Parametric On-Line Algorithms for Packing Rectangles and Boxes[†]

F. K. Miyazawa
Instituto de Computação — Universidade Estadual de Campinas
Caixa Postal 6176 — 13084-971 — Campinas, SP — Brazil — fkm@ic.unicamp.br

Y. Wakabayashi
Instituto de Matemática e Estatística — Universidade de São Paulo
Rua do Matão, 1010 — 05508–900 — São Paulo, SP — Brazil — yw@ime.usp.br

Abstract

We present approximation algorithms for the following problems: the *two-dimensional bin packing* (2BP), the *three-dimensional packing problem* (TPP) and the *container packing problem* (3BP). We consider the special case in which the items to be packed are small and must be packed on-line. We say an item is *small* if each of its dimension is at most $\frac{1}{m}$ of the respective dimension of the recipient, where m is an integer greater than 1. These problems are denoted by $2\mathrm{BP_m}$, $\mathrm{TPP_m}$ and $3\mathrm{BP_m}$, and the performance of the algorithms are given in terms of the parameter m. To our knowledge, the parametric on-line algorithms we present here have the best so far achieved asymptotic performance bounds for these problems. For $2\mathrm{BP_m}$ and $\mathrm{TPP_m}$ we present algorithms with performance bound close to $(m+2)/m + 1/(m+1)^2$, and for $3\mathrm{BP_m}$ we describe an algorithm with performance bound close to $(m+3)/m + 2/m^2 + 1/(m+1)^2$. For $m = 2$ (respectively $m = 3$) these bounds are 2.112 and 3.112 (respectively 1.73 and 2.285). The results on $2\mathrm{BP_m}$ and $3\mathrm{BP_m}$ extend the results on multidimensional on-line packing presented by Coppersmith and Raghavan [6]. The result on $\mathrm{TPP_m}$ improves the bound $(m+1)/(m-1)$ due to Li and Cheng [15]. All these algorithms can be implemented to run in $O(n \log n)$ time, where n is the number of items to be packed.

Keywords: Packing, on-line packing, parametric packing, approximation algorithms, three-dimensional packing.

1 Introduction

We present fast approximation algorithms for packing problems where the items to be packed are small compared to the dimension of the recipient. More precisely, for each

[†]This work has been partially supported by CNPq: Proc. 464114/00-4, Proc. 470608/01-3, ProNEx Project 664107/1997-4, and individual research grants Proc. 300301/98-7 and Proc. 304527/89-0.

packing problem a *parameter* m (a positive integer) is given meaning that the input list consists of items (rectangles, boxes) whose each of its dimension is at most $1/m$ of the respective dimension of the recipient. In case $m \geq 2$, we say that the item is small: this is the case we are interested (the case $m = 1$ is just the general problem with no restriction on the input list).

We consider here on-line algorithms. Given a list $L = (e_1, e_2, \ldots, e_n)$ of items to be packed, an algorithm $\mathcal{A}$ to pack L is said to be *on-line* if (i) $\mathcal{A}$ packs items in the order given by the list L; (ii) $\mathcal{A}$ packs each item e_i without knowledge of any item e_j, $j > i$; and (iii) $\mathcal{A}$ never moves an item already packed. An algorithm that is not on-line is called *off-line*. A packing problem for which one is looking for an on-line or off-line algorithm is called an on-line or off-line problem, respectively.

The problems we consider are all NP-hard and have many applications, specially in job scheduling. One such problem, that can be formulated as a packing problem [16, 18], is to find an assignment of jobs to resources (*e.g.* memory or processors), where each job uses a small amount of resource during a continuous interval of time. The objective is to find a scheduling that minimizes the total time needed to execute all jobs. On-line algorithms are very important in these applications, as most of the time the operating system must schedule jobs very fast and it does not know about further jobs users can start. The use of exact algorithms in this context turns out to be infeasible, and fast approximation algorithms are highly desirable. Moreover, in most of the cases, each job requires just a small part of the resource that is available.

We consider the following parametric problems:

1. *Two-dimensional Bin Packing Problem* ($2\mathrm{BP_m}$): Given a list L of rectangles with dimensions at most $1/m$, $m \geq 2$, and rectangles of unit dimensions $R = (1, 1)$, called bins, pack the rectangles of L into a minimum number of bins.

2. *Three-dimensional Packing Problem* ($\mathrm{TPP_m}$): Given a list L of boxes, with bottom dimensions at most $1/m$, $m \geq 2$, and a box B of width 1, length 1 and infinite height, $B = (1, 1, \infty)$, pack the boxes of L into B such that the height of the packing is minimized.

3. *Container Packing Problem* ($3\mathrm{BP_m}$): Given a list L of boxes with dimensions at most $1/m$, $m \geq 2$, and boxes of unit dimensions $B = (1, 1, 1)$, called containers, pack the boxes of L into a minimum number of containers.

Li and Cheng [16] proved that $2\mathrm{BP_m}$ is NP-hard. Since this is a special case of the other problems, all the others are also NP-hard.

We present approximation algorithms for these problems: polynomial time algorithms that guarantee a certain quality of the solution found, compared to the optimum packing. Given an algorithm $\mathcal{A}$ for one of the above problems and a list of items L for the respective problem, we denote by $\mathcal{A}(L)$ the height or the number of recipients (depending on which problem is considered) of the packing generated by the algorithm $\mathcal{A}$ applied to the list L. We denote by $\mathrm{OPT}(L)$ the corresponding value of an optimum packing. We say that an algorithm $\mathcal{A}$ has *asymptotic performance bound* α if there exists a constant β such that for any instance L we have $\mathcal{A}(L) \leq \alpha \cdot \mathrm{OPT}(L) + \beta$.

For a survey on approximation algorithms for packing problems and some classic algorithms we mention here the reader is referred to Coffman, Garey and Johnson [3, 4].

Some results on the parametric versions of these problems are the following. Most of them are on off-line algorithms. In the early seventies, Johnson [11, 12] presented several off-line approximation algorithms for the one-dimensional case, some with bounds $(m+3)/(m+2)$. More recently, Csirik [7] proved that the First Fit Decreasing (FFD) algorithm has asymptotic performance bound $(m+3)/(m+2) - 1/(m(m+1)(m+2))$, when m is odd, and $(m+3)/(m+2) - 2/(m(m+1)(m+2))$, when m is even, $m \geq 5$. Coffman, Garey, Johnson and Tarjan [5] presented an algorithm with asymptotic performance bound $(m+2)/(m+1)$ for the *Strip Packing Problem*. In 1990, Li and Cheng [15] presented an algorithm for $\mathrm{TPP_m}$ with asymptotic performance bound $(m+1)/(m-1)$, $m \geq 2$.

Not as many results have been obtained for on-line problems. Johnson *et al.* [11, 13] proved that for the on-line one-dimensional bin packing problem the asymptotic performance bound of the First Fit (FF) and the Best Fit algorithm is $(m+1)/m$, Galambos [9] showed that for $m = 2$ and $m = 3$ the Harmonic algorithm H_M has bounds 1.423 and 1.302, respectively, for sufficiently large values of M.

We present here algorithms for the *two-dimensional bin packing problem* ($2\mathrm{BP_m}$), and the *three-dimensional packing problem* ($\mathrm{TPP_m}$) with asymptotic performance bound close to $(m+2)/m + 1/(m+1)^2$. For the *container packing problem* ($3\mathrm{BP_m}$) we describe an algorithm with asymptotic performance bound close to $(m+3)/m + 2/m^2 + 1/(m+1)^2$. The results on $2\mathrm{BP_m}$ and $3\mathrm{BP_m}$ extend the results of Coppersmith and Raghavan [6] for the on-line parametric two-dimensional bin packing and the container packing problem. The result on $\mathrm{TPP_m}$ improves the bound $(m+1)/(m-1)$ due to Li and Cheng [15].

For the general case, the best bounds known for (off-line) algorithms for these problems are the following. For the strip packing problem, an algorithm of Baker, Brown and Katseff [1] with bound 1.25; for the $2BP_m$, an algorithm of Chung, Garey and Johnson [2] with bound 2.125; for TPP_1, an algorithm of Miyazawa and Wakabayashi [18] with bound 2.67; and for $3BP_m$, algorithms with bound 4.84 of Csirik and van Vliet [8] and of Li and Cheng [14].

In the next section we present some definitions, notation and two known basic algorithms. In the subsequent sections we describe the parametric algorithms we designed, and show their asymptotic performance bounds.

2 Definitions, Notation and Basic Algorithms

We consider the Euclidean space $\mathbb{R}^3$ with the xyz coordinate system to represent the packings. An item (or recipient) e has its dimensions defined by $x(e)$, $y(e)$ and $z(e)$, where each of these dimensions is the measure in the corresponding axis of the xyz coordinate system. For the two-dimensional bin packing problem, one of these coordinate axes is not considered, and for the one-dimensional bin packing problem only one of these coordinates is considered. The reader is referred to [18] for a formal definition of three-dimensional packing.

The *area* of a rectangle r is denoted by $S(r)$, and the *volume* of a box b is denoted by $V(b)$. If $\mathcal{P}$ is a packing then we denote by $H(\mathcal{P})$ the *height* of $\mathcal{P}$, and by $\#(\mathcal{P})$ the *number of bins* used by $\mathcal{P}$. If $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_v$ are packings of disjoint lists $L_1, L_2, \ldots, L_v$, respectively, we denote the *concatenation* of these packings by $\mathcal{P}_1 \| \mathcal{P}_2 \| \ldots \| \mathcal{P}_v$.

If $\mathcal{P}$ is a packing of a list L and the inequality $\#(\mathcal{P}) \leq \frac{1}{s} \cdot S(L) + C$ holds, then we say s an *area guarantee* of $\mathcal{P}$. Analogously, if $\#(\mathcal{P}) \leq \frac{1}{v} \cdot V(L) + C$ holds, then we say v is a *volume guarantee* of $\mathcal{P}$.

In the three-dimensional packing problems we assume that there is no box with height greater than some constant Z.

Given a function $f : C \to \mathbb{R}$ and a subset $C' \subseteq C$, we denote by $f(C')$ the sum $\sum_{e \in C'} f(e)$.

The notation below is convenient to represent subsets of a list of items.

$$\mathcal{X}[p,q] := \{e : \ p < x(e) \leq q\}, \qquad \mathcal{Y}[p,q] := \{e : \ p < y(e) \leq q\}, \qquad \mathcal{Z}[p,q] := \{e : \ p < z(e) \leq q\},$$

$$\mathcal{C}\,[p_1, q_1 \ ; \ p_2, q_2] \ := \ \mathcal{X}[p_1, q_1] \cap \mathcal{Y}[p_2, q_2],$$

$$\mathcal{C}\,[p_1,q_1\ ;\ p_2,q_2\ ;\ p_3,q_3] \quad := \quad \mathcal{X}[p_1,q_1] \cap \mathcal{Y}[p_2,q_2] \cap \mathcal{Z}[p_3,q_3],$$

$$\mathcal{C}_m := \mathcal{C}\left[0,\frac{1}{m}\ ;\ 0,\frac{1}{m}\right], \quad \mathcal{C}_y^x := \{e:\ x(e) \geq y(e)\}, \quad \mathcal{C}_x^y := \{e:\ x(e) < y(e)\}.$$

2.1 Basic Algorithms for the One-dimensional Bin Packing Problem

In the description of the algorithms of the next sections we shall use some known on-line algorithms. Two of these algorithms are the NF (Next Fit) and the FF (First Fit) algorithms, both for the One-dimensional On-line Bin Packing Problem ($1\mathrm{BP_m}$). This is the following problem: given a list of items $L = (e_1, \ldots, e_n)$, each with length $x(e_i) \leq 1/m$, $m \geq 2$, and bins B with length 1, find a packing of the items of L into bins B that uses a minimum number of bins B.

Let us first describe the algorithm NF. Given a list of items $L = (e_1, \ldots, e_n)$, the algorithm considers each item in the order given by L and verifies whether it can be packed into the current bin B_i (the first one being B_1). While this is possible, the next items are packed into B_i. When an item cannot be packed into B_i this item is packed into a new bin B_{i+1} that becomes the current bin. The algorithm halts when all items have been packed, returning the packing $(B_1, \ldots, B_k)$, where B_k is the last bin that has been created.

Note that the algorithm NF always tries to pack an item in the last bin (current bin), and once it is not the current bin anymore, the algorithm never visits it again. The algorithm FF can be seen as an improvement of the algorithm NF, as it tries to pack an item in any of the previously generated bins. It can be described as follows. Let $\{B_1, B_2, \ldots, B_k\}$ be the packing generated by the algorithm FF, where e_{i-1} is the last item that has been packed. To pack the next item e_i, the algorithm finds the smallest index j, $1 \leq n$, such that e_i can be packed into the bin B_j. If there is no such a bin, the item e_i is packed into a new bin B_{k+1}. This process is repeated until all the items of L have been packed.

The algorithms NF and FF can be implemented to run in linear time and $O(n \log n)$ time respectively. They have asymptotic performance bound 2 and 1.7, respectively [10, 11].

The algorithms we shall describe use the above on-line algorithms as subroutines. For simplicity, we may describe them as being off-line, assuming implicitly that a transformation into an on-line algorithm can be done. To illustrate one such transformation, consider the following algorithm.

Algorithm MNF

// *For the One-dimensional Bin Packing Problem* ($1\mathrm{BP_m}$).

// *Input:* A list $L \subset \mathcal{X}[0, \frac{1}{m}]$.

// *Output:* A packing of L into bins B.

1 (Partition L into sublists) Let $L_i \leftarrow L \cap \mathcal{T}_i, \quad i = 1, 2, 3,$
where $\mathcal{T}_1 = \mathcal{X}[\frac{1}{m+1}, \frac{1}{m}]$, $\mathcal{T}_2 = \mathcal{X}[\frac{1}{m+2}, \frac{1}{m+1}]$ and $\mathcal{T}_3 = \mathcal{X}[0, \frac{1}{m+2}]$.

2 $\mathcal{P} \leftarrow \mathrm{NF}(L_1) \cup \mathrm{NF}(L_2) \cup \mathrm{NF}(L_3)$.

3 Return $\mathcal{P}$.

End Algorithm.

The algorithm MNF is off-line, but can be easily transformed into an on-line algorithm, as follows (consider the sets $\mathcal{T}_i$ as above).

Algorithm MNF′

// *For the One-dimensional Bin Packing Problem* ($1\mathrm{BP_m}$).

// *Input:* A list $L \subset \mathcal{X}[0, \frac{1}{m}]$, $L = (e_1, \ldots, e_n)$.

// *Output:* A packing of L into bins B.

1 $\mathcal{P}_i \leftarrow \emptyset$, $i = 1, 2, 3$. (Each $\mathcal{P}_i$ will be a packing of the items in $L \cap \mathcal{T}_i$.)

2 for $k \leftarrow 1$ to n do

2.1 Let i be an integer such that $e_k \in \mathcal{T}_i$, $i \in \{1, 2, 3\}$.

2.2 Pack e_k into a bin in $\mathcal{P}_i$ using the algorithm NF. That is, if possible pack the item e_k into the current bin in $\mathcal{P}_i$; if necessary, take a new bin and make it the current bin in $\mathcal{P}_i$.

3 Return $\mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3$.

End algorithm.

We note that the above example shows that in the on-line version, once the type (size) of an item is checked, a decision of *where* and *how* to pack it can be made immediately: certain bins (rooms) —either new or already existing— are used; and this information is given by the off-line description. Thus, the computational time complexity of all on-line algorithms presented here will be the same as of the corresponding off-line version.

The transformation of the off-line algorithms of the next sections into on-line algorithms can be done as illustrated above. Since the direct description of on-line algorithms may hide the ideas behind them, and makes the analysis more complicated, we prefered to leave to the reader the design of the corresponding on-line versions. In the statements of the theorems on the performance of an algorithm we refer to its on-line version.

3 Two-dimensional Bin Packing Problem

In this section we present an on-line parametric algorithm to pack rectangles into two-dimensional bins of unit capacity. The on-line algorithm with the best known asymptotic performance bound for this problem is due to Li and Cheng [14] and Csirik and van Vliet [8] with an approximation factor of 2.86. First, we describe the algorithm $\mathrm{FF}_p^{(2)}$, with parameter $0 < p < 1$, presented by Li and Cheng [14] which we use as subroutine for our algorithm. This algorithm uses a rounding set $\mathcal{S}_p = \{1, p, p^2, \ldots, p^i, \ldots\}$. The height $y(r)$ of each rectangle r is rounded up to the nearest value in $\mathcal{S}_p$, say $p^i \in \mathcal{S}_p$, and it is then called an i-rectangle. Considering their width $x(r)$, such rectangles are packed (as one-dimensional items) into levels of height p^i, called i-levels. In the following, we describe this algorithm.

Algorithm $\mathrm{FF}_p^{(2)}$

// *For the Two-dimensional Bin Packing Problem* ($2\mathrm{BP}_\mathrm{m}$).

// *Input:* A list of rectangles $L = (r_1, \ldots, r_n)$.

// *Output:* A packing of L into bins $R = (1, 1)$.

1 For $j \leftarrow 1$ to n do

- **1.1** Let i be an integer such that r_j is an i-rectangle.
- **1.2** Consider the rectangle r_j as an item of width $x(r_j)$ and the i-levels generated until this step as one-dimensional bins of length 1. Use the algorithm FF to pack r_j into the i-levels.
- **1.3** If a new i-level is generated in step 1.2, say $p^i \in \mathcal{S}_p$, then use the algorithm FF to pack this i-level inside the current two-dimensional bins. Otherwise, pack the level into a new bin.

2 Return the packing generated in step 1.

End Algorithm.

This algorithm packs the rectangles side by side along the x-axis, and is also denoted by $\mathrm{FF}_p^{(x2)}$. The analogous version of the algorithm $\mathrm{FF}_p^{(2)}$ that generates a packing with strips in the y direction is denoted by $\mathrm{FF}_p^{(y2)}$.

Li and Cheng proved that the algorithm $\mathrm{FF}_p^{(2)}$ has an asymptotic performance bound that can be made as close to 2.89 as desired, but for that one has to take p close to 1 and such that $p^s = \frac{1}{2}$ for some s.

We are now ready to describe the main algorithm of this section, called $2\mathrm{D}_m$. The basic idea behind this algorithm is to subdivide the list L into sublists, then use the algorithm $\mathrm{FF}_p^{(2)}$ with an appropriate value of p to each of these sublists. The list L is subdivided into sublists $L_1, \ldots, L_8$, as indicated in Figure 1. A formal description of these sublists is given in the description of the algorithm.

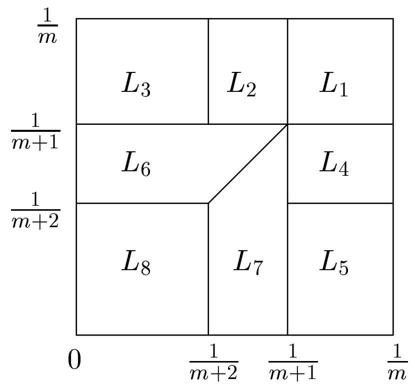

Figure 1: Partition of the list L performed by the algorithm $2\mathrm{D}_m$.

Algorithm $2\mathrm{D}_m$

// *For the Two-dimensional Bin Packing Problem* ($2\mathrm{BP}_\mathrm{m}$).

// *Input:* A list of rectangles $L = (r_1, \ldots, r_n) \subset \mathcal{C}_m$.

// *Output:* A packing of L into bins $R = (1, 1)$.

1 Subdivide the list L into sublists $L_1, \ldots, L_8$ in the following way:

$$\begin{aligned}
L_1 &\leftarrow L \cap \mathcal{C}\left[\tfrac{1}{m+1}, \tfrac{1}{m} \; ; \; \tfrac{1}{m+1}, \tfrac{1}{m}\right], \\
L_2 &\leftarrow L \cap \mathcal{C}\left[\tfrac{1}{m+2}, \tfrac{1}{m+1} \; ; \; \tfrac{1}{m+1}, \tfrac{1}{m}\right], \\
L_3 &\leftarrow L \cap \mathcal{C}\left[0, \tfrac{1}{m+2} \; ; \; \tfrac{1}{m+1}, \tfrac{1}{m}\right], \\
L_4 &\leftarrow L \cap \mathcal{C}\left[\tfrac{1}{m+1}, \tfrac{1}{m} \; ; \; \tfrac{1}{m+2}, \tfrac{1}{m+1}\right], \\
L_5 &\leftarrow L \cap \mathcal{C}\left[\tfrac{1}{m+1}, \tfrac{1}{m} \; ; \; 0, \tfrac{1}{m+2}\right], \\
L_6 &\leftarrow L \cap \mathcal{C}\left[0, \tfrac{1}{m+1} \; ; \; \tfrac{1}{m+2}, \tfrac{1}{m+1}\right] \cap \mathcal{C}_x^y, \\
L_7 &\leftarrow L \cap \mathcal{C}\left[\tfrac{1}{m+2}, \tfrac{1}{m+1} \; ; \; 0, \tfrac{1}{m+1}\right] \cap \mathcal{C}_y^x, \\
L_8 &\leftarrow L \cap \mathcal{C}_{m+2}.
\end{aligned}$$

2 $\mathcal{P}_i \leftarrow \mathrm{FF}_{1/m}^{(x2)}(L_i), \quad i = 1, 2, 3.$

3 $\mathcal{P}_i \leftarrow \mathrm{FF}_{1/m}^{(y2)}(L_i), \quad i = 4, 5.$

4 $\mathcal{P}_6 \leftarrow \mathrm{FF}_{1/(m+1)}^{(x2)}(L_6).$

5 $\mathcal{P}_7 \leftarrow \mathrm{FF}_{1/(m+1)}^{(y2)}(L_7).$

6 $\mathcal{P}_8 \leftarrow \mathrm{FF}_p^{(2)}(L_8), \quad$ for $\quad p_m = \frac{m(m+2)}{(m+1)^2}.$

7 $\mathcal{P} \leftarrow \mathcal{P}_1 \cup \ldots \cup \mathcal{P}_8.$

8 Return $\mathcal{P}$.

End Algorithm.

The following theorem shows that the algorithm $2\mathrm{D}_m$ has asymptotic performance bound $\frac{m+2}{m} + \frac{1}{(m+1)^2}$. The drawback of this algorithm is the additive constant β that depends on m. When m increases, the asymptotic performance bound becomes very close to 1, but the value of β increases. For simplicity, in the following we denote the value p_m without the subscript m. Note that fixing p as a constant—instead of a function of m—the factor we loose due to the rounding also turns out to be constant. The value of p, defined in step 6 and used to generate packing $\mathcal{P}_8$ by the algorithm $\mathrm{FF}_p^{(2)}$, was chosen to be the minimum value that guarantees that packing $\mathcal{P}_8$ has the same area guarantee as of packings $\mathcal{P}_2, \ldots, \mathcal{P}_7$.

The following lemma is used in the proof of the next theorem and also in the proof of Theorem 4.1 and Theorem 5.1.

Lemma 3.1 *Suppose A, B, a, b are real numbers such that $a > 0$ and $0 < A < B < 1$. Then*

$$\frac{a+b}{\max\{a,\, A\cdot a + B\cdot b\}} \le 1 + \frac{1-A}{B}.$$

Proof. Let $\alpha(a,b) := (a+b)/\max\{a,\, A\cdot a + B\cdot b\}$. Suppose $A\cdot a + B\cdot b \le a$. In this case, $b/a \le 1 - A/B$ and $\alpha(a,b) = (a+b)/a = 1 + b/a \le 1 + (1-A)/B$.

Assume now that $A\cdot a + B\cdot b \ge a$. In this case, $\alpha(a,b) = (a+b)/(A\cdot a + B\cdot b)$. Note that since $A\cdot a + B\cdot b \ge a$, we cannot have $b \le 0$. In fact, if $b \le 0$, then $a \le A\cdot a + B\cdot b \le A\cdot a < a$, a contradiction. Thus, we must have $b > 0$.

Dividing by a both the numerator and the denominator of the expression that defines $\alpha(a,b)$, we have $\alpha(a,b) = (1 + b/a)/(A + B\cdot b/a)$. Thus, it suffices to study the function $H(\mu) = (1+\mu)/(A + B\cdot\mu)$, where $\mu > 0$ and $A + B\cdot\mu \ge 1$. This last inequality comes from the hypothesis that $A\cdot a + B\cdot b \ge a$ and $\mu = b/a$.

Since $H'(\mu) < 0$ and $H''(\mu) > 0$ when $A < B$, it follows that H is a decreasing convex function when $A < B$. Since $\mu \ge (1-A)/B$, the function $H(\mu)$ attains its maximum at $\mu = (1-A)/B$. Thus, $H(\mu) \le H((1-A)/B) = 1 + (1-A)/B$.

Putting together the results obtained above, we obtain $\alpha(a,b) \le 1 + (1-A)/B$.

$\square$

Theorem 3.2 *For any list L of rectangles with dimensions at most $\frac{1}{m}$,*

$$2\mathrm{D}_m(L) \le \alpha_m \cdot \mathrm{OPT}(L) + \beta_m,$$

where $\alpha_m \le \frac{m+2}{m} + \frac{1}{(m+1)^2}$ and $\beta_m = (m+2)^2 + 7$.

Proof. Denote by $\mathcal{P}_{2-8}$ the packing $\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_8$. For the list L_1 the algorithm $2\mathrm{D}_m$ packs m^2 rectangles in each bin, except perhaps in the last. In this case we have an optimum packing for the list L_1 with area guarantee $(m/(m+1))^2$. Thus, the following inequalities hold

$$\begin{aligned} \mathrm{OPT}(L_1) &= \#(\mathcal{P}_1), && (1)\\ \#(\mathcal{P}_1) &\le \left(\frac{m+1}{m}\right)^2 S(L_1) + 1. && (2) \end{aligned}$$

For the lists $L_2, \ldots, L_7$, the algorithm applies the corresponding version of the algorithm $\mathrm{FF}_p^{(2)}$ (according to steps 2–5), obtaining an area guarantee of at least $m/(m+2)$. That is,

$$\#(\mathcal{P}_i) \leq \left(\frac{m+2}{m}\right) S(L_i) + 1\,,\; i = 2, \ldots, 7. \tag{3}$$

For the packing $\mathcal{P}_8$, we prove, in what follows, the following inequality:

$$\#(\mathcal{P}_8) \leq \left(\frac{m+2}{m}\right) S(L_8) + (m+2)^2. \tag{4}$$

Let L_8^k be the set of rectangles in L_8 with width in $(p^{k+1}, p^k]$, $k \geq 0$, $p = \frac{m(m+2)}{(m+1)^2}$ and n_k the number of strips generated. Since each strip, except perhaps the last, is filled with boxes attaining length at least $1 - \frac{1}{m+2}$,

$$\begin{aligned} S(L_8^k) &= \sum_{r \in L_8^k} x(r) \cdot y(r) \\ &> p^{k+1} \cdot \left(1 - \frac{1}{m+2}\right) \cdot (n_k - 1) \\ &> p^{k+1} \cdot \left(\frac{m+1}{m+2}\right) \cdot (n_k - 1)\,. \end{aligned}$$

Therefore,

$$\begin{aligned} S(L_8) &= \sum_{k \geq 0} S(L_8^k) \\ &> \sum_{k \geq 0} p^{k+1} \cdot \left(\frac{m+1}{m+2}\right) \cdot (n_k - 1) \\ &= \left(\frac{m+1}{m+2}\right) p \left(\sum_{k \geq 0} p^k n_k - \sum_{k \geq 0} p^k\right). \end{aligned}$$

Since $\sum_{k \geq 0} p^k n_k$ is the sum of the widths of all strips in each bin and the width filled by strips in each bin is at least $\left(1 - \frac{1}{m+2}\right)$, except perhaps in the last, we have:

$$\begin{aligned} S(L_8) &\geq \left(\frac{m+1}{m+2}\right) p \left(\left(1 - \frac{1}{m+2}\right)\left(\#(\mathcal{P}_8) - 1\right) - \frac{1}{1-p}\right) \\ &\geq \left(\frac{m+1}{m+2}\right) p \left(\frac{m+1}{m+2}\right)\left(\#(\mathcal{P}_8) - 1\right) - \left(\frac{m+1}{m+2}\right)\left(\frac{p}{1-p}\right). \end{aligned}$$

Now using the fact that $p = \frac{m(m+2)}{(m+1)^2}$, and $\frac{p}{1-p} = m(m+2)$, substituting these values in the last inequality we obtain

$$\begin{aligned} S(L_8) &\geq \frac{m}{m+2}\Big(\#(\mathcal{P}_8) - 1 - (m+1)(m+2)\Big) \\ &\geq \frac{m}{m+2}\Big(\#(\mathcal{P}_8) - m^2 - 3m - 3\Big) \\ &\geq \frac{m}{m+2}\Big(\#(\mathcal{P}_8) - (m+2)^2\Big). \end{aligned}$$

From inequalities (3) and (4), we have

$$S(L \setminus L_1) \geq \left(\frac{m}{m+2}\right)\big(\#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_8) - (m+2)^2 - 6\big)\,. \tag{5}$$

Note that we have the final packing of L divided into two parts. For one part we have an asymptotic optimum packing with area guarantee $(m/(m+1))^2$, and for the other part we have a packing with area guarantee $m/(m+2)$.

Let $a := \#(\mathcal{P}_1) - 1$ and $b := \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_8) - (m+2)^2 - 6$. If $a \leq 0$ then the result holds, since inequality (5) implies

$$\begin{aligned} \#(\mathcal{P}) &= \#(\mathcal{P}_1) + \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_8) \\ &\leq 1 + \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_8) \\ &\leq \left(\frac{m+2}{m}\right) S(L \setminus L_1) + (m+2)^2 + 7 \\ &\leq \alpha_m \cdot S(L) + (m+2)^2 + 7 \\ &\leq \alpha_m \cdot \mathrm{OPT}(L) + (m+2)^2 + 7. \end{aligned}$$

So, assume that $a > 0$, and consider the following inequality that can be derived from (2) and (5):

$$S(L) \geq \left(\frac{m}{m+1}\right)^2 \cdot a + \left(\frac{m}{m+2}\right) \cdot b\,. \tag{6}$$

Set $A := \left(\frac{m}{m+1}\right)^2$ and $B := \frac{m}{m+2}$. Since $\mathrm{OPT}(L) \geq S(L)$, from inequality (6) we conclude that $\mathrm{OPT}(L) \geq A \cdot a + B \cdot b$. This inequality together with the fact that $\mathrm{OPT}(L) \geq \mathrm{OPT}(L_1) = \#(\mathcal{P}_1) = a + 1$, give us the following inequality:

$$\mathrm{OPT}(L) \geq \max\{a,\, A \cdot a + B \cdot b\}\,. \tag{7}$$

Since $\#(\mathcal{P}) = \#(\mathcal{P}_1) + \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_8) = a + b + (m+2)^2 + 7$, we have

$$\#(\mathcal{P}) \leq \alpha_m \cdot \mathrm{OPT}(L) + \beta_m,$$

where $\alpha_m = (a+b)/\max\{a,\ A \cdot a + B \cdot b\}$ and $\beta_m = (m+2)^2 + 7$.

Note that $0 < A < B < 1$. Thus, we can apply Lemma 3.1 and conclude that $\alpha_m \leq 1+(1-A)/B$. Substituting the values of A and B we have $\alpha_m \leq (m+2)/m+1/(m+1)^2$. □

4 Three Dimensional Packing Problem

Li and Cheng [15] were the first authors to develop approximation algorithms for the three-dimensional packing problem. In [17] they present an on-line algorithm with asymptotic performance bound that can be made as close to 2.89 as desired, the best one so far. In this section we present a family of algorithms, denoted by $\mathrm{TP}_{m,p}$, for the parametric version $\mathrm{TPP_m}$, and we show that the asymptotic performance bound of these algorithms can be made as close to $(m+2)/m + 1/(m+1)^2$ as desired.

The algorithm $\mathrm{TP}_{m,p}$ uses a rounding strategy that is similar to the strategy presented by Li and Cheng in [17]. In this strategy the height of each box is rounded up to the nearest value $p^i \cdot Z$, for $i \geq 0$ and $0 < p < 1$. The value Z is an upper bound for the height of any box in L. Each box of height $p^i \cdot Z$, called i-box, is packed into a level of height $p^i \cdot Z$, called i-level. The packing in levels is made by the algorithm $2\mathrm{D}_m$ (for $2\mathrm{BP_m}$).

Algorithm $\mathrm{TP}_{m,p}$

// *For the Three-dimensional Packing Problem* ($\mathrm{TPP_m}$).

// *Input:* A list of boxes $L = (b_1, \ldots, b_n)$, $b_i \in \mathcal{C}_m$.

// *Output:* A packing of boxes in L into a box $B = (1, 1, \infty)$.

1 $\mathcal{P} \leftarrow \emptyset$.

2 For $i \leftarrow 1$ to n do

2.1 Let $j \geq 0$ be an integer such that b_i is a j-box.

2.2 Let $\mathcal{N}_j$ be the set of j-levels generated so far.

2.3 Use the algorithm $2\mathrm{D}_m$ to pack b_i in levels $\mathcal{N}_j$, visualizing each level of $\mathcal{N}_j$ as a rectangle (bin) of unit dimensions and b_i as a rectangle $(x(b_i), y(b_i))$. If necessary, generate a new j-level into $\mathcal{P}$ to pack b_i.

3 Return $\mathcal{P}$.

End algorithm.

The packing generated by the algorithm $\mathrm{TP}_{m,p}$ is constituted of two parts: a partial packing with asymptotic bound that can be made very close to the optimum and volume guarantee $\frac{1}{p}\left(\frac{m}{m+1}\right)^2$, of boxes in $L \cap \mathcal{C}\left[\frac{1}{m+1}, \frac{1}{m} ; \frac{1}{m+1}, \frac{1}{m}\right]$; and another packing with volume guarantee $\frac{1}{p}\left(\frac{m}{m+2}\right)$, of the remaining boxes. As p can be taken very close to 1, we can have a performance bound close to the one obtained for the algorithm $2\mathrm{D}_m$. This is shown in the next theorem.

Theorem 4.1 *Let L be a list of boxes with bottom dimensions at most $\frac{1}{m}$ and height at most Z. Then, for any real number p, $0 < p < 1$, we have*

$$\mathrm{TP}_{m,p}(L) \leq \alpha_{m,p} \cdot \mathrm{OPT}(L) + \beta_{m,p} Z,$$

where $\lim_{p \to 1} \alpha_{m,p} = \alpha(\mathrm{TP}_{m,p}) \leq \frac{m+2}{m} + \frac{1}{(m+1)^2}$ *and* $\beta_{m,p} = \frac{(m+1)^2+7}{1-p}$.

Proof. For $j \geq 0$ let L^j be the set of j-boxes. For $i = 1, \ldots, 8$ let L_i^j be a partition of L^j as defined in step 1 of the algorithm $2\mathrm{D}_m$, and set $L_i := \cup_j L_i^j$. Let $\mathcal{P}_i^j$ be the set of levels generated in the packing of the boxes in L_i^j; let $\mathcal{P}_i = \mathcal{P}_i^0 \| \mathcal{P}_i^1 \| \ldots$, and let $\#(\mathcal{P}_1^j)$ be the number of levels in packing $\mathcal{P}_1^j$. Denote by $\mathcal{P}_{2-8}$ the packing $\mathcal{P}_2 \| \ldots \| \mathcal{P}_8$. Since we cannot pack more than m^2 boxes of $\cup_j \mathcal{P}_1^j$ side by side in the same level and at most one j-level of $\mathcal{P}_1^j$ can have less than m^2 boxes, we have

$$\begin{aligned}
\mathrm{OPT}(L) &\geq \mathrm{OPT}(L_1) \\
&\geq \sum_j p^{j+1} Z(\#(\mathcal{P}_1^j) - 1) \\
&= p \cdot \left(H(\mathcal{P}_1) - \frac{Z}{1-p} \right).
\end{aligned} \tag{8}$$

Considering volume inequalities, we have

$$\begin{aligned}
V(L_1) &= \sum_j V(L_1^j) \\
&\geq \sum_j p^{j+1} \cdot Z \cdot S(L_1^j) \\
&\geq \sum_j p^{j+1} \cdot Z \cdot \left(\#(\mathcal{P}_1^j) - 1\right) \cdot \left(\frac{m}{m+1}\right)^2 \qquad \text{(from inequality (2))}
\end{aligned}$$

$$\geq \left(\frac{m}{m+1}\right)^2 \cdot p \left(\sum_j p^j \cdot Z \cdot \#(\mathcal{P}_1^j) - Z \sum_j p^j\right)$$

$$\geq \left(\frac{m}{m+1}\right)^2 \cdot p \left(H(\mathcal{P}_1) - \frac{Z}{1-p}\right). \tag{9}$$

Analogously, for the sublists $L_2, \ldots, L_8$, using inequalities (3) and (4) we have

$$V(L_2 \cup \ldots \cup L_8) \geq \left(\frac{m}{m+2}\right) \cdot p \left(H(\mathcal{P}_2 \| \ldots \| \mathcal{P}_8) - \frac{(m+2)^2+6}{1-p} Z\right). \tag{10}$$

Let $a := H(\mathcal{P}_1) - \frac{Z}{1-p}$ and $b := H(\mathcal{P}_{2-8}) - \frac{(m+2)^2+6}{1-p} Z$. If $a \leq 0$ then the result holds, since inequality (10) implies

$$\begin{aligned} H(\mathcal{P}) &= H(\mathcal{P}_1) + H(\mathcal{P}_2 \| \ldots \| \mathcal{P}_8) \leq \frac{Z}{1-p} + H(\mathcal{P}_2 \| \ldots \| \mathcal{P}_8) \\ &\leq \frac{1}{p}\left(\frac{m+2}{m}\right) V(L \setminus L_1) + \frac{(m+2)^2+7}{1-p} Z \\ &\leq \frac{1}{p}\left(\frac{m+2}{m} + \frac{1}{(m+1)^2}\right) \cdot V(L) + \frac{(m+2)^2+7}{1-p} Z \\ &\leq \frac{1}{p}\left(\frac{m+2}{m} + \frac{1}{(m+1)^2}\right) \cdot \mathrm{OPT}(L) + \frac{(m+2)^2+7}{1-p} Z. \end{aligned}$$

Now assume that $a > 0$, and consider the following inequality that can be derived from (9) and (10):

$$V(L) \geq \left(\frac{m}{m+1}\right)^2 \cdot p \cdot a + \left(\frac{m}{m+2}\right) \cdot p \cdot b. \tag{11}$$

Set $A := \left(\frac{m}{m+1}\right)^2$ and $B := \frac{m}{m+2}$. Thus, from (11) we get $\mathrm{OPT}(L) \geq V(L) \geq A \cdot p \cdot a + B \cdot p \cdot b$. Since from (8) we have $\mathrm{OPT}(L) \geq p \cdot a$, combining these results we have

$$\mathrm{OPT}(L) \geq p \cdot \max\{a,\ A \cdot a + B \cdot b\}\ .$$

Proceeding analogously as in the previous section, we have

$$H(\mathcal{P}) \leq \alpha_{p,m} \cdot \mathrm{OPT}(L) + \beta_{m,p} \cdot Z, \tag{12}$$

where $\alpha_{p,m} = \frac{1}{p} \cdot (a+b)/\max\{a,\ A \cdot a + B \cdot b\}$ and $\beta_{p,m} = \frac{(m+2)^2+7}{1-p}$.

Now we can apply Lemma 3.1 and conclude that $\alpha_{p,m} \leq \frac{1}{p} \cdot \left(\frac{m+2}{m} + \frac{1}{(m+1)^2}\right)$. Thus, $\lim_{p \to 1} \alpha_{p,m} \leq \frac{m+2}{m} + \frac{1}{(m+1)^2}$, and the proof is complete.

□

5 Container Packing Problem

In 1989 Coppersmith and Raghavan [6] presented the first approximation algorithm for the container packing problem. They presented an algorithm with asymptotic performance bound 6.25. The algorithms with the best asymptotic performance bound known for this problem, 4.84, is due to Li and Cheng [14] and Csirik and van Vliet [8]. In this section we present a parametric on-line algorithm with asymptotic performance bound $(m+3)/m + 2/m^2 + 1/(m+1)^2$.

First, we describe an algorithm used as subroutine for the $3\mathrm{BP_m}$, called H3D (Hybrid 3-D bin packing). This algorithm uses the same strategy used by the algorithm HFF (Hybrid First Fit) presented by Chung, Garey and Johnson [2]. It uses an algorithm for $\mathrm{TPP_m}$ to generate levels, and then uses an algorithm for $1\mathrm{BP_m}$ to pack the levels into containers.

Algorithm H3D

// *Input Subroutines:* A level oriented algorithm $\mathcal{A}_{\mathrm{TPP}}$ for $\mathrm{TPP_m}$ and algorithm $\mathcal{A}_{\mathrm{UNI}}$ for $1\mathrm{BP_m}$.

// *Input:* A list of boxes L.

// *Output:* A packing of L into containers $B = (1, 1, 1)$.

1 $\mathcal{P} \leftarrow \mathcal{A}_{\mathrm{TPP}}(L)$.

2 Let $\mathcal{N}$ be the set of levels in $\mathcal{P}$.

3 Apply algorithm $\mathcal{A}_{\mathrm{UNI}}$ to pack the levels of $\mathcal{N}$ into containers B. Each level $N \in \mathcal{N}$, with height z_N, is seen as a one-dimensional item of height z_N, and each container B is seen as a one-dimensional bin of height 1. Let $\mathcal{P}_{\mathrm{H3D}}$ be the packing generated with this procedure.

4 Return $\mathcal{P}_{\mathrm{H3D}}$.

End algorithm.

Note that if the input subroutines (algorithms $\mathcal{A}_{\mathrm{UNI}}$ and $\mathcal{A}_{\mathrm{TPP}}$) are on-line, then the algorithm H3D can also be transformed into an on-line algorithm.

The next algorithm uses the algorithm H3D, with subroutines $2\mathrm{D}_m$ and FF, and therefore we assume it to be on-line. The input list L is partitioned into five sublists, $L_1, \ldots, L_5$, and the algorithm maintains five types of containers, one for each sublist. The bins of the final packing are partitioned into five types, one for each sublist, i.e., the boxes of sublist

L_i are packed into bins of type i, $i = 1, \ldots, 5$. Denote by $\overline{b}$ a box defined as the box b with height rounded up in the following way: the height of the boxes for list L_i, $i = 1, \ldots, 4$ are rounded up to a value $h_i := 1/(m + i - 1)$; for the boxes in sublist L_5, we define a value q_m, $0 < q_m < 1$, and round up the height of a box $b \in L_5$ to the nearest value in the set $\mathcal{S}_q$. Once a box b has its height rounded up, it is packed in a level of height $\overline{b}$ by means of the algorithm $2\mathrm{D}_m$, considering all levels of this height. If a new level of heigth $\overline{b}$ is generated, this level is packed into bins of the same type, using the algorithm FF. In order not to loose a constant factor because of the roundings, as considered for the parameter p in the algorithm $2\mathrm{D}_m$, q_m is taken as a function of m.

Algorithm $3\mathrm{D}_m$

// *For the Three-dimensional Bin Packing Problem* ($3\mathrm{BP_m}$).

// *Input:* A list of boxes L with dimensions at most $\frac{1}{m}$.

// *Output:* A packing of L into containers $B = (1, 1, 1)$.

1 Subdivide the list L into sublists $L_1, \ldots, L_5$ in the following way

$$L_1 \leftarrow L \cap \mathcal{C}\left[\tfrac{1}{m+1}, \tfrac{1}{m}\ ;\ \tfrac{1}{m+1}, \tfrac{1}{m}\ ;\ \tfrac{1}{m+1}, \tfrac{1}{m}\right];$$

$$L_2 \leftarrow L \cap \mathcal{Z}[\tfrac{1}{m+1}, \tfrac{1}{m}] \setminus L_1;$$

$$L_3 \leftarrow L \cap \mathcal{Z}[\tfrac{1}{m+2}, \tfrac{1}{m+1}];$$

$$L_4 \leftarrow L \cap \mathcal{Z}[\tfrac{1}{m+3}, \tfrac{1}{m+2}];$$

$$L_5 \leftarrow L \setminus (L_1 \cup \ldots \cup L_4).$$

2 $\mathcal{P}_i \leftarrow \mathrm{H3D}(\mathrm{TP}_{m,q_i}, \mathrm{FF}, L_i)$, for $i = 1, \ldots, 5$, with $q_1 = \frac{1}{m}$, $q_i = \frac{1}{m+i-2}$, for $i = 2, \ldots, 4$ and $q_5 = \frac{(m+1)(m+3)}{(m+2)^2}$;

5 $\mathcal{P} \leftarrow \mathcal{P}_1 \cup \ldots \cup \mathcal{P}_5$.

6 Return $\mathcal{P}$.

End algorithm.

We note that this algorithm,a s describer, is not on-line, but its transformation into an on-line algorithm is simple.

Theorem 5.1 *Let L be any list of boxes with dimensions at most $\frac{1}{m}$, $m \geq 2$. Then*

$$3\mathrm{D}_m(L) \leq \alpha_m \cdot \mathrm{OPT}(L) + \mathcal{O}(m^4),$$

where $\alpha_m \leq \frac{m+3}{m} + \frac{2}{m^2} + \frac{1}{(m+1)^2}$.

Proof. Denote by $\mathcal{P}_{2-5}$ the packing $\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_5$. For the list L_1 the algorithm packs m^3 rectangles in each bin, except perhaps in the last. In this case we have an optimum packing for the list L_1 with volume guarantee $(m/(m+1))^3$. Therefore, the following inequalities can be obtained:

$$\begin{aligned} \mathrm{OPT}(L_1) &= \#(\mathcal{P}_1), && (13)\\ \#(\mathcal{P}_1) &\le \left(\frac{m+1}{m}\right)^3 V(L_1) + 1. && (14) \end{aligned}$$

Let N_2 be the number of levels generated by the algorithm $\mathrm{TP}_{m,p}$ in the packing of L_2. Since the algorithm TP uses the algorithm $2\mathrm{D}_m$ to pack boxes of L_2 into levels of height $\frac{1}{m}$, we have

$$\begin{aligned} V(L_2) &\ge \left(\frac{1}{m+1}\right) \cdot S(L_2)\\ &\ge \left(\frac{1}{m+1}\right) \left[\left(\frac{m}{m+2}\right) \cdot N_2 - \beta_m\right] \quad \text{(from inequality 5)}\\ &\ge \left(\frac{1}{m+1}\right)\left(\frac{m}{m+2}\right) \cdot (m \cdot (\#(\mathcal{P}_2) - 1) - \beta_m)\\ &= \left(\frac{m^2}{(m+1)(m+2)}\right) \#(\mathcal{P}_2) - \gamma'_m, \end{aligned}$$

where γ'_m is a $\mathcal{O}(m)$ function. The same inequality can be proved for packings $\mathcal{P}_3$ and $\mathcal{P}_4$. That is,

$$V(L_i) \ge \frac{m^2}{(m+1)(m+2)} \#(\mathcal{P}_i) - \gamma'_m, \quad i = 3, 4.$$

Now, consider the packing $\mathcal{P}_5$. Denote by N_5^i the number of i-levels generated in the packing of L_5^i, $i \ge 0$. These levels are generated by the algorithm $\mathrm{TP}_{m,q}$ (with $q := q_5$) which uses the algorithm $2\mathrm{D}_m$ to pack the boxes into levels. Therefore, by Theorem 3.2 we have

$$\begin{aligned} V(L_5^i) &\ge q^{i+1} S(L_5^i)\\ &\ge q^{i+1} \left[\left(\frac{m}{m+1}\right)^2 N_5^i - \beta_m\right] \quad \text{(from inequalities (3) and (5))}. \end{aligned}$$

Therefore,

$$V(L_5) = \sum_{i \ge 0} V(L_5^i)$$

$$
\begin{aligned}
&\geq \sum_{i\geq 0} q^{i+1}\left[\left(\frac{m}{m+1}\right)^2 N_5^i - \beta_m\right] \\
&= q\left[\left(\frac{m}{m+1}\right)^2 \sum_{i\geq 0} q^i N_5^i - \beta_m/(1-q)\right] \\
&\geq q\left[\left(\frac{m}{m+1}\right)^2 \left(1-\frac{1}{m+3}\right)(\#(\mathcal{P}_5)-1) - \beta_m/(1-q)\right].
\end{aligned}
$$

Substituing q, we have

$$
\#(\mathcal{P}_5) \leq \frac{(m+1)(m+2)}{m^2} V(L_5) + \gamma''_m,
$$

where $\gamma''_m = \mathcal{O}(m^4)$.

Now we can consider the final packing divided into two parts. In one part we have an optimum packing with volume guarantee $A := (m/(m+1))^3$, and the other part we have a packing with volume guarantee $B := m^2/((m+1)(m+2))$.

Let $a := \#(\mathcal{P}_1) - 1$ and $b := \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_5) - (3\gamma'_m + \gamma''_m)$. If $a \leq 0$ then the result holds, since we have

$$
\begin{aligned}
\#(\mathcal{P}) &= \#(\mathcal{P}_1) + \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_5) \\
&\leq 1 + \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_5) \\
&\leq \left(\frac{(m+1)(m+2)}{m^2}\right) V(L \setminus L_1) + \gamma_m \\
&\leq \left(\frac{m+3}{m} + \frac{2}{m^2}\right) V(L) + \gamma_m,
\end{aligned}
$$

where $\gamma_m := 3\gamma'_m + \gamma''_m + 1$.

If $a > 0$, from inequality (13) and the volume inequalities obtained for each sublist, we have

$$
\mathrm{OPT}(L) \geq \max\{a,\, A\cdot a + B\cdot b\}. \tag{15}
$$

Since $\#(\mathcal{P}) = \#(\mathcal{P}_1) + \#(\mathcal{P}_2 \cup \ldots \cup \mathcal{P}_5) = a + b + \gamma_m$, we have

$$
\#(\mathcal{P}) \leq \alpha_m \cdot \mathrm{OPT}(L) + \gamma_m,
$$

where $\alpha_m = (a+b)/\max\{a,\, A\cdot a + B\cdot b\}$.

Since $0 < A < B < 1$, we can apply Lemma 3.1 and conclude that $\alpha_m \leq 1 + (1-A)/B$. Substituting the values of A and B, we have $\alpha_m \leq \frac{m+3}{m} + \frac{2}{m^2} + \frac{1}{(m+1)^2}$.

∎

6 Conclusion

We note that the off-line version of the main algorithms presented in this paper are based on the algorithm FF (and its variants). And, as we mentioned, FF can be implemented to run in $O(n \log n)$, where n is the number of itens to be packed. Furthermore, the computational complexity of the off-line algorithms is maintained for their on-line versions. Thus, the corresponding parametric on-line algorithms can be implemented to run in $O(n \log n)$.

The asymptotic performance bounds of these algorithms are summarized in the Table shown below. In the first column we indicate the best previously known bounds (for the general case), and in the other columns we indicate the bounds achieved by the parametric on-line algorithms. We note that for $m > 1$ these are the best known bounds achieved by on-line algorithms for the problems considered.

Problem	general case	$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$
2D bin packing	2.125 [2]	3.25	2.112	1.73	1.54	1.428	1.354	1.302
3D Packing	2.67 [18, 19]	3.25	2.112	1.73	1.54	1.428	1.354	1.302
Container packing	4.84 [8, 14]	6.25	3.112	2.285	1.915	1.708	1.576	1.486

Table 1: Asymptotic performance bounds achieved by the algorithms presented here

References

[1] B. S. Baker, D. J. Brown, and H. P. Katseff. A $\frac{5}{4}$ algorithm for two-dimensional packing. Journal of Algorithms, 2 (1981) 348–368.

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

[3] E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing - an updated survey. In G. Ausiello, M. Lucertini, and P. Serafini, editors, Algorithms design for computer system design, pages 49–106, Spring-Verlag, New York, 1984.

[4] E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson. Approximation algorithms (ed. D. Hochbaum), chapter Approximation algorithms for bin packing - a survey. PWS, 1997.

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

[6] D. Coppersmith and P. Raghavan. Multidimensional on-line bin packing: algorithms and worst-case analysis. Operations Research Letters, 8 (1) (1989) 17–20.

[7] J. Csirik. The parametric behavior of the first-fit decreasing bin packing algorithm. Journal of Algorithms, 15 (1993) 1–28.

[8] J. Csirik and A. van Vliet. An on-line algorithm for multidimensional bin packing. Operations Research Letters, 13 (1993) 149–158.

[9] G. Galambos. Parametric lower bound for on-line bin-packing. SIAM Journal on Algebraic and Discrete Methods, 7 (1986) 362–367.

[10] M. R. Garey, R. L. Graham, and J. D. Ullman. Worst-case analysis of memory allocation algorithms. In Proc. 4th Annual ACM Symp. on the Theory of Computing, pages 143–150, 1972.

[11] D. S. Johnson. Near-optimal bin packing algorithms. PhD thesis, Massachusetts Institute of Technology, Cambridge, Mass., 1973.

[12] D. S. Johnson. Fast algorithms for bin packing. Journal of Computer and System Sciences, 8 (1974) 272–314.

[13] D. S. Johnson, A. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. SIAM Journal on Computing, 3 (1974) 299–325.

[14] K. Li and K-H. Cheng. A generalized harmonic algorithm for on-line multidimensional bin packing. TR UH-CS-90-2, University of Houston, January 1990.

[15] K. Li and K-H. Cheng. On three-dimensional packing. SIAM Journal on Computing, 19 (1990) 847–867.

[16] K. Li and K-H. Cheng. Static job scheduling in partitionable mesh connected systems. Journal of Parallel and Distributed Computing, 10 (1990) 152–159.

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

[18] F. K. Miyazawa and Y. Wakabayashi. An algorithm for the three-dimensional packing problem with asymptotic performance analysis. Algorithmica, 18 (1) (1997) 122–144.

[19] F.K. Miyazawa and Y. Wakabayashi. Approximation algorithms for the orthogonal z-oriented 3-D packing problem. SIAM Journal on Computing, 29 (3) (2000) 1008–1029.