BIN PACKING IN MULTIPLE DIMENSIONS: INAPPROXIMABILITY RESULTS AND APPROXIMATION SCHEMES

NIKHIL BANSAL[1], JOSÉ R. CORREA[2], CLAIRE KENYON[3], AND MAXIM SVIRIDENKO[1]

[1] *IBM T.J. Watson Research Center,*
Yorktown Heights, NY 10598, USA. {nikhil,sviri}@us.ibm.com
[2] *School of Business, Univesidad Adolfo Ibáñez,*
Santiago, Chile. correa@uai.cl
[3] *Laboratoire d'Informatique, Ecole Polytechnique,*
91128 Palaiseau Cedex, France. kenyon@lix.polytechnique.fr

Abstract. We study the multidimensional generalization of the classical Bin Packing problem: Given a collection of d-dimensional rectangles of specified sizes, the goal is to pack them into the minimum number of unit cubes.

A long history of results exists for this problem and its special cases. Currently, the best known approximation algorithm for packing two-dimensional rectangles achieves a guarantee of 1.69 in the asymptotic case (i.e., when the optimum uses a large number of bins) [3]. An important open question has been whether 2−dimensional bin packing is essentially similar to the 1−dimensional case in that it admits an asymptotic polynomial time approximation scheme (APTAS) [12, 17] or not. We answer the question in the negative and show that the problem is APX hard in the asymptotic sense.

On the positive side, we give the following results: First, we consider the special case where we have to pack d-dimensional cubes into the minimum number of unit cubes. We give an asymptotic polynomial time approximation scheme for this problem. This represents a significant improvement over the previous best known asymptotic approximation factor of $2 - (2/3)^d$ [21] (1.45 for $d = 2$ [11]), and settles the approximability of the problem. Second, we give a polynomial time algorithm for packing arbitrary rectangles into at most OPT square bins with sides of length $1 + \varepsilon$, where OPT denotes the minimum number of unit bins required to pack these rectangles. Interestingly, this result does not have an additive constant term i.e., is not an asymptotic result. As a corollary, we obtain a polynomial time approximation scheme for the problem of placing a collection of rectangles in a minimum area encasing rectangle, settling also the approximability of this problem.

1. Introduction

In the *two-Dimensional Bin Packing Problem* rectangles of specified size (width, height) have to be packed into larger squares (bins). The most interesting and well-studied version of this problem is the so called *orthogonal packing without rotation* where each rectangle must be packed parallel to the edges of a bin. The goal is to find a feasible packing, i.e. a packing where rectangles do not overlap, using the smallest number of bins.

Bin packing and its d-dimensional variants have been extensively studied since the 60's both in the context of offline approximation algorithms and online algorithms (see e.g. [1, 3, 4, 6, 8, 11, 13, 18, 19, 21, 24, 30]). Detailed surveys can be found in [7, 10]. Throughout this paper we only consider offline algorithms.

Date: May 2005.

Key words and phrases. Bin packing, Approximation Algorithms, Polynomial time approximation schemes, Hardness of approximation.

Extended abstracts of different parts of this work appeared in the proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA 2004).

Clearly the NP-hardness of two-dimensional bin-packing follows from that of 1-dimensional bin packing (which is a special case, when all the heights are exactly 1). Furthermore it is known that given a collection of rectangles (in fact, squares), it is NP-hard to decide in polynomial time whether these can be packed in a single bin or require two bins [24]. Hence, no $2-\varepsilon$ approximation algorithm for the problem can exist unless $P = NP$. These kind of hardness of approximation results are typical in bin packing problems, nevertheless one can usually prove much stronger performance guarantees when considering asymptotic analysis. Therefore, the standard measure used to analyze the performance of a packing algorithm A is the asymptotic approximation ratio R_A^∞ defined as

$$\begin{aligned} R_A^n &= \max_L \{ \frac{A(L)}{\mathrm{OPT}(L)} : \mathrm{OPT}(L) = n\}, \\ R_A^\infty &= \lim_{n\to\infty} \sup R_A^n, \end{aligned}$$

where L ranges over the set of all problem instances and $A(L)$ (resp. $\mathrm{OPT}(L)$) denotes the number of bins used by A (resp. the optimal solution). Thus, the notion of asymptotic approximation ratio allows us to ignore the anomalous behavior of the algorithm for small instances. A problem admits an asymptotic approximation scheme, APTAS for short, if for any $\varepsilon > 0$, there is a polynomial time algorithm with an asymptotic approximation ratio of $1 + \varepsilon$.

For 1-dimensional bin packing, Fernandez de La Vega and Lueker [12] gave the first asymptotic approximation scheme. Later, this was improved by Karmarkar and Karp [17] to give an algorithm which uses $\mathrm{OPT} + O(\log^2(\mathrm{OPT}))$ bins. For the two-dimensional case, the first results were obtained by [6] who gave a 2.125 approximation algorithm. For a long time this was the best known, until a $2+\varepsilon$ (for any $\varepsilon > 0$) approximation was obtained (implicitly) by Kenyon and Rémila [19]. The recent breakthrough is an elegant 1.691 approximation algorithm due to Caprara [3]. Interestingly, many more results are known for some special cases in which there is a restriction on how the rectangles can be packed in a bin. Two particular cases that are widely studied are *Strip Packing* and *Shelf Packing* (details about these can be found in [3] and the references therein). While clever asymptotic approximation schemes are known for some of these special cases, it was unclear whether the general two-dimensional bin packing problem admits an approximation scheme.

For the special case of packing squares in square bins (which is also NP-hard by [24]) only algorithms with constant factor approximation ratios were known prior to our work. The first guarantee better than 2.125 was obtained by Ferreira et al. [13] who gave a 1.988-approximation algorithm. This was later improved to $14/9 + \varepsilon = 1.55 + \varepsilon$ by Kohayakawa et al. [21] and by Seiden and van Stee [30]. Recently, Caprara gave an algorithm and showed that it has approximation ratio in the interval $[1.490, 1.507]$, provided a conjecture is true [3]. The most recent (and best) known guarantee prior to our work is due to Epstein and van Stee [11] who give an $16/11 + \varepsilon = 1.454 + \varepsilon$ algorithm for the two-dimensional case. The algorithm in [21] also works in the d-dimensional case and its asymptotic approximation ratio of $2 - (2/3)^d$ was the best known approximation ratio for $d > 2$ dimensions prior to our work.

Another relevant problem in the context of two-dimensional packing is the *Minimum Rectangle Placement Problem*: Find the minimum area rectangle in which a given set of rectangles can be placed. This problem has many applications in scheduling, and, maybe the most prominent one, in VLSI design. Despite the significant number of approximate and exact heuristic methods that have been proposed for this problem (see e.g. [22, 27] and references therein), very few results that prove worst case approximation guarantees are known. Exceptions are the work of Kleitman and Krieger [20] and the work by Novotny [28], that consider the special case of packing squares into a minimum area rectangle.

A related multidimensional packing problem is *Vector Bin packing*, described as follows: Given a set of n rational, d-dimensional vectors $p_1, \ldots, p_n$ from $[0,1]^d$, find a partition of the vectors into sets $A_1, \ldots, A_m$ such that $||\bar{A}_i||_\infty \le 1$ for $1 \le i \le m$, where $\bar{A}_i = \sum_{j \in A_i} p_j$ is the sum of the vectors in A_i. The objective is to minimize m, the size of the partition. For $d = 1$, the vector bin packing problem is identical to the classical 1-dimensional bin packing, but this is not true for $d > 1$. Chekuri and Khanna [5] showed an interesting connection between d-dimensional vector bin packing (for arbitrary d) and graph coloring, which implies that vector bin packing is hard to approximate within $O(d^{\frac{1}{2}-\varepsilon})$ for any $\varepsilon > 0$. Woeginger [34] showed that the problem is APX hard even for the case of $d = 2$. The best known result for this problem is a $(1 + \varepsilon d + O(\ln \varepsilon^{-1}))$-approximation for any fixed $\varepsilon > 0$ [5].

Another closely related problem is the packing problem where we are given a set of rectangles with non-negative profits. The goal is to maximize the total profit of rectangles that can be packed into a larger rectangle (or square by scaling). The most recent paper on this problem is due to Jansen and Zhang [15]. They describe a few constant factor approximation algorithms for this problem and provide references on the state of the art for it.

1.1. **Our Results.** We now state the main result in each section of the paper.

1. In Section 2 we prove that the 2−dimensional bin packing problem does not admit an asymptotic approximation scheme. This directly implies the non-existence of an asymptotic PTAS for all $d \ge 2$.

2. We give the first asymptotic approximation scheme for packing squares into square bins in Section 3. More generally, we give an APTAS for packing d-dimensional hypercubes into bins for any fixed d. Specifically we prove the following:

Theorem 1.1. *There exists an algorithm A which, given a list I of n d-dimensional cubes and a positive number ε, produces a packing of I into $A(I)$ copies of $[0,1]^d$ such that:*

$$A(I) \le \lceil (1+\varepsilon)\mathrm{OPT}(I) \rceil + 1.$$

The running time of A is $O(n \log(n))$ for fixed d and ε.

We will also show that a slight variation of the algorithm achieves an absolute approximation guarantee of 2, matching a recent result by Van Stee [32]. Note that this algorithm achieves the best possible non-asymptotic ratio. This follows directly due to the impossibility of distinguishing in polynomial time whether a collection of squares can be packed in a single bin or requires two bins [13].

3. In Section 4 we design a resource augmented approximation scheme for the general two-dimensional rectangle packing problem, i.e., an algorithm which, given $\varepsilon > 0$, packs rectangles into bins of size $(1+\varepsilon) \times (1+\varepsilon)$. Our result is:

Theorem 1.2. *There exists an algorithm A which, given a list I of n rectangles and a positive number ε, produces a packing of I into $A(I)$ copies of $[0, 1+\varepsilon]^2$ such that:*

$$A(I) \le \mathrm{OPT}(I),$$

where $\mathrm{OPT}(I)$ is the minimum number of unit cubes in which I can be packed. The running time of A is polynomial in n for fixed ε.

Unfortunately, it turns out that the running time of our algorithm is $O(n^{2^{2^{\tilde{O}(1/\varepsilon)}}})$, which is rather impractical.

4. Observe that the result in Theorem 1.2 above is not asymptotic (there is no additive term in the expression for the number of bins used by the algorithm). We use this observation to give a polynomial time approximation scheme (PTAS) for the minimum rectangle placement problem. This is described in Section 5.

Theorem 1.3. *There exists an algorithm A which, given a list I of n rectangles and a positive number ε, finds a rectangle R, in which all rectangles in I fit and such that*

$$\mathrm{Vol}(R) \leq (1+\varepsilon)\mathrm{OPT}(I).$$

Here $\mathrm{Vol}(R)$ is the area of rectangle R and $\mathrm{OPT}(I)$ is the area of the minimum area rectangle in which I can be packed. The running time of A is polynomial in n for fixed ε.

1.2. **Techniques.** Our result for the APX hardness of two-dimensional bin packing has ideas similar to those used by Woeginger [34] to show the APX hardness of two-dimensional vector packing. However, our construction is much more involved than in [34] as there is much less structure in how the rectangles can be packed in a bin for two-dimensional bin packing as compared with that for vector packing. For example, given a collection of rectangles (or equivalently vectors) $p_1, \ldots, p_n$, it is NP-hard to decide whether these rectangles can be packed in a single bin, whereas this is trivial for vector bin packing (to do this, simply sum up the vectors and verify that all coordinates are at most 1).

For hypercube packing, most previous approaches that obtain a constant factor approximation [13, 30, 21, 11] use the classical techniques used for 1-dimensional bin packing. That is, classify the objects into large and small. Find a packing of the large objects using rounding and exhaustive search, and then pack the remaining small objects. In the case when $d > 1$, the above approach does not directly yield an approximation scheme for the following reason: The gaps left in the bin after packing the large objects can have arbitrary structure, and it is not clear how to pack the small objects in these gaps without wasting a constant fraction of the space. Our approach, extends the ideas of Fernandez de la Vega and Lueker [12] and is based on a technique originally used by Sevastianov and Woeginger [31] in the context of some problems in shop-scheduling. We partition the objects into 3 sets: large, medium and small, such that the medium objects do make up a significant portion of the input instance, and at the same time this gives us a sufficient gap between the sizes of the large objects and the small objects. We pack the medium objects separately, and then show how to pack the large and the small objects together.

We now consider the case of packing rectangles using resource augmentation. The main difficulty of obtaining a PTAS in this case is that there is no natural total order on rectangles. Hence, the rounding techniques used by Fernandez de la Vega and Lueker [12] and for the square packing above do not seem to extend here. We adopt the following approach. Since we are allowed to enlarge the bin sizes from $[0, 1]^2$ to $[0, 1+\varepsilon]^2$, we show that it is not a problem if we round the sides of rectangles that are both tall and wide (i.e., whose height and width are both large than a small constant δ). This enables to reduce the rectangles that are both wide and tall to a constant number of types. The main problem is to deal with the rectangles which are either very wide and flat or very thin and tall. For either kind of these rectangles we show that we can essentially use the strip-packing algorithm of Kenyon and Rémila [19]. Our algorithm thus relies on an appropriate partition of rectangles into "large", "small", "horizontal", "vertical", and "medium", in such a way that the medium rectangles are negligible. It only remains to mix gracefully the various kinds of rectangles; this requires discretizing a near-optimal solution appropriately so as to be able to "guess" not only where the large rectangles go, but also the areas used to pack horizontal rectangles and the areas used to pack vertical rectangles.

Our result for the minimum encasing rectangle is obtained by using the previous algorithm for packing rectangles using resource augmentation. Specifically we use the previous algorithm to decide whether a list of rectangles do not fit in a given rectangle or they fit in a slightly larger rectangle. We then do exhaustive search over all possible rectangles and select the one of minimum area. This procedure takes care of all instances having at least one "wide" and one "tall" rectangle. The rest of the instances are solved by a slight variation of a strip packing type result in [8].

2. APX Hardness of two-dimensional bin packing

We give an approximation preserving reduction from the Maximum Bounded 3-Dimensional matching problem (MAX-3-DM). The MAX-3-DM problem is defined as follows.

Input: Three sets $X = \{x_1, \dots, x_q\}, Y = \{y_1, \dots, y_q\}$ and $Z = \{z_1, \dots, z_q\}$. A subset $T \subseteq X \times Y \times Z$ such that any element in X, Y, Z occurs in one, two or three triples in T. Note that this implies that $q \leq |T| \leq 3q$.

Goal: Find a maximum cardinality subset T' of T such that no two triples in T' agree in any coordinate.

Measure: Cardinality of T'.

Kann [16] was the first who proved the MAX SNP hardness of the MAX-3-DM problem. Recently, explicit lower bounds on the approximation ratio of Max-3-DM have also been obtained [2, 14]. Petrank [29] (Theorem 4.4) proved a refined hardness result, where he showed that it is NP-hard to distinguish between instances where $|T'| = q$ and instances where $|T'| \leq (1-\varepsilon)q$ for some constant $\varepsilon > 0$.

Given an instance I of MAX-3-DM we will construct an instance of two-dimensional bin packing. The following is a high level description of the steps involved. First, we create a numerical version of the MAX-3-DM instance by associating an integer with each element in X, Y, Z and T. These integers will have the property that any four of them sum up exactly to a number B, if and only if the integers correspond to a triple $t_l = (x_i, y_j, z_k)$. Let x'_i (resp. y'_j, z'_k, t'_l) denote the integer corresponding to x_i (resp. x_j, z_k, t_l). Next, with each integer we will associate two rectangles. Each rectangle has width close to $1/4$ and height close to $1/2$. Each of the rectangles will have an area of about $1/8$, so no more than 8 rectangles can fit in a bin. The exact width and height of a rectangle is a small perturbation around $1/4$ and $1/2$ respectively, depending of the value of the integer associated it. Typically, one of the rectangles will be "thin" and "tall", and the other rectangle will be "wide" and "short". These perturbations are chosen such that the following property is satisfied. If $t_l = (x_i, y_j, z_k)$ or equivalently $t'_l + x'_i + y'_j + z'_k = B$, then the eight rectangles corresponding to x'_i, y'_j, z'_k and t'_l can fit in a bin. Otherwise, each bin is suboptimal in the sense that it either contains at most 7 rectangles, or contains at most 3 "tall" and "thin" rectangles. We begin by describing the construction and then prove some useful structural properties of this construction. We then show how these properties together with the APX hardness of MAX-3-DM implies the asymptotic hardness of 2-dimensional bin packing.

We begin by defining the integers corresponding to x_i, y_j, z_k, t_l. Let $r = 32q$. Define

$$\begin{aligned} x'_i &= ir^3 + i^2 r + 1, \quad &\text{for} \quad 1 \leq i \leq q, \\ y'_j &= jr^6 + j^2 r^4 + 2, \quad &\text{for} \quad 1 \leq j \leq q, \\ z'_k &= kr^9 + k^2 r^7 + 4, \quad &\text{for} \quad 1 \leq k \leq q. \end{aligned}$$

For every triple $t_l = (x_i, y_j, z_k)$ in T, we define

$$t'_l = r^{10} - x'_i - y'_j - z'_k + 15 = r^{10} - kr^9 - k^2 r^7 - jr^6 - j^2 r^4 - ir^3 - i^2 r + 8.$$

Let $\delta = 1/500$ and let $c = (r^{10} + 15)/\delta$. Observe that $0 < x'_i, y'_j, z'_k < \delta c/10$ and $t'_l < \delta c$ holds for all i, j, k, l. Intuitively, think of c as a scaling factor much larger than x'_i, y'_j, z'_k and b'_l.

We now describe the rectangles in our instance. A rectangle of width w and height h will be denoted by (w, h). For each element $x_i \in X$ (resp. $y_i \in Y$ and $z_i \in Z$), we define a pair of rectangles $a_{x,i}, a'_{x,i}$ (resp. $a_{y,i}, a'_{y,i}$ and $a_{z,i}, a'_{z,i}$) as follows (the reader is encourage to look at Figure 1):

$$a_{x,i} = \left(\frac{1}{4} - 4\delta + \frac{x'_i}{c}, \frac{1}{2} + 4\delta - \frac{x'_i}{c}\right) \quad \text{and} \quad a'_{x,i} = \left(\frac{1}{4} + 4\delta - \frac{x'_i}{c}, \frac{1}{2} - 4\delta + \frac{x'_i}{c}\right),$$

$$a_{y,i} = \left(\frac{1}{4} - 3\delta + \frac{y'_i}{c}, \frac{1}{2} + 3\delta - \frac{y'_i}{c}\right) \quad \text{and} \quad a'_{y,i} = \left(\frac{1}{4} + 3\delta - \frac{y'_i}{c}, \frac{1}{2} - 3\delta + \frac{y'_i}{c}\right),$$

$$a_{z,i} = \left(\frac{1}{4} - 2\delta + \frac{z'_i}{c}, 1/2 + 2\delta - \frac{z'_i}{c}\right) \quad \text{and} \quad a'_{z,i} = \left(\frac{1}{4} + 2\delta - \frac{z'_i}{c}, \frac{1}{2} - 2\delta + \frac{z'_i}{c}\right).$$

As $x'_i/c, y'_i/c$ and z'_i/c are all no more than $\delta/10$, observe that for each pair of rectangles the first rectangle is "thin" and "tall" and the second rectangle is "wide" and "short". Also observe how the perturbations in height and width are related to each other (this is made precise in Observations 2.1 and 2.2 below).

Next for each $t_l \in T$ we define two rectangles b_l and b'_l as

$$b_l = \left(\frac{1}{4} + 8\delta + \frac{t'_l}{c}, \frac{1}{2} + \delta - \frac{t'_l}{c}\right) \quad \text{and} \quad b'_l = \left(\frac{1}{4} - 8\delta - \frac{t'_l}{c}, \frac{1}{2} - \delta + \frac{t'_l}{c}\right).$$

Finally we define D to be a collection of $|T| - q$ dummy rectangles d_i such that $d_i = (3/4 - 10\delta, 1)$. These dummy rectangles will not play a significant role and will just serve as "garbage collectors".

Formally, we say that two rectangles a and a' are buddies iff $\{a, a'\}$ is $\{a_{x,i}, a'_{x,i}\}$ or $\{a_{y,j}, a'_{y,j}\}$ or $\{a_{z,k}, a'_{z,k}\}$ for $1 \le i, j, k \le q$ or $\{a, a'\} = \{b_l, b'_l\}$ for some $1 \le l \le |T|$.

Let $A_x = \{a_{x,1}, \ldots, a_{x,q}\}$ and $A'_x = \{a'_{x,1}, \ldots, a'_{x,q}\}$. A_y, A'_y, A_z and A'_z are defined similarly to be the set of rectangles $a_{y,i}$, $a'_{y,i}, a_{z,i}$ and $a'_{z,i}$ respectively. Let A denote the collection $A_x \cup A_y \cup A_z$ and $A' = A'_x \cup A'_y \cup A'_z$ and finally let $B = \{b_1, \ldots, b_{|T|}\}$ and $B' = \{b'_1, \ldots, b'_{|T|}\}$. Having all the notation in place, we first list some observations about these rectangles that follow directly from the choice of sizes of these rectangles.

Observation 2.1. *For each rectangle $a \in A \cup A'$, $w(a) + h(a) = 3/4$. For any $b \in B$, $w(b) + h(b) = 3/4 + 9\delta$ and for any $b' \in B'$, $w(b') + h(b') = 3/4 - 9\delta$. In particular, this implies that for any $b \in B$ and $b' \in B'$, $h(b) + h(b') + w(b) + w(b') = 3/2$.*

Observation 2.2. *For any two rectangles a, a' in $A \cup A' \cup B \cup B'$, $h(a) + h(a') = 1$ iff a and a' are buddies.*

Observation 2.3. *The width (resp. height) of any rectangle in the instance is at least $1/4 - 10\delta$ (resp. at least $1/2 - 5\delta$), and hence the area of any rectangle is at least $1/8 - 25/4\delta > 1/9$. Thus no bin can have more than 8 rectangles.*

Observation 2.4. *For any packing of squares in a bin, every vertical line (i.e. of the form $x = c$), intersects at most one rectangle from $A \cup B$. This follows as the height of each rectangle in $A \cup B$ is strictly greater than $1/2$.*

This observation implies that any bin can contain at most four rectangles in $A \cup B$. Moreover, any bin can contain at most three rectangles in B as the width of each rectangle in $A \cup B$ is more than $1/5$, and the width of each rectangle in B is more than $1/4$. Finally, it is easy to see that,

Observation 2.5. *If a rectangle $d_i \in D$ lies in some bin S, then S contains at most 2 other rectangles and at most one of them is a rectangle from $A \cup B$.*

Next, we give two less obvious consequences (Lemmas 2.6 and 2.7) of the choices of sizes of these rectangles. The following definition is needed only for the next two lemmas. For a rectangle a, we now define $\Delta(a)$ which allows us to relate back the rectangle to the integers x'_i, y'_j, z'_k or b'_l. For a rectangle $a_{x,i} \in A_x$, let $\Delta(a_{x,i}) = x'_i$ and for $a'_{x_i} \in A'_x$, $\Delta(a'_{x,i}) = -x'_i$. Similarly, $\Delta(a_{y,j}) = y'_j$, $\Delta(a_{z,k}) = z'_k$, $\Delta(b_l) = t'_l$ and $\Delta(a'_{y,j}) = -y'_j$, $\Delta(a'_{z,k}) = -z'_k$, $\Delta(b'_l) = -t'_l$.

Lemma 2.6. *For any three rectangles $a_1, a_2, a_3 \in A$ and $b \in B$, we have that $w(a_1) + w(a_2) + w(a_3) + w(b) = 1$ iff there is a triple $t_l = (x_i, y_j, z_k)$ in the instance I of MAX-3-DM and $\{a_1, a_2, a_3, b\} = \{a_{x,i}, a_{y,j}, a_{z,k}, b_l\}$.*

Proof. The "if" part follows directly from the choice of the sizes of the rectangles. In particular, if $t_l = (x_i, y_j, z_k)$, then

$$w(a_{x,i}) + w(a_{y,j}) + w(a_{z,k}) + w(b_l) = 1 - \delta + \frac{(x_i' + y_j' + z_k' + t_l')}{c} = 1 - \delta + \frac{r^{10} + 15}{c} = 1.$$

For the "only if" part of the lemma, we first observe that

$$w(a_1) + w(a_2) + w(a_3) + w(b) = 1 - \delta + \frac{\Delta(a_1) + \Delta(a_2) + \Delta(a_3) + \Delta(b)}{c}.$$

Thus, if $w(a_1) + w(a_2) + w(a_3) + w(b) = 1$, it must be that

$$\sum_{i=1}^{3} \Delta(a_i) + \Delta(b) = \delta c = r^{10} + 15.$$

Consider the quantity $\sum_{i=1}^{3} \Delta(a_i) + \Delta(b)$ modulo r. Since r is a multiple of 32, it follows that there is exactly one rectangle each from A_x, A_y, A_z and B since $15 = 1 + 2 + 4 + 8$ and this is the only way to represent 15 as a multi-sum (with possible repetitions) of four numbers from the set $\{1, 2, 4, 8\}$. Next, considering the sum $\sum_{i=1}^{3} \Delta(a_i) + \Delta(b)$ modulo r^2 it follows that i and l is such that $x_i \in t_l$. Similarly, considering modulo r^4 and r^7, it follows that $y_j \in t_l$ and $z_k \in t_l$. As x_i, y_j and z_k determine t_l uniquely, the result follows. □

Lemma 2.7. *Let $\mathcal{R} = \{a_1, a_2, a_3, a_4\}$ be a set of any four rectangles lying in $A \cup A'$, such that no two of them are buddies. Then $\sum_{i=1}^{4} w(a_i) \neq 1$.*

Proof. Suppose for the sake of contradiction that $\sum_{i=1}^{4} w(a_i) = 1$. As $0 \leq |\Delta(a_i)| \leq \delta c/10$, it must be that $\sum_{i=1}^{4} \Delta(a_i) = 0$. We first show that there cannot be more than one rectangle from $A_x \cup A_x'$ in the set $\mathcal{R}$ (later we will show that the same argument works for $A_y \cup A_y'$ and $A_z \cup A_z'$), which contradicts that $\mathcal{R}$ contains four rectangles in $A \cup A'$. Consider the coefficient of r and r^3 in $\sum_{i=1}^{4} \Delta(a_i)$. These coefficients depend on rectangles in $(A_x \cup A_x') \cap \mathcal{R}$ only. Let i_1, i_2, i_3, i_4 denote the indices of rectangles from $(A_x \cup A_x') \cap \mathcal{R}$ (where an index is 0, if fewer than 4 occur). Since no two rectangles are buddies, we cannot have both a_{x,i_k} and a'_{x,i_k} in $\mathcal{R}$. So, for each i_k, $1 \leq k \leq 4$, we associate a variable $\alpha(i_k)$, where $\alpha(i_k) = 1$ if $a_{x,i_k} \in \mathcal{R}$ and -1 if $a'_{x,i_k} \in \mathcal{R}$.

Since the coefficients of r and r^3 sum to 0, we must have that $\sum_{k=1}^{4} \alpha(i_k) i_k = 0$ and $\sum_{k=1}^{4} \alpha(i_k) i_k^2 = 0$, with the constraints that all non-zero i_k's are distinct (since no two rectangles are buddies). We now claim that the only feasible solution to this system is $i_k = 0$ for all $1 \leq k \leq 4$. As all the i_k's are distinct, we need at least 3 of the i_k's to be non-zero, else we cannot have $\sum_k \alpha(i_k) i_k = 0$. Also, it is not the case that all $\alpha(i_k)$ are either -1, or all are +1. Under these constraints, when there are exactly three non-zero i_k's, without loss of generality, we have the following system of equations: $i_1 + i_2 = i_3$ and $i_1^2 + i_2^2 = i_3^2$ such that $i_1, i_2, i_3 > 0$. However, squaring the first equation and subtracting the second from it implies that $2i_1 i_2 = 0$, which contradicts that all of i_1, i_2 and i_3 are non-zero. Thus, there is no solution to this system of equations.

Finally, we consider the case when all the i_k's are non-zero. As not all $\alpha(i_k)$ have the same sign, without loss of generality we have only the following two cases.

(1) $i_1 + i_2 = i_3 + i_4$ and $i_1^2 + i_2^2 = i_3^2 + i_4^2$,
(2) $i_1 + i_2 + i_3 = i_4$ and $i_1^2 + i_2^2 + i_3^2 = i_4^2$.

As all i_1, i_2, i_3 and i_4 are non-zero, the last case clearly does not have a solution. For the first case, rewrite the equations as $i_1 - i_3 = i_4 - i_2$ and $i_1^2 - i_3^2 = i_4^2 - i_2^2$. As i_1, i_2, i_3 and i_4 are all pairwise distinct, this implies that $i_1 + i_3 = i_4 + i_2$. Together with, $i_1 + i_2 = i_3 + i_4$, this implies that $i_1 = i_4$ which gives a contradiction. Repeating a similar argument for $A_y \cup A_y'$ and $A_z \cup A_z'$ by considering the coefficients of r^4 and r^6 and that of r^7 and r^9 respectively, it follows that there can be at most one rectangle each from $A_y \cup A'_y$ and $A_z \cup A'_z$ in $\mathcal{R}$, which contradicts the assumption that $\mathcal{R}$ contains four rectangles. □

We now have enough structural results about how rectangles can be packed in a bin. This will allow us to show that if a bin does not contain eight rectangles corresponding to a triple, then it is sub-optimally packed. We make the notion of "sub-optimally packed" precise with the following definition of a good bin.

Definition 2.8. Given a packing of the bins, call a bin *good* if it contains exactly 8 rectangles and additionally it has exactly 4 rectangles from $A \cup B$.

The following crucial lemma characterizes the structure of good bins.

Lemma 2.9. *A bin is good if and only if it contains the rectangles $a_{x,i}, a_{y,j}, a_{z,k}, b_l$ and the corresponding rectangles $a'_{x,i}, a'_{y,j}, a'_{z,k}, b'_l$ such that $t_l = (x_i, y_j, z_k)$ corresponds to a triple in the MAX-3-DM instance.*

Proof. We first show that the rectangles corresponding to a triple can be packed in a bin. Starting from the bottom left corner of the bin and moving towards the right, we pack the rectangles $a_{x,i}$,$a_{y,j}$, $a_{z,k}$ and b_l. Each of these rectangles is placed such that it touches the bottom of the bin.

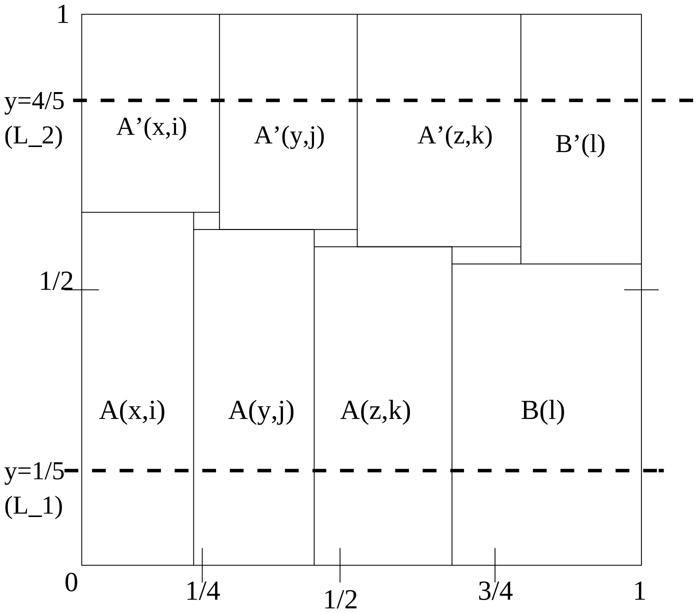

FIGURE 1. Packing of the rectangles corresponding to a triple

Figure 1 shows the packing. It is easy to verify that these four rectangles can be packed as described, as

$$w(a_{x,i}) + w(a_{y,j}) + w(a_{z,k}) + w(b_l) = 1 - \delta + (x'_i + y'_j + z'_k + t'_l)/c = 1.$$

Next we observe that each of the rectangles $a'_{x,i}, a'_{y,j}, a'_{z,k}$ and b'_l can be placed in the remaining gaps (as shown in Figure 1). Clearly, $a'_{x,i}$ can be placed on top of $a_{x,i}$ because $h(a_{x,i}) + h(a'_{x,i}) = 1$ and as $h(a_{x,i}) > h(a_{y,j}) > h(a_{z,k}) > h(b_l)$, this allows $a'_{x,i}$ to extend horizontally beyond $a_{x,i}$. Arguing similarly and observing that $w(a'_{x,i}) + w(a'_{y,j}) + w(a'_{z,k}) + w(b'_l) = 1$, it is easy to see that rectangles $a'_{y,j}, a'_{z,k}$ and b'_l also fit.

We now show that any good bin must correspond to a triple. We first give a way for labeling the 8 rectangles in a good bin. Consider the lines $L_1 = \{y = 1/5\}$ and $L_2 = \{y = 4/5\}$. It is easy to see that in any packing of a bin with 8 rectangles, each rectangle must intersect exactly one of L_1 or L_2. This follows as any rectangle has height at most $1/2 + 1/50 < 3/5$ and at least $1/2 - 1/50 > 2/5$. Moreover, as any rectangle has width strictly larger than $1/5$, it follows that each L_1 and L_2 intersects exactly 4 rectangles. Let $\{a_1, a_2, a_3, a_4\}$ denote the rectangles that intersect L_1 such that a_i is to the left of a_j for $i < j$. Similarly let $\{a_5, a_6, a_7, a_8\}$ denote the rectangles that intersect L_2 in the left to right order. Thus, we have that

$$w(a_1) + w(a_2) + w(a_3) + w(a_4) \leq 1, \tag{1}$$

and that

$$w(a_5) + w(a_6) + w(a_7) + w(a_8) \leq 1. \tag{2}$$

Finally, as the width of each rectangle is at least $1/5$, for each $1 \leq i \leq 4$, it must be that rectangle a_i overlaps with a_{i+4} in an x-coordinate (i.e. there is a vertical line $x = c$ for some c that intersects both a_i and a_{i+4}). Implying that

$$h(a_i) + h(a_{i+4}) \leq 1 \qquad \text{for } 1 \leq i \leq 4. \tag{3}$$

We now show that any good bin must contain exactly one rectangle from B.

(1) Suppose that at least two rectangles in B lie in a good bin. As each rectangle in B has width at least $1/4 + 8\delta$, two such rectangles use at least $1/2 + 16\delta$. Thus the width left for rectangles from A is at most $1/2 - 16\delta$ (we cannot put rectangles from A on top of the rectangles from B), and hence at most one rectangle from A can fit. Similarly, if exactly three rectangles from B lie in the bin, there cannot be any rectangle from A. Thus in either case there are at most three rectangles from $A \cup B$ which contradicts the fact that the bin is good.

(2) Suppose that no rectangle in B lies in a good bin. We claim the bin cannot have more than three rectangles from A. For the sake of contradiction suppose $r_1, r_2, r_3, r_4 \in A$ lie in the bin. Then no rectangle from B' can lie in the bin, because any rectangle in A has height at least $1/2 + 2\delta$ while the height of any rectangle in B' is at least $1/2 - \delta$ and moreover any rectangle in B' must overlap in some x-coordinate with some r_i for $1 \leq i \leq 4$. Thus all the eight rectangles lie in $A \cup A'$.

Adding (1), (2) and (3) we have that $\sum_{i=1}^{8}(w(a_i)+h(a_i)) \leq 6$. Moreover from Observation 2.1, as $w(a)+h(a) = 3/4$ for each rectangle $a \in A \cup A'$, we have that $\sum_{i=1}^{8}(w(a_i)+h(a_i)) = 6$, thus it must be the case that each of the inequalities (1), (2) and (3) must hold with equality. By Observation 2.2, this implies that a_i and a_{i+4} are buddies for each $i = 1, \ldots, 4$. This in particular implies that no two rectangles are buddies among the rectangles a_1, a_2, a_3 and a_4. Therefore by Lemma 2.7, it is impossible that $\sum_{i=1}^{4} w(a_i) = 1$, and hence we have a contradiction.

Thus, any good bin can contain exactly one rectangle from B. To finish the proof, we show that if b_l lies in a good bin, then it must contain rectangles corresponding to the triple $t_l = (x_i, y_j, z_k)$.

First, by definition, any good bin with exactly one rectangle b_l from B must contain exactly three other rectangles from A. Since any rectangle from B' cannot overlap in an x-coordinate with any rectangle in A, it follows that there can be *at most* one rectangle from B', which must overlap with b_l. Next, we show that there has to be *at least* one rectangle from B'. Suppose there are no rectangles from B', then we claim that the total width of all the rectangles must be strictly larger than 2, which is not possible. To see this, since there is no rectangle from B', there must be four rectangles from A'. As the width of any rectangle in A' (resp. B) is strictly more than $1/4 + \delta$ (resp. $1/4 + 8\delta$), any four rectangles from A' together with any rectangle from B take up width strictly larger than $5/4 + 12\delta$. But, as the width of any rectangle in A is at least $1/4 - 4\delta$, we do not have sufficient total width to pack three rectangles from A, which contradicts the fact that the bin is good.

Hence there is exactly one rectangle from B'. Call it $b'_{l'}$. By Observation 2.1, we have that $h(b_l)+w(b_l)+h(b'_{l'})+w(b'_{l'}) = 3/2$. Moreover, observing that $w(a)+h(a) = 3/4$ for each $a \in A \cup A'$, it follows that each of the inequalities (1), (2) and (3) is satisfied with equality. To complete the argument, suppose b_l intersects line L_1, let a_1, a_2, a_3 denote the rectangles in $A \cup A'$ which also intersect L_1. Thus, we have that $w(a_1) + w(a_2) + w(a_3) + w(b_l) = 1$. None of the a_i, $1 \leq i \leq 3$ can lie in A', because otherwise it is always the case that $w(a_1) + w(a_2) + w(a_3) + w(b_l) > 1$. Since all a_1, a_2 and a_3 lie in A, by Lemma 2.6 it follows that these rectangles correspond to a triple $t_l = (x_i, y_j, z_k)$. □

Theorem 2.10. *There is no Asymptotic PTAS for the two-dimensional Bin Packing Problem unless $P = NP$.*

Proof. If the MAX-3-DM problem has a matching consisting of q triples, then we can get a bin packing solution which uses $|T|$ bins as follows. For each of the triples in the matching, create a good bin as described in Lemma 2.9. For each t_l not in the matching, we put b_l and b'_l along with a dummy rectangle, and hence we use $q + (|T| - q) = |T| \leq 3q$ bins.

Assume now that every feasible solution of the MAX-3-DM problem has at most $(1-\varepsilon)q$ triples. We will show that any solution to the corresponding bin packing problem uses at least $(1+\varepsilon/33)|T|$ bins. Consider any feasible solution to the bin-packing instance. There will be exactly $n_d = |T| - q$ bins with dummy objects. Let n_g denote the number of good bins. Since the set of good bins corresponds to some feasible solution by Lemma 2.9 we have $n_g \leq (1-\varepsilon)q$. Among the bins which are not good let n_{b_1} denote the number of bins (other than the bins with dummy objects) which contain at most 7 rectangles and let n_{b_2} denote the rest of the bins (note that these are precisely the bins that have eight rectangles but 3 or fewer rectangles from $A \cup B$).

Since any solution must cover all the rectangles in $A \cup B$ and any bin with a dummy rectangle can have at most one rectangle from $A \cup B$, we have that

$$4n_g + 4n_{b_1} + 3n_{b_2} + n_d \geq 3q + |T|.$$

Equivalently,

$$4n_g + 4n_{b_1} + 3n_{b_2} \geq 4q.$$

Finally, since all the rectangles in $A \cup A' \cup B \cup B'$ must be covered, we have that

$$8n_g + 7n_{b_1} + 8n_{b_2} + 2n_d \geq 6q + 2|T|.$$

Equivalently,

$$8n_g + 7n_{b_1} + 8n_{b_2} \geq 8q.$$

Adding the inequalities above, $12n_g + 11n_{b_1} + 11n_{b_2} \geq 12q$. Equivalently, $n_g + n_{b_1} + n_{b_2} \geq 12q/11 - n_g/11$. Adding the bins with dummy objects, this implies that the total number of bins used is at least $|T| - q + 12q/11 - n_g/11 = |T| + (q - n_g)/11 \geq |T| + \varepsilon q/11 \geq |T|(1 + \varepsilon/33)$.

Now if there is an APTAS for two-dimensional bin packing, then for every $\varepsilon > 0$, there exists an algorithm A_ε and a constant c_ε, such that for instances I if $|\mathrm{OPT}(I)| > c_\varepsilon$, then $A_\varepsilon \leq (1 + 2\varepsilon)|\mathrm{OPT}(I)|$. Thus for any $\varepsilon > 0$, if $q > c_\varepsilon$ we can distinguish between two instances of the MAX-3-DM problem with $|T'| = q$ and $|T'| \leq (1 - 66\varepsilon)q$, which is an NP-hard problem by [29]. □

3. Hypercube Packing

In this section we describe our asymptotic approximation scheme for packing a collection of d-dimensional cubes into minimum number of unit cubes. Our result will consist of three parts. In section 3.2 we show how to pack small cubes into rectangular regions without wasting too much space. In section 3.3 we show how to compute an almost optimum packing of large cubes into unit bins. Finally, in section 3.4 we show how to combine these two ideas together to get a close to optimum packing of large and small cubes simultaneously. We begin with some preliminaries.

3.1. **Definitions and Preliminaries.** We adopt two standard assumptions in multidimensional bin packing: first, items are allowed to "touch" i.e., they can intersect in a face; second, items cannot be rotated (*orthogonal packing without rotation*).

A d-dimensional cube is given by a positive number a, representing the length of its side. Since we will pack squares into unit squares, we need to assume $a \leq 1$. Given an input list I of n d-dimensional cubes with sides of length $a_i \in (0,1]$ for $i = 1, \ldots, n$, the volume of the list is defined as: $\mathrm{Vol}(I) = \sum_{i=1}^{n} a_i^d$. Given two cubes and their positions in the space, we say that they are *nonoverlapping* if their interiors are disjoint.

A packing of I into k bins is a positioning of the cubes into k copies $H_1, \ldots, H_k$ of the unit hypercube $[0,1]^d$, so that no two cubes overlap. The d-dimensional cube packing problem consists of finding a packing of I into the minimum number of bins, $\mathrm{OPT}(I)$. We denote by $A(I)$ the number of bins algorithm A uses to pack I.

We now establish a simple result that will be needed later.

Lemma 3.1. *Let P be a packing of m d-dimensional cubes in $[0,1]^d$. Then the unused space in the bin, denoted by $[0,1]^d \setminus P$, can be divided into at most $(2m)^d$ nonoverlapping d-dimensional rectangles.*

Proof. Extending each facet of each cube in P, we obtain a grid in $[0,1]^d$ consisting of $(2m+1)^d$ cells. By pushing cubes towards the origin, in Lemma 3.1 we can assume without loss of generality that in the packing P there is a cube touching each of the hyperplanes $x_i = 0$, for $i = 1, \ldots, d$. Thus, $[0,1]^d \setminus P$ can be seen as the union of no more than $(2m)^d$ nonoverlapping d-dimensional rectangles. □

We can make a stronger statement for the two-dimensional case.

Observation 3.2. *(The two-dimensional case)*
Let P be a packing of m squares in $[0,1]^2$. Then, $[0,1]^2 \setminus P$ can be divided into at most $3m$ nonoverlapping rectangles.

Proof. For each square in P draw a horizontal line at its top and bottom until it intersects some other square as in Figure 1. Assuming the bottom-most square is at height 0, we have drawn $2m-1$ horizontal lines. This partitions the unit square into at most $3m$ rectangles (two to the side of each square and another to the top of each square in P) plus the original m squares. □

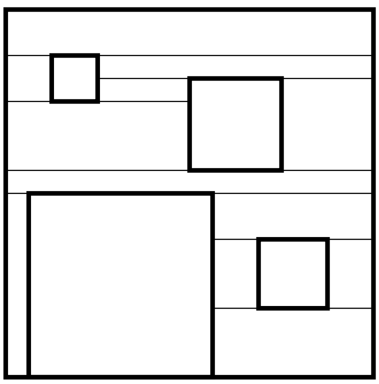

FIGURE 2. Decomposing the used space in the bin into rectangular regions

3.2. **Packing Small Cubes.** In this section we look at multidimensional cube packing in case all cubes are small. In this setting we are given an input list $S = (a_1, \ldots, a_n)$ of cubes with sides $a_i \leq \delta$ for some positive (small) constant δ. We will use a multidimensional version of the Next-Fit-Decreasing-Height shelf heuristic (NFDH). Coffman, Garey, Johnson, and Tarjan [8] analyzed this heuristic in the context of strip packing, pointing out properties closely related to what we extend here to higher dimensions.

Let us assume that the cubes are sorted according to non-increasing order of sizes. In two dimensions, the NFDH algorithm is a level algorithm which uses the next-fit approach to pack the sorted list of cubes. The cubes are packed, left-justified on a shelf until the next rectangle will not fit. This rectangle is used to define a new shelf and the packing continues on this shelf. The earlier shelves are not revisited. In general, for higher dimensions, we define the NFDH heuristic inductively. Assume we know how to perform NFDH in $d-1$ dimensions. The d-dimensional NFDH heuristic will consider a facet $\mathcal{F}$ of the bin and pack cubes on it using its $(d-1)$-dimensional version. For that, one of the dimensions (say the height) of the d-dimensional cubes is ignored and these cubes are seen as $(d-1)$-dimensional cubes. After it finishes packing in this facet, it will cut off from the bin a d-dimensional rectangle (*shelf*) with base $\mathcal{F}$ and height a_1. It will proceed packing the remaining list in the rest of the bin. The following lemma illustrates a fundamental property of NFDH. It is a consequence of a result of Meir and Moser [25] but we include a proof for completeness.

Lemma 3.3. *Let C be a set of d-dimensional cubes (where $d \geq 2$) of side smaller than δ. Consider the NFDH heuristic applied to C. If NFDH cannot place any other cube in a rectangle R of size $r_1 \times r_2 \times \cdots \times r_d$ (with $r_i \leq 1$), the total wasted (unfilled) volume in that bin is at most:*

$$\delta \sum_{i=1}^{d} r_i.$$

Proof. Let $h_1, h_2, \ldots, h_l$ denote the height of the shelves generated by the algorithm. Similarly, let c_i and d_i denote the side of the largest and smallest cube respectively in shelf i. Clearly, since the cubes are packed according to non-increasing sizes, we have that $c_{i+1} \leq d_i$ for all $i = 1, \ldots, l-1$. Similarly, we have that $\sum_{i=1}^{l} h_i \geq r_d - c_{l+1} \geq r_d - d_l$, otherwise the algorithm would open another shelf.

We will prove the required result by induction on the number of dimensions. For $d = 2$, from the result of [8], we know that the wasted volume is at most $\delta(r_1 + r_2)$. Suppose the result is true for $d-1$, hence for each $(d-1)$-dimensional facet, the wasted volume along the first $d-1$ dimensions is at most $\delta \sum_{i=1}^{d-1} r_i$. We now bound the wasted volume in d dimensions. First, the volume wasted in the topmost shelf (that cannot be packed with any cube) is at most $d_l \prod_{i=1}^{d-1} r_i$. Second, for each shelf $i \leq l$, the wasted volume in the $(d-1)$-dimensional facet contributes at most $h_i \cdot \delta(\sum_{j=1}^{d-1} r_j)$ and the additional volume wasted along the d^{th} dimension can be bounded by $(c_i - d_i) \cdot \prod_{j=1}^{d-1} r_j$. Adding all the contributions of all the shelves and the topmost unpacked shelf, we get that the total wasted volume is at most $c_1 \cdot \prod_{j=1}^{d-1} r_j + \delta(\sum_{j=1}^{d-1} r_j)$. Without loss of generality, suppose that $r_1 \leq r_2 \leq \ldots \leq r_d$. Since $c_1 \leq \delta$ and $\prod_{j=1}^{d-1} r_j \leq r_d$, the total waste is bounded by $\delta(\sum_{i=1}^{d} r_i)$, which implies the result. ☐

3.3. **Packing Large Cubes.** We now concentrate on the case of packing large cubes. In this case we are given an input list L of n cubes whose sides are at least δ. For simplicity, we split the analysis into two steps (as in [12, 33]).

Lemma 3.4. *Suppose the input list L contains large cubes only, say $a_i \geq \delta$ for all $i = 1, \ldots, n$, and there are only K different cube sizes for some constant K. Then, we can solve the cube packing problem optimally in time $O(polylog(n))$.*

Proof. Clearly the number of cubes that fit in a bin is bounded by $M = \lfloor 1/\delta^d \rfloor$. For a cube of type i, we denote its side length by α_i. Given a packing of cubes in a bin, its bin type will be described by the number of cubes of each type in that bin. This will be denoted by a vector $(x_1, x_2, \ldots, x_K)$ which means that there are x_1 cubes of type 1, x_2 cubes of type 2, up to x_K cubes of type K , in that bin. Clearly, there are at most M^K (a constant) bin types. We now show that, given a vector $(x_1, x_2, \ldots, x_K)$ such that $x_i \leq M$ for each $1 \leq i \leq K$, we can test in constant time whether it denotes a valid bin type. To see this, we show that there are only a constant number of possible positions for each cube in a bin at which it can be placed. We can then test in constant time whether a vector is valid, by exhaustively trying all possible positions for the cubes in that vector.

If a given vector of cubes fit in a bin, without loss of generality we can assume that their vertices lie at coordinates belonging to the set:

$$\mathcal{C} = \left\{ \sum_{i=1}^{K} \lambda_i \alpha_i : \lambda_i \in \mathbb{Z}_+ \text{ and } 0 \leq \sum_{i=1}^{K} \lambda_i \alpha_i \leq 1 \right\}.$$

Indeed, given a general placement of the cubes in the bin, fix a dimension $l = 1, \ldots, d$ and sort the cubes by their leftmost point in their $l-$th coordinate. Now, consider each cube in this sorted order and push them one-by-one to the left until the cube touches another cube, which always happens at a coordinate in $\mathcal{C}$. By doing this for all coordinates we obtain the claimed result. Let C denote the cardinality of the set $\mathcal{C}$. Since $\alpha_i \geq \delta$, any point $\sum_{i=1}^{K} \lambda_i \alpha_i$ in $\mathcal{C}$ will satisfy $\lambda_i \leq 1/\delta$ for all $i = 1, \ldots, K$. Thus $C \leq (1/\delta)^K$, which is a (huge) constant. Thus there are at most $(C^d)^M$ ways to specify the position of each possible cube in a bin.

As mentioned above, we can thus check in constant time if a given vector $(x_1, \ldots, x_K)$ is a bin type or not. Thus, by exhaustive enumeration we can generate in constant time a list $\mathcal{T} = \{T_1, \ldots, T_Q\}$ consisting of all valid bin types.

It only remains to see how to find an optimal packing. To this end we will use an integer program of size $O(\text{polylog}(n))$ and with fixed number of variables and constraints. For $1 \leq j \leq K$, let T_{ij} denote the number type j cubes in bin type T_i and n_j denote the number of cubes of type j in the problem instance. Let x_i correspond to the number of bins that have configuration corresponding to the type T_i. We can formulate an integer program (denoted by IP) in the x variables as follows:

$$\min \sum_{i=1}^{Q} x_i \tag{4}$$

$$\begin{aligned} \sum_{i=1}^{Q} T_{ij} x_i &\geq n_j && \text{for } j = 1, \ldots, K \\ x_i &\geq 0 && \text{for } i = 1, \ldots, Q \\ x_i &\in \mathbb{Z}_+ && \text{for } i = 1, \ldots, Q. \end{aligned}$$

Clearly the size of IP is $O(\log(n))$. Moreover, it only has a constant number of variables. Therefore, by Lenstra's algorithm for integer programming in fixed dimension [23], we can find an optimum integer solution to IP in time $O(\text{polylog}(n))$. It is straightforward to see that such a solution represents an optimal packing. □

Lemma 3.5. *Suppose the input list L contains large cubes ($a_i \geq \delta$ for all $i = 1, \ldots, n$) only. Then, for any $\varepsilon > 0$, there is a $(1+\varepsilon)$-approximation algorithm for packing cubes into a minimum number of unit cubes that runs in time $O(n \log(n))$.*

Proof. Let L denote the given instance list. The algorithm is as follows:

(i) Sort the n cubes in non-increasing order of sizes and partition them in $K = \lceil 1/(\varepsilon\delta^d) \rceil$ groups, each containing at most $Q = \lceil n/K \rceil$ cubes.

(ii) Construct an instance J by rounding up all items in each group to the largest cube in the group. By construction, the cubes in J have at most K distinct sizes.

(iii) Apply the algorithm of Lemma 3.4 to solve the rounded instance and output the packing found by it.

To analyze the algorithm, we use the argument of Fernandez de la Vega and Lueker [12]. Consider two instances J and J' derived from the sorted input list L. As in the algorithm, J is constructed by rounding up all items in each group to the largest cube in the group. On the other hand, J' is constructed by rounding down all items in each group to the smallest cube in the group. Clearly

$$\mathrm{OPT}(J') \leq \mathrm{OPT}(L) \leq \mathrm{OPT}(J).$$

Now, each cube in group g of J' has size at least that of the cubes in group $g+1$ of J. Thus, possibly placing each of the largest Q cubes from J in a bin by itself, it follows that

$$\mathrm{OPT}(J) \leq \mathrm{OPT}(J') + Q \leq \mathrm{OPT}(L) + Q \leq \lceil (1+\varepsilon)\mathrm{OPT}(L) \rceil.$$

The last inequality follows directly from $\mathrm{OPT}(L) \geq n\delta^d$ together with $Q = \lceil n/K \rceil \leq \lceil n\varepsilon\delta^d \rceil \leq \lceil \varepsilon \mathrm{OPT}(L) \rceil$. □

3.4. **Packing small and large cubes together.** In this section we prove Theorem 1.1. Although we already know how to construct almost optimal packings of only large and only small cubes separately, we cannot directly apply the algorithms in the previous sections to construct a packing in the general case. One extra step will be required to allow the combination of the previously seen algorithms.

For the sake of simplicity in the notation we work in this section only in the two-dimensional case. The extension to higher dimensions is straightforward and discussed at the end of the section.

The general asymptotic polynomial time approximation scheme for square packing is as follows:

Algorithm for square packing

(1) Let I be the input list and $\varepsilon > 0$ fixed. Let $r = \lceil 1/\varepsilon \rceil$. Consider the sequence $\varepsilon, \varepsilon^3, \ldots, \varepsilon^{2^i-1}, \ldots$ for $i = 1, \ldots, r+1$. Let $M_i = \{j : a_j \in [\varepsilon^{2^{i+1}-1}, \varepsilon^{2^i-1})\}$, for $i = 1, \ldots, r$.
(2) Take $M := M_i$ for some index $1 \leq i \leq r$ satisfying $\mathrm{Vol}(M_i) \leq \varepsilon \mathrm{Vol}(I)$. Define the set of large items as $L = \{j : a_j \geq \varepsilon^{2^i-1}\}$ and the set of small items as $S = \{j : a_j < \varepsilon^{2^{i+1}-1}\}$.
(3) Find an almost optimal packing of L as in Lemma 3.5.
(4) Partition the remaining space in the opened bins into rectangular regions using Observation 3.2. Use NFDH to pack as many square in S as possible into these rectangular free space. Let $S' \subset S$ denote the subset of the small items that could not be packed (S' could possibly be empty).
(5) Open new bins and use NFDH to pack $M \cup S'$.

From now on, we refer to this algorithm as algorithm A. We show that it achieves the desired approximation ratio and running time as given in Theorem 1.1.

Analysis of the Running Time. First, it requires time $O(n \log n)$ to sort the cubes according to their sizes. Next, it is easy too see that steps (1) and (2) take at most $O(n)$ time. By Lemma 3.5, step (3) requires $O(\mathrm{polylog}(n))$ time. Finally, the running time in steps (4) and (5) is dominated by the running time of the NFDH shelf heuristic which is also dominated by the time to sort the list of cubes and hence is at most $O(n \log n)$.

Analysis of the Algorithm. First observe that since we have $\lceil 1/\varepsilon \rceil$ groups M_i, there exists an i such that M_i satisfies $\mathrm{Vol}(M_i) \leq \varepsilon \mathrm{Vol}(I)$.

To prove that algorithm A satisfies the result in Theorem 1.1, we need to distinguish two cases depending on what the set S' was after step (4). In both cases we will show that the bound in the theorem holds.

(i) *After step (4), S' is empty.* In this case, by Lemma 3.5, the number of bins algorithm A has opened by step (4) is bounded by

$$A(L \cup S) = A(L) \leq \lceil (1+\varepsilon)\mathrm{OPT}(L) \rceil \leq \lceil (1+\varepsilon)\mathrm{OPT}(I) \rceil.$$

It only remains to account for the bins used to pack the cubes in M. Since the size of each cube in M is at most ε, and these are packed using NFDH, the number of bins required by algorithm A to pack M is at most

$$\lceil \mathrm{Vol}(M)/(1-2\varepsilon) \rceil \leq \lceil \varepsilon \mathrm{OPT}(I)/(1-2\varepsilon) \rceil \leq \lceil 2\varepsilon \mathrm{OPT}(I) \rceil$$

new bins. The last step of the inequality follows by assuming that $\varepsilon < 1/4$ (clearly, if the result holds for $\varepsilon < 1/4$, then it also holds for any $\varepsilon' \geq 1/4$). It follows that the total number of bins used by A was no more than

$$\lceil (1+3\varepsilon)\mathrm{OPT}(I) \rceil + 1.$$

(ii) *After step (4), S' is non-empty.* In this case we derive a volume argument for the bins opened to pack L. Consider a bin obtained after step (4) and let $L' \subseteq L$ denote the set of large cubes packed in this bin. Clearly $|L'| \leq (1/\varepsilon^{(2^i-1)})^2 = 1/\varepsilon^{2(2^i-1)}$. From Observation 3.2, $[0,1]^2 \setminus L'$ can be decomposed into no more than $3/\varepsilon^{(2^i-1)2}$ rectangles, these are filled in step (4) of the algorithm (by NFDH) with cubes from the set S. Lemma 3.3 tells us that for each rectangle in the partition we will cover everything but a volume of at most $2\varepsilon^{2^{i+1}-1}$ (since each rectangle in the partition has sides length no more than 1). Adding over all rectangles, the total wasted volume can be bounded by $2\varepsilon^{2^{i+1}-1} \times 3/\varepsilon^{(2^i-1)2} \leq 6\varepsilon$. This last bound implies that for each bin containing cubes in L, at least a fraction $(1-6\varepsilon)$ of its volume is filled with cubes from $L \cup S$. Moreover, by the argument in the previous case, the cubes in $M \cup (S')$ can packed such that $(1-2\varepsilon)$ fraction of volume is used except possibly for the last bin.

$$A(I) \leq \left\lceil \frac{\mathrm{Vol}(I)}{1-6\varepsilon} \right\rceil \leq (1+12\varepsilon)\mathrm{OPT}(I) + 1,$$

The last step follows by assuming without loss of generality that $\varepsilon < 1/12$. This proves Theorem 1.1 for the two-dimensional case.

The Higher Dimensional Analysis. The asymptotic approximation scheme in the d-dimensional case is almost the same as the one described above. The only difference lies in the sequence we need to consider in step (1) used to define L, M and S (the large, medium and small cubes). In d dimensions, we need to use the decomposition in Lemma 3.1. Hence, steps (1) and (2) should be replaced by:

(1') Let I be the input list and $\varepsilon > 0$. Consider the sequence (α_i) satisfying $\alpha_0 = \varepsilon$ and $\alpha_{i+1} = \alpha_i^{2d}\varepsilon$. Such sequence is:

$$\alpha_i = \varepsilon^{\frac{(2d)^{(i+1)}-1}{2d-1}} \qquad \text{for } i \geq 0.$$

Let $M_i = \{j : a_j \in [\alpha_{i+1}, \alpha_i)\}$.

(2') Take $M := M_i$ for some i such that $\text{Vol}(M_i) \leq \varepsilon\text{OPT}(I)$. Define the set of large items as $L = \{j : a_j \geq \alpha_i\}$ and the set of small items as $S = \{j : a_j < \alpha_{i+1}\}$.

Clearly, the running time of the algorithm is again polynomial in the input size if d and ε are fixed constant.

Finally, the analysis of the algorithm is exactly the same in case **(i)** and very similar in case **(ii)**. The only difference is that we will need to use the bounds from Lemma 3.1 and the definition of large and small cubes in the d-dimensional setting.

An Exact 2-Approximation Algorithm. Recently, Van Stee [32] gave a (non-asymptotic) 2-approximation algorithm for square packing. Due to the impossibility of distinguishing in polynomial time whether a collection of squares can be packed in a single bin or requires two bins [13] this constant is the best that can be obtained. We describe in what follows how a slight variation of our algorithm also achieves such approximation result, even for hypercubes.

Take ε small (in particular $\varepsilon < 1/12$ so that Theorem 1.1 holds). Let I be the instance and $\text{Vol}(I)$ be the total volume of the instance. The new algorithm should check in step (1) whether $\text{Vol}(I) > 1$ or $\text{Vol}(I) \leq 1$. In the former case the algorithm continues exactly as described before. In the latter case, step (3) in the algorithm is replaced by: *(3') Find an optimum packing of L.* Note that, from Lemma 3.4, this new step (3') can be executed in polynomial time since L only contains a constant number of items.

Clearly if $\text{Vol}(I) > 1$, then $\text{OPT}(I) \geq 2$ implying that

$$\lceil (1+\varepsilon)\text{OPT}(I) \rceil + 1 \leq 2 \cdot \text{OPT}(I).$$

Thus, in this case, our algorithm is a 2-approximation. On the other hand, if $\text{Vol}(I) \leq 1$ the modified algorithm will find a packing using no more than $\text{OPT}(I) + 1 \leq 2 \cdot \text{OPT}(I)$ bins (specifically it will use at most $\text{OPT}(I) + \lceil \text{Vol}(M)/(1-2\varepsilon) \rceil \leq \text{OPT}(I) + \lceil \varepsilon/(1-2\varepsilon) \rceil = \text{OPT}(I) + 1$ bins in case $S' = \emptyset$, and at most $\lceil \text{Vol}(I)/(1-6\varepsilon) \rceil \leq \lceil 1/(1-6\varepsilon) \rceil \leq 2$ bins in case $S' \neq \emptyset$).

4. Packing Rectangles

By our results in Section 2 we know that we cannot hope to get an asymptotic approximation scheme for the general problem of packing rectangles. Hence, we now consider a relaxed version of the problem, where we are allowed to pack in slightly larger square bins of size $1+\varepsilon$. We will give an algorithm that uses no additional bins than those required by the optimum possible packing using unit size bins. We begin by describing our algorithm.

4.1. The Algorithm.

The following algorithm packs any list I of rectangles into no more than $\text{OPT}(I)$ bins of size $(1+15\varepsilon) \times (1+15\varepsilon)$, where $\text{OPT}(I)$ is the minimum number of unit squares in which I can be packed. Of course, to obtain the exact result in Theorem 1.2 it is enough to reassign $\varepsilon \leftarrow \varepsilon/15$ and apply the algorithm to this newly defined ε.

The algorithm will first decompose the input into "large," "horizontal," "vertical," "medium," and "small," rectangles in such a way that medium rectangles are negligible. So we will pack medium rectangles in thin strips that will be added to the bins along their sides. Large, horizontal and vertical rectangles will be rounded so that there are few different large rectangles, few different widths of horizontal rectangles, and few different heights of vertical rectangles. Then, a bin of size $(1+O(\varepsilon)) \times (1+O(\varepsilon))$ will be seen as a grid consisting of cells of size $\varepsilon\varepsilon' \times \varepsilon\varepsilon'$. We will guess labelings of these cells and attempt to pack all rectangles of each type (large, horizontal and vertical) into cells having the corresponding label. After that, small rectangles will be packed in the remaining space using the NFDH heuristic. Eventually a correct labeling together with a desired packing will be found.

Algorithm for rectangle packing

Input. Let I denote the input list consisting of n rectangles to be packed and let $\varepsilon > 0$. Assume that the i^{th} rectangle has width a_i and height b_i, with $0 \leq a_i, b_i \leq 1$. Denote also by $\mathrm{Vol}(I) = \sum_{i=1}^{n} a_i b_i$, the total area of the input.
Partitioning the Input. For $j \in \{1, 2, \ldots, 2/\varepsilon\}$, let M_j denote the set of rectangles in I such that $a_i \in (\varepsilon^{2(j+1)+1}, \varepsilon^{2j+1}]$ or $b_i \in (\varepsilon^{2(j+1)+1}, \varepsilon^{2j+1}]$. Let $j_0 \in \{1, 2, \ldots, 2/\varepsilon\}$ be such that the total area of the rectangles of M_{j_0} is minimum. Let $\varepsilon' = \varepsilon^{2j_0+1}$, and define the partition $I = M_{j_0} \cup L \cup H \cup V \cup S$, where:

- $L = \{i : a_i > \varepsilon' \text{ and } b_i > \varepsilon'\}$
- $S = \{i : a_i < \varepsilon'\varepsilon^2 \text{ and } b_i < \varepsilon'\varepsilon^2\}$
- $H = \{i : a_i > \varepsilon' \text{ and } b_i < \varepsilon'\varepsilon^2\}$
- $V = \{i : a_i < \varepsilon'\varepsilon^2 \text{ and } b_i > \varepsilon'\}$

Rounding the Input. For each rectangle in I, we round every side greater than ε' up to the nearest multiple of $\varepsilon'\varepsilon$. Denote by I' the instance containing only the rounded rectangles, i.e., $L \cup H \cup V$.
Let C denote the number of distinct rectangles in L: thus L contains ℓ_i rectangles of type i, for $1 \leq i \leq C$. Also, note that the rectangles in H have a constant number of widths and the rectangles in V have a constant number of heights.
Rounding and Partitioning the Output. We define bin types by decomposing squares of size $(1 + 2\varepsilon) \times (1 + 2\varepsilon)$ as follows:

- We consider all possible packings of (large) rectangles of type i, $1 \leq i \leq C$, into a $(1 + 2\varepsilon) \times (1 + 2\varepsilon)$ square, such that the corners of the rectangles are placed at coordinates that are integer multiples of $\varepsilon'\varepsilon$.
- For each possible packing into a $(1 + 2\varepsilon) \times (1 + 2\varepsilon)$ square, we decompose the area of the bin which is still uncovered as a union of small square cells of side length $\varepsilon'\varepsilon$ (where each square is positioned at integer multiples of $\varepsilon'\varepsilon$), and consider all possible labelings of each cell with labels H or V.

Each packing of large rectangles together with a labeling of the uncovered area defines a bin type. Let K denote the total number of bin types (which is a constant that we will estimate later).
Main Loop. For each $(n_1, \ldots, n_K)$ such that $\sum_j n_j \leq n$, we attempt to construct a packing of $I' \cup S \cup M_{j_0}$ using n_j bins of type j, such that the rectangles from L are packed in the spaces reserved for them in the bin type, the cells labeled H are only used for rectangles from $H \cup S$ and the cells labeled V are only used for rectangles from $V \cup S$. Almost all rectangles in I' are packed in steps (1) to (3), rectangles in S are packed in step (4), while the rest is packed in step (5). This is done as follows.

(1) To decide whether the rectangles from L can be placed, we check that for every rectangle type i, $1 \leq i \leq C$, the number ℓ_i of type i rectangles in I' is less than or equal to the total space available for them:
$$\ell_i \leq \sum_{1 \leq j \leq K} n_j \cdot \left(\begin{array}{c} \text{number of type } i \text{ rectangles} \\ \text{positioned in type } j \text{ bins} \end{array} \right).$$

(2) We use the following algorithm for packing the rectangles from H.

(a) For each bin type j, consider the union U of the cells labeled H. Drawing horizontal lines at y-coordinates integer multiples of $\varepsilon'\varepsilon$, we can interpret U as a union of horizontal strips of height $\varepsilon'\varepsilon$ and width multiple of $\varepsilon'\varepsilon$. For each integer multiple ℓ of $\varepsilon'\varepsilon$, let $h_\ell^{(j)}$ denote the sum of the heights of the strips of width ℓ in a type j bin. Let h_ℓ denote the total height of the strips of width ℓ in the packing which we are currently constructing,

$$h_\ell = \sum_{1 \le j \le K} n_j h_\ell^{(j)}.$$

(b) We now consider the problem of packing rectangles from H with rounded widths into two-dimensional bins of height h_ℓ and width ℓ, moreover the number of different width types for rectangles in H is bounded above by a constant. To do this we will solve the following fractional strip-packing problem. Consider all *configurations* $(w_1, w_2, \ldots)$ of widths (including empty configuration) which are multiples of $\varepsilon'\varepsilon$ and sum to at most $1+\varepsilon$. Note that the number of such configurations is constant. Let A_{ir} denote the number of occurrences of the width $i\varepsilon'\varepsilon$ in configuration r. Let B_i denote the sum of all heights of the rectangles of H whose width equals $i\varepsilon'\varepsilon$. We define one variable $x_r^{(\ell)}$ for each strip width ℓ and for each configuration r whose widths sum to at most ℓ. We find, in polynomial time, a basic feasible solution to the following system of linear constraints, if it exists.

$$\begin{cases} (\forall i) & \sum_{\ell,r} A_{ir} x_r^{(\ell)} & \geq & B_i \\ (\forall \ell) & \sum_r x_r^{(\ell)} & \leq & h_\ell \\ (\forall r, \ell) & x_r^{(\ell)} & \geq & 0 \end{cases}$$

(c) We place rectangles from H in the configurations thus defined, proceeding in a greedy fashion.

(d) We cut back bins of height h_l into strips of height $\varepsilon'\varepsilon$ and place them back into the bins. If rectangle is cut we throw this rectangle away from the packing and pack it later.

(3) Similarly, we pack the rectangles of V into the parts of the bins labeled V.

Let $M_H \subseteq H$ denote the set of rectangles which either did not fit in the fractional packing after (c) or are cut in the process (d). Analogously, we define the set of rectangles $M_V \subseteq V$ which remained unpacked after step (3).

(4) We pack the rectangles of S into all the $\varepsilon'\varepsilon \times \varepsilon'\varepsilon$ cells which have available space, using the Next Fit Decreasing Height (NFDH) algorithm.

(5) We expand each bin by adding 13 thin $1 \times \varepsilon$ horizontal strips and also 13 thin $\varepsilon \times 1$ vertical strips and use them to pack the rectangles from $M_{j_0} \cup M_H \cup M_V$ using an $O(1)$-approximation algorithm such as NFDH in the horizontal strips and Next Fit Decreasing Width (NFDW) in the vertical strips.

Output. We output the best packing among all feasible packings of I thus constructed.

4.2. **Analysis of the Running Time.** The running time is relatively easy to analyze. As $j_0 \leq 2/\varepsilon$, we have that $\varepsilon' \geq \varepsilon^{4/\varepsilon} = \Omega(1)$. The number C of large rectangle types is at most $(1/\varepsilon'\varepsilon)^2 = O(1)$. A bin type can be defined by labeling each $\varepsilon'\varepsilon \times \varepsilon'\varepsilon$ cell by H, V, or $i \leq C$, thus the number K of bin types is at most $(C+2)^{\left(\frac{1+2\varepsilon}{\varepsilon'\varepsilon}\right)^2} = O(1)$, and so the number of iterations through the main loop is at most n^K which is polynomial in n. Note that since $\log(1/\varepsilon') \leq (4/\varepsilon)\log(1/\varepsilon)$, then $\left(\frac{1+2\varepsilon}{\varepsilon'\varepsilon}\right)^2 \log(C+2) = \tilde{O}(\frac{1}{(\varepsilon')^2})$. Thus, we estimate the number of iterations as

$$n^K = n^{(C+2)^{\left(\frac{1+2\varepsilon}{\varepsilon'\varepsilon}\right)^2}} = n^{2^{\left(\frac{1+2\varepsilon}{\varepsilon'\varepsilon}\right)^2 \log(C+2)}} = n^{2^{\tilde{O}(1/(\varepsilon')^2)}} = n^{2^{2^{\tilde{O}(1/\varepsilon)}}}.$$

The number of strip widths is at most $1/\varepsilon'\varepsilon = O(1)$. The number of configurations is at most $2^{2/(\varepsilon'\varepsilon)} = O(1)$. Multiplying, the number of variables in the linear program is $O(1)$. The number of constraints is at most $2/(\varepsilon'\varepsilon) = O(1)$. The coefficients $A_{i,r}$ are bounded by $O(1)$ and B_i are written on at most $O(\log(n))$ bits, hence the linear program can be solved efficiently in polylogarithmic time.

The Next Fit Decreasing Height and Next Fit Decreasing Width algorithms run in time $O(n\log(n))$.

Overall, we conclude that the algorithm thus runs in time:

$$O\left(n\log(n) \cdot \text{polylog}(n) \cdot n^{2^{2^{\tilde{O}(1/\varepsilon)}}}\right) = n^{2^{2^{\tilde{O}(1/\varepsilon)}}}.$$

4.3. **Analysis of Correctness.**

Lemma 4.1. *The area of M_{j_0} satisfies*

$$\text{Vol}(M_{j_0}) \leq \varepsilon \text{Vol}(I).$$

Proof. Each rectangle of I belongs to at most two sets M_j, thus $\sum_{1\leq j\leq 2/\varepsilon} \text{Vol}(M_j) \leq 2\text{Vol}(I)$. The minimum area is less than the average area, which is bounded by $\varepsilon\text{Vol}(I)$. □

Lemma 4.2. *Let I' denote the rounded input. If I can be packed into* OPT *unit size bins, then $I' \cup S \cup M_{j_0}$ can be packed into* OPT *bins of size $(1+2\varepsilon)\times(1+2\varepsilon)$, in such a way that any rectangle with $a'_i \geq \varepsilon'$ is positioned at an x-coordinate which is an integer multiple of $\varepsilon'\varepsilon$, and any rectangle with $b'_i \geq \varepsilon'$ is positioned at a y-coordinate which is an integer multiple of $\varepsilon'\varepsilon$.*

Proof. Consider the optimal packing of I. Define a partial order $\prec_H$ on rectangles as the transitive closure of the relation: i is in relation with i' if rectangle i, when translated horizontally to the right by $2\varepsilon'\varepsilon$, intersects i'. Note that any chain in $\prec_H$ contains at most $1/\varepsilon'$ rectangles with $a_i \geq \varepsilon'$.

Consider a linear order $\leq$ which extends $\prec_H$. We take the rectangles (such that $a_i > \varepsilon'$) one by one in that order and deal with i as follows: we extend i horizontally to the right so that its width becomes a'_i, we translate it horizontally to the right to that it is positioned at an integer multiple of $\varepsilon'\varepsilon$; then, for each $i' \geq i$ in increasing order, if i' intersects some rectangle $i'' \leq i'$ then we translate i' to the right by $2\varepsilon'\varepsilon$. This construction produces a feasible packing since rectangles are shifted every time infeasibility occurs.

By induction we can show that the number of times a rectangle i is shifted is upper bounded by the number of rectangles in a longest chain in the partially ordered set $\{j \prec_H i : a_j > \varepsilon'\}$ under the partial order $\prec_H$. Since such chains contain at most $1/\varepsilon'$ rectangles, each rectangle is translated at most $1/\varepsilon'$ times, hence in total by at most $(2\varepsilon'\varepsilon)/\varepsilon' = 2\varepsilon$. Thus, the final packing fits in bins of size $(1+2\varepsilon)\times 1$.

We then proceed similarly for rounding the b_is into b'_i. □

Lemma 4.3. *Consider a packing of I' into bins of size $(1+2\varepsilon)\times(1+2\varepsilon)$, satisfying the condition of Lemma 4.2. Consider any cell $\mathcal{C} = [m\varepsilon'\varepsilon, (m+1)\varepsilon'\varepsilon] \times [p\varepsilon'\varepsilon, (p+1)\varepsilon'\varepsilon]$ in any bin. Then either*

$H \cap \mathcal{C} = \emptyset$ or $V \cap \mathcal{C} = \emptyset$, i.e., if a rectangle in H intersects cell $\mathcal{C}$, then no rectangle in V can intersect it.

Proof. Assume, for a contradiction, that there are rectangles $i \in H \cap \mathcal{C} \neq \emptyset$ and $i' \in V \cap \mathcal{C} \neq \emptyset$. Since i has width and starting point integer multiples of $\varepsilon'\varepsilon$, it must be that i spans the whole width of $\mathcal{C}$. Similarly, i' must span the whole height of $\mathcal{C}$. However in this case i and i' must intersect, which gives a contradiction. □

Lemma 4.4. *The areas of M_H and M_V can be estimated as:*

$$\mathrm{Vol}(M_H) = O(\varepsilon)\mathrm{Vol}(I) \quad \textit{and} \quad \mathrm{Vol}(M_V) = O(\varepsilon)\mathrm{Vol}(I).$$

Proof. Let us only prove the first equation, the second is analogous. Consider the sets M_c and M_d denoting the rectangles that were discarded in steps (c) and (d) respectively. Then $M_c \cup M_d = M_H$. We prove that both $\mathrm{Vol}(M_c) = O(\varepsilon)\mathrm{Vol}(I)$ and $\mathrm{Vol}(M_d) = O(\varepsilon)\mathrm{Vol}(I)$.

To see the first equality, we recall the strip packing analysis by Kenyon and Rémila [19]. Consider a fractional strip packing (i.e. a solution to the linear program in step (2)(b)) $x_r^{(\ell)}$ for all r and ℓ. Clearly such solution has at most $(2/\varepsilon'\varepsilon)$ nonzero coordinates. Fix ℓ and let $x_1^{(\ell)}, \ldots, x_k^{(\ell)}$ be the nonzero variables corresponding to that ℓ. To construct an integer strip packing we proceed as follows: Let $x_j^{(\ell)} > 0$ be the variable corresponding to the current configuration. This configuration will be used between levels $l_j^\ell = (x_1^{(\ell)} + \varepsilon'\varepsilon^2) + \cdots + (x_{j-1}^{(\ell)} + \varepsilon'\varepsilon^2)$ and $l_{j+1}^\ell = l_j^\ell + x_j^{(\ell)} + \varepsilon'\varepsilon^2$. For each i such that $A_{ij} \neq 0$ we draw A_{ij} columns of width $i\varepsilon'\varepsilon$ going from level l_j^ℓ to level l_{j+1}^ℓ. After this is done for all ℓ and all configurations, we take all columns of width $i\varepsilon'\varepsilon$ and start filling them up with the corresponding rectangles of the same width in a greedy manner; as the maximum height of a rectangle in H is $\varepsilon'\varepsilon$, it is not difficult to see that all rectangles fit. Now, we take all rectangles whose top end belongs to the interval $(l_j^\ell - \varepsilon'\varepsilon^2, l_j^\ell]$, for any ℓ and j, and set them aside (they are put in M_c). The total area of the removed rectangles is clearly no more than

$$2 \cdot (\text{ number of constraints }) \cdot (\text{ max item height }) = 2 \cdot \frac{2}{\varepsilon'\varepsilon} \cdot \varepsilon'\varepsilon^2 = 4\varepsilon.$$

This proves that a fractional strip packing where strips of width ℓ are of height h_ℓ (the linear program in step (2)(b)), can be turned into an integer strip packing of almost all rectangles where strips of width ℓ are of height no more than h_ℓ. We can conclude that the total area of rectangles in H that may not fit after step (2)(c) is bounded by 4ε. In other words, $\mathrm{Vol}(M_c) \leq 4\varepsilon\mathrm{Vol}(I)$.

Noting that rectangles in H have height smaller than $\varepsilon'\varepsilon^2$ and the cut strips of step (2)(d) are of height $\varepsilon'\varepsilon$, the area of the rectangles put aside in step (2)(d) is no more than a fraction ε of the area of H. Therefore, $\mathrm{Vol}(M_d) = \varepsilon\mathrm{Vol}(I)$ and the total area of M_H is no more than a fraction $O(\varepsilon)$ (indeed 5ε) of the total area of the instance. □

Lemma 4.5. *The rectangles in $M_{j_0} \cup M_H \cup M_V$ fit into the thin strips added along the bins in step (5).*

Proof. Clearly the whole area of $M_{j_0} \cup M_H \cup M_V$ is no more than $O(\varepsilon)\mathrm{Vol}(I)$. Moreover, we can partition $M_{j_0} \cup M_H \cup M_V$ into two sets A and B such that:

- A contains only rectangles with $a_i < \varepsilon' < \varepsilon^2$ and $\mathrm{Vol}(A) \leq \mathrm{Vol}(M_V) + \mathrm{Vol}(M_{j_0}) \leq 6\varepsilon\mathrm{Vol}(I)$ (A contains M_V and part of M_{j_0}).
- B contains only rectangles with $b_i < \varepsilon' < \varepsilon^2$ and $\mathrm{Vol}(B) \leq \mathrm{Vol}(M_H) + \mathrm{Vol}(M_{j_0}) \leq 6\varepsilon\mathrm{Vol}(I)$ (B contains M_H and part of M_{j_0}).

As all rectangles in A have width smaller than ε^2, for small enough ε, the NFDW heuridtic packs A into the $13 \cdot \mathrm{OPT}(I)$ added strips of size $\varepsilon \times 1$. On the other hand, NFDH does the work for the rectangles in B. Note that the number of thin strips that we need to add to the $(1+2\varepsilon) \times (1+2\varepsilon)$ bins depends on the algorithm used to pack rectangles in $M_{j_0} \cup M_H \cup M_V$. □

Lemma 4.6. *Consider the two-dimensional NFDH heuristic applied to rectangles in S. When NFDH cannot place any other rectangle in a bin of size $a \times b$ then the total unused area in that bin is no more than:*

$$\varepsilon'\varepsilon^2 \cdot (a+b).$$

Proof. The result is very similar to Lemma 3.3 and to a result in [8]. □

We are now ready to give the overall analysis of the algorithm.

Proof of Theorem 1.2. Consider an input list I and let I' be the rounded input. From Lemma 4.2 we know that there is a packing of $I' \cup S \cup M_{j_0}$ into no more than $\mathrm{OPT}(I)$ bins of size $(1+2\varepsilon) \times (1+2\varepsilon)$. Consider then the optimal packing in such bins for $I' \cup S \cup M_{j_0}$ satisfying the conditions of Lemma 4.2. By Lemma 4.3 we know that in such packing all cells intersect either a rectangle in L, H or V but not two of them. In other words in a packing of $I' \cup S \cup M_{j_0}$ satisfying the conditions of Lemma 4.2 each cell is labeled either V, H, or i for $i = 1, \ldots, C$ (where C was the number of distinct large rectangles); or will have no label at all.

Clearly, our algorithm will eventually guess a labeling of the cells that coincides with the labeling in the optimal packing. At that point the algorithm will find a feasible packing of all rectangles in L and almost all rectangles in H and V. By Lemmas 4.4 and 4.5 the unpacked rectangles M_{j_0}, M_H and M_V are of small area and they can be packed in the extra space added in step (5) of the algorithm.

It only remains to see that the small rectangles S will be successfully packed by NFDH. We prove that is not possible that in step (4) a new bin is opened if already $\mathrm{OPT}(I)$ bins of size $(1+2\varepsilon) \times (1+2\varepsilon)$ have been used. We do this by a volume argument. Suppose by contradiction that such a new bin is opened in step (4). At this step we distinguish four types of $\varepsilon'\varepsilon \times \varepsilon'\varepsilon$ cells: the ones completely filled with a rectangle in L, the ones filled with only rectangles in S, the ones filled only with rectangles in H or V, and the ones partly filled with rectangles in H or V and partly with rectangles in S. By the arguments in Lemmas 4.4 and 4.6 all cells are almost filled. Namely a fraction $(1-2\varepsilon)$ of their area is filled. Overall this implies that a fraction $(1-2\varepsilon)$ of the first $\mathrm{OPT}(I)$ bins is filled, and then the total area that has been filled in the first $\mathrm{OPT}(I)$ bins of size $(1+2\varepsilon) \times (1+2\varepsilon)$ is at least

$$(1-2\varepsilon)\mathrm{OPT}(I)(1+2\varepsilon)^2 > (1-2\varepsilon)(1+4\varepsilon)\mathrm{OPT}(I) > \mathrm{OPT}(I).$$

Where the inequality follows since $\varepsilon < 1/4$.

5. A Related Rectangle Packing Problem

The algorithm presented in Section 4 is closely related to the optimization version of the following question posed by Moser [26]:

Determine the smallest number x such that any system of squares with total area 1 may be packed in parallel into a rectangle of area x.

Bounds for this problems have been obtained by Kleitman and Krieger [20] and by Novotny [28]. Indeed, the first authors proved that any such system of squares can be packed in a rectangle of size $\sqrt{2} \times \sqrt{4/3}$, while the latter author proved that they can be packed in a rectangle of area no more than 1.53.

Although we do not obtain absolute bounds (i.e., independent of the input) for this problem, our algorithm can easily be adapted to obtain approximate solutions for a related optimization problem: the minimum rectangle placement problem. The problem can be formally stated as follows:

Given a system of n rectangles with total area 1, determine the smallest number x such that all n rectangles may be packed in parallel into a rectangle R of area x.

We now proceed to prove Theorem 1.3, i.e., we present a PTAS for the minimum rectangle placement problem.

Proof of Theorem 1.3. Let I denote the instance. For each rectangle, let $a_i \in (0,1]$ denote its width and $b_i \in (0,1]$ its height; the total area of the instance, denoted by $\mathrm{Vol}(I)$, equals 1. To solve this problem note first that the techniques in Section 4 can be easily adapted to solve the underlying approximate decision problem: Given n rectangles with sides $a_i, b_i \in (0,1]$, given a rectangle R of size $a \times b$ and given $\delta > 0$, determine if all n rectangles can be packed in a rectangle of size $[(1+\delta)a+\delta] \times [(1+\delta)b+\delta]$ or they cannot be packed in R.

Consider a given $\varepsilon > 0$, and let $\alpha = \min\{\max_{i=1,\ldots,n} a_i, \max_{i=1,\ldots,n} b_i\}$. We distinguish two cases:

(i) $\alpha > \varepsilon$. In this case, we ensure that the sides of the minimum area rectangle in which all n items can be packed are at least ε. It is then enough to solve the decision problem, taking $\delta = \varepsilon^2$, for all rectangles R such that:
- Their lower-left corner is $(0,0)$ and their upper-right corner is on or below the curve $xy = 4\mathrm{Vol}(I) = 4$ (since the NFDH shelf packing heuristic can always pack I in a square of area $4\mathrm{Vol}(I)$).
- The coordinates of their upper-right corner are at least ε.
- Their sides length are integer multiples of ε^2.

The number of such rectangles is clearly no more than the total number of points with coordinates multiples of ε^2 contained in the square $[\varepsilon, 4/\varepsilon] \times [\varepsilon, 4/\varepsilon]$, which is less than $4/\varepsilon^3 \cdot 4/\varepsilon^3 = 16/\varepsilon^6$ (a slightly more careful analysis shows that the number of rectangles with the above properties is less than $\int_\varepsilon^{1/\varepsilon} \frac{4}{x} dx \cdot \frac{1}{\varepsilon^4} = \frac{4}{\varepsilon^4} \ln\left(\frac{1}{\varepsilon^2}\right) \le \frac{8}{\varepsilon^4} \ln\left(\frac{1}{\varepsilon}\right)$). Since $16/\varepsilon^6$ is constant, it suffices to solve the decision problem for a constant number of rectangles: we can then choose the one with minimum area. Since in this case a and b are guaranteed to be large, $[(1+\delta)a+\delta] \le (1+2\varepsilon)a$ and $[(1+\delta)b+\delta] \le (1+2\varepsilon)b$. Therefore the best rectangle is guaranteed to have area within a factor of $(1+O(\varepsilon))$ that of the optimal one.

(ii) $\alpha \le \varepsilon$. In this case either all rectangles are flat or all are narrow. Without loss of generality assume they are all narrow. We will pack all rectangles using the FFDH heuristic for strip packing in a strip of width $\sqrt{\varepsilon}$. A slight adaptation of the following theorem – essentially shown in [8] – proves that all items can be placed in a rectangle of area roughly equal to $\mathrm{Vol}(I)$.

Theorem 5.1. *If all rectangles in I have width less than or equal to δ, then the height $FFDH(I)$ achieved by the FFDH heuristic (on a strip of width 1) satisfies:*

$$FFDH(I) \le (1+2\delta)\mathrm{Vol}(I) + 1.$$

In our case the width of the strip is $\sqrt{\varepsilon}$ and then $FFDH$ finds a packing in a rectangle of size:

$$\sqrt{\varepsilon} \times \left[(1+2\sqrt{\varepsilon})\frac{\mathrm{Vol}(I)}{\sqrt{\varepsilon}} + 1\right].$$

The area of such a rectangle is then

$$(1+2\sqrt{\varepsilon})\mathrm{Vol}(I) + \sqrt{\varepsilon} = (1+3\sqrt{\varepsilon}).$$

Again the result follows in this case.

6. Open Problems

Even though the result on cube packing in this paper holds in general (fixed) dimension this is not the case for the results on rectangle packing in Sections 4 and 5. An interesting open question is therefore to generalize Theorems 1.2 and 1.3 to higher dimensional rectangle packing. We have not been able to do so since the interaction of say 3-dimensional rectangles which are long in one direction but short in the other two seems much more complicated than in the two-dimensional case. Another interesting open problem is to find explicit asymptotic inapproximability gaps for the two-dimensional rectangle packing problem. For the d-dimensional rectangle packing problem, the best known approximation algorithms have an asymptotic approximation ratio of $(1.691...)^d$ [9] (which is exponential in d). However, the only lower bound on the achievable approximation ratio follows from our hardness result in Section 2. It would be very interesting to close this gap, in particular, to find out whether the lower bound construction can be extended to higher dimensions to yield a bound exponential in d, or whether an algorithm for d-dimensional rectangle packing with an approximation ratio sub-exponential in d can be obtained.

Acknowledgments. The authors would like to thank David Johnson and two anonymous referees for many invaluable comments and suggestions regarding the presentation of the paper.

The research of the second author was mostly done while he was visiting Ecole Polyechnique, supported by Institut Universitaire de France and by an AMAFC Franco-Chilean cooperation grant.

References

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

[2] P. Berman and M. Karpinski. 2003. Improved Approximation Lower Bounds on Small Occurrence Optimization. *Electronic Colloquium on Computational Complexity (ECCC)*, 10(008).

[3] A. Caprara. 2002. Packing two-dimensional bins in harmony. In *Proceeding of the 43rd IEEE Symposium on Foundations of Computer Science (FOCS)*, 490–499.

[4] A. Caprara, A. Lodi, and M. Monaci. 2003. Fast Approximation Schemes for the Two-Stage, Two-Dimensional Bin Packing Problem. Research Report OR/03/6 DEIS.

[5] C. Chekuri and S. Khanna. 1999. On multi-dimensional packing problems. In *Proceeding of the 10th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 185–194.

[6] F. R. K. Chung, M. R. Garey, and D. S. Johnson. 1982. On packing two-dimensional bins. *SIAM Journal on Algebraic and Discrete Methods*, 3:66–76.

[7] E. G. Coffman, M. R. Garey, and D. S. Johnson. 1996. Approximation algorithms for bin packing: a survey. In *D. Hochbaum, editor, Approximation algorithms for NP-hard problems*, 46–93. PWS Publishing, Boston.

[8] E. G. Coffman, M. R. Garey, D. S. Johnson and R. E. Tarjan. 1980. Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM Journal on Computing* 9:808–826.

[9] J. Csirik and A. van Vliet. 1993. An on-line algorithm for multidimensional bin packing. In *Operations Research Letters*, 13:149–158.

[10] J. Csirik and G. Woeginger. 1998. On-line packing and covering problems. In *A. Fiat and G. Woeginger Eds., Online Algorithms: The State of the Art*, Springer LNCS 1442, 147–177.

[11] L. Epstein and R. van Stee. 2004. Optimal online bounded space multidimensional packing. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 207–216.

[12] W. Fernandez de la Vega and G. Lueker. 1981. Bin packing can be solved within $1 + \varepsilon$ in linear time. *Combinatorica*, 1:349–355.

[13] C. E. Ferreira, F. K. Miyazawa, and Y. Wakabayashi. 1999. Packing squares into squares. *Pesquisa Operacional*, 19:223–237.

[14] E. Hazan, S. Safra, O. Schwartz. 2003. On the Complexity of Approximating k-Dimensional Matching. In *Proceedings of 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems and of the 7th International Workshop on Randomization and Computation (RANDOM-APPROX)*, Spinger LNCS 2764, 83–97.

[15] K. Jansen and G. Zhang. 2004. On rectangle packing: maximizing benefits. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 197–206.

[16] V. Kann. 1991. Maximum bounded 3-dimensional matching is MAX SNP-complete. *Information Processing Letters*, 37:27–35.

[17] N. Karmarkar and R. M. Karp. 1982. An efficient approximation scheme for the one-dimensional bin-packing problem. In *Proceeding of the 23rd IEEE Symposium on Foundations of Computer Science (FOCS)*, 312–320.

[18] C. Kenyon and E. Rémila. 1996. Approximate Strip Packing. In *Proceeding of the 37th IEEE Symposium on Foundations of Computer Science (FOCS)*, 31–36.

[19] C. Kenyon and E. Rémila. 2000. A Near-Optimal Solution to a Two-Dimensional Cutting Stock Problem. *Mathematics of Operations Research* 25: 645–656.

[20] D. J. Kleitman and M. Krieger. 1975. An Optimal Bound for Two-Dimensional Packing. In *Proceeding of the 16th IEEE Symposium on Foundations of Computer Science (FOCS)*, 163–168.

[21] Y. Kohayakawa, F.K. Miyazawa, P. Raghavan, and Y. Wakabayashi. 2001. Multidimensional Cube Packing. In *Proceeding of the Brazilian Symposium on Graphs, Algorithms and Combinatorics*. *Algorithmica*, to appear.

[22] R. Korf. 2003. Optimal Rectangle Packing: Initial Results. In *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS)*, 287–295.

[23] H. W. Lenstra. 1983. Integer Programming With a Fixed Number of Variables. *Mathematics of Operations Research*, 8:538–548.

[24] J. Y. T. Leung, T. W. Tam, C. S. Wong, G. H. Young, and F. Y. L. Chin. 1990. Packing Squares into a Square. *Journal of Parallel and Distributed Computing*, 10: 271–275.

[25] A. Meir and L. Moser. 1968. On Packing of Squares and Cubes. *Journal of Combinatorial Theory*, Series A, 5:126–134.

[26] L. Moser. 1965. Poorly Formulated Unsolved Problems of Combinatorial Geometry. Mimeographed.

[27] H. Murata, K. Fujiyoshi, S. Nakatake and Y. Kajitani. VLSI Module Placement Based on Rectangle-Packing by the Sequence-Pair. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15: 1518-1524, 1996.

[28] P. Novotny. 1996. On Packing of Squares Into a Rectangle. *Archivum Mathematicum* 32:75–83.

[29] E. Petrank. 1994. The hardness of approximation: gap location. *Computational Complexity*, 4:133–157.

[30] S. Seiden and R. van Stee. 2003. New bounds for multi-dimensional packing. *Algorithmica*, 36:261–293.

[31] S. Sevastianov and G. Woeginger. 1998. Makespan minimization in open shops: a polynomial time approximation scheme. Networks and matroids; Sequencing and scheduling. *Mathematical Programming Series B*, 82:191–198.

[32] R. Van Stee. 2004. An approximation algorithm for square packing. *Operations Research Letters*, 32:535–539.

[33] V. Vazirani. 2001. *Approximation Algorithms*. Springer, Berlin.

[34] G. Woeginger. 1997. There is no asymptotic PTAS for two-dimensional vector packing. *Information Processing Letters*, 64:293–297.