Approximation Algorithms for 3D Orthogonal Knapsack*

Florian Diedrich[1], Rolf Harren[2], Klaus Jansen[1],
Ralf Thöle[1], Henning Thomas[1]

[1] Institut für Informatik, Christian-Albrechts-Universität zu Kiel,
Olshausenstr. 40, 24098 Kiel, Germany
[2] Fachbereich Informatik, Universität Dortmund,
Otto-Hahn-Str. 14, 44227 Dortmund, Germany

Abstract. We study non-overlapping axis-parallel packings of 3D boxes with profits into a dedicated bigger box where rotation is forbidden; we wish to maximize the total profit. Since this optimization problem is NP-hard, we focus on approximation algorithms. We obtain fast and simple algorithms with approximation ratios $9 + \epsilon$ and $8 + \epsilon$ as well as an algorithm with approximation ratio $7 + \epsilon$ that uses more sophisticated techniques; these are the smallest approximation ratios known for this problem. *Topics:* Algorithms, computational and structural complexity.

1 Introduction

Given a list $L = \{R_1, \ldots, R_n\}$ of boxes with sizes $R_i = (x_i, y_i, z_i)$ and positive profits p_i for each $i \in \{1, \ldots, n\}$ and a dedicated box $Q = (a, b, c)$, we study non-overlapping axis-parallel packings of sublists of L into Q which we call *feasible*. For simplicity call Q a *bin*. We wish to select a sublist that permits a packing and maximizes the profit. This problem will be called the *orthogonal three-dimensional knapsack problem* or *OKP-3* for short and we denote the optimal profit by OPT. It is a natural generalization of the knapsack problem (KP) which is known to be NP-hard. This makes an exact algorithm with a polynomial worst-case runtime bound impossible unless P = NP holds. W.l.o.g. we assume $a = b = c = 1$ and that each $R_i \in L$ can be packed by otherwise removing *infeasible* boxes and scaling in $O(n)$ time.

    Related problems. Different geometrically constrained two- and three-dimensional packing problems were studied, resulting in three main directions. In *strip packing* the target area is a strip of infinite height; the objective is to

minimize the height of the packing. For the 2D case, [22] yields an approximation ratio of 5/2; in [1], an asymptotic approximation ratio of 5/4 was obtained. The best known absolute approximation ratio of 2 was obtained with different techniques in [21,23]. In [13], an asymptotic fully polynomial time approximation scheme (AFPTAS – see [24] for a definition) was presented. For the 3D case, research has focused mainly on the asymptotic approximation ratio [20]. An asymptotic ratio of $2 + \epsilon$ was obtained in [10]; this was improved to 1.691 by Bansal et al. [3]. In [17] the *on-line* version was studied, resulting in a competitive ratio of 29/10. In *bin packing* the objective is to minimize the number of identical bins. For the 1D case, an asymptotic polynomial time approximation scheme (APTAS – see [24]) was presented in [7], while in [18] the currently best known asymptotic approximation ratio of 11/9 for the popular FFD algorithm is proved. For the 2D case, an asymptotic approximation ratio of 1.691 was obtained in [4]. In [2] it was proved that d-dimensional bin packing does not admit an APTAS for $d \geq 2$ and therefore no FPTAS, but an APTAS for packing *d*-dimensional cubes into the minimum number of unit cubes was presented. In the *knapsack* scenario the number of bins is a fixed constant [5], usually 1. For the 2D case, [11] yields an approximation ratio of $2 + \epsilon$. Classical 1D knapsack problems are relatively well understood, see [12,19] for surveys. Although the problems are closely related, results cannot be transferred directly. One main difference between bin/strip packing and knapsack packing is that in the first setting all boxes of the instance must be packed but in the latter a selection of items is needed.

Previous results and applications. Harren [8] obtained a ratio of $9/8 + \epsilon$ for the special case of packing cubes into a cube and proved the APX-completeness of the general case [9]. A *cutting stock* application is cutting blocks with given profits from larger pieces of material to maximize the profit; another application is the problem of selecting boxes to be transported in a container. Besides these, the problem is motivated from multiprocessor scheduling on grid topology. In this perspective, for a time slice of fixed duration, a set of jobs to be executed must be chosen and each job requires a subgrid of prespecified rectangular shape. For a special case of this application, in [25] an on-line algorithm is presented; See [6] for a study of similar problems.

New results. Our contribution is a fast and simple $(9 + \epsilon)$-approximation algorithm based on strip packing (Section 2) which is refined to an $(8 + \epsilon)$-approximation algorithm in Section 3. Both of these have practical running times. In Section 4 we obtain a $(7 + \epsilon)$-approximation algorithm using more costly techniques before concluding with open problems in Section 5.

2   An Algorithm Based on Strip Packing

We approximately solve a relaxation by selecting $L' \subseteq L$ that is at least near-optimal and has a total volume of at most 1 which is partitioned into 9 sublists. For each of these a packing into the bin will be generated. Out of these one with maximum profit is chosen, resulting in a $(9 + \epsilon)$-approximation algorithm.

More precisely L' will be packed into a strip $[0,1] \times [0,1] \times [0,\infty)$ by a level-oriented algorithm, improving a result from [16]. We partition the strip into packings of sublists of L' and among these return one with maximum profit. For each box R_i the rectangle (x_i, y_i) is called the *base rectangle* of R_i, denoted as $br(R_i)$. Such a rectangle (x_i, y_i) is called *big* $:\Leftrightarrow x_i \in (1/2, 1] \wedge y_i \in (1/2, 1]$, *long* $:\Leftrightarrow x_i \in (1/2, 1] \wedge y_i \in (0, 1/2]$, *wide* $:\Leftrightarrow x_i \in (0, 1/2] \wedge y_i \in (1/2, 1]$, and *small* $:\Leftrightarrow x_i \in (0, 1/2] \wedge y_i \in (0, 1/2]$. For each list L of boxes use $V(L) := \sum_{R_i \in L} x_i y_i z_i$ to denote the total volume of L and for each list L of rectangles $r_i = (x_i, y_i)$ use $A(L) := \sum_{r_i \in L} x_i y_i$ to denote the total area of L. Furthermore, $P(L) := \sum_{R_i \in L} p_i$ denotes the total profit of L. Finally, for each list L of boxes use $H(L)$ to denote the height of a packing of L where the packing itself will be clear from the context. We use the following theorem from [11] which is a refinement of the main result from [23].

Theorem 1. *Let L be a list of n rectangles such that $A(L) \leq 1/2$ holds and no long rectangles or no wide rectangles occur in L. Then L permits a feasible packing into the unit square which can be generated in time $O(n \log^2 n / \log \log n)$.*

First we apply the modified strip packing algorithm, then we construct the partition of the strip. The strip packing algorithm uses Theorem 1 to obtain an *area guarantee* for each but the last level, improving a result from [16].

Algorithm A.

1. Partition L into two sublists $L_1 := \{R_i | br(R_i) \text{ is long}\}$ and $L_2 := L \setminus L_1$. W.l.o.g. let $L_1 = \{R_1, \ldots, R_m\}$ and $L_2 = \{R_{m+1}, \ldots, R_n\}$.
2. Generate the packing for L_1 as follows.
   2.1. Find the boxes R_i in L_1 for which the area of $br(R_i)$ is greater than $1/4$ which are $R_{p+1}, \ldots, R_m$ w.l.o.g. Stack these on top of one another in direction z, each on its own level.
   2.2. Sort the remaining boxes $R_1, \ldots, R_p$ in non-increasing order of z_i, resulting in a list L_1'.
   2.3. Partition L_1' into consecutive sublists $L_1'', \ldots, L_v''$ where the total base area of each sublist is as close to $1/2$ as possible but not greater. Pack each of these sublists on a level by itself using Theorem 1. Stack all of these levels on top of one another in direction z.
3. Generate the packing for L_2 in a similar way as for L_1 by Theorem 1. The resulting steps are called Steps $3.1 - 3.3$.
4. Concatenate the packings of L_1 and L_2 to obtain a packing of L.

Theorem 2. *For each list L of n boxes Algorithm A generates a packing of height at most $4V(L) + Z_1 + Z_2$ where Z_1 and Z_2 are the heights of the first levels generated in Steps 2.3 and 3.3. The construction can be carried out in time $O(n \log^2 n / \log \log n)$.*

The proof is omitted due to page limitations; the result can be obtained by replacing the area bound $7/32$ by $1/4$ in the proof of Theorem 4 from [16]. The second part is a partition applied to the output of Algorithm A; see Figure 1.

Algorithm B.

1. Set $\delta := \epsilon/(9+\epsilon)$. Use an FPTAS for KP from [12,14] to select $L' \subseteq L$ such that $V(L') \leq 1$ and $P(L') \geq (1-\delta)$OPT holds, where OPT denotes the optimum of the generated KP instance.
2. Use Algorithm A to generate a packing of L' into the strip but separate the first levels generated in Steps 2.3 and 3.3. Pack these into a bin each.
3. By Theorem 2 the remaining strip has a height of at most $4V(L') \leq 4$. Consider the three cutting unit squares $[0,1] \times [0,1] \times \{i\}$ for $i \in \{1,2,3\}$. Generate a partition of the region $[0,1] \times [0,1] \times [0,4]$ into 7 subsets, namely 4 subsets which are each positioned in the regions $[0,1] \times [0,1] \times [i-1,i]$ for $i \in \{1,\ldots,4\}$ but not intersecting any of the unit squares and 3 subsets of boxes which each intersect with one of the three cutting unit squares.
4. Out of the sets generated in Steps 2 and 3 return one with maximum profit.

Each set generated in Steps 2 and 3 permits a feasible packing into the unit cube which is available as a byproduct of Algorithm A. L' is partitioned into at most 9 subsets by Algorithm B, as illustrated in Figure 1.

Theorem 3. *Algorithm B is a $(9+\epsilon)$-approximation algorithm for OKP-3 with running time $O(\mathrm{T_{KP}}(n,\epsilon) + n\log^2 n/\log\log n)$, where $\mathrm{T_{KP}}(n,\epsilon)$ is the running time of the FPTAS used for KP from [12,14]. Furthermore this bound is tight.*

Proof. Clearly $9+\epsilon$ is an upper bound for the ratio and the running time is dominated by solving the knapsack instance and by Algorithm A. For the following instance this bound can be attained. We have 10 boxes $R_1 := (1/2, 1/2, 2/15)$, $R_2 := (3/4, 1/3, 2/15)$, $R_3 := (1, 2/7, 3/4)$, $R_4 := \ldots := R_7 := (1, 2/7, 1/2)$, $R_8 := (1, 2/7, 1/4+2/15)$, $R_9 := (1, 2/7, 2/15)$ and $R_{10} := (1,1,1)$. Furthermore $p_1 := \ldots := p_9 := 1/9 - \epsilon/[9(9+\epsilon)]$ and $p_{10} := 1$. Let $S_1 := \{R_1, \ldots, R_9\}$ and $S_2 := \{R_{10}\}$. It is clear that S_2 is an optimal solution; elementary calculation shows $V(S_1) = 1$ and $P(S_1) = 1 - \delta$, hence S_1 may be selected in Step 1 of Algorithm B. Applying Algorithm B and assuming that the boxes are stacked in increasing order of index in Step 2.1 of Algorithm A, we obtain 9 bins each containing an item with profit $1/(9+\epsilon)$. $\square$
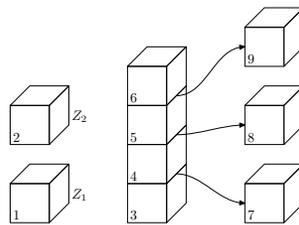

Fig. 1. At most 9 bins are generated by Algorithm B

Note that only the subset that is returned needs to be packed level-wise using the algorithm from Theorem 1 while the discarded subsets need not to be arranged. Algorithm B can be used to solve the special cases where we wish to maximize the number of selected boxes or the volume by setting $p_i := 1$ or $p_i := x_i y_i z_i$ for each $i \in \{1, \ldots, n\}$. This also holds for the other two algorithms that we present. In [14] approximation algorithms for various knapsack problems are found. Using these, Algorithm B can be generalized by replacing the KP solver in Step 1, yielding algorithms for *unbounded* OKP-3 and *multiple-choice* OKP-3; see [14] for notions and details. Algorithm B can be modified to yield a ratio of 18 with a much better running time by using a 2-approximation algorithm for classical KP. Call R_i *small* $:\Leftrightarrow x_i \in (0, 1/2] \wedge y_i \in (0, 1/2] \wedge z_i \in (0, 1/2]$. We obtain a criterion for packability of a list of boxes; the proof is omitted due to space restrictions.

Lemma 1. *Each list L of small boxes for which $V(L) \leq 1/8$ holds can be packed into the unit cube in time $O(n \log^2 n / \log \log n)$.*

3 A Refined Construction

In Algorithm A and the proof of Theorem 2, the area bound $1/2$ from Theorem 1 was used. We separate boxes with base area greater than $1/4$, resulting in the area guarantee $1/2 - 1/4 = 1/4$ for each level generated in Steps 2.2 and 2.3 except the last ones. The height bound can be improved if this area guarantee is improved. We have arbitrarily chosen direction z to be the axis for level generation, but any direction $d \in \{x, y, z\}$ will do. The two levels of height Z_1 and Z_2 are the result of partitioning the instance. We study the packing of small boxes more closely; Algorithm C is applied only to lists of small boxes.

Algorithm C.

1. Find the boxes R_i in L for which the area of $br(R_i)$ is greater than $1/10$ which are $R_1, \ldots, R_m$ w.l.o.g. Sort these in non-increasing order of z_i, resulting in a list L_1. Arrange these in groups of 4 boxes each, except for the last group. Each group can be put on a seperate level by placing the boxes into the corners of the level. Stack these levels on top of one another in direction z.
2. Sort the remaining boxes $R_{m+1}, \ldots, R_n$ in non-increasing order of z_i, resulting in a list L_2.
3. Partition L_2 into consecutive sublists $L_1'', \ldots, L_v''$ where the total base area of each sublist is as close to $1/2$ as possible but not greater. Pack each of these sublists on a level by itself using Theorem 1. Stack all of these levels on top of one another in direction z.
4. Concatenate the packings of L_1 and L_2 to obtain a packing of L.

Note that we formed two groups and obtain an area guarantee of $2/5$ for each layer except the last ones generated in Steps 1 and 3. To avoid confusion we point out that the area guarantee does not hold for the *last* generated layers, while the summands Z_1 and Z_2 in Theorem 2 are the heights of the respective *first* layers. Similar to the proof of Theorem 2, we obtain the following results.

Theorem 4. *For each list L of n small boxes Algorithm C generates a feasible packing of height at most $5/2V(L) + Z_1 + Z_2$ where $Z_1 \leq 1/2$ and $Z_2 \leq 1/2$ are the heights of the first levels generated in Steps 1 and 3. The construction can be carried out in time $O(n \log^2 n / \log \log n)$.*

Lemma 2. *Each list L of n small boxes with $V(L) \leq 1$ permits a feasible packing into at most 5 bins. The construction can be carried out algorithmically in time $O(n \log^2 n / \log \log n)$; the bound of 5 is tight for the used construction.*

Proof. Use Algorithm C to arrange L in a strip, but separate the first levels generated in Steps 1 and 3. Since L contains only small boxes, these two levels can be packed together into a bin. By Theorem 4, the remaining strip has a height of at most $5/2$. Consider the two cutting unit squares $[0,1] \times [0,1] \times \{i\}$ for $i \in \{1,2\}$. Generate a partition of the region $[0,1] \times [0,1] \times [0,5/2]$ into 5 subsets, namely first 3 subsets which are each positioned in the regions $[0,1] \times [0,1] \times [i-1,i]$ for $i \in \{1,2\}$ as well as the region $[0,1] \times [0,1] \times [2,5/2]$ but not intersecting any of the unit squares, and furthermore 2 subsets of boxes which each intersect with one of the two cutting unit squares. The first three sets can be packed into one bin each. Since L contains only small boxes, the last two sets can be arranged together into one additional bin. We have at most 5 bins; see Figure 2. The running time is dominated by Algorithm C and thus bounded by $O(n \log^2 n / \log \log n)$. To show the tightness of the bound let $\gamma := 1/500$ and consider the instance L consisting of $R_1 := \ldots := R_{29} := (1/2, 1/5 + \gamma, 1/3 + \gamma)$, $R_{30} := (\gamma, \gamma, 1/2)$, $R_{31} := \ldots := R_{33} := (1/2, 1/5, \gamma)$ and $R_{34} := (1/2, 1/5 - 2\gamma^2, \gamma)$. Note that $V(L) = 29/30 + 122/15\gamma + 15\gamma^2 - \gamma^3 < 29/30 + 1/60$. Application of Algorithm C results in 9 layers with height greater than $1/3$, which means that the layers cannot be arranged in less than 5 bins. □

Now we refine Algorithm B to yield a better approximation ratio, but we need to solve a stronger relaxation. For any direction $d \in \{x,y,z\}$ a box R_i is called *d-big* $:\Leftrightarrow d_i \in (1/2,1]$ and we use X, Y and Z to denote the set of boxes that are d-big for the corresponding direction. Any box that is d-big for every direction $d \in \{x,y,z\}$ will be called a *big* box.

Lemma 3. *Let L be a list of n boxes in which no small boxes and at most 3 big boxes occur. Then L can be partitioned into sets X', Y' and Z' in time $O(n)$, such that each of these contains at most one big box and the x-projections of boxes in X', the y-projections of boxes in Y' and the z-projections of boxes in Z' contain no long or no wide rectangles.*
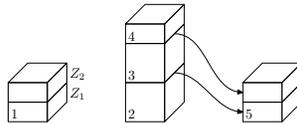

Fig. 2. The small boxes can be packed into at most 5 bins

Proof. Remove the at most 3 big boxes from L and distribute them in X', Y' and Z' such that in each of these sets at most one big rectangle occurs. Set $X' := X' \cup \{R_i \in L | x_i > 1/2, z_i \le 1/2\}$, $Y' := Y' \cup \{R_i \in L | y_i > 1/2, x_i \le 1/2\}$ and finally $Z' := Z' \cup \{R_i \in L | z_i > 1/2, y_i \le 1/2\}$ to obtain the claim.   □

To avoid repetition, we enumerate the cases in the analysis only.

Algorithm D.

1. Set $\delta := \epsilon/(8+\epsilon)$. Use a PTAS for non-geometric 4D KP from [12,14] to select $L' \subseteq L$ such that $P(L') \ge (1 - \delta)\text{OPT}$ where OPT denotes the optimum of the integral linear program

$$\text{maximize } \sum_{i=1}^{n} p_i R_i \text{ subject to } R \in P$$

   where R_i is an indicator variable for the box of the same name and the polytope P of nonnegative integers is defined by the constraints

$$\sum_{i=1}^{n} x_i y_i z_i R_i \le 1, \quad \sum_{R_i \in X} y_i z_i R_i \le 1, \quad \sum_{R_i \in Y} x_i z_i R_i \le 1, \quad \sum_{R_i \in Z} x_i y_i R_i \le 1.$$

2. Partition L' into at most 8 subsets which permit a feasible packing as described below. Out of these, return one with maximum profit.

Theorem 5. *Algorithm D is an $(8 + \epsilon)$-approximation algorithm for OKP-3 with running time $O(\text{T}_{4\text{DKP}}(n, \epsilon) + n \log^2 n / \log \log n)$, where $\text{T}_{4\text{DKP}}(n, \epsilon)$ is the running time of the PTAS used for 4D KP from [12,14]. Furthermore this bound is tight.*

Proof. We have not imposed a bound on the number of big boxes in the relaxation; due to the area conditions there are at most 3 big boxes in the selected set. *Case 1:* There is a direction $d \in \{x, y, z\}$ such that the d-projection area of all d-big boxes in L' is larger than or equal to $1/2$. In this case all d-big boxes can be packed into at most 3 bins with a construction from [11], which can be carried out in time $O(n \log^2 n / \log \log n)$, resulting in a volume of at least $1/4$ being packed. The total volume of the remaining boxes is bounded by $3/4$ and each remaining box has a d-height of at most $1/2$. We apply Algorithm A in direction d which results in a strip of d-height at most 3 and two additional levels of d-height at most $1/2$ each. All of these sets can be packed into at most 5 bins, generating at most 8 bins in total. *Case 2:* For all $d \in \{x, y, z\}$ the total projection area of all d-big boxes is smaller than $1/2$. By Lemma 3 we partition the set $\{R_i \in L' | R_i \text{ is not small}\}$ into sets X', Y' and Z' such that the total projection area of X', Y' and Z' for the corresponding direction is not greater than $1/2$ and the x-projections of boxes in X', the y-projections of boxes in Y' and the z-projection of boxes in Z' contain no long or no wide rectangles, respectively, and each of these sets contains at most one big box.

By Theorem 1 the sets X', Y' and Z' can be packed into at most one bin each, resulting in at most 3 bins in total. Let S denote the set of small boxes; these are not yet packed. Clearly $V(S) \leq 1$ holds, so by Lemma 2 the set S can be packed into at most 5 bins, which results in at most 8 bins in total. The runtime bound follows from the fact that we can distinguish between the two cases in time $O(n)$. For the tightness of the bound, consider the instance L in which $R_1, \ldots, R_{34}$ are as in the proof of Lemma 2, $R_{35} := (1, 1, 1/180)$, $R_{36} := (1, 1/180, 1)$, $R_{37} := (1/180, 1, 1)$, and $R_{38} := (1, 1, 1)$. The profits are defined by $p_i := 1/[9(8 + \epsilon)]$ for $i \in \{1, \ldots, 4, 30, \ldots, 34\}$, $p_i := 1/[8(8 + \epsilon)]$ for $i \in \{5, \ldots, 28\}$, $p_i := 1/(8 + \epsilon)$ for $i \in \{29, 35, 36, 37\}$ and $p_{38} := 1$. Let $S_1 := L \setminus \{R_{38}\}$ and $S_2 := \{R_{38}\}$. Since $P(S_1) = 8/(8+\epsilon) = (1-\delta) < 1 = P(S_2)$, S_2 is an optimal solution. Elementary calculation verifies that S_1 may be chosen in Step 1 of Algorithm D. Application of Algorithm D leads to *Case 2* in the analysis above, where $X' = \{R_{35}\}$, $Y' = \{R_{37}\}$ and $Z' = \{R_{36}\}$. Each of these sets is packed into a separate bin. The remaining items are small and are packed into 5 bins by the proof of Lemma 2. In total, 8 bins are generated; the profits are chosen such that each bin yields a profit of exactly $1/(8 + \epsilon)$. □

4   Enumerations and a Shifting Technique

The algorithms above generate cutting areas in the strip, resulting in subsets that have to be re-packed. We permit further loss of profit by discarding more boxes to remove inconvenient layers; the loss will be suitably bounded. The improvement will be at the cost of a considerably larger running time due to a large enumeration. Since the running time is not practicable anyway we omit the run time analysis of this approach. First we remove sets intersecting the cutting areas and the additional layers with a shifting technique.

Lemma 4. *Let $L = \{R_1, \ldots, R_n\}$ be a list of boxes with $z_i \leq \epsilon$ for each $R_i \in L$. Suppose L admits a packing into a strip of height at most h and let m be a positive integer. Then we can create m gaps of shape $[0,1] \times [0,1] \times [0,\epsilon]$ in the packing by deleting boxes such that for the remaining list $L' \subseteq L$ the inequality $P(L') \geq (1 - 2(m+1)\epsilon/h)P(L)$ holds. The construction can be done in time polynomial in n.*

Proof. We partition the strip into regions of height ϵ and eventually one region of smaller height. More precisely we define $p := \lceil h/\epsilon \rceil$ and partition the strip of height h into p regions $S_1, \ldots, S_p$ of shape $[0,1] \times [0,1] \times [0,\epsilon]$ where the uppermost region is of possibly smaller height. Then for each $i \in \{1, \ldots, p\}$ let $T_i = \{R_j \in L \mid R_j \cap S_i \neq \emptyset\}$ and let $U_1, \ldots, U_{m+1}$ be the $m+1$ sets out of $T_1, \ldots, T_p$ which have the smallest profit. Removing these from the packing causes a loss of profit which is $2(m+1)/pP(L)$ at most; we remove $m+1$ sets since we might select the uppermost region; in this way we assert that we have at least m regions of height ϵ. Let L' be the set of remaining boxes; finally $2(m+1)/pP(L) \leq 2(m+1)(h/\epsilon)^{-1}P(L) = 2(m+1)\epsilon/hP(L)$ holds. □

Note that the construction above can be carried out in any direction.

Theorem 6. *Let $L = \{R_1, \ldots, R_n\}$ be a list of boxes with $z_i \leq \epsilon$ for each $R_i \in L$ and $V(L) \leq \alpha$ with $\alpha \in [1/4, 1]$ holds. Then it is possible to select $L'' \subseteq L$ such that $P(L'') \geq (1 - 12\epsilon)P(L)$ holds and L'' admits a feasible packing into at most $\lceil 4\alpha \rceil$ bins. The construction can be carried out in time polynomial in n.*

Proof. Use Algorithm A to pack L into a strip of height at most $h := 4\alpha$ and two additional layers L_1 and L_2 by Theorem 2. Let L' be the set of boxes in L arranged in the strip; the following construction is illustrated in Figure 3. Use Lemma 4 to generate at most 5 suitable gaps in the strip, resulting in a loss of profit of at most $12\epsilon/hP(L') \leq 12\epsilon/hP(L)$; since $\alpha \in [1/4, 1]$, this loss is bounded by $12\epsilon P(L)$. The remaining set of boxes in the strip and L_1 and L_2 is denoted as L''. Consider the 3 cutting unit squares $[0, 1] \times [0, 1] \times \{i\}$ for $i \in \{1, 2, 3\}$ and L_3, L_4 and L_5 be the sets of boxes in the strip that intersect with these unit squares, respectively. W.l.o.g. none of the sets $L_1, \ldots, L_5$ is empty; otherwise it is removed from consideration. Note that each of the sets $L_1, \ldots, L_5$ can be arranged on a layer of height at most ϵ, so we generate a feasible packing by arranging them into the 5 gaps. In the resulting packing, the 3 cutting unit squares $[0, 1] \times [0, 1] \times \{i\}$ for $i \in \{1, 2, 3\}$ do not intersect with any box. Furthermore, all layers $L_1, \ldots, L_5$ are merged in the strip; the packing can be rearranged into $\lceil 4\alpha \rceil$ bins. $\square$
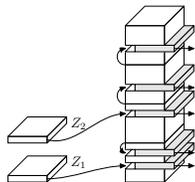

Fig. 3. The shifting technique described in Theorem 6

Similar as before for any $d \in \{x, y, z\}$ a box R_i is called d-ϵ-*big* $:\Leftrightarrow d_i \in (\epsilon, 1]$, and d-ϵ-*small* $:\Leftrightarrow d_i \in (0, \epsilon]$. We explain the details in the proof only.

Algorithm E.

1. Set $\delta := \epsilon/[35(7 + \epsilon)]$, let $L_1 := \{R_i | R_i$ is d-δ-big for each $d \in \{x, y, z\}\}$ and $L_2 := L \setminus L_1$.
2. For each $L_3 \subseteq L_1$ such that $|L_3| \leq \lfloor 1/\delta^3 \rfloor$ use an exact algorithm to verify whether L_3 is feasible. Store feasible L_3 of maximum total profit.
3. Use an FPTAS for classical KP from [12,14] to select $L_4 \subseteq L_2$ such that $V(L_4) \leq 1$ and $P(L_4) \geq (1 - \delta)$OPT holds.
4. Use the construction described below to select $L_5 \subseteq L_4$ which can be packed into at most 6 bins under a small loss of profit.

5. Out of the at most 7 sets generated in Step 2 and Step 4 return one with maximum profit.

Theorem 7. *Algorithm E is a $(7+\epsilon)$-approximation algorithm for OKP-3. Furthermore, this bound is asymptotically tight in the sense that it cannot be improved for ϵ arbitrary small.*

Proof. Note that $\lfloor 1/\delta^3 \rfloor$ is an upper bound for the number of boxes from L_1 in a feasible solution since δ^3 is a lower bound for the volume of each $R_i \in L_1$. Step 2 can be carried out in time polynomial in δ and thus polynomial in $1/\epsilon$ using an exact optimization algorithm as in [2]. We show that in Step 4 at most 6 sets are generated, resulting in at most 7 bins in total. Partition L_2 into 3 subsets X', Y' and Z' such that in each of these all boxes R_i are d-ϵ-small for the correspondig direction; note that $V(X') + V(Y') + V(Z') \leq 1$ holds. We apply the construction from Theorem 6 in each of the three directions. Study the following cases, where $V(X') \geq V(Y') \geq V(Z')$ holds w.l.o.g. *Case 1:* $V(X') \in (3/4, 1]$. The boxes in X' can be packed into at most 4 bins. We have $V(Y') + V(Z') \leq 1/4$. This means $V(Y') \leq 1/4$ and $V(Z') \leq 1/4$ holds. Consequently Y' and Z' can be packed into one bin each, resulting in at most 7 bins in total. *Case 2:* $V(X') \in (1/2, 3/4]$. The boxes in X' can be packed into at most three bins. Furthermore $V(Y') + V(Z') < 1/2$, which means that $V(Y') < 1/2$ holds. Consequently the boxes in Y' can be packed into at most 2 bins. Furthermore $V(Z') < 1/4$ holds and finally the boxes in Z' can be packed into 1 bin; this generates at most 7 bins in total. *Case 3:* We have $V(X') \in [0, 1/2]$. The boxes in X' can be packed into at most two additional bins. Furthermore $V(Y') \leq 1/2$ and $V(Z') \leq 1/2$ holds. This means that the boxes in Y' and Z' can be packed into at most two bins each. In total at most 7 bins are generated. In each of these cases at most 7 bins are generated; now we prove the ratio. Fix an optimal solution S and let P_1^* be the profit of boxes in $S \cap L_1$ and let P_2^* be the profit of boxes in $S \cap L_2$. Consequently $P_1^* + P_2^* \geq$ OPT holds. Let P_1 be the profit of the set that is stored in Step 2 and let P_2 be the profit of the set that is selected in Step 3. By construction we have $P_1 \geq P_1^*$ and $P_2 \geq (1 - \delta)P_2^*$. Furthermore, by threefold application of the construction from Theorem 6 the loss of profit in P_2 is bounded by $36\delta P_2$. The profit of the set returned in Step 5 is at least

$$
\begin{aligned}
(P_1 + P_2)/7 &\geq (P_1^* + (1 - \delta)(1 - 36\delta)P_2^*)/7 \\
&\geq (P_1^* + P_2^*)(1 - \delta)(1 - 36\delta)/7 \\
&= \text{OPT}(1 - \delta)(1 - 36\delta)/7 \geq \text{OPT}/(7 + \epsilon)
\end{aligned}
$$

which proves the claimed approximation ratio. The instance for the tightness of the bound is omitted for space reasons.   □

5   Conclusion

We contributed approximation algorithms for an NP-hard combinatorial optimization problem, where the runtimes of the simpler algorithms are practical. It

is an open problem whether here an algorithm with a ratio less than $7 + \epsilon$ exists. We are interested in a reduction of the running time, especially for Algorithm E. In [15] it was proved that it is NP-complete to decide whether a set of squares can be packed into the unit square. However, it is an open problem whether checking the feasibility of cubes into the unit cube is NP-complete. Lemma 4 reminds of the main result from [23], but is less flexible and structural. Further research is necessary to generalize the main result from [23] to the 3D case.

References

1. B. S. Baker, D. J. Brown, and H. P. Katseff. A 5/4 algorithm for two-dimensional packing. *Journal of Algorithms*, 2(4):348–368, 1981.

2. N. Bansal, J. R. Correa, C. Kenyon, and M. Sviridenko. Bin packing in multiple dimensions: inapproximability results and approximation schemes. *Mathematics of Operations Research*, 31:31–49, 2006.

3. N. Bansal, X. Han, K. Iwama, M. Sviridenko, and G. Zhang. Harmonic algorithm for 3-dimensional strip packing problem. accepted at the ACM-SIAM Symposium on Discrete Algorithms (SODA), 2007.

4. A. Caprara. Packing two-dimensional bins in harmony. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 490–499, 2005.

5. F. Diedrich. Approximative Algorithmen für Rucksackprobleme. Diploma thesis, Institut für Informatik und Praktische Mathematik der Christian-Albrechts-Universität zu Kiel, 2004.

6. A. Feldmann, J. Sgall, and S.-H. Teng. Dynamic scheduling on parallel machines. *Theoretical Computer Science (Special Issue on Dynamic and On-line Algorithms)*, 130(1):49–72, 1994.

7. W. Fernandez de la Vega and G. Lueker. Bin packing can be solved within $1 + \epsilon$ in linear time. *Combinatorica*, 1(4):349–355, 1981.

8. R. Harren. Approximating the orthogonal knapsack problem for hypercubes. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 238–249, 2006.

9. R. Harren. Approximation Mehrdimensionaler Packungsprobleme. Diploma thesis, Universität Dortmund, 2006.

10. K. Jansen and R. Solis-Oba. An asymptotic approximation algorithm for 3d-strip packing. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 143–152, 2006.

11. K. Jansen and G. Zhang. Maximizing the total profit of rectangles packed into a rectangle. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 197–206, 2004.

12. H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, 2004.

13. C. Kenyon and E. Rémila. A near-optimal solution to a two dimensional cutting stock problem. *Mathematics of Operations Research*, 25:645–656, 2000.

14. E. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4:339–356, 1979.

15. J. Y.-T. Leung, T. W. Tam, C. S. Wong, G. H. Young, and F. Y. L. Chin. Packing squares into a square. *Journal of Parallel and Distributed Computing*, 10:271–275, 1990.

16. K. Li and K.-H. Cheng. On three-dimensional packing. *SIAM Journal of Computation*, 19(5):847–867, 1990.

17. K. Li and K.-H. Cheng. Heuristic algorithms for on-line packing in three dimensions. *Journal of Algorithms*, 13:589–605, 1992.

18. R. H. Li and M. Y. Yue. The proof of FFD(L) $\leq$ (11/9)OPT(L) + (7/9). *Chinese Science Bulletin*, 42:1262–1265, 1997.

19. S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. Wiley, 1990.

20. F. K. Miyazawa and Y. Wakabayashi. An algorithm for the three-dimensional packing problem with asymptotic performance analysis. *Algorithmica*, 18:122–144, 1997.

21. I. Schiermeyer. Reverse-fit: A 2-optimal algorithm for packing rectangles. In *Proceedings of the Second Annual European Symposium on Algorithms (ESA)*, pages 290–299, 1994.

22. D. D. K. Sleator. A 2.5 times optimal algorithm for packing in two dimensions. *Information Processing Letters*, 10(1):37–40, 1980.

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

24. V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.

25. D. Ye and G. Zhang. Online scheduling of parallel jobs with dependencies on 2-dimensional meshes. In *Proceedings of the 14th Annual International Symposium on Algorithms and Computation (ISAAC)*, pages 329–338, 2003.