nearest larger number of the form $1/q$ where $q \in \mathbb{Z}$. Thus, for $\alpha_i \in (1/(q+1), 1/q]$, $f_k(\alpha_i) := 1/q$, for $q \in [k-1]$. Otherwise, $f_k(\alpha_i) := \alpha_i$. It is well-known [9] that, for any sequence $\alpha_1, \alpha_2, \ldots, \alpha_n$, with $\alpha_i \in (0, 1]$ and $\sum_{i=1}^n \alpha_i \leq 1$, for a small enough ε, we have $\sum_{i=1}^n f_{1/\varepsilon}(\alpha_i) \leq T_\infty + \varepsilon \approx 1.691$.

We first round the item heights in I using $f_{1/\varepsilon}$ to obtain a new set of items I^∞ and obtain a 3D Strip Packing of them using the algorithm by [38]. Let $\mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty)$ be the optimal number of $1 \times 1 \times T_\infty$-sized bins needed to pack all items in I^∞. Then, it is easy to see that $\mathrm{OPT}_{\text{3D-SP}}(I^\infty) \leq T_\infty \mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty)$. [2] Then we have $\frac{3}{2}\mathrm{OPT}_{\text{3D-SP}}(I^\infty) \leq \frac{3T_\infty}{2}\mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty) \leq \frac{3T_\infty}{2}\mathrm{OPT}_{\text{3D-BP}}(I)$. The last inequality follows from harmonic rounding.

Now we need to ensure that the *tall* items in I^∞ packed in the strip with height $\frac{3}{2}\mathrm{OPT}_{\text{3D-SP}}(I^\infty)$ are not cut by the cutting planes at integral heights – we call this *tall-not-sliced* property. A similar idea was used by Bansal, Han, Iwama, Sviridenko, and Zhang [9] to obtain an alternate $(T_\infty + \varepsilon)$-approximation for 2D-BP. However, 3D packing is much more involved. For this, we exploit the structural properties from the packing by [35]. First, we show that the strip can be divided into $O_\varepsilon(1)$ cuboids such that, for each cuboid, the corresponding items packed inside are *similar*. Next, we show that we can pack almost all *tall* items in I^∞ of the same height ($1/q$ for some $q \in [k-1]$) at heights that are multiples of $1/q$ and incur only a small additive loss. This will ensure that none of these items are cut by planes at integral heights. For items with big width and depth, we use a linear program to assign items to the containers. For other items (except a small volume of them), the packing is based on variants of the Next-Fit-Decreasing-Height (NFDH) algorithm [15]. Finally, we show that we can pack the remaining items in the remaining free regions and an additional $O_\varepsilon(1)$ bins. This provides an improved guarantee for 3D-BP after nearly two decades.

▶ **Theorem 4.** *For any $\varepsilon > 0$, there exists a polynomial-time algorithm for 3D-BP with an asymptotic approximation ratio $(3T_\infty/2 + \varepsilon) \approx 2.54$.*

Organization of the paper. In Section 2, we present some preliminaries needed for our results. Section 3 provides absolute approximation algorithms for 3D-BP and 3D-SP, and we prove Theorems 1 and 2. Section 4 deals with the asymptotic approximation algorithm for 3D-BP and establishes Theorem 4. In Section 5, we discuss results related to 3D-MVBB and prove Theorem 3. Finally, Section 7 ends with a conclusion. An overview of all results can be found in Table 1.

---

[2] For simplicity, we are ignoring the $O_\varepsilon(1)$ in the following discussion in this section.

Problem	Absolute Approximation Ratio		Asymptotic Approximation Ratio	
	Previous Best	**Our Result**	**Previous Best**	**Our Result**
3D-BP	13 [44]	6 (WR: 5)	$T_\infty^2 + \varepsilon < 2.86$ [12]	$3T_\infty/2 + \varepsilon < 2.54$
3D-SP	$46/7 \approx 6.58$ [44]	6	$3/2 + \varepsilon$ [35]	–
3D-MVBB	$46/7 + \varepsilon$ [44, 6]	$3 + \varepsilon$	$46/7 + \varepsilon$ [44, 6]	$1 + \varepsilon$

▦ **Table 1** Summary of results. WR denotes the case when 90° rotation around any axis is allowed.

1.2 Related work

For $d > 3$, Caprara [12] gave an algorithm with AAR of T_∞^{d-1} for both d-dimensional Bin Packing and Strip Packing. Sharma [55] gave T_∞^{d-1}-asymptotic approximation for these two problems when the items can be orthogonally rotated about all or a subset of axes. For 3D-MVBB, if the items are allowed to be rotated by 90 degrees about any axis, Alt and Scharf [6] gave a 17.738-approximation. Another related problem is the 3D Knapsack problem, in which each item additionally has an associated profit, and the goal is to obtain a maximum profit packing inside a unit cube knapsack. The authors in [20] have given a $(7 + \varepsilon)$-approximation algorithm. For other related problems, we refer the readers to the surveys on approximation algorithms for multidimensional packing [5, 14].

2 Preliminaries

We define width, depth, and height along x, y, z axes, respectively. Let I be the given set of n items, where each item $i \in I$ is an axis-aligned cuboid having height, width, and depth equal to h_i, w_i, d_i, respectively. Let $h_{\max}, w_{\max}, d_{\max} \in (0, 1]$ be the maximum height, width, and depth of an item in I, respectively. Given a box $B := [0, W] \times [0, D] \times [0, H]$, if the (bottom-left-back corner of) item i is placed (by translation) at (x_i, y_i, z_i) then it occupies the region: $[x_i, x_i + w_i] \times [y_i, y_i + d_i] \times [z_i, z_i + h_i]$, and the packing is feasible if $x_i \in [0, W - w_i], y_i \in [0, D - d_i], z_i \in [0, H - h_i]$. In this placement, we define the top, right, and back faces of item i to be $[x_i, x_i + w_i] \times [y_i, y_i + d_i] \times \{z_i + h_i\}$, $\{x_i + w_i\} \times [y_i, y_i + d_i] \times [z_i, z_i + h_i]$, and $[x_i, x_i + w_i] \times \{y_i + d_i\} \times [z_i, z_i + h_i]$, respectively. Analogously, bottom, left, and front faces are defined. Two items do not overlap if their interiors are disjoint. The volume of item i is $v(i) := h_i w_i d_i$. For any set T, let $v(T)$ denote the total volume of items in T. We define $\text{OPT}_\Pi(I)$ to be the value of the optimal solution for problem Π on instance I.

2.1 Algorithms for 3D Packing

We now state three results on 3D packing that will be crucial for our results. The first two results give a volume-based guarantee.

▶ **Theorem 5** ([44]). *Let T be a set of 3D items where each item has height bounded by $h_{\max}$.*
  (i) *All items in T can be packed into a strip with 1×1 base and height $4v(T) + 8h_{\max}$.*
  (ii) *If further, each item has either width or depth (or both) not exceeding $1/2$, then all items in T can be packed inside a strip with 1×1 base and height $3v(T) + 8h_{\max}$.*

Proof. We first prove (ii). Let $T_w := \{i \in T \mid w_i \leq 1/2\}$ and $T_d := T \setminus T_w$, so that $d_i \leq 1/2$, for all $i \in T_d$. We further classify the items of T_w based on their base area as follows: let $T_{w\ell} \subseteq T_w$ be the items whose base area exceeds $1/6$ and let $T_{ws} := T_w \setminus T_{w\ell}$. For packing

items in $T_{w\ell}$, we first sort them in non-increasing order of heights and pack them in layers, with two items in each layer placed one beside the other (except possibly the last one). Note that this is possible since $w_i \leq 1/2$ for $i \in T_w$. Let k denote the number of layers and let $h'_1, \ldots, h'_k$ be the heights of the layers so that h'_i's are non-increasing. Since each layer (except possibly the last) has a base area more than $1/3$, we have $v(T_{w\ell}) > \frac{1}{3} \sum_{i=2}^{k} h'_i$, and therefore the height of the packing of $T_{w\ell}$ is at most $3v(T_{w\ell}) + h_{\max}$.

Next, we pack items of T_{ws}. To this end, we again sort the items in non-increasing order of heights and group them into maximal groups of base area at most $1/2$. Since the base area of each item is at most $1/6$, the total base area of each group (except possibly the last) is at least $1/2 - 1/6 = 1/3$. We pack each group in a layer using Steinberg's algorithm [56]. Similar to the packing of $T_{w\ell}$, the total height of the layers of T_{ws} is bounded by $3v(T_{w\ell}) + h_{\max}$.

Analogously, we obtain a packing of items in T_d in layers and place them above the layers of T_w. Altogether, we obtain a packing of T inside a strip of height $3v(T) + 4h_{\max}$.

We now turn to the proof of (i). For this we classify T as follows: let $T_\ell := \{i \in T \mid w_i > 1/2 \text{ and } d_i > 1/2\}$, and $T_s := T \setminus T_\ell$. We pack the items of T_ℓ in layers, with one item per layer. Since the base area of each item in T_ℓ exceeds $1/4$, the height of this packing is at most $4v(T_\ell)$. For the set T_s, we use the packing algorithm of (ii) that yields a packing into a strip of height $3v(T_s) + 4h_{\max}$. Placing this strip above the packing of T_ℓ, we obtain a packing of T inside a strip of height $4v(T) + 4h_{\max}$, completing the proof. ◀

▶ **Theorem 6** ([50]). *Given a set of 3D items T, there is a polynomial-time algorithm that places these items into at most $8v(T) + O(1)$ bins.*

The last result is regarding the asymptotic approximation of 3D-SP.

▶ **Theorem 7** ([35]). *Given a set of 3D items I where each item has height bounded by $h_{\max}$, for any constant $\varepsilon > 0$, there is a polynomial-time algorithm that returns a packing of I into a strip of height at most $(3/2 + \varepsilon)\mathrm{OPT}_{3D\text{-}SP}(I) + \varepsilon + O_\varepsilon(1)h_{\max}$.*

2.2 Harmonic transformation

Lee and Lee [41] introduced harmonic transformation in the context of online bin packing. The harmonic transformation with parameter k is defined by the following function f_k:
For $\alpha_i \in (1/(q+1), 1/q]$, $f_k(\alpha_i) := 1/q$, for $q \in [k-1]$. Otherwise, $f_k(\alpha_i) := \frac{k}{k-1}\alpha_i$.
Intuitively, the function f_k rounds up $\alpha_i \in (1/k, 1]$ to the nearest larger number of the form $1/q$ where $q \in \mathbb{Z}$.

Now we define harmonic constant T_∞. Let $t_1 := 1$ and $t_{i+1} = t_i(t_i + 1)$ for $i \in \mathbb{Z}_{\geq 2}$. The sequence $t_i + 1$ is also known as Sylvester's sequence (where each term is the product of the previous terms, plus one). Let $m(k)$ be the integer such that $t_{m(k)} \leq k \leq_{m(k)+1}$. Now T_k is defined as $\sum_{q=1}^{m(k)} \frac{1}{t_q} + \frac{k}{t_{(m(k)+1)}\cdot(k-1)}$, and $T_\infty := \lim_{k\to\infty} T_k$. Thus $T_\infty = \sum_{i=1}^{\infty} \frac{1}{t_i} = 1 + \frac{1}{2} + \frac{1}{6} + \cdots \approx 1.69103$. Note that $T_k \leq T_\infty + \frac{1}{(k-1)}$.

Lee and Lee [41] showed that that, for any sequence $\alpha_1, \alpha_2, \ldots, \alpha_n$, with $\alpha_i \in (0, 1]$ and $\sum_{i=1}^{n} \alpha_i \leq 1$, we have $\sum_{i=1}^{n} f_k(\alpha_i) \leq T_k$. In fact, $\lim_{k\to\infty} \sum_{i=1}^{n} f_k(\alpha_i) \leq T_\infty \approx 1.691$.

In fact, Bansal, Han, Iwama, Sviridenko, and Zhang [9] showed the above inequality is true even if we define $f_k(\alpha_i) := \alpha_i$ for $\alpha_i \leq 1/k$.

2.3 Next-Fit-Decresing-Height (NFDH)

The Next-Fit-Decreasing-Height (NFDH) algorithm is a shelf-based approach for packing 2D items into a strip of fixed width w. Given a set of items I, the algorithm first sorts them in

decreasing order of height. It then places the items sequentially from left to right on the floor of the strip (or the current shelf) until the next item no longer fits, i.e., adding the next item would cause the total width of items on the shelf to exceed w. At this point, a new shelf is created by drawing a horizontal line at the height of the tallest item on the current shelf. The process then continues on the new shelf, following the same placement rule, until all items have been packed. For more details on the algorithm, we refer to [15, 14].

▶ **Lemma 8** ([15]). *Given a 2D rectangular box of height h and width w, and a set of 2D items with maximum height $h_{\max}$ and maximum width $w_{\max}$, it is possible to place any subset of items with a total area of at most $(h - h_{\max})(w - w_{\max})$, into the box using NFDH.*

2.4   Generalized Assignment Problem (GAP)

In the Generalized Assignment Problem, we are given a set of k knapsacks, each with an associated capacity $\{c_j\}_{j \in [k]}$, and a set of n items, where each item $i \in [n]$ has size s_{ij} and profit p_{ij} for knapsack j. The goal is to obtain a maximum-profitable packing of a subset of the items into the knapsacks that respects the knapsack capacities, i.e., the total size of the items packed inside each knapsack does not exceed the capacity of the knapsack. For the general case of GAP, there is a tight $(\frac{e}{e-1} + \varepsilon)$-approximation [25]. However for the special case when $k = O(1)$, there exists a PTAS.

▶ **Theorem 9** ([26]). *For any $\varepsilon > 0$, there is an algorithm for GAP with k knapsacks running in $n^{O(k/\varepsilon^2)}$ time that returns a packing of profit at least $(1 - \varepsilon)\mathrm{OPT}$.*

3   Absolute 6-approximation for 3D-BP

In this section, our goal is to prove Theorem 1. Let $K > 0$ be a large constant such that the algorithm of Caprara [12] already yields an absolute 6-approximation when $\mathrm{OPT}_{\text{3D-BP}} > K$. Our goal is to obtain a 6-approximation for the case when $\mathrm{OPT}_{\text{3D-BP}} \leq K$. Let $\lambda = 1/40$, and $\delta < \lambda$ be a sufficiently small constant. The following lemma follows from a standard shifting argument.

▶ **Lemma 10.** *There exists a polynomial-time computable $\mu \leq \delta$ such that the total volume of the items that have at least one of the dimensions in the range $(\mu^4, \mu]$ is at most δ.*

Proof. Let $\mu_0 = \delta$, and for $j \in [3K/\delta]$, define $\mu_j = \mu_{j-1}^4$. Let $I_j \subseteq I$ be the items having at least one dimension in the range $(\mu_j, \mu_{j-1}]$. Note that since we are in the case when $\mathrm{OPT}_{\text{3D-BP}} \leq K$, we have $v(I) \leq K$. Observing that each item can belong to at most three of the sets $\{I_j\}_{j \in [3K/\delta]}$, we have $\sum_{j \in [3K/\delta]} v(I_j) \leq 3K$, and thus there must exist an index j^* for which $v(I_{j^*}) \leq \delta$. Setting $\mu = \mu_{j^*-1}$ completes the proof.   ◄

We classify the items depending on their dimensions: let L be the items whose height, width and depth all exceed μ (called *large* items), I_h be the items with height at most μ^4, I_w be the remaining items having width at most μ^4 and I_d be the remaining items with depth at most μ^4. Finally, let I^{rem} be the remaining items, each having at least one of the dimensions in the range $(\mu^4, \mu]$. Note that $v(I^{\text{rem}}) \leq \delta$ owing to Lemma 10. We further classify the items of I^{rem} in a similar way – let $I_h^{\text{rem}} \subseteq I^{\text{rem}}$ be the items with height at most μ, $I_w^{\text{rem}} \subseteq I^{\text{rem}} \setminus I_h^{\text{rem}}$ be the remaining items with width at most μ, and $I_d^{\text{rem}} = I^{\text{rem}} \setminus (I_h^{\text{rem}} \cup I_w^{\text{rem}})$.

In the remainder of the section, we prove the following result.
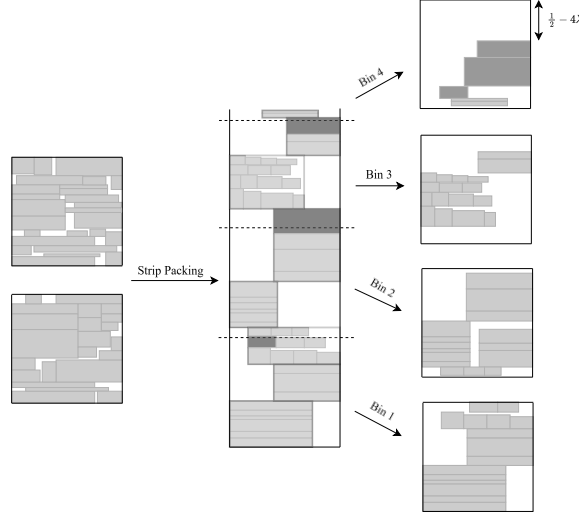
■ **Figure 1** Packing from Lemma 12 (only the front view is shown for simplicity) for $k = 2$. The dark gray items are sliced while cutting out $\lfloor 3k/2 \rfloor + 1$ bins from the Strip Packing solution. Finally, sliced items are packed into the empty regions of the last bin.

▶ **Proposition 11.** *If there exists a packing of all items into $k \leq K$ bins, then a packing using at most $6k$ bins can be computed in polynomial time.*

For this, we first show the following lemma, which follows from a simple application of Theorem 7.

▶ **Lemma 12.** *Let T be a set of items, each having a height (analogously width, depth) of at most μ, and suppose that there exists a packing of T into $k \leq K$ bins. Then, it is possible to compute a packing of T using $\lfloor 3k/2 \rfloor + 1$ bins in polynomial time. Further, one of these bins has an empty strip with 1×1 base and height (analogously width, depth) $1/2 - 4\lambda$.*

Proof. Since there exists a packing of T into k bins, the optimal Strip Packing height of the items of T is also bounded by k. Using Theorem 7 with $\varepsilon = \lambda/K$, we first obtain a packing of T into a strip of height at most $(3/2 + \lambda/K)k + \lambda/K + O_\lambda(1)\mu \leq 3k/2 + 3\lambda$, since $\mu \leq \delta$ and δ is sufficiently small. Next, we cut the strip at integral heights and let T' be the set of items that are sliced by this process. The remaining items of T are thus packed into $\lceil 3k/2 + 3\lambda \rceil = \lfloor 3k/2 \rfloor + 1$ bins. Note also that the last bin is filled up to a height of at most $1/2 + 3\lambda$. We pack the items of T' into the last bin, which further occupies a height of at most $\mu(\lfloor 3k/2 \rfloor + 1) \leq \lambda$. Hence, the empty region inside the last bin has a height of at least $1/2 - 4\lambda$. ◀

We divide the proof of Proposition 11 into two cases depending on $v(L)$.

3.1 Case 1: $v(L) > 64\delta K$

In this case, for some $j \in \{h, w, d\}$, the total volume of the items in I_j must not exceed $(k - 64\delta K)/3 \leq (1/3 - 21\delta)k$ – w.l.o.g. assume that $j = h$. We first pack the items of $I_w \cup I_w^{\mathrm{rem}}$ and $I_d \cup I_d^{\mathrm{rem}}$ into $\lfloor 3k/2 \rfloor + 1$ bins each, using Lemma 12. Our goal next is to pack the items of $L \cup I_h \cup I_h^{\mathrm{rem}}$ into $2k$ bins. To this end, we further classify the items of I_h depending on their width and depth. Let $I_{h,\ell} := \{i \in I_h \mid w_i, d_i > 1/2\}$ and let $I_{h,s} := I_h \setminus I_{h,\ell}$. We first pack the items of $I_{h,s}$ into k bins using the following lemma by applying Theorem 5.
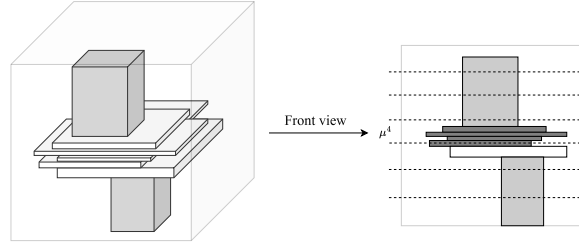
■ **Figure 2** The light gray items are items of L. The dark gray items are deleted in order to position the upper large item at a multiple of μ^4.

▶ **Lemma 13.** *The items of $I_{h,s}$ can be completely packed into k bins where each bin additionally has an empty strip with 1×1 base and height 59δ.*

Proof. We arbitrarily group the items of $I_{h,s}$ into maximal groups of volume at most $(1/3 - 20\delta)$ each. Since the volume of each item is bounded by μ^4, each group has a volume of at least $1/3 - 20\delta - \mu^4 > 1/3 - 21\delta$, and therefore the number of groups is at most k (since $v(I_{h,s}) \le v(I_h) \le (1/3 - 21\delta)k$). Using Theorem 5, the items in each group can be packed in a bin within a height of $3 \cdot (1/3 - 20\delta) + 8\mu^4 \le 1 - 59\delta$, leaving an empty region of height 59δ as claimed. ◀

Now consider the optimal packing inside k bins restricted to the items of $I_{h,\ell} \cup L$. Our goal is to compute a packing of all these items, barring a subset of items from $I_{h,\ell}$ that have a total volume of at most $O(\mu)k$. For this, we first discretize the positions of the large items inside the bins. We focus on the packing inside any one bin.

▶ **Lemma 14.** *By discarding items of $I_{h,\ell}$ having a total volume of at most 2μ, the number of distinct positions of the items of L can be assumed to be polynomially-bounded.*

Proof. Let us start with the optimal packing inside k bins restricted to the items of $I_{h,\ell} \cup L$. For discretizing the x- and y-positions, we push all items as much to the left and front as possible. Then the distance of the left (resp. front) face of an item of L from the left (resp. front) face of the bin can be written as the sum of the widths (resp. depths) of at most $1/\mu$ large items and at most one item of $I_{h,\ell}$. Thus, there are polynomially many choices.

In order to discretize the positions along the z-axis, we draw horizontal planes inside the bin at heights as integral multiples of μ^4 and consider the large items inside the bin from bottom to top. For each $i \in L$ in the order of increasing height of their bases, we discard items of $I_{h,\ell}$ in order to pull the item i downwards until the bottom face of i hits one of the horizontal planes at heights multiples of μ^4 or the top face of another large item lying below i (see Figure 2). At the end of this process, it can be seen that the distance between the bottom face of any large item and the base of the bin can be written as the sum of a multiple of μ^4 and the heights of at most $1/\mu$ items of L. The total volume of the discarded items is bounded by $(1/\mu^3) \cdot 2\mu^4 = 2\mu$. ◀

We next draw horizontal planes passing through the top and bottom faces of each large item and discard the items of $I_{h,\ell}$ that are intersected by these planes. The volume of these discarded items is bounded by $(2/\mu^3) \cdot \mu^4 = 2\mu$. This partitions the bin into at most $2/\mu^3 + 1$ *slots*, where each slot is penetrated from top to bottom by at most $1/\mu^2$ large items (see Figure 3). Note also that no large item begins or ends in the interior of a slot. Together with Lemma 14, we thus have the following result.
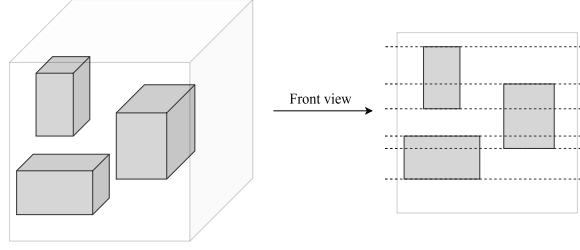
Figure 3 The regions between two consecutive dotted lines correspond to slots.

▶ **Lemma 15.** *There exists a subset $I'_{h,\ell} \subseteq I_{h,\ell}$ with $v(I'_{h,\ell}) \geq v(I_{h,\ell}) - 4\mu k$ such that the items of $I'_{h,\ell}$ are completely packed inside the slots formed by the large items in the k bins.*

Proof. By Lemma 14, we discard items of $I_{h,\ell}$ having a volume of $2\mu k$, in order to discretize the positions of the large items. Also, the volume of items discarded while partitioning the k bins into slots is bounded by $2\mu k$. The remaining items of $I_{h,\ell}$, that have a volume of at least $v(I_{h,\ell}) - 4\mu k$ are packed inside the slots in the k bins. ◀

Our algorithm essentially tries to compute a packing close to the one guaranteed by the above lemma. As mentioned before, we obtain a packing into k bins of all items of L, and a large volume subset of $I_{h,\ell}$ that is packed inside the slots formed by the large items.

▶ **Lemma 16.** *In polynomial-time, it is possible to compute a set $I''_{h,\ell} \subseteq I_{h,\ell}$ with $v(I''_{h,\ell}) \geq v(I_{h,\ell}) - 5\delta k$, and a packing of all items in $I''_{h,\ell} \cup L$ into k bins.*

Proof. We first guess the positions (from polynomially many choices) of all the (at most K/μ^3) large items inside the k bins, and create slots by extending their top and bottom faces. Our goal is to pack a maximum volume subset of $I_{h,\ell}$ into these slots by creating an instance of the Generalized Assignment Problem (GAP) with $O(1)$ knapsacks, for which there exists a PTAS [26] (see Section 2.4 for details). Each slot corresponds to a knapsack with a capacity equal to the height of the slot. For an item $i \in I_{h,\ell}$, the size of i for a slot equals its height h_i if the item fits inside the slot, given the relative positions of the large items inside it, and ∞ otherwise. The profit of an item is the same as its volume. By Lemma 15, the optimal packing packs items of $I_{h,\ell}$ having a volume of at least $v(I_{h,\ell}) - 4\mu k$. We apply the PTAS for GAP with parameter δ/K to our instance, yielding a packing of a subset $I''_{h,\ell} \subseteq I_{h,\ell}$ into the slots formed by the items of L, whose volume is least $(1 - \delta/K)(v(I_{h,\ell}) - 4\mu k) \geq v(I_{h,\ell}) - 5\delta k$. ◀

It remains to pack the items of $I_{h,\ell} \setminus I''_{h,\ell}$ and I_h^{rem}. Intuitively, they have a small volume, and hence, we can pack them into the empty regions inside the already-existing bins.

▶ **Lemma 17.** *The items of $I_{h,\ell} \setminus I''_{h,\ell}$ can be completely packed by using a height of 25δ from each of the empty regions inside the bins that were used to pack the items of $I_{h,s}$. Further, the items in I_h^{rem} can be packed within a height of 12δ inside one of the bins.*

Proof. Recall that each of the k bins used for packing items of $I_{h,s}$ had an empty strip of height 59δ by Lemma 13. Consider the items of $I_{h,\ell} \setminus I''_{h,\ell}$ that have a volume of at most $5\delta k$ by Lemma 16. We arbitrarily group them into maximal groups of volume not exceeding 6δ each. Since each item in $I_{h,\ell}$ has a volume of at most μ^4, the volume of each group is at least $6\delta - \mu^4 > 5\delta$, and thus there are at most k groups. Using Theorem 5, the items in each such group can be packed within a height of $4 \cdot 6\delta + 8\mu^4 \leq 25\delta$.

Note that there is still an empty region of height $59\delta - 25\delta = 34\delta$ left in each bin. Since $v(I_h^{\mathrm{rem}}) \leq \delta$ by Lemma 10, the items of I_h^{rem} can be packed such that the packing has a
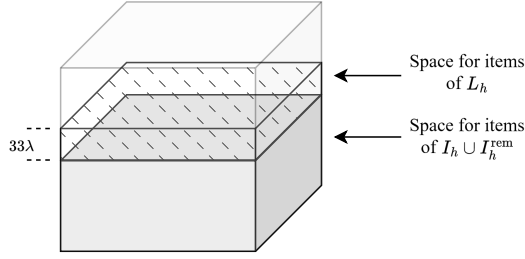
Figure 4 Packing inside bin B_h

height of at most $4\delta + 8\mu \le 12\delta$ using Theorem 5, and hence we can place them inside the remaining empty region inside any of these bins.   ◀

Altogether, we used $2 \cdot (\lfloor 3k/2 \rfloor + 1) \le 4k$ bins for packing items in $I_w \cup I_w^{\mathrm{rem}} \cup I_d \cup I_d^{\mathrm{rem}}$ and $2k$ bins for packing the items of $L \cup I_h \cup I_h^{\mathrm{rem}}$, resulting in at most $6k$ bins overall.

3.2   Case 2: $v(L) \le 64\delta K$

In this case, we first pack the items of $I_h \cup I_h^{\mathrm{rem}}, I_w \cup I_w^{\mathrm{rem}}$ and $I_d \cup I_d^{\mathrm{rem}}$ into $\lfloor 3k/2 \rfloor + 1$ bins each, using Lemma 12 (note that these items have height, width and depth bounded by μ, respectively). Let B_h, B_w, B_d be the bins having empty strips of height, width, and depth $1/2 - 4\lambda$, respectively, that are guaranteed by Lemma 12. Intuitively, since the large items have a very small volume, they can be completely packed inside these empty strips.

▶ **Lemma 18.** *The items in L can be completely packed inside three strips, each having a 1×1 base aligned with the xy-, yz-, and zx-planes, respectively. The strips have height, width, and depth of 33λ, respectively, and therefore they fit inside the bins B_h, B_w, and B_d.*

Proof. First, note that for sufficiently small δ, we have $\delta K \le \lambda^3$, and since $v(L) \le 64\delta K \le 64\lambda^3$, it follows that one of the dimensions of each large item must be at most $(64\lambda^3)^{1/3} = 4\lambda$. We classify the items of L into three groups – let $L_h \subseteq L$ be those items with height at most 4λ, $L_w \subseteq L \setminus L_1$ be the items with width at most 4λ and $L_d = L \setminus (L_1 \cup L_2)$ be the remaining items, each having depth not exceeding 4λ.

Next, since $v(L_h) \le v(L) \le 64\lambda^3$, Theorem 5 implies that the items in L_h can be completely packed within a strip with 1×1 base and a height of at most $4 \cdot 64\lambda^3 + 8 \cdot 4\lambda \le 33\lambda$ (see Figure 4). Analogously, we pack the items of L_w and L_d into strips of width and depth 33λ, respectively.   ◀

Overall, we obtain a packing into $3 \cdot (\lfloor 3k/2 \rfloor + 1) \le 6k$ bins, establishing Proposition 11.

Overall algorithm: We first run the algorithm of Caprara [12] that already returns a 6-approximate solution when $\mathrm{OPT}_{\text{3D-BP}} > K$. Next, for each guessed value of $\mathrm{OPT}_{\text{3D-BP}} = k \le K$, we run the algorithm of Proposition 11. For this, we first compute a value of μ using Lemma 10 and classify the items as discussed. Next, we divide into two cases depending on the volume of the large items. If $v(L) > 64\delta K$, we find $j \in \{h, w, d\}$ for which the volume of the items in I_j does not exceed $(1/3 - 21\delta)k$; w.l.o.g we take $j = h$. We pack the items of $I_w \cup I_w^{\mathrm{rem}}$ and $I_d \cup I_d^{\mathrm{rem}}$ into $\lfloor 3k/2 \rfloor + 1$ bins each using Lemma 12. We classify items of I_h into $I_{h,\ell}$ and $I_{h,s}$ depending on their width and depth and obtain a packing of $I_{h,s}$ into k bins, ensuring each of these bins has an empty strip of height 59δ. Next we compute a set $I_{h,\ell}'' \subseteq I_{h,\ell}$ such that $v(I_{h,\ell} \setminus I_{h,\ell}'') \le 5\delta k$, and pack the items of $I_{h,\ell}'' \cup L$ into k bins via a reduction to the Generalized Assignment Problem (Lemma 16). Finally, the items in

$(I_{h,\ell} \setminus I''_{h,\ell}) \cup I_h^{\text{rem}}$ are packed into the empty spaces inside the bins for $I_{h,s}$ using Lemma 17. For the other case when $v(L) \leq 64\delta K$, we pack items in $I_h \cup I_h^{\text{rem}}, I_w \cup I_w^{\text{rem}}$ and $I_d \cup I_d^{\text{rem}}$ into $\lfloor 3k/2 \rfloor + 1$ bins each, using Lemma 12, ensuring one of the bins has a sufficiently large empty strip, and then pack items of L inside these empty strips using Lemma 18.

3.3 Implication on 3D-SP

We now establish Theorem 2. We use the following observation from our 3D-BP algorithm.

▶ **Lemma 19.** *If there exists a packing of all items into a single bin, then it is possible to compute a packing into 6 bins in polynomial-time, where one of the bins is filled up to a height of at most $1 - 22\delta$.*

Proof. For the case when $v(L) > 64\delta K$, our algorithm first finds an index $j \in \{h, d, w\}$ for which $v(I_j) \leq 1/3 - 21\delta$ holds. If $j \neq h$, we already have a bin that is filled up to a height of at most $1/2 + 4\lambda$ by Lemma 12. Consider the case when $j = h$. By Lemma 13, there was an empty strip of height 59δ inside the bin used to pack items of $I_{h,s}$. Out of this, a height of $25\delta + 12\delta = 37\delta$ was used to pack the items of $(I_{h,\ell} \setminus I''_{h,\ell}) \cup I_h^{\text{rem}}$ by Lemma 17. Thus the height of the remaining empty strip inside the bin is $59\delta - 37\delta = 22\delta$. Finally, for the case when $v(L) \leq 64\delta K$, by Lemma 18, the bin B_h is filled up to a height of only at most $1/2 + 4\lambda + 33\lambda = 1/2 + 37\lambda$, and we are done. ◀

Consider now the optimal Strip Packing of the input instance. Since the height of this packing must lie in $[h_{\max}, nh_{\max}]$, we can assume the optimal height to be of the form $h_{\max}(1 + \varepsilon)^j$, by losing only a factor of $1 + \varepsilon$. We scale the height of each item by the guessed height so that all items now fit inside a $1 \times 1 \times 1$ bin. Using Lemma 19, we compute a packing into 6 bins and stack these bins one on top of the other along the height so that the resulting height of the packing is at most $6 - 22\delta$. This establishes Theorem 2 with $\rho = 22\delta$.

The above result holds for the case when the strip is unbounded along the z-axis. If, instead, the strip could be extended along any of the x-, y- or z-axes and the goal was to minimize the length of the strip along that direction, then we improve further.

▶ **Lemma 20.** *If there exists a packing of all items into a single bin, then for any $\varepsilon > 0$, it is possible to compute a packing into 6 bins in polynomial time, where one of the bins is filled up to a length (height/width/depth) of at most $1/2 + O(\varepsilon)$.*

Proof. First note that, although we ran our 3D-BP algorithm with $\lambda = 1/80$, we could as well execute it with parameter ε, assuming $\varepsilon < \lambda$. Recall that the proof of Proposition 11 was divided into two cases depending on the volume of the large items. For the case when $v(L) > 64\delta K$, we first found an index $j \in \{h, w, d\}$ for which $v(I_j) \leq 1/3 - 21\delta$ – w.l.o.g assume $j = h$. Then we packed the items of $I_w \cup I_w^{\text{rem}}$ and $I_d \cup I_d^{\text{rem}}$ into 2 bins each using Lemma 12, which ensures that one of the two bins in each case has an empty strip of length $1/2 + 4\varepsilon$, and we are done in this case. For the other case when $v(L) \leq 64\delta K$, the bin B_h (resp. B_w, B_d) was filled up to a height (resp. width, depth) of at most $1/2 + 4\varepsilon + 33\varepsilon = 1/2 + 37\varepsilon$, and we are done. ◀

W.l.o.g. assume that one of the bins is filled up to height $1/2 + O(\varepsilon)$. Thus, stacking the bins along the z-axis would be of height at most $11/2 + O(\varepsilon)$, implying the following result.

▶ **Corollary 21.** *For any $\varepsilon > 0$, there exists a polynomial-time $(11/2 + \varepsilon)$-approximation for 3D-SP, if the strip can be extended along any of the x-, y- or z-axes.*

4 Asymptotic $(\frac{3}{2} \cdot T_\infty + \varepsilon)$-approximation for 3D-BP

In this section, we prove Theorem 4 and present an improved asymptotic approximation algorithm for 3D-SP. We will utilize ideas from the algorithm for 3D-SP by Jansen and Prädel [35] which packs I into strip height of $(3/2 + \varepsilon) \cdot \mathrm{OPT}_{\text{3D-SP}}(I) + O_\varepsilon(1)h_{\max}$. As mentioned earlier, the naive approach of cutting the strip at integral height will result in $(3 + \varepsilon)$-approximation. Instead, we exploit the structural properties of the solution provided by the algorithm, along with the *harmonically rounded heights* of the items, to ensure that items with a height larger than ε are not sliced.

Recall the definition of harmonic rounding: For $\alpha_i \in (1/(q+1), 1/q]$, $f_k(\alpha_i) := 1/q$, for $q \in [k-1]$; and for $\alpha_i \in (0, 1/k], f_k(\alpha_i) := \alpha_i$. Also, if $\sum_{i=1}^{n} \alpha_i \leq 1$, then $\lim_{k \to \infty} \sum_{i=1}^{n} f_k(\alpha_i) \leq T_\infty \approx 1.691$. In the following, we assume $k = 1/\varepsilon$ to be large enough such that $\sum_{i=1}^{n} f_{1/\varepsilon}(\alpha_i) \approx T_\infty$, and define $f_{1/\varepsilon}$ to be f.

Let I^∞ be the instance derived from the given 3D-BP instance I by applying harmonic rounding f to the heights of the items. Thus an item $i \in I$ becomes an item of I^∞ with width, depth, height to be $w_i, d_i, f(h_i)$, respectively. Let $\mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty)$ denote the minimum number of bins with dimensions $1 \times 1 \times T_\infty$ required to pack all items from I^∞, and let $\mathrm{OPT}_{\text{3D-SP}}(I^\infty)$ denote the minimum height to pack all items from I^∞ into a strip with unit square base and unbounded height. The following lemma connects packing of I^∞ with I:

▶ **Lemma 22.** $\mathrm{OPT}_{3D\text{-}SP}(I^\infty) \leq T_\infty \cdot \mathrm{OPT}_{3D\text{-}BP}^{T_\infty}(I^\infty) \leq T_\infty \cdot \mathrm{OPT}_{3D\text{-}BP}(I)$.

Proof. First, we show that $\mathrm{OPT}_{\text{3D-BP}}^\infty(I^\infty) \leq \mathrm{OPT}_{\text{3D-BP}}(I)$. Take the optimal solution for 3D-BP, then increase the item heights due to the harmonic rounding and extend the height of the bin by T_∞. As the height of each combination of items that are on top of each other in the solution $\mathrm{OPT}_{\text{3D-BP}}(I)$ adds up to at most 1, the sum of rounded heights will be bounded by T_∞, end hence the packing still fits into the bin.

Next, we prove that $\mathrm{OPT}_{\text{3D-SP}}(I^\infty) \leq T_\infty \cdot \mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty)$. Consider the optimal solution to corresponding to $\mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty)$. We can create a solution to 3D-SP by stacking all the bins on top of each other. As each bin has a height of at most T_∞, the total height of the packing is bounded by $T_\infty \cdot \mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty)$ and hence $\mathrm{OPT}_{\text{3D-SP}}(I^\infty) \leq T_\infty \cdot \mathrm{OPT}_{\text{3D-BP}}^{T_\infty}(I^\infty)$.   ◀

We transform the instance I^∞ for 3D-SP into an instance for 2D-BP, similar to [35], which then uses structural results from a 2D-BP algorithm [36]. Given ε, we round up the item heights of I^∞ to the next multiple of $\frac{\varepsilon v(I^\infty)}{n}$. We create an instance $I_{\text{2D-BP}}$ of 2D-BP, by introducing for each item $i \in I^\infty$ with rounded height $k\frac{\varepsilon v(I^\infty)}{n}$, exactly k rectangles with width w_i and depth d_i. The following lemma bounds the incurred loss.

▶ **Lemma 23.** $\frac{\varepsilon v(I^\infty)}{n}\mathrm{OPT}_{2D\text{-}BP}(I_{2D\text{-}BP}) \leq (1 + \varepsilon)\mathrm{OPT}_{3D\text{-}SP}(I^\infty)$.

Proof. We prove that $\frac{\varepsilon v(I^\infty)}{n}\mathrm{OPT}_{\text{2D-BP}}(I_{\text{2D-BP}}) \leq (1 + \varepsilon)\mathrm{OPT}_{\text{3D-SP}}(I^\infty)$. First, note that by rounding the item heights to the next larger multiple of $\frac{\varepsilon v(I^\infty)}{n}$ the total packing height of the optimal solution for 3D-SP is increased by at most $n \cdot \frac{\varepsilon v(I^\infty)}{n} = \varepsilon v(I^\infty)$, as there are at most n items on top of each other. As each packing for 3D-SP given $I_{\text{3D-SP}}$ has a height of at least $v(I^\infty)$, the height of the instance is increased by at most $\varepsilon v(I^\infty) \leq \varepsilon \mathrm{OPT}_{\text{3D-SP}}(I^\infty)$.

Note that we can assume that in an optimal solution, each item either stands on the bottom or directly on top of another item. We cut the optimal solution for the rounded instance at multiples of $\frac{\varepsilon v(I^\infty)}{n}$. Using these cuts, there is only one item on top of each other. Therefore, these slices represent a solution for $I_{\text{2D-BP}}$ and hence $\frac{\varepsilon v(I^\infty)}{n}\mathrm{OPT}_{\text{2D-BP}}(I_{\text{2D-BP}}) \leq (1 + \varepsilon)\mathrm{OPT}_{\text{3D-SP}}(I^\infty)$.   ◀

4.1 Essentials of the 2D-BP algorithm

The main ingredient for the 2D-BP algorithm by Jansen and Prädel [36] is a restructuring theorem. It states that each packing can be rearranged into packing with an iterable structure. This rearrangement comes at the cost of introducing more bins to the packing.

First, the items are classified into big, vertical, horizontal, intermediate, and tiny based on a suitably chosen constant $\delta \in [\varepsilon^{O_\varepsilon(1)}, \varepsilon]$. Let $\mu = \delta^4$. An item i is *big* if $w_i \geq \delta$ and $d_i \geq \delta$; *vertical* if $d_i \geq \delta$ but $w_i < \mu$; *horizontal* if $w_i \geq \delta$ but $d_i < \mu$; *tiny* if $d_i < \mu$ and $w_i < \mu$; and *intermediate* if either $w_i \in [\mu, \delta)$ or $d_i \in [\mu, \delta)$. Using standard argument (see Lemma 43 in Appendix A), we show that the area of intermediate items is at most $\varepsilon \cdot \text{area}(I)$.

Jansen and Prädel [36] showed the existence of a structured packing that we call *k-2D-container-packing*. In a *k-2D-container-packing*, one can consider a rounded-up instance $\tilde{I}$ from $I_{2\text{D-BP}}$, where the widths and depths of big items, widths of horizontal items, and depths of vertical items from $I_{2\text{D-BP}}$ are rounded up to $O(1/\delta^2)$ values. Let $\mathcal{T}, \mathcal{W}, \mathcal{D}$ be the set of different types of (rounded) large items, widths of wide items, and depths of vertical items in $\tilde{I}$, respectively. Then $|\mathcal{T}|$ is $O(1/\delta^4)$, and $|\mathcal{W}|, |\mathcal{D}|$ are $O(1/\delta^2)$.

In a *k-2D-container-packing*, each bin of the packing is partitioned into containers. Furthermore, vertical items are allowed to be sliced along y-axis, horizontal items may be sliced along x-axis, and tiny items may be sliced in both directions. The containers are of five types, and only specific types of items from $\tilde{I}$ are allowed to be packed in the corresponding containers:

- (i) *Big containers:* Each such container contains only one (rounded up) big item and has the size of this big item.
- (ii) *Horizontal containers:* Each such container has a width $w \in \mathcal{W}$ and a depth that is a multiple of μ, and contains only horizontal items with width w. Per bin, the total width of these containers is bounded by $O(k)$.
- (iii) *Vertical containers:* Each such container has a depth $d \in \mathcal{D}$, a height that is a multiple of μ, and contains only vertical items with depth d. Per bin, the total depth of these containers is bounded by $O(k)$.
- (iv) *Tiny containers:* Each such container contains only tiny items and has a width and depth that is a multiple of μ. Each bin contains at most $O(k)$ of these containers.
- (v) *Intermediate containers:* These containers contain only intermediate items. There will be an extra $O(\varepsilon)\text{OPT}_{2\text{D-BP}}$ bins reserved separately to pack the intermediate items.

We refer to a bin configuration as a valid way to split one bin into containers with regard to the considered k-2D-container-packing. Jansen and Prädel [36] showed the existence of a good *2D-container-packing* (see Appendix A for omitted details).

▶ **Theorem 24** ([36]). *Given some instance I for 2D-BP, there is an algorithm that finds in polynomial time a rounded-up instance $\tilde{I}$ with $|\mathcal{T}|, |\mathcal{D}|, |\mathcal{W}| \in O(1/\delta^4)$ and a $(1/\delta^3)$-2D-container-packing of $\tilde{I}$ into B bins with at most $O_\varepsilon(1)$ different bin-configurations that fulfills $B \leq (\frac{3}{2} + O(\varepsilon))\text{OPT}_{2\text{-BP}}(I) + O_\varepsilon(1)$.*

Although this theorem is not explicitly stated in [36], it encapsulates the core idea of the paper. For completeness, we provide a proof in Appendix A.

4.2 The algorithm for 3D-BP

In this subsection, we describe how the items are filled into the bins when a (ε/μ)-2D-container-packing is given. An overview of the complete algorithm can be found in Algorithm 1. To prove the following Lemma, we consider the big, horizontal and vertical, as well as tiny items separately.

■ **Algorithm 1** Asymptotic $((3/2)T_\infty + \varepsilon)$-approximation for 3D-BP with input I and $\varepsilon > 0$.

---

1 Create I^∞ by harmonically rounding the item heights of I.

2 Create $I_{\text{2D-BP}}$ after rounding the heights of the items in I^∞ to multiples of $\frac{\varepsilon v(I^\infty)}{n}$.

3 Find a rounded instance $\tilde{I}$ and the $(1/\delta^3)$-2D-container-packing into B 2D bins with at most $O_\varepsilon(1)$ bin-configurations.

4 Place I^∞ into 3D containers corresponding to the 2D-BP solution as discussed in Section 4.2.

5 **return** the packing.
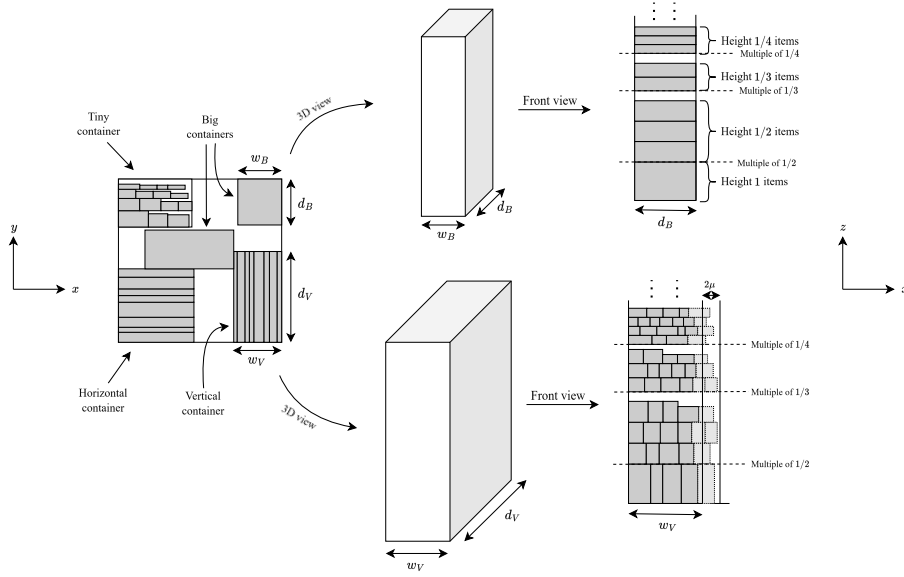
■ **Figure 5** The left figure depicts a 2D container-packing, which forms the base of a 3D configuration. The middle figure shows two 3D containers corresponding to two containers (one big and one horizontal) in 2D container-packing. On the right, the packing that ensures the *tall-not-sliced* property is shown. The light gray rectangles are repacked into additional bins.

▶ **Lemma 25.** *Given a rounded 2D-BP instance $\tilde{I}$ derived from the 3D-BP instance I^∞ and an (ε/μ)-2D-container-packing of $\tilde{I}$ into B 2D bins using k 2D bin-configurations, a packing of the items I^∞ into $(1 + O(\varepsilon))\frac{\varepsilon v(I^\infty)}{n}B + O(\log(1/\varepsilon)k + |\mathcal{T}|)$ 3D bins can be found in polynomial time.*

Let C be one of the k 2D bin configurations used, and let x_C denote the multiplicity (possibly fractional) of C in the solution consisting of B 2D bins. We create a 3D configuration of height $\lceil x_C \cdot \frac{\varepsilon v(I^\infty)}{n} \rceil$, whose base is identical to C. Note that the rounding to an integer height increases the total height of the configurations by at most k, i.e., the total height of all 3D configurations is bounded by $\frac{\varepsilon v(I^\infty)}{n}B + k$. The 2D containers of C raised by the corresponding height of the configuration will be referred to as 3D containers and will be denoted by the same type as the corresponding 2D containers.

Our goal is to pack the items of I^∞ into these 3D containers while ensuring the *tall-not-sliced* property. Items with a rounded height larger than ε are called *tall* items, and the remaining items are called *short*. The classification of items into big, vertical, horizontal, tiny, and intermediate, depending on the dimensions of their top faces, as defined in the

previous subsection, continues to hold.

Placing big items: The different slices of a big 3D item might be rounded differently by the 2D-BP algorithm. However, using a linear program, we can assign all big 3D items except for $O(|\mathcal{T}|)$ to the big 3D containers without violating the container heights. The $O_\varepsilon(1)$ unassigned big items will be packed using $O_\varepsilon(1)$ additional bins. To formalize this process, consider the set of 3D items I_B whose slices are big 2D items in $\tilde{I}$. Further, for any $t \in \mathcal{T}$, denote by K_t the set of 3D containers that are designated for big items with rounded size of type t and let $\tilde{I}_t$ denote the set of 3D items with rounded size t.

▶ **Lemma 26.** *Given a rounded (2D) instance $\tilde{I}$ of $I_{2\text{D-BP}}$, in polynomial time, it is possible to compute an assignment of all but $|\mathcal{T}|$ big 3D items to rounded sizes $t \in \mathcal{T}$, such that for each $t \in \mathcal{T}$, the total height of assigned 3D items is bounded by $\frac{\varepsilon v(I^\infty)}{n}|\tilde{I}_t|$.*

Proof. We introduce an LP to decide which items in I_B will be placed into which containers. For some 3D item $i \in I_B$, let $\mathcal{T}_i$ denote the set of types the item can be rounded to. We introduce variables $x_{i,t}$ that represent, for each big item i and type $t \in \mathcal{T}_i$, the total height of i that is rounded to size t. As $\tilde{I}$ is a rounding that assigns each of the big 3D item slices of height $\frac{\varepsilon v(I^\infty)}{n}$ to a rounded size, the assignment LP

$$\sum_{t \in \mathcal{T}_i} x_{i,t} = h_i \qquad\qquad \forall i \in I_B$$

$$\sum_{i \in I_B} x_{i,t} \leq \frac{\varepsilon v(I^\infty)}{n}|\tilde{I}_t| \qquad\qquad \forall t \in \mathcal{T}$$

$$x_{i,t} \geq 0 \qquad\qquad \forall i \in I_B, t \in \mathcal{T}_i$$

has a basic solution with at most $|I_B| + |\mathcal{T}|$ non-zero components. As the number of variables and constraints are $O_\varepsilon(1)$, such a solution can be computed in polynomial time. As each item needs at least one non-zero component at most $|\mathcal{T}| = O_\varepsilon(1)$ items are assigned fractionally. ◀

For each of the rounded base types $t \in \mathcal{T}$, let $B_t \subseteq I^\infty$ denote the set of 3D items that are assigned to have a rounded base $t \in \mathcal{T}$ by the basic solution of the assignment LP. We place each of the $|\mathcal{T}|$ fractionally placed big items into individual bins. In the next step, we assign the items in B_t to containers K_t.

▶ **Lemma 27.** *By extending the height of one container in K_t by $O(\log(1/\varepsilon))$ and all other containers by ε, it is possible to place all the items B_t into the containers K_t while maintaining the tall-non-sliced property.*

Proof. Observe that $h(K_t) \geq h(B_t)$, as the given (ε/μ)-2D-container-packing is feasible and hence places all items in I_t resulting in $h(B_t) \leq \frac{\varepsilon v(I^\infty)}{n}|\tilde{I}_t| \leq h(K_t)$. Let $H_k := \sum_{i=1}^{k} 1/i$ denote the k-th harmonic number. Note that $H_{1/\varepsilon} \leq \ln(1/\varepsilon) + 1$. We will build one stack of all items in B_t that fulfills the tall-non-sliced property

The algorithm sorts the items B_t by height and stacks them on top of each other in order of nonincreasing heights. Remember that the heights of the tall items in B_t are rounded harmonically, i.e., all items have a height of the form $1/q$ for some $q \in \mathbb{N}$. For each $q \in \{3, \ldots, 1/\varepsilon\}$, let i_q be the first item in the stack (from bottom to top) that has a height of $1/q$. Iterating q from 3 to $1/\varepsilon$, the algorithm shifts up i_q and all the following items by at most $1/q$ such that the bottom of i_q is aligned to a height that is an integer multiple of $1/q$. By this step, the height of the stack is increased by at most $H_{1/\varepsilon} - \frac{3}{2}$. Further, it fulfills the tall-non-sliced property, as items with heights $1/q > \varepsilon$ never overlap integer heights.

In the next step, the algorithm iterates the containers in K_t, and for each container $K \in K_t$, it cuts the remaining stack at height $h(K)$ and places all items in this part into the container. As all the containers in K_t have an integer height, no tall item is cut in this process. If a small item is cut, the rest of the item is placed on top of the container, effectively extending the height of the container by ε. If $K \in K_t$ is the last container, all the residual items are placed inside it. As the stack has a height of at most $h(K_t) + H_{1/\varepsilon} - \frac{3}{2}$, the height of this last container is increased by at most $O(\log(1/\varepsilon))$.   ◀

Next, we show that the big items can be placed into the containers while maintaining the tall-non-sliced property.

▶ **Lemma 28.** *By extending the height of all containers for big items by $O(\log(1/\varepsilon))$, it is possible to place them into their containers and $O(|\mathcal{T}|)$ additional ones, while maintaining the tall-non-sliced property.*

Proof. We place each of the $|\mathcal{T}|$ fractionally placed big items from the proof of Lemma 26 into individual bins. In the next step, we assign the items B_t to containers K_t as done by Lemma 27. As a consequence, all tall items can be placed into their containers whose heights have been extended by $O(\log(1(\varepsilon))$ and $O(|\mathcal{T}|)$ additional ones.   ◀

Placing vertical and horizontal items. Note that for horizontal and vertical items, we use the same procedure, except that we rotate the strip and the items by 90 degrees along the z-axis.

For each $d \in \mathcal{D}$, let $\mathcal{V}_d$ denote the set of 3D items whose 2D counterparts have been rounded to be vertical items with rounded depth d and let K_d be the set of all containers for vertical items with rounded depth d. Since the depth of each container in K_d equals the vertical item depth d, we simplify the placement by considering only the xz-plane, i.e., the front face of the containers and the items. Let $\mathcal{V}_{d,xz}$ denote the set of all front faces of items in $\mathcal{V}_d$ and let $K_{d,xz}$ denote the set of front faces of all containers K_d for these vertical items. Algorithm 2 specifies how to place almost all 2D items $\mathcal{V}_{d,xz}$ into the 2D containers $K_{d,xz}$. Note that the y-coordinate of the 3D item corresponds to the y-coordinate of the 3D container.

The idea of the algorithm is to iteratively choose any empty container-face $K \in K_{d,xz}$ and pick a set of 2D items V_d' to be placed that have at least the area of K. To make sure that all items V_d' can be placed into the target area, the container-face K is extended on both sides. This extension is chosen large enough such that all items V_d' fit, when using the NFDH algorithm to place them. We use the notation $\mathtt{NFDH}(I, a \times b)$ to denote that the set of items I is placed into the rectangular area $a \times b$ with width a and height b using the NFDH algorithm. The algorithm returns a pair of vectors (x^N, z^N) specifying the x- and z-coordinates of the lower-left corner for each placed item. In the next step, the items overlapping the width of the container are removed and remembered in a set V_d''. Later, we will see that the total area of these removed items is not too large. Remember that the NFDH algorithm places groups of items on shelves. To ensure the tall-not-sliced property, these shelves are shifted up such that they start at a height that is an integer multiple of the tallest item on the shelf.

▶ **Lemma 29.** *Algorithm 2 places all vertical items $\mathcal{V}_{d,xz}$ except for V_d'' into the containers $K_{d,xz}$ whose heigths have been extended by $H_{1/\varepsilon} + 1$.*

Proof. First, we prove that it is possible to place all items from V_d' into the area $(w(K) + 2\mu) \times (h(K)+2)$. As the width of the items in $\mathcal{V}_{d,xz}$ is bounded by $w_{\max} \leq \mu$, and their height is bounded by $h_{\max} \leq 1$, it holds that $\mathrm{area}(V_d') \leq \mathrm{area}(K)+\mu = w(K)h(K)+\mu$. By Lemma 8,

---

Algorithm 2 Algorithm to place the vertical items into containers.

1 Input: $K_{d,xz}, \mathcal{V}_{d,xz}$.

2 Sort the items $\mathcal{V}_{d,xz}$ by height, $V_d'' \leftarrow \emptyset$.

3 while $\mathcal{V}_{d,xz} \neq \emptyset$ **do**

4   Choose $K \in K_{d,xz}$, $K_{d,xz} \leftarrow K_{d,xz} \setminus \{K\}$, $V_d' \leftarrow \emptyset$.

5   **while** $\text{area}(V_d') < \text{area}(K)$ *and* $\mathcal{V}_{d,xz} \neq \emptyset$ **do**

6 $i \leftarrow \arg\max_{i \in \mathcal{V}_{d,xz}} h_i$, $V_d' \leftarrow V_d' \cup \{i\}$, $\mathcal{V}_{d,xz} \leftarrow \mathcal{V}_{d,xz} \setminus \{i\}$.

7 $(x^N, z^N) \leftarrow \texttt{NFDH}(V_d', (w(K) + 2\mu) \times (h(K) + 2))$.

8 $V_d'' \leftarrow V_d'' \cup \{i \in V_d' \mid w_i + x_i^N > w(K)\}$.

9   Sequentially shift up all NFDH shelves such that every shelf whose tallest item has a height of $1/q > \varepsilon$ is aligned at an integer multiple of $1/q$. Extend the height of the container as needed.

10 return the set of packings into the different areas and V_d''.

NFDH places any set of items V_d' into an area $w \times h$ if $(w - w_{\max}) \cdot (h - h_{\max}) \geq \text{area}(V_d')$. This condition is fulfilled as $(w(K) + 2\mu - w_{\max}) \cdot (h(K) + 2 - h_{\max}) \geq (w(K) + \mu) \cdot (h(K) + 1) \geq w(K)h(K) + \mu$. Therefore, all the items from V_d' can be placed into the area $(w(K) + 2\mu) \times (h(K) + 2)$.

Next, we show that the algorithm does not run out of containers K before all items in $\mathcal{V}_{d,xz}$ are packed. Let W_d be the total width of 2D items with rounded depth d in $\tilde{I}$. Each of these items corresponds to an item slice with height $\frac{\varepsilon v(I^\infty)}{n}$. Hence, the total area of the items in $\mathcal{V}_{d,xz}$ is bounded by $W_d \cdot \frac{\varepsilon v(I^\infty)}{n}$. On the other hand, as we are given a feasible (ε/μ)-2D-container-packing, we know that the total width of containers for vertical items with rounded depth d is at least W_d. Since each configuration gets assigned a height of at least $\frac{\varepsilon v(I^\infty)}{n}$, it holds that $\text{area}(K_{d,xz}) \geq \text{area}(\mathcal{V}_{d,xz})$. In each step, the algorithm selects a set of items V_d' with $\text{area}(V_d') > \text{area}(K)$, ensuring that containers do not run out before the last item is placed.

Finally, we need to prove that the height $H_{1/\varepsilon} - 1$ is a large enough extension to the height of the boxes to ensure that each shelf whose tallest item has a height of $1/q > \varepsilon$ is aligned to a height that is a multiple of $1/q$. Note that as the items are placed in order of their height, there will not be a gap between shelves containing items of height 1 and items of height 1/2. Therefore, the total height of gaps created between shelves is bounded by $\sum_{i=3}^{1/\varepsilon} 1/i = H_{1/\varepsilon} - \frac{3}{2} \in O(\log(1/\varepsilon))$. ◄

Next, we show that iteratively using Algorithm 2 for all rounded sizes $d \in \mathcal{D}$ gives a packing of all the vertical items into the container and a few additional bins.

▶ **Lemma 30.** *Enlarging the height of the 3D vertical containers by $O(\log(1/\varepsilon))$ allows all vertical items to be placed into their containers and at most $O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n} B + k)$ additional bins while maintaining the tall-not-sliced property.*

Proof. Note that any solution given by Algorithm 2 already fulfills the tall-not-sliced property, as all shelves whose tallest item has a height of $1/q > \varepsilon$ is aligned at an integer multiple of $1/q$. Therefore, any item on such a shelf cannot cross any integer height.

It remains to place the items from $V'' = \bigcup_{d \in \mathcal{D}} V_d''$. For any $d \in \mathcal{D}$ and $K \in K_d$, only items that are placed right of $w(K)$ are added to V_d''. As each placed item has a width of at most μ, a total width of at most 3μ is removed at each height. Regarding the height, we added a height of 2 to each container. So the total height of the 3D configurations is bounded

by $(\frac{\varepsilon v(I^\infty)}{n}B + 3k)$. Further, as we consider an (ε/μ)-2D-container-packing, the total depth of containers for vertical items per configuration is bounded by $O(\varepsilon/\mu)$. Therefore, the items in V'' have a total volume of at most $(O(\varepsilon/\mu) \cdot 3\mu) \cdot (\frac{\varepsilon v(I^\infty)}{n}B + 3k) = O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n}B + k)$. Hence, by Theorem 6, these items can be placed into $O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n}B + k)$ extra bins. ◄

Placing tiny items. The tiny items are placed by iteratively selecting a subset of items and packing them into the containers using the NFDH algorithm. We show that almost all tiny items can be packed into their designated 3D containers, except for a small-volume subset that can be accommodated in at most $O(\delta)\mathrm{OPT}_{\mathrm{2D\text{-}BP}} + O(1)$ additional bins.

Let $\mathcal{S}$ denote the set of 3D items whose base is classified as tiny by the 2D-BP algorithm. Further, let K_S denote the set of containers for tiny items. As the small items are not rounded to a constant number of sizes in any dimension, we will take a different approach than for the horizontal and vertical items. We sort the small items by height and iteratively pick a container $K \in K_S$ to fill the items inside. For this container K, we iteratively choose a set of items $\mathcal{S}_{K,\ell}$, for some $\ell \in \mathbb{N}$, by greedily taking the tallest small items until the top faces of the items have a larger area than the area of the top face of the container. The top-faces $\mathcal{S}_{K,\ell}^{\mathrm{flat}}$ of the items $\mathcal{S}_{K,\ell}$ are then placed into the area $(w(K) + 2\mu) \times (d(K) + 2\mu)$ using the NFDH algorithm. In the next step, the items overlapping the container borders are removed and added to the set $\mathcal{S}''$. The items in $\mathcal{S}_{K,\ell}$ are then actually placed into the container by assigning them their relative xy-position in the box and all of them a height such that their top faces align with the same height h. We call the placement of such a set $\mathcal{S}_{K,\ell}$ a layer and they are numbered starting at 1. After the last layer intersects the height $h(K) + 1$, we close the container. But before we consider the next container, the algorithm ensures the tall-not-sliced property by shifting the layers upwards, such that the z-axis of the layer is aligned with an integer multiple of the tallest item from the layer.

🟨 **Algorithm 3** Algorithm to place the tiny items into containers.

---

1 **Input:** $\mathcal{S}$, K_S.

2 Sort the items $\mathcal{S}$ by height, $\mathcal{S}'' \leftarrow \emptyset$.

3 **while** $\mathcal{S} \neq \emptyset$ **do**

4   Pick $K \in K_S$, $K_S \leftarrow K_S \setminus \{K\}$, $h \leftarrow 0$, $\ell \leftarrow 0$.

5   **while** $h \leq h(K) + 1$ *and* $\mathcal{S} \neq \emptyset$ **do**

6 $\mathcal{S}_{K,\ell} \leftarrow \emptyset$, $\mathcal{S}_{K,\ell}^{\mathrm{flat}} \leftarrow \emptyset$, $\mathcal{S}_{K,\ell}^{\mathrm{area}} \leftarrow 0$, $h \leftarrow h + \max\{h_i | i \in \mathcal{S}\}$.

7   **while** $\mathcal{S}_{K,\ell}^{\mathrm{area}} < w(K) \cdot d(K)$ *and* $\mathcal{S} \neq \emptyset$ **do**

8 $i = \arg\max_{i' \in \mathcal{S}} h_{i'}$, $\mathcal{S}_{K,\ell} \leftarrow \mathcal{S}_{K,\ell} \cup \{i\}$, $\mathcal{S} \leftarrow \mathcal{S} \setminus \{i\}$.

9 $\mathcal{S}_{K,\ell}^{\mathrm{flat}} \leftarrow \mathcal{S}_{K,\ell}^{\mathrm{flat}} \cup \{(w_i, d_i)\}$, $\mathcal{S}_{K,\ell}^{\mathrm{area}} \leftarrow \mathcal{S}_{K,\ell}^{\mathrm{area}} + w_i d_i$.

10 $(\mathbf{x}^N, \mathbf{y}^N) \leftarrow \mathrm{NFDH}(\mathcal{S}_{K,\ell}^{\mathrm{flat}}, (w(K) + 2\mu) \times (d(K) + 2\mu))$.

11 $\mathcal{S}'' \leftarrow \mathcal{S}'' \cup \{i \in \mathcal{S}_{K,\ell} \mid (x_i^N + w_i > w(K)) \vee (y_i^N + d_i > d(K))\}$.

12 $\forall i \in \mathcal{S}_{K,\ell} \setminus \mathcal{S}'' : (x_i, y_i, z_i) \leftarrow (x_K + x_i^N, y_K + y_i^N, h - h_i)$.

13 $h_s \leftarrow 0$.

14   **for** $\ell' = 1$ **to** ℓ **do**   // ensure tall-non-sliced property

15 $i = \arg\max_{i' \in \mathcal{S}_{K,\ell'}} h_{i'}$.

16   **if** $h_i > \varepsilon$ **then**

17 $q \leftarrow 1/h_i$, $q' \leftarrow \lceil q \cdot (h_i + h_s + z_i) \rceil$, $h_s \leftarrow h_s + (q'/q - (h_i + h_s + z_i))$.

18 $\forall i' \in \mathcal{S}_{K,\ell} \setminus \mathcal{S}'' : (x_{i'}, y_{i'}, z_{i'}) \leftarrow (x_{i'}, y_{i'}, q'/q - h_{i'})$.

19 **return** $(\mathbf{x}, \mathbf{y}, \mathbf{z}) \in ([0,1) \times [0,1) \times \mathbb{R}_+)^{|\mathcal{S}|}$, $\mathcal{S}''$.

▶ **Lemma 31.** *Given $\mathcal{S}$ and K_S, Algorithm 3 places all items in $\mathcal{S}$ into the containers K_S whose heights have been extended by $H_{1/\varepsilon} + 1$ except for $\mathcal{S}''$ while fulfilling the tall-not-sliced property.*

Proof. First, we show that all the upper faces of the items in $\mathcal{S}'$ can be placed into the area $(w(K) + \mu) \times (d(K) + \mu)$. By Lemma 8, it is possible to place items with a total area of at least $(w(K) + 2\mu - w_{\max}) \times (d(K) + 2\mu - d_{\max})$ into the area $(w(K) + 2\mu) \times (d(K) + 2\mu)$. As tiny items have a width and a depth that is bounded by μ, NFDH places items with a total volume of at most $w(K)d(K) + \mu(w(k) + d(K)) + \mu^2$. As each tiny item has an upper face with an area of at most μ^2, it holds that $\mathcal{S}'_{\text{area}} \leq w(K)d(K) + \mu^2$ when placing the items into the rectangular area $(w(K) + 2\mu) \times (d(K) + 2\mu)$.

Next, we show that $\mathcal{S} = \emptyset$ before we try to choose a container K from $\emptyset$. As we are given a (ε/μ)-2D-container packing, the total area of tiny items S_{area} in the 2D-BP instance is at most as large as the total area of the containers for small items $K_{S,\text{area}}$. As a consequence, $v(K_S) \geq \frac{\varepsilon v(I^\infty)}{n} K_{S,\text{area}} \geq v(\mathcal{S})$, i.e., the total volume of containers for tiny items is at least as large as the total volume of tiny items. As for each $\mathcal{S}_{K,\ell'}$, in container K, a total area of $w(K)d(K)$ is chosen, it remains to calculate the loss due to the fact that item in $\mathcal{S}_{K,\ell'}$ might have different heights. Let $h_{K,\ell',\max}$ denote the maximum height in $\mathcal{S}_{K,\ell'}$ and $h_{K,\ell',\min}$ the corresponding minimum height. Then, due to the property of NFDH, the total volume we lose in K is bounded by $\sum_{\ell'=1}^{L}(h_{K,\ell',\max} - h_{K,\ell',\min})w(K)d(K) \leq \sum_{\ell'=1}^{L}(h_{K,\ell',\max} - h_{K,\ell'+1,\max})w(K)d(K)$, where we define $h_{K,\ell+1,\max} = 0$. Note that $\sum_{\ell'=1}^{L}(h_{\ell',\max} - h_{\ell'+1,\max})w(K)d(K) = h_{K,1,\max}w(K)d(K) \leq w(K)d(K)$. Note that for each container K that is closed before $\mathcal{S} = \emptyset$ in the algorithm, it holds that $h(K) + 1 < \sum_{\ell=1}^{L} h_{\ell,\max} \leq h(K) + 2$. As a consequence, the algorithm chooses items with a total volume of at least $(h(K) + 1)w(K)d(K) - w(K)d(K) = h(K)w(K)d(K) = v(K)$ to be placed into the container K. As $v(K_S) \geq v(\mathcal{S})$, all items can be placed before the algorithm runs out of containers. Note that the height of each container needs to be extended by at most $2 \in O(\log(1/\varepsilon))$ for this process.

Finally, we need to prove that the height $O(\log(1/\varepsilon))$ is a large enough extension to the height of the boxes to ensure that each layer of items $\mathcal{S}_{K,\ell'}$, whose tallest item has a height of $1/q > \varepsilon$, is aligned to a height that is a multiple of $1/q$. Note that as the items are placed in order of their heights, there will not be a gap between layers containing items of height 1 and items of height $1/2$. Therefore, the total height of gaps created between layers is bounded by $h_s \leq \sum_{q=3}^{1/\varepsilon} 1/q = H_{1/\varepsilon} - \frac{3}{2} \in O(\log(1/\varepsilon))$. Hence, the height of all containers needs to be extended by at most $O(\log(1/\varepsilon))$ to accommodate all layers. ◀

It remains to place the items from $\mathcal{S}''$. The next lemma shows that all the tiny items can be placed into their containers and a few additional bins.

▶ **Lemma 32.** *By enlarging the 3D tiny containers by $O(\log(1/\varepsilon))$ along the height, it is possible to compute a packing of a subset of the tiny items, ensuring the tall-not-sliced property. The remaining tiny items can be packed into $O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n}B + k)$ additional bins.*

Proof. By Lemma 31, we know that given $\mathcal{S}$ and K_S, Algorithm 3 places all items in $\mathcal{S}$ into the containers K_S whose heights have been extended by $H_{1/\varepsilon} + 1 = O(\log(1/\varepsilon))$, except for $\mathcal{S}''$, while fulfilling the tall-not-sliced property. Therefore, it remains to prove that the items from $\mathcal{S}''$ can be placed into $O(\varepsilon)B + O(\varepsilon)(B + k)$ extra bins. Note that per container $K \in K_S$, the volume attributed to $\mathcal{S}''$ is bounded by $3\mu \cdot w(K) \cdot (h(K) + 2) + 3\mu \cdot d(K) \cdot (h(K) + 2) \leq 6\mu(h(K) + 2)$. As the given packing is a (ε/μ)-2D-container packing, there are at most $O(\varepsilon/\mu)$ containers for tiny items per configuration. As a consequence, per configuration C,

we lose a volume of at most $6\mu(h(C)+2) \cdot O(\varepsilon/\mu) = O(\varepsilon)(h(C)+2)$. Hence the total volume of $\mathcal{S}''$ is bounded by $O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n}B + 3k) = O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n}B + k)$. Therefore, the items $\mathcal{S}''$ can be placed in $O(\varepsilon)(B+k)$ additional containers. ◀

These three lemmas about placing big, vertical, horizontal, and tiny items give us the tools to prove Lemma 25.

Proof of Lemma 25. Note that for all item types big, horizontal, vertical, and tiny, the containers have to be increased by a height of at most $O(\log(1/\varepsilon))$. Hence the total height of the 3D configurations is bounded by $\frac{\varepsilon v(I^\infty)}{n}B + O(\log(1/\varepsilon))k$. Further, the large items need at most $|\mathcal{T}|$ extra bins while the horizontal and vertical and small items need at most $O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n}B + k)$ extra bins. We use Theorem 6 to place the medium items into $O(\varepsilon)\text{OPT}_{\text{3D-BP}} + O(1) \le O(\varepsilon)\frac{\varepsilon v(I^\infty)}{n}B + O(1)$ additional bins, as they have a volume of at most $O(\varepsilon)\frac{\varepsilon v(I^\infty)}{n}\text{OPT}_{\text{2D-BP}} \le O(\varepsilon)\text{OPT}_{\text{3D-BP}}(I)$.

In the next step, we transform the packings into the 3D configurations into packings into 3D bins by slicing each configuration at all integer heights. The items sliced by this process all have a height of at most ε, and, hence, we can pack items sliced by $1/\varepsilon$ lines inside a single bin. As the total height of the configurations is bounded by $\frac{\varepsilon v(I^\infty)}{n}B + O(\log(1/\varepsilon))k$, this splitting introduces at most $(1+O(\varepsilon))(\frac{\varepsilon v(I^\infty)}{n}B + O(\log(1/\varepsilon))k)$ new bins in total. Therefore, the total number of bins the algorithm produces is bounded by $(1+O(\varepsilon))(\frac{\varepsilon v(I^\infty)}{n}B + O(\log(1/\varepsilon))k) + |\mathcal{T}| + O(\varepsilon)(\frac{\varepsilon v(I^\infty)}{n}B + k) = (1+O(\varepsilon))\frac{\varepsilon v(I^\infty)}{n}B + O(\log(1/\varepsilon)k + |\mathcal{T}|)$. ◀

By Theorem 24, we can find $\tilde{I}$ and a $(1/\delta^3)$-2D-container-packing of $\tilde{I}$ into B bins with at most $O_\varepsilon(1)$ different bin-configurations that fulfills $B \le (\frac{3}{2} + O(\varepsilon))\text{OPT}_{\text{2D-BP}}(I) + O_\varepsilon(1)$. As a consequence, since $\mu \le \varepsilon\delta^3$, by Lemma 25, we can find a packing of I^∞ into at most $(\frac{3}{2} + O(\varepsilon))\frac{\varepsilon v(I^\infty)}{n} \cdot \text{OPT}_{\text{2D-BP}}(I_{\text{2D-BP}}) + O_\varepsilon(1)$ 3D bins, because $\frac{\varepsilon v(I^\infty)}{n} \le \varepsilon$ and $O(\log(1/\varepsilon)k + |\mathcal{T}|) \in O_\varepsilon(1)$. Since by Lemma 22 and Lemma 23, we have that $\frac{\varepsilon v(I^\infty)}{n}\text{OPT}_{\text{2D-BP}}(I_{\text{2D-BP}}) \le (1+\varepsilon)\text{OPT}_{\text{3D-SP}}(I^\infty) \le (1+\varepsilon)T_\infty\text{OPT}_{\text{3D-BP}}(I)$, the solution generated by the algorithm uses at most $(\frac{3}{2} + O(\varepsilon))T_\infty \cdot \text{OPT}_{\text{3D-BP}}(I) + O_\varepsilon(1)$ bins establishing Theorem 4. ◀

5 Algorithms for 3D-MVBB

In this section, we establish Theorem 3. Note that by the discussion in Section 3, Corollary 21 already implies an absolute approximation ratio of $11/2 + \varepsilon$ for 3D-MVBB, improving on the result of [44]. We now improve the approximation factor to $(3 + \varepsilon)$. First, as previously described, we guess the dimensions of the optimal bounding box within factors of $1 + \varepsilon$ and then scale the dimensions of each item by the corresponding (guessed) dimensions of the bounding box so that all items now fit inside a $1 \times 1 \times 1$ bin.

5.1 Absolute approximation

We first compute a value of μ in order to classify the items using the following lemma.

▶ **Lemma 33.** *There exists a polynomial-time computable $\mu \le \varepsilon$ such that the total volume of the (scaled) items having at least one of the dimensions in the range $(\mu^6, \mu]$ is at most ε.*

Proof. Similar to the proof of Lemma 10, we define $\mu_0 = \varepsilon$, and for $j \in [3/\varepsilon]$, define $\mu_j = \mu_{j-1}^6$. Let I_j be the items with one of the dimensions lying in the range $(\mu_j, \mu_{j-1}]$. Since $v(I) \le 1$, and each item belongs to at most three of the sets $\{I_j\}_{j \in [3/\varepsilon]}$, we have $\sum_{j \in [3/\varepsilon]} v(I_j) \le 3$, and therefore there exists a j^* for which $v(I_{j^*}) \le \varepsilon$. We set $\mu = \mu_{j^*-1}$ and are done. ◀

We now classify the items as follows – let L be the items whose height, width and depth all exceed μ, which we will refer to as *large* items. Let I_h be the items with height at most μ^6, I_w be the remaining items with width at most μ^6 and I_d be the remaining items having depth at most μ^6. Finally, let I^{rem} be the items having at least one dimension in the range $(\mu^6, \mu]$, whose total volume is bounded by ε by Lemma 33. These items are further subdivided as follows: let $I_h^{\mathrm{rem}} \subseteq I^{\mathrm{rem}}$ be the items with height at most μ, $I_w^{\mathrm{rem}} \subseteq I^{\mathrm{rem}} \setminus I_h^{\mathrm{rem}}$ be the items with width at most μ, and $I_d^{\mathrm{rem}} = I^{\mathrm{rem}} \setminus (I_h^{\mathrm{rem}} \cup I_w^{\mathrm{rem}})$ be the remaining items with depth at most μ.

▶ **Lemma 34.** *The items of I_h^{rem} (resp. $I_w^{\mathrm{rem}}, I_d^{\mathrm{rem}}$) can be packed inside a strip with 1×1 base and height (resp. width, depth) at most 12ε.*

Proof. Since $v(I_h^{\mathrm{rem}}) \leq \varepsilon$ and the height of each item in I_h^{rem} is bounded by μ, they can be packed within a height of $4\varepsilon + 8\mu \leq 12\varepsilon$ by Theorem 5. Analogous claims hold for I_w^{rem} and I_d^{rem}. ◀

We now consider the packing of the items in $L \cup I_h$ and obtain the following result.

▶ **Proposition 35.** *In polynomial time, we can compute a packing of $L \cup I_h$ inside a cubic bin of side length $1 + O(\varepsilon)$.*

Proof. We classify the items of I_h based on their width and depth into big, horizontal, vertical and small items as follows – an item i is *big* if both $w_i, d_i > \mu$, *horizontal* if $w_i > \mu$ and $d_i \leq \mu^6$, *vertical* if $d_i > \mu$ and $w_i \leq \mu^6$ and *tiny* if both $w_i, d_i \leq \mu^6$. Using a result of Bansal et al. [8], who gave an APTAS for 2D-BP when the sides of each bin can be augmented by $1 + O(\varepsilon)$, we can discretize the x- and y-coordinates of all items of L and the non-tiny items of I_h.

▶ **Lemma 36** ([8]). *By enlarging the width and depth of the bin by μ, we can ensure that the width and the x-coordinate of any item with $w_i > \mu$ is rounded up to an integer multiple of $\mu^2/2$, and the depth and the y-coordinate of any item with $d_i > \mu$ is rounded up to an integer multiple of $\mu^2/2$.*

In a similar way, by extending the height of the bin by μ, we can discretize the heights of the large items. Also, we can discretize the z-coordinates of all the items.

▶ **Lemma 37.** *Assume μn is an integer so that μ^2 is an integer multiple of μ/n. By enlarging the height of the bin by 2μ, we can assume the following.*
- *The heights of all items are integer multiples of μ/n, and they are positioned at z-coordinates that are integer multiples of μ/n.*
- *The heights of the large items are rounded up to integer multiples of $\mu^2/2$. Further, they are placed at z-coordinates that are integer multiples of $\mu^2/2$.*

Since there are only at most $1/\mu^3$ large items and $O(1/\mu^6)$ possible positions for them inside the bin by Lemmas 36 and 37, we can guess all of them in $O_\mu(1)$ time. Assuming we have guessed all of them correctly, we now have a bin with at most $1/\mu^3$ holes, and our goal is to pack the items of I_h in the space outside the holes. For this, we adopt the same approach as described in Section 4. We slice the items of I_h at heights of μ/n, thus obtaining an instance of 2D-BP, where the bins now have holes. By Lemmas 36 and 37, the positions of the items inside the (enlarged) bins are already discretized, and thus the number of possible 3D configurations is only $O(1/\mu^4)$.

It remains to pack the items of I_h into their respective 3D containers. This is done similar to [35]. First the height of each configuration is extended by the maximum height of an

item in I_h, which is bounded by μ^6. Since the total number of configurations is bounded by $O(1/\mu^4)$, the increase in the height of the packing is at most $O(\mu^2)$. Using a result of Lenstra et al. [42], all the big items can be packed inside the extended big containers (extended by μ^6 along the height) by stacking them one above the other. The horizontal items of a particular rounded width are packed into horizontal containers of the same width using NFDH. For this, the depth of the container is extended by μ^6, and the horizontal items are packed into the extended containers using NFDH. Analogously, the vertical items are packed into extended vertical containers. For packing tiny items into tiny containers, again the width and depth of the container are extended by μ^6, and the items are packed using 3D-NFDH (Algorithm 3).

Finally, it remains to remove the extensions of the containers and pack the items intersecting the extensions at the top of all the configurations. This is done similar to [35], which shows that the extensions can be packed within a height of $O(\mu^2)$. Thus the overall height of the packing of $L \cup I_h$ is at most $1 + 2\mu + O(\mu^2) \leq 1 + O(\varepsilon)$. ◄

As a special case, Proposition 35 applied separately to I_w and I_d also yields packings into cubic bins of side length $1 + O(\varepsilon)$. Together with Lemma 34, we obtain a packing of all input items into a bounding box of volume $3 + O(\varepsilon)$, establishing the absolute approximation guarantee of Theorem 3.

5.2 APTAS

The technique described in the previous subsection (Lemma 37 together with [35]) gives an APTAS for 3D-SP with resource augmentation.

▶ **Lemma 38** ([8, 35]). *For any $\varepsilon > 0$, it is possible to compute a packing into a strip whose width and depth are augmented by a factor of $1 + O(\varepsilon)$ and whose height is at most $(1 + \varepsilon)\mathrm{OPT}_{3D\text{-}SP} + \varepsilon + O_\varepsilon(1)h_{\max}$.*

We run the above algorithm with the base of the strip aligned along the xy-, yz- and zx-planes, respectively, and return the bounding box with the minimum volume. Let δ be a small constant depending on ε. Note that clearly, the height, width, and depth of the optimal bounding box must be at least $h_{\max}, w_{\max}$ and $d_{\max}$, respectively. Therefore, the volume of the optimal box is lower bounded by $h_{\max}w_{\max}d_{\max}$. In the asymptotic case, we can assume $\mathrm{OPT} >> h_{\max}w_{\max}d_{\max}$. Hence, in order to establish an APTAS, we need that the volume of the box returned by our algorithm is at most $(1 + O(\varepsilon))\mathrm{OPT}_{3D\text{-}MVBB} + O_\varepsilon(1)h_{\max}w_{\max}d_{\max}$. We divide the analysis into two cases depending on the values of $h_{\max}, w_{\max}, d_{\max}$.

Case 1: $h_{\max}, w_{\max}, d_{\max}$ all exceed δ - Consider the Strip Packing along the z-axis. Since $w_{\max}, d_{\max} > \delta$, the height of the packing is at most $1 + 2\varepsilon + O_\varepsilon(1) \cdot (1/\delta^2)h_{\max}w_{\max}d_{\max}$. Hence, the volume of the bounding box obtained is also at most $1 + O(\varepsilon) + O_\varepsilon(1)h_{\max}w_{\max}d_{\max}$, and we have an APTAS for this case.

Case 2: At least one out of $h_{\max}, w_{\max}, d_{\max}$ does not exceed δ - W.l.o.g. assume that $h_{\max} \leq \delta$. Then the packing along the z-axis will have a height bounded by $1 + 2\varepsilon + O_\varepsilon(1) \cdot \delta \leq 1 + 3\varepsilon$, for sufficiently small δ. Therefore, the bounding box also has a volume of at most $1 + O(\varepsilon)$, and we, in fact, obtain a PTAS in this case.

Overall, we have the following theorem.

▶ **Theorem 39.** *For any $\varepsilon > 0$, there is a polynomial-time algorithm that returns a packing into a bounding box of volume $(1 + \varepsilon)\mathrm{OPT}_{3D\text{-}MVBB} + O_\varepsilon(1)h_{\max}w_{\max}d_{\max}$.*

6 3D-BP with rotation

In this section, we show the following result.

▶ **Theorem 40.** *There exists a polynomial-time 5-approximation for 3D-BP with rotations.*

Proof. The work of Sharma [55] gave a $(T_\infty^2 + \varepsilon)$-asymptotic approximation for 3D-BP with rotations, building on the algorithm of Caprara [12]. Let K be a large constant so that the algorithm of Sharma gives an absolute approximation ratio of 5 when $\text{OPT}_{\text{3D-BP}} > K$. From now on, we assume $\text{OPT}_{\text{3D-BP}} = k \leq K$ and that we have correctly guessed the value of k. Our goal is to obtain a packing of all items using at most $5k$ bins. Define $\mu = \frac{1}{12^4 K}$ and let L be the items having all dimensions exceeding μ, which we will refer to as the *large* items from now on.

We greedily create at most $4k$ groups of items from $I \setminus L$, where each group has a volume in $[1/4 - 3\mu, 1/4 - 2\mu]$ except for the last one, which can have a smaller volume if it contains all residual items from $I \setminus L$. Finding such groups is possible as one dimension of each item in $I \setminus L$ is bounded by μ, and hence the volume of each item in $I \setminus L$ is bounded by μ. Each such group can be packed into a single bin using Theorem 5 since $4(1/4 - 2\mu) + 8\mu = 1$.

If all items of $I \setminus L$ are already packed into the first $4k$ bins, we pack the items of L into an additional k bins by brute-force enumeration. Thus we obtain a packing of I into $5k$ bins, and are done.

Otherwise, we have packed a volume of at least $(1 - 12\mu)k$ into the first $4k$ bins. Let T denote the set of unpacked items of $I \setminus L$. Note that since $\text{OPT}_{\text{3D-BP}} = k$, we have $v(L \cup T) \leq 12\mu k$.

▷ **Claim 41.** One of the dimensions of each item of L does not exceed $1/12$.

Proof. Since $v(L) \leq 12\mu k \leq 1/12^3$, every large item must have a side of length at most $1/12$. ◀

We orient each large item so that its height is at most $1/12$. Using Theorem 5, the large items can be packed inside a strip with 1×1 base and height $4 \cdot 12\mu k + 8/12 < 3/4$. Finally, the items of T are also packed in a strip of height at most $4 \cdot 12\mu k + 8\mu < 1/4$ using Theorem 5. Thus, the items in $L \cup T$ can all be packed inside a single bin. Overall, we obtain a packing of I into $4k + 1 \leq 5k$ bins and are done. ◀

7 Conclusion

We obtained an improved approximation for 3D-BP, 3D-SP, 3D-MVBB. Our framework is quite general and should extend to other cases. E.g., for the case with rotations, we expect that our techniques should be easily extendable to provide similar asymptotic guarantees. We also expect that the asymptotic approximation algorithm for 3D-BP should extend to d-dimensions ($d > 3$) and provide a $3T_\infty^{d-2}/2$-approximation. The existence of a PTAS (or hardness) for 3D-MVBB is still open. It is also interesting to obtain improved guarantees in pseudopolynomial-time (when the input numeric data is polynomially bounded in n).

References
---

1   ICRA virtual manufacturing automation competition. `https://www.icra2013.org/index0c2d.html?page_id=231`.
2   OPTIL.io. `https://www.optil.io/optilion/problem/3017`.

3   Sara Ali, António Galrão Ramos, Maria Antónia Carravilla, and José Fernando Oliveira. On-line three-dimensional packing problems: A review of off-line and on-line solution approaches. *Computers & Industrial Engineering*, 168:108–122, 2022.

4   Sam D Allen, Edmund K Burke, and Graham Kendall. A hybrid placement strategy for the three-dimensional strip packing problem. *European Journal of Operational Research*, 209(3):219–227, 2011.

5   Helmut Alt. Computational aspects of packing problems. *Bulletin of EATCS*, 1(118), 2016.

6   Helmut Alt and Nadja Scharf. Approximating smallest containers for packing three-dimensional convex objects. *International Journal of Computational Geometry & Applications*, 28(2):111–128, 2018.

7   Nikhil Bansal, Alberto Caprara, and Maxim Sviridenko. A new approximation method for set covering problems, with applications to multidimensional bin packing. *SIAM Journal on Computing*, 39(4):1256–1278, 2010.

8   Nikhil Bansal, José R Correa, Claire Kenyon, and Maxim Sviridenko. Bin packing in multiple dimensions: inapproximability results and approximation schemes. *Mathematics of Operations Research*, 31(1):31–49, 2006.

9   Nikhil Bansal, Xin Han, Kazuo Iwama, Maxim Sviridenko, and Guochuan Zhang. A harmonic algorithm for the 3d strip packing problem. *SIAM Journal on Computing*, 42(2):579–592, 2013.

10   Nikhil Bansal and Arindam Khan. Improved approximation algorithm for two-dimensional bin packing. In *SODA*, pages 13–25, 2014.

11   Eberhard E Bischoff and Michael D Marriott. A comparative evaluation of heuristics for container loading. *European Journal of Operational Research*, 44(2):267–276, 1990.

12   Alberto Caprara. Packing d-dimensional bins in d stages. *Mathematics of Operations Research*, 33(1):203–215, 2008.

13   Miroslav Chlebík and Janka Chlebíková. Hardness of approximation for orthogonal rectangle packing and covering problems. *Journal of Discrete Algorithms*, 7(3):291–305, 2009.

14   Henrik I Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79, 2017.

15   Edward G Coffman, Jr, Michael R Garey, David S Johnson, and Robert Endre Tarjan. Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM Journal on Computing*, 9(4):808–826, 1980.

16   Teodor Gabriel Crainic, Guido Perboli, and Roberto Tadei. Extreme point-based heuristics for three-dimensional bin packing. *INFORMS Journal on Computing*, 20(3):368–384, 2008.

17   Teodor Gabriel Crainic, Guido Perboli, and Roberto Tadei. Ts2pack: A two-level tabu search for the three-dimensional bin packing problem. *European Journal of Operational Research*, 195(3):744–760, 2009.

18   János Csirik and André Van Vliet. An on-line algorithm for multidimensional bin packing. *Operations Research Letters*, 13(3):149–158, 1993.

19   Wenceslas Fernandez de la Vega and George S. Lueker. Bin packing can be solved within 1+epsilon in linear time. *Combinatorica*, 1(4):349–355, 1981.

20   Florian Diedrich, Rolf Harren, Klaus Jansen, Ralf Thöle, and Henning Thomas. Approximation algorithms for 3d orthogonal knapsack. *Journal of Computer Science and Technology*, 23(5):749–762, 2008.

21   Thai Ha Duong. *Heuristics approaches for three-dimensional strip packing and multiple carrier transportation plans*. PhD thesis, University of Nottingham, 2015.

22   Leah Epstein and Rob van Stee. Optimal online bounded space multidimensional packing. In *SODA*, pages 214–223, 2004.

23   Oluf Faroe, David Pisinger, and Martin Zachariasen. Guided local search for the three-dimensional bin-packing problem. *INFORMS journal on Computing*, 15(3):267–283, 2003.

24   Carlos E Ferreira, Flavio K Miyazawa, and Yoshiko Wakabayashi. Packing squares into squares. *Pesquisa Operacional*, 19(2):223–237, 1999.

25   Lisa Fleischer, Michel X Goemans, Vahab S Mirrokni, and Maxim Sviridenko. Tight approximation algorithms for maximum separable assignment problems. *Mathematics of Operations Research*, 36(3):416–431, 2011.

26   Waldo Gálvez, Fabrizio Grandoni, Salvatore Ingala, Sandy Heydrich, Arindam Khan, and Andreas Wiese. Approximating geometric knapsack via L-packings. *ACM Transactions on Algorithms*, 17(4):1–67, 2021.

27   John A George and David F Robinson. A heuristic for packing boxes into a container. *Computers & Operations Research*, 7(3):147–156, 1980.

28   Paul C Gilmore and Ralph E Gomory. Multistage cutting stock problems of two and more dimensions. *Operations Research*, 13(1):94–120, 1965.

29   Rolf Harren. Approximation algorithms for orthogonal packing problems for hypercubes. *Theoretical Computer Science*, 410(44):4504–4532, 2009.

30   Rolf Harren, Klaus Jansen, Lars Prädel, and Rob Van Stee. A $(5/3+\varepsilon)$-approximation for strip packing. *Computational Geometry*, 47(2):248–267, 2014.

31   Rolf Harren, Klaus Jansen, Lars Praedel, Ulrich M Schwarz, and Rob van Stee. Two for one: tight approximation of 2d bin packing. *International Journal of Foundations of Computer Science*, 24(08):1299–1327, 2013.

32   Sören Henning, Klaus Jansen, Malin Rau, and Lars Schmarje. Complexity and inapproximability results for parallel task scheduling and strip packing. *Theory of Computing Systems*, 64(1):120–140, 2020.

33   Mhand Hifi, Imed Kacem, Stéphane Nègre, and Lei Wu. A linear programming approach for the three-dimensional bin-packing problem. *Electronic Notes in Discrete Mathematics*, 36:993–1000, 2010.

34   Klaus Jansen, Arindam Khan, Marvin Lira, and K. V. N. Sreenivas. A PTAS for packing hypercubes into a knapsack. In *ICALP*, pages 78:1–78:20, 2022.

35   Klaus Jansen and Lars Prädel. A new asymptotic approximation algorithm for 3-dimensional strip packing. In *SOFSEM*, pages 327–338, 2014.

36   Klaus Jansen and Lars Prädel. New approximability results for two-dimensional bin packing. *Algorithmica*, 74:208–269, 2016.

37   Klaus Jansen and Malin Rau. Closing the gap for pseudo-polynomial strip packing. In *ESA*, pages 62:1–62:14, 2019.

38   Klaus Jansen and Roberto Solis-Oba. An asymptotic approximation algorithm for 3 d-strip packing. In *SODA*, pages 143–152, 2006.

39   Zhihong Jin, Takahiro Ito, and Katsuhisa Ohno. The three-dimensional bin packing problem and its practical algorithm. *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing*, 46(1):60–66, 2003.

40   Claire Kenyon and Eric Rémila. A near-optimal solution to a two-dimensional cutting stock problem. *Mathematics of Operations Research*, 25(4):645–656, 2000.

41   C. C. Lee and D. T. Lee. A simple on-line bin-packing algorithm. *Journal of the ACM*, 32(3):562–572, 1985.

42   Jan Karel Lenstra, David B Shmoys, and Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46:259–271, 1990.

43   Keoin Li and Kam Hoi Cheng. Heuristic algorithms for on-line packing in three dimensions. *Journal of Algorithms*, 13(4):589–605, 1992.

44   Keqin Li and Kam-Hoi Cheng. On three-dimensional packing. *SIAM Journal on Computing*, 19(5):847–867, 1990.

45   Xueping Li, Zhaoxia Zhao, and Kaike Zhang. A genetic algorithm for the three-dimensional bin packing problem with heterogeneous bins. In *IIE Annual Conference*, page 2039, 2014.

46   Andrea Lodi, Silvano Martello, and Daniele Vigo. Heuristic algorithms for the three-dimensional bin packing problem. *European Journal of Operational Research*, 141(2):410–420, 2002.

47 Yiping Lu, Danny Z Chen, and Jianzhong Cha. Packing cubes into a cube is np-complete in the strong sense. *Journal of Combinatorial Optimization*, 29:197–215, 2015.

48 Daniel Mack and Andreas Bortfeldt. A heuristic for solving large bin packing problems in two and three dimensions. *Central European Journal of Operations Research*, 20:337–354, 2012.

49 Batoul Mahvash, Anjali Awasthi, and Satyaveer Chauhan. A column generation-based heuristic for the three-dimensional bin packing problem with rotation. *Journal of the Operational Research Society*, 69(1):78–90, 2018.

50 Silvano Martello, David Pisinger, and Daniele Vigo. The three-dimensional bin packing problem. *Operations Research*, 48(2):256–267, 2000.

51 Aram Meir and Leo Moser. On packing of squares and cubes. *Journal of Combinatorial Theory*, 5(2):126–134, 1968.

52 Flavio Keidi Miyazawa and Yoshiko Wakabayashi. An algorithm for the three-dimensional packing problem with asymptotic performance analysis. *Algorithmica*, 18(1):122–144, 1997.

53 Flavio Keidi Miyazawa and Yoshiko Wakabayashi. Packing problems with orthogonal rotations. In *LATIN*, pages 359–368, 2004.

54 Francisco Parreño, Ramón Alvarez-Valdés, José Fernando Oliveira, and José Manuel Tamarit. A hybrid grasp/vnd algorithm for two-and three-dimensional bin packing. *Annals of Operations Research*, 179:203–220, 2010.

55 Eklavya Sharma. Harmonic algorithms for packing d-dimensional cuboids into bins. In *FSTTCS*, pages 32:1–32:22, 2021.

56 A Steinberg. A strip-packing algorithm with absolute performance bound 2. *SIAM Journal on Computing*, 26(2):401–409, 1997.

57 Tony Wauters, Jannes Verstichel, and Greet Vanden Berghe. An effective shaking procedure for 2d and 3d strip packing problems. *Computers & Operations Research*, 40(11):2662–2669, 2013.

58 Lijun Wei, Wee-Chong Oon, Wenbin Zhu, and Andrew Lim. A reference length approach for the 3d strip packing problem. *European Journal of Operational Research*, 220(1):37–47, 2012.

A   Structural result for 2D-BP

In this section, we focus on the 2D-BP algorithm, and we restate all the ingredients to prove the main structure lemma for 2D-BP from [36].

▶ **Theorem 24** ([36]). *Given some instance I for 2D-BP, there is an algorithm that finds in polynomial time a rounded-up instance $\tilde{I}$ with $|\mathcal{T}|, |\mathcal{D}|, |\mathcal{W}| \in O(1/\delta^4)$ and a $(1/\delta^3)$-2D-container-packing of $\tilde{I}$ into B bins with at most $O_\varepsilon(1)$ different bin-configurations that fulfills $B \leq (\frac{3}{2} + O(\varepsilon))\mathrm{OPT}_{2D\text{-}BP}(I) + O_\varepsilon(1)$.*

The proof works in three steps. First, we prove that a suitable rounding of the items exists, and then we prove that each packing of the rounded items can be rearranged into a $(1/\delta^3)$-2D-container-packing. Finally, we present an algorithm to find such a $(1/\delta^3)$-2D-container-packing.

A.1   Existence of a suitable rounding for the items in 2D-BP

To round the items, it will become necessary to slice horizontal items horizontally (along x-axis) and vertical items vertically (along y-axis). Later, when constructing the container packing, it will become necessary to slice the tiny items in any direction. We denote by $\mathrm{OPT}_{2D\text{-}BP}^{\mathrm{slice}}(I)$ an optimal solution for a 2D-BP instance I, where we allow this kind of slicing of the items. In this section, we prove the following lemma, which states that there exists a suitable rounding of the items of the given 2D-BP instance.

▶ **Lemma 42.** *Given $\varepsilon > 0$ and a set of items I and an optimal packing into $\mathrm{OPT}_{2D\text{-}BP}(I)$ bins, we can find a rounded instance $\tilde{I}$ where the big items are rounded into $1/\delta^4$ types $\mathcal{T}$ such that either the width or the depth is a multiple of δ^2, the vertical items have $O(1/\varepsilon \cdot \log(1/\delta))$ different depths $\mathcal{D}$, and the horizontal items have at most $O(1/\varepsilon \cdot \log(1/\delta))$ different widths $\mathcal{W}$, such that $\mathrm{OPT}_{2D\text{-}BP}^{\mathrm{slice}}(\tilde{I}) \leq ((3/2) + O(\varepsilon))\mathrm{OPT}_{2D\text{-}BP}(I) + O(1)$. The rounded instance $\tilde{I}$ is one of $n^{O_\varepsilon(1)}$ possibilities to round the items.*

First, note that we can find the right value δ to classify the items by size in polynomial time. Further, we can assume that $1/\varepsilon$ and $1/\delta$ are integer values.

▶ **Lemma 43** ([36])**.** *There exists a polynomial-time computable $\delta \in [\varepsilon^{O_\varepsilon(1)}, \varepsilon]$ such that the total area of items with width in $[\delta^4, \delta)$ or depth in $[\delta^4, \delta)$ is bounded by $\varepsilon \cdot \mathrm{area}(I)$ and $1/\delta$ is a multiple of 24.*

Rounding vertical and horizontal items

Rounding the width of horizontal items and the height of vertical items is classically done by linearly or geometrically grouping the items. Indeed, this is also possible when considering 2D-BP. For completeness, we add the proof here. The rounding will lead to a total area of at most $O(\varepsilon)\mathrm{OPT}_{2D\text{-}BP}$ of vertical and horizontal items that cannot be placed inside the original bins. The main work is to argue that these items can be placed into $O(\varepsilon)\mathrm{OPT}_{2D\text{-}BP} + O(1)$ additional bins.

Note that the rounding can be adjusted such that all the slices of a 3D item whose top faces have been classified as vertical or horizontal are rounded to the same depth or height, respectively.

▶ **Lemma 44.** *Let any packing of the items I into B bins be given. Assume we are allowed to slice the vertical items vertically and the horizontal items horizontally. Then we can find a packing into at most $B + O(\varepsilon\mathrm{OPT}_{2D\text{-}BP}) + O(1)$ bins, where the depths of vertical and widths of horizontal items are rounded to $O(\frac{1}{\varepsilon}\log(\frac{1}{\delta}))$ sizes.*

Proof. The main idea to round the items is to use geometric grouping, introduced by [19]. We do the rounding exemplarily for horizontal items, but the rounding for vertical items works analogously when flipping depth and width. The first step is to group the horizontal items by width into groups $G_i := \{i \in I | d_i < \delta, 1/2^i < w_i \leq 1/2^{i-1}]\}$, for each $i \in \{1, 2, \ldots, \lceil \log(1/\delta) \rceil\}$. In the next step, for each group G_i, a linear grouping step rounds the width in this group to $2/\varepsilon$ sizes: All items in group G_i are sorted by width and stacked on top of each other such that the widest item is at the bottom. Let $h(G_i)$ be the total depth of this stack, and let $\sigma(i)$ denote the depth at which the item starts in the stack. The stack is now partitioned into subgroups $G_{i,j}$ for each $j \in \{1, \ldots, \lceil 2/\varepsilon \rceil\}$ such that $G_{i,j}$ contains all items with $(j-1)\varepsilon h(G_i)/2 < \sigma(i) + i_h \leq j\varepsilon h(G_i)/2$. Items overlapping some multiple of $j\varepsilon h(G_i)/2$ are sliced at this line. In the rounding step, the widths of all the items in a group $G_{i,j}$ are rounded to the width of the widest item in the group, except for the items that have the smallest width inside the group, which will keep their width, if it is the same as the widest width in the next group $G_{i,j+1}$. Note that when allowing to horizontally slice items, the rounded items from group $G_{i,j}$, for each $j > 1$, can be placed instead of the rounded items from group $G_{i,j-1}$, since the widest original width in group $G_{i,j-1}$ has a width that is at least as large as the rounded width of each item in $G_{i,j}$.

However, the items from the group $G_{i,1}$ cannot be placed in the place of other items from any other group. These items will be placed into new bins. Note that the items from group $G_{i,1}$ have a width of at most $\frac{1}{2^{i-1}}$. Therefore, it is possible to place 2^{i-1} of them next to

each other in one bin. As we are allowed to slice these items horizontally, we can pack them into a container of width 1 and depth $\frac{\varepsilon h(G_i)}{2} \cdot \frac{1}{2^{i-1}} = \frac{\varepsilon h(G_i)}{2^i}$. Note that the total area of the items in group G_i is at least $\frac{\varepsilon h(G_i)}{2^i}$. Therefore, the total depth of all these containers for groups G_i is bounded by

$$\sum_{i=1}^{\lceil \log(1/\delta) \rceil} \frac{\varepsilon \cdot h(G_i)}{2^i} \leq \varepsilon \sum_{i=1}^{\lceil \log(1/\delta) \rceil} \frac{h(G_i)}{2^i} \leq \varepsilon \cdot \text{area}(I) \leq \varepsilon \cdot \text{OPT}_{\text{2D-BP}}.$$

As we are allowed to slice the horizontal items horizontally, we can fill each bin to the top and therefore introduce at most $\lceil \varepsilon \text{OPT}_{\text{2D-BP}} \rceil$ new bins.   ◀

Rounding big items

Rounding the big items presents a bigger challenge compared to rounding the horizontal or vertical items. To round the big items, the entire packing has to be rearranged to allow one side of the items to be rounded to a multiple of $\delta^2/2$. The main ingredient for the structural result for 2D-BP is the following restructuring theorem, which states that each bin can be restructured to have one of two properties, while not increasing the number of used bins too much.

▶ **Property 45.** *The width and the x-coordinate of each item in the bin of width at least δ is a multiple of $\delta^2/2$.*

▶ **Property 46.** *The depth and the y-coordinate of each item in the bin of depth at least δ is a multiple of $\delta^2/2$.*

▶ **Theorem 47** ([36]). *For any δ, with $1/\delta$ being a multiple of 24, and for any solution that fits into m bins, we are able to round up the widths or the depths of the rectangles so that they fit into $(3/2 + 5\varepsilon)m + 37$ bins, while the packing of each of the bins satisfies either Property 45 or Property 46.*

Property 45 and Property 46 allow one side of the items to be rounded to a multiple of δ^2. The main idea behind rounding the other side is to use linear grouping for each rounded width and each rounded depth.

▶ **Lemma 48.** *Consider a packing into B bins where each bin fulfills Property 45 or Property 46. By adding at most $O(\varepsilon)\text{OPT}_{\text{2D-BP}}(I) + O(1)$ bins, the sizes of the big items can be rounded to at most $O(1/(\delta^3 \varepsilon))$ sizes.*

Proof. First, the big items are partitioned into groups. Let L_i^w denote that set of big items where the width has been rounded to $i\delta^2/2$ for some $i \in \mathbb{N}$, where $i < 2/\delta^2$. Similarly, let L_i^d denote the set of big items where the depth has been rounded to $i\delta^2/2$ for some $i \in \mathbb{N}$, where $i < 2/\delta^2$. In the next step, within each group L_i^w, the depths are rounded using linear grouping, while within each group L_i^d, the widths are rounded using linear grouping.

Consider the set L_i^d for some $i \in \mathbb{N}$. Sort set L_i^d by width and partition it onto $\lceil 2/(\varepsilon\delta) \rceil$ groups $L_{i,1}^h, \ldots, L_{i,\lceil 2/(\varepsilon\delta) \rceil}^h$, such that each group contains $\lfloor \varepsilon\delta |L_i^d|/2 \rfloor$ items, except for the last group $L_{i,\lceil 2/\delta^2 \rceil}^h$ containing the items with the smallest widths. This group can contain fewer items. In each group, the widths are rounded to the size of the largest width in that group. Note that in the given packing, for each group, the rounded items can be placed in place of the original items in the group with the next larger width. The last group of each

has L_i^d will be placed into new bins. Note that as there are at most $\lfloor \varepsilon\delta|L_i^d|/2 \rfloor$ items in the set, and each has a width of at most 1 and a depth of $i\delta^2/2$ it holds that

$$\text{area}(L_{i,1}^h) \leq \lfloor \varepsilon\delta|L_i^d| \rfloor \cdot i\delta^2/2 \leq \varepsilon \cdot \text{area}(L_i^d).$$

Note that the last inequality holds as each item in L_i^d has a width of at least δ.

We get a similar result when rounding the items in L_i^w. As a consequence, the total area of the items that have to be placed into additional bins is bounded by $\varepsilon \cdot \text{area}(L)$, where L is the set of big items. We use Steinberg's algorithm [56] to place the items into a box with width 1 and depth $2\varepsilon \cdot \text{area}(L)$. This box is then cut at every integer depth to create bins from the box. The items cut by the line will be placed into extra bins. Therefore, we can place these items into at most $\lceil 4\varepsilon\text{area}(L) \rceil \leq \lceil 4\varepsilon\text{OPT}_{\text{2D-BP}}(I) \rceil$ bins.   ◄

Note that when just considering the items, we do not know which of them are rounded to have widths that are multiples of δ^2 and which of them are rounded to have depths that are multiples of δ^2. In [36], it is shown that there is only a constant number of possible rounding for big items, and one of them corresponds to the one derived by the method in Lemma 48.

▶ **Lemma 49** ([36]). *There is an algorithm that iterates at most $n^{O_\varepsilon(1)}$ different roundings for the big items, where one of them is guaranteed to be the same, that can be derived from the optimal solution.*

The main idea to find this rounding is to iterate over the possible subsets of items that are the $O(1/\delta^4)$ items responsible for the rounded width or depth in the linear grouping step. Further, for each rounded depth and width that is a multiple of δ^2, the total number of items is guessed that are rounded this way. After this guessing step, the big items are assigned to the rounded classes using a flow network. As there are only $n^{O_\varepsilon(1)}$ possible choices, the algorithm needs to iterate over at most this many number of different roundings.

Proof of Lemma 42. By Lemma 44, we can round the depths for vertical items and the widths for horizontal items to $O((1/\varepsilon)\log(1/\delta))$ different sizes while increasing the number of used bins by only $O(\varepsilon\text{OPT}_{\text{2D-BP}}) + O(1)$. Further, by combining Theorem 47 and Lemma 48, we know that the big items can be rounded to at most $O(1/\delta^4)$ types while increasing the number of bins by at most $(\frac{1}{2} + O(\varepsilon))\text{OPT}_{\text{2D-BP}} + O(1)$. This proves that the items can be rounded as described in the Lemma statement while maintaining $\text{OPT}_{\text{2D-BP}}^{\text{slice}}(\tilde{I}) \leq ((3/2) + O(\varepsilon))\text{OPT}_{\text{2D-BP}}(I) + O(1)$. Finally, Lemma 49 guarantees that the right rounding can be computed in polynomial time.   ◄

A.2   Existence of a container-based packing for 2D-BP

We define a *preliminary container-based packing* as follows. Each bin is partitioned into at most $O(1/\delta^3)$ rectangular regions called containers. Further vertical items may be sliced vertically, horizontal items may be sliced horizontally, and tiny items may be sliced in both directions. The containers have the following properties:

- There are at most $1/\delta^2$ container containing only one big item i and having size of i.
- There are at most $3/\delta^3$ containers that contain only horizontal and tiny items.
- There are at most $3/\delta^3$ containers that contain only vertical and tiny items.
- The container contains only intermediate items and spans the complete bin.

▶ **Lemma 50.** *Consider a packing of items I into B bins, where each bin satisfies either Property 45 or Property 46. By adding at most $O(\varepsilon)B + O(1)$ bins, the packing can be transformed into a preliminary container-based packing.*

Proof. In the first step, we remove all the intermediate items from the bin. Note that these items have a total area of at most $\varepsilon\mathrm{OPT}_{\text{2D-BP}}(I)$. We use Steinberg's algorithm [56] to place the items into a bin of width 1 and depth at most $2\varepsilon\mathrm{OPT}_{\text{2D-BP}}$. This box is then partitioned into $\lceil 2\varepsilon\mathrm{OPT}_{\text{2D-BP}}\rceil$ boxes of depth 1. The items on each cut are placed into an extra bin. Therefore, the total number of bins added is bounded by $2\lceil 2\varepsilon\mathrm{OPT}_{\text{2D-BP}}(I)\rceil \leq O(\varepsilon)B + O(1)$.

Next, we describe how to partition the bins into containers. Consider a bin that satisfies Property 45, i.e., the width and the x-coordinate of each item with width at least δ has a width that is a multiple of $\delta^2/2$. First, we introduce containers for the big items. Each big item gets its own container and it is positioned exactly where the big item is placed. As each bin can contain at most $1/\delta^2$ of these items, the total number of these containers is also bounded by $1/\delta^2$.

Next, we consider all vertical strips of width $\delta^2/2$ from left to right. For each of these strips, we create at most $1/\delta$ containers for vertical items, with their left and right borders defined by the strip's boundaries. To introduce the depth-aligned borders parallel to the x-axis, we iterate the packing from depth 0 to depth 1. At the smallest depth where a vertical item is first encountered, we open a new container by placing a border parallel to the x-axis at this depth. The other border of this container is then set parallel to the x-axis at the next smallest depth where the scan encounters a big or a horizontal item. Once a big or horizontal item is encountered, the scan continues until either depth 1 is reached or another vertical item appears. In the latter case, a new container is opened, and the process is repeated. This method ensures that no horizontal item is sliced, and vertical items are only sliced vertically along box borders, as all big and horizontal items are aligned to the $\delta^2/2$ grid with their width. As vertical items have a depth of at least δ, the total number of containers for vertical items created this way is bounded by $\frac{1}{\delta}\cdot\frac{2}{\delta^2}=\frac{2}{\delta^3}$.

Finally, we create containers for the horizontal items. First, note that at this point, all vertical and big items are covered by containers. Therefore, we only need to partition the residual area into rectangular containers. To do that, we consider the y-coordinates of all boxes of containers for vertical and big items. We expand these upper and lower borders to the left and right until they meet another container. These horizontal lines partition the residual area into rectangular containers. As they do not intersect containers for vertical or big items, no vertical item is sliced horizontally by these containers. Note that each of these horizontal lines is adjacent to at most 3 containers for horizontal items. As the total number of these lines is bounded by $4/\delta^3$ and each container requires two lines (upper and lower border) to be complete, there are at most $6/\delta^3$ containers for horizontal items.

Note that this partition can be done analogously to any container that fulfills Property 46 by changing the role of vertical and horizontal containers.   ◄

In the next step, the containers for vertical and horizontal items are divided into sub-containers, such that they only contain one type of item. This will reduce the total number of possible container configurations further.

Recall the definition of a *k-2D-container-packing*: Each bin of the packing is partitioned into containers. The containers are of five types, and only specific types of items from $\tilde{I}$ are allowed to be packed in the corresponding containers:

- (i) *Big container:* Each such container contains only one (rounded up) big item and has the size of this big item.
- (ii) *Horizontal containers:* each such container has a width $w \in \mathcal{W}$ and a depth that is a multiple of μ, and contains only horizontal items with width w. The total width of these containers per bin is bounded by $O(k)$.

- (iii) *Vertical containers:* each such container has a depth $d \in \mathcal{D}$, a height that is a multiple of μ, and contains only vertical items with depth d. The total depth of these containers per bin is bounded by $O(k)$.

- (iv) *Tiny containers:* contain only tiny items and have a width and depth that is a multiple of μ. Each bin contains at most $O(k)$ of these containers.

- (v) *Intermediate containers:* These containers contain only intermediate items. There will be an extra $O(\varepsilon)\mathrm{OPT}_{\text{2D-BP}}$ bins reserved separately to pack the intermediate items.

In the following, we prove that such a *k-2D-container-packing* can be generated from a preliminary container-based packing. Further, note that Property 45 implicitly states that each item that has a width larger than δ is further rounded up to be a multiple of $\delta^2/2$ if property. We reverse this rounding to multiples of $\delta^2/2$ for the horizontal items and vertical items (for Property 46). Instead, for them, we consider the rounded width from the first rounding step in Lemma 44.

▶ **Lemma 51.** *Consider a preliminary container-based packing into B bins. By adding at most $O(\max\{\varepsilon, \mu/\delta^3\} \cdot B)$ bins, the packing can be transformed into a $(1/\delta^3)$-2D-container-packing for $\tilde{I}$.*

Proof. Assume we are given a preliminary container-based packing. The containers for the big items already have the desired properties. Therefore, this proof focuses on partitioning the other containers into smaller containers for vertical, horizontal, and tiny items.

Horizontal Containers. Consider the containers for horizontal items. First, we round down the depths of these containers to multiples of μ. As there are only $O(1/\delta^3)$ containers per bin, rounding down the depth of each container for horizontal items in this bin to the next multiple of μ removes a total depth of at most $O(\mu/\delta^3)$ from all containers of this bin combined. In this process, we slice some of the horizontal items horizontally. The total depth of the removed parts over all bins adds up to at most $O(\mu/\delta^3)B$. We create for these removed parts $O(\mu/\delta^3)B \geq 1$ new containers that each fill a complete bin.

In the next step, we round the widths of these containers using geometric grouping. We round the widths to $\frac{2}{\varepsilon} \cdot \log(\frac{1}{\delta})$ different sizes, using the same process as to round the horizontal items. Due to the rounding, we need to place some of the containers into new bins. Note that the total area of the containers that need to be placed into new bins is bounded by $O(\varepsilon)B$, as they have a total area of at most B. As a consequence, we introduce at most $O(\varepsilon)B \geq 1$ new bins that each are completely filled by a container for horizontal items. Denote by K_H the set of rounded containers.

To partition the containers K_H into sub-containers for the 2D-container-based packing, we use a linear program. Recall that $\mathcal{W}$ denotes the set of rounded widths for horizontal items. Let $\mathcal{W}_K$ denote the set of rounded widths of the containers K_H. For each $w \in \mathcal{W}$, denote by $d_{H,w}$ the total depth of horizontal items with rounded width w and for each $w' \in \mathcal{W}_K$ denote by $d_{K,w'}$ the total depth of containers with rounded width w'. Further, for each $w' \in \mathcal{W}_K$, a *horizontal arrangement* C_H denotes a multiset of rounded horizontal item widths from $\mathcal{W}$ that adds up to at most w', i.e., these items fit next to each other into a container of width w'. Let $\mathcal{C}_{w'}$ be the set of all corresponding arrangements. For a given arrangement $C_H \in \mathcal{C}_{w'}$ and rounded horizontal item width $w \in \mathcal{W}$, we denote by $C_H(w)$ the multiplicity of width w in the multiset C_H. The variable x_{C_H} represents the depth of

arrangement $C_H \in \mathcal{C}_{w'}$. The configuration LP for horizontal items has the following form:

$$\sum_{C_H \in \mathcal{C}_{w'}} x_{C_H} = d_{K,w'} \qquad\qquad \forall w' \in \mathcal{W}_K$$

$$\sum_{w' \in \mathcal{W}_K} \sum_{C_H \in \mathcal{C}_{w'}} x_{C_H} \cdot C_H(w) = d_{H,w} \qquad\qquad \forall w \in \mathcal{W}$$

$$x_{C_H} \geq 0 \qquad\qquad \forall C_H \in \bigcup_{w' \in \mathcal{W}_K} \mathcal{C}_{w'}$$

Note that a solution to the LP exists, as the transformed packing into the containers K_H can be sliced into horizontal strips at the borders (that are parallel to the x-axis) of all horizontal item slices in the containers. The items contained in a strip represent one (horizontal) arrangement C_H for the (horizontal) container. If we set the depth of this arrangement x_{C_H} to the sum of the depths of all strips containing exactly this set of items in containers with the same rounded depth, we get a solution to the above LP. A basic solution to the LP has at most $|\mathcal{W}| + |\mathcal{W}_K| \leq O(\log(1/\delta)/\varepsilon) \leq O(1/\delta^2)$ non-zero components and can be found in $O_\varepsilon(1)$ time.

We use the solution to the LP to fill the containers for horizontal items. First, we increase the depth of each arrangement x_{C_H} to the next larger multiple of μ and introduce for each arrangement a new container of depth μ that will be placed into an extra bin. As there are at most $O(\log(1/\delta)/\varepsilon)$ arrangements, the total depth of added horizontal containers is bounded by $O(\mu \log(1/\delta)/\varepsilon) \leq O(\delta^2)$. We iterate over the arrangements and fill them into the containers until they reach the depth of the container or are completely packed. If one container is full, we continue to fill the next one. If the arrangement is completely packed, but the container is not full, the container is split horizontally at that point, and we continue to fill the next fitting arrangement into this container. As there are at most $O(\frac{1}{\delta^2})$ arrangements, we add at most $O(\frac{1}{\delta^2})$ new containers for horizontal items by this step. Note that the total number of horizontal containers per bin is still bounded by $O(1/\delta^3)$ and hence their total width is bounded by $O(1/\delta^3)$.

In the next step, we partition each horizontal container into at most $1/\delta$ sub-containers. Consider a container with width $w' \in \mathcal{W}_K$ and depth d that is filled with the arrangement C_H. As the arrangements specify a set of horizontal items that can be placed next to each other inside the container, we partition the container vertically into sub-containers, each of depth d, such that, for each width $w \in C_H$, there is one sub-container that has width w. We arrange the sub-containers such that they are placed next to each other. Not that after this step, the total width of sub-containers for horizontal items is still bounded by $O(1/\delta^3)$.

Note that the widths in the arrangement might add up to something smaller than the width of the original container. This leaves a free space where a container for tiny items can be placed. This container will get the depth of the container, which is a multiple of μ, and a width that is the largest multiple of μ that still fits next to the sub-containers for horizontal items. This step introduces at most $O(1/\delta^3)$ container for small items.

Vertical Containers. Note that we can partition the containers for vertical items into sub-containers similarly as for the horizontal items by just swapping width and depth. This creates $O_\varepsilon(1)$ (sub-)containers for vertical items that have a total depth of at most $O(1/\delta^3)$, have a width that is a multiple of μ and a height $d \in \mathcal{D}$.

Tiny Containers. As mentioned before, in each container for horizontal items and each container for vertical items, we introduce a container for tiny items with width and depth that is a multiple of μ. Hence, their number is bounded by $O(1/\delta^3)$. Note that the totally free area inside the containers for horizontal and vertical items is at least as large as the

total area of the tiny items because the LP solution covers exactly the area of the horizontal (and vertical) items. Due to the rounding of the width of sub-containers for tiny items in the containers for horizontal items, we lose an area of at most $O(\mu/\delta^3)$ per bin, as there are at most $O(1/\delta^3)$ container for horizontal items per bin with a total depth of at most $O(1/\delta^3)$. Similarly, in containers for vertical items, we lose a total area of $O(\mu/\delta^3)$ for tiny items. Therefore, a total area of at most $O(\mu/\delta^3)B$ of the tiny items that were contained in the containers for horizontal items cannot be covered by the newly introduced containers for tiny items. Therefore, we can use $O(\mu/\delta^3)B$ new bins to cover the residual area for tiny items. ◀

To find a *k-2D-container-packing* for all items, it is a useful tool to have an algorithm that can decide for a set of items if these items can be placed into one bin, that fulfills the requirements to be a bin of a *k-2D-container-packing*. In the next step, we prove that such a decision can indeed be made in polynomial time.

▶ **Lemma 52.** *Given a set of rounded items I, it is possible to decide in $O(n) + O_{\delta,\mu,k}(1)$ time if this set of items can be placed into a k-2D-container-packing that consists of one bin.*

Proof. First, note that for each rounded horizontal item width $w \in \mathcal{W}$, the total height of items of this width has to be bounded by k. Otherwise, the items have a total area larger than 1 and trivially do not fit into one bin. Furthermore, the number of containers for horizontal items is bounded by k/δ as each container has a width of at least δ. As all containers for horizontal items have a depth that is a multiple of μ and there are at most k/δ containers, there are only $O_{\delta,\mu,k}(1)$ possible choices for all the container heights of containers covering the horizontal items with width w. Similarly, for each vertical item depth $\in \mathcal{D}$, there are only $O_{\delta,\mu,k}(1)$ possibilities to cover these items with containers. Further, the containers for tiny items have depths and widths that are multiples of μ, and per bin, there exist at most $O(k)$ of them. Therefore, there are only $O_{\delta,\mu,k}(1)$ possibilities to choose container sizes for tiny items.

Given a choice for container sizes for big, horizontal, vertical, and tiny items, we can check whether these containers can be placed into one bin in $O_{\delta,\mu,k}(1)$ time, as there are at most $O_{\delta,\mu,k}(1)$ container. ◀

A.3 Algorithm to find a container-based packing for 2D-BP

A *2D bin configuration* $C \in \mathbb{N}^{|\mathcal{T}|+|\mathcal{D}|+|\mathcal{W}|+1}$ is a vector that specifies a set of items that can be placed into one bin of a 2D-container-packing. This vector specifies for each big item type $t \in \mathcal{T}$, the number $n(t, C)$ of items of this type in the container; for each horizontal item width $w \in \mathcal{W}$, the number $D(w, C)$ where the total depth of containers for these items is $D(w, C) \cdot \delta^4$; for each vertical item depth $d \in \mathcal{D}$, the number $V(d, C)$ where the total width of containers for these items is $V(d, C) \cdot \delta^4$; and for tiny items, the number $S(C)$ where the total area of containers for tiny items is $S(C) \cdot \delta^8$. Let $\mathcal{C}$ be the set of all 2D bin configurations. Then $|\mathcal{C}|$ is $O_\varepsilon(1)$ as the sum of entries has an upper bound of $O(1/\delta^8)$. By Lemma 52, we can verify in polynomial time if a given vector translates to a configuration C.

To find a 2D-container-packing, the algorithm from [36] utilizes an integer program, to find $\mathrm{OPT}_{2\text{D-BP}}^C(\tilde{I})$. However, for our purposes, the linear programming (LP) relaxation of this program is sufficient. For each big item type $t \in \mathcal{T}$, let n_t denote the number of items with this type. Similarly, let W_d denote the total width of vertical items with rounded depth $d \in \mathcal{D}$, and D_w denote the total depth of horizontal items with rounded width $w \in \mathcal{W}$. Further, let S_{area} denote the total area of tiny items in I. We introduce variables x_C that

denote for each configuration $C \in \mathcal{C}$, the *amount* of this configuration in the solution, e.g., if $x_C = 2.7$, the configuration C appears 2.7 times in the optimal LP solution. The LP is defined as follows.

$$\min \sum_{C \in \mathcal{C}} x_C$$

$$\text{s.t.} \quad \sum_{C \in \mathcal{C}} n(t, C) x_C \geq n_t \qquad \forall t \in \mathcal{T}$$

$$\sum_{C \in \mathcal{C}} V(d, C) \delta^4 x_C \geq W_d \qquad \forall d \in \mathcal{D}$$

$$\sum_{C \in \mathcal{C}} D(w, C) \delta^4 x_C \geq D_w \qquad \forall w \in \mathcal{W}$$

$$\sum_{C \in \mathcal{C}} S(C) \delta^8 x_C \geq S_{\text{area}}$$

$$x_C \geq 0 \qquad \forall C \in \mathcal{C}$$

Let $\text{OPT}_{\text{2D-BP}}^{LP}(\tilde{I})$ denote the optimal solution for the LP. A basic solution to this LP uses only $O_\varepsilon(1)$ different configurations, as there are only $O_\varepsilon(1)$ types of containers in total.

▶ **Lemma 53.** $\text{OPT}_{2D\text{-}BP}^{LP}(\tilde{I}) \leq \text{OPT}_{2D\text{-}BP}^{C}(\tilde{I})$ *and in* $n^{O_\varepsilon(1)}$ *time a basic solution* x *with* $\sum_{C \in \mathcal{C}} x_C = \text{OPT}_{2D\text{-}BP}^{LP}(\tilde{I})$ *can be found.*

Proof. We show that $\text{OPT}_{2D\text{-}BP}^{LP}(\tilde{I}) \leq \text{OPT}_{2D\text{-}BP}^{C}(\tilde{I})$ and in $O_\varepsilon(1)$ time a basic solution x with $\sum_{C \in \mathcal{C}} x_C = \text{OPT}_{2D\text{-}BP}^{LP}(\tilde{I})$ can be found. Each 2D-container-packing can be transformed to be a solution to the LP by creating the corresponding configuration for each bin and adding one to the entry of x corresponding to the constructed configuration, implying $\text{OPT}_{2D\text{-}BP}^{LP}(\tilde{I}) \leq \text{OPT}_{2D\text{-}BP}^{C}(\tilde{I})$. The number of variables is in $O_\varepsilon(1)$ and the number of constraints is in $O_\varepsilon(1)$. Therefore, a basic optimal solution can be found in $O_\varepsilon(1)$ time, e.g., even by using the simplex algorithm. ◀

Now we have all the tools to prove Theorem 24. In the proof, we discuss the algorithm that given some instance I for 2D-BP, finds in polynomial time a rounded instance $\tilde{I}$ with $|\mathcal{T}|, |\mathcal{D}|, |\mathcal{W}| \in O(1/\delta^4)$ and a $(1/\delta^3)$-2D-container-packing of $\tilde{I}$ into B bins with at most $O_\varepsilon(1)$ different bin-configurations that fulfills $B \leq (\frac{3}{2} + O(\varepsilon))\text{OPT}_{\text{2D-BP}}(I) + O_\varepsilon(1)$.

Proof of Theorem 24. The basic idea of the algorithm is to consider each of the $O_\varepsilon(1)$ possible roundings to create $\tilde{I}$ described in Lemma 42. For each of these $\tilde{I}$, the algorithm finds the optimal solution to the configuration LP. Note that the set of all feasible configurations can be found in polynomial time using Lemma 52. For each solution, x to the LP, the algorithm creates $\lceil x_C \rceil$ bins for each $C \in \mathcal{C}_{\tilde{I}}$. Note that the set of possible configurations differs for different roundings $\tilde{I}$. For each of these $\lceil x_C \rceil$ bins, the algorithm creates the same $(1/\delta^3)$-2D-container packing by using the algorithm from Lemma 52 to create a 2D-container-packing. Finally, it returns the $(1/\delta^3)$-2D-container packing with the smallest number of bins over all the different roundings for I. Let B be the smallest number of required bins.

First, note that the number of configurations is bounded by $O_\varepsilon(1)$, as a basic optimal solution to the configuration-LP has at most $O_\varepsilon(1)$ non-zero components. Further, by Lemma 53 it holds that $\text{OPT}_{\text{2D-BP}}^{\text{LP}}(\tilde{I}) \leq \text{OPT}_{\text{2D-BP}}^{C}(\tilde{I})$ where $\text{OPT}_{\text{2D-BP}}^{C}(\tilde{I})$ refers to an optimal $(1/\delta^3)$-2D-container packing for $\tilde{I}$. Further, Lemma 42 states that for one of the iterated rounded instances $\tilde{I}$ it holds that $\text{OPT}_{\text{2D-BP}}^{C}(\tilde{I}) \leq (\frac{3}{2} + O(\varepsilon))\text{OPT}_{\text{2D-BP}}(I) + O_\varepsilon(1)$.

Finally, as there are at most $O_\varepsilon(1)$ different configurations in the basic LP solution, it holds that $B \leq \text{OPT}^{\text{LP}}_{\text{2D-BP}}(\tilde{I}) + O_\varepsilon(1)$. Hence, in total, the solution returned by the algorithm uses at most $(\frac{3}{2} + O(\varepsilon))\text{OPT}_{\text{2D-BP}}(I) + O_\varepsilon(1)$ bins with at most $O_\varepsilon(1)$ different configurations. ◄