

O conteúdo do presente relatório é de única responsabilidade do(s) autor(es).
The contents of this report are the sole responsibility of the author(s).

**Parametric On-Line Algorithms
for Packing Rectangles and Boxes**

F. K. Miyazawa Y. Wakabayashi

Relatório Técnico IC-00-14

Agosto de 2000

Parametric On-Line Algorithms for Packing Rectangles and Boxes*

F. K. Miyazawa[†]

Y. Wakabayashi[‡]

Abstract

We present approximation algorithms for the following problems: the *two-dimensional bin packing*, the *three-dimensional packing problem* and the *container packing problem*. 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. To our knowledge, the asymptotic performance bound of these algorithms are the best so far obtained for this special case (parametric on-line). For the above problems, (in the respective order) the algorithms we describe have bounds 2.112, 2.112, and 3.112, for $m = 2$; and bounds 1.73, 1.73 and 2.285, for $m = 3$.

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 packing problem a *parameter* (a positive integer), say m , is given meaning that the input list consists of items whose each of its dimension is at most $1/m$ of the respective dimension of the recipient. So if $m = 2$, the items have size at most half of the size 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).

An algorithm to pack this kind of items is also said to be a *parametric algorithm*, due to the parameter m . We consider here on-line algorithms: given a list $L = (e_1, e_2, \dots, e_n)$ 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 , $j > i$;

*This work has been partially supported by Project ProNEx 107/97 (MCT/FINEP), FAPESP (Proc. 96/4505-2) and CNPq individual research grants: Proc. 300301/98-7 (fkm@dcc.unicamp.br) and Proc. 304527/89-0 (yw@ime.usp.br).

[†]Instituto de Computação — Universidade Estadual de Campinas — C. Postal 6176 — 13083-970 — Campinas-SP — Brazil.

[‡]Instituto de Matemática e Estatística — Universidade de São Paulo — Rua do Matão, 1010 – Cidade Universitária — 05508-900 — São Paulo-SP — Brazil.

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 correspondingly an on-line or off-line problem.

The problems we consider have many applications, in special in job scheduling [16, 18, 19]. 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. Therefore, the use of exact algorithms turn out to be infeasible, and fast approximation algorithms are highly desirable for these applications. Moreover, in most of the cases, each job do not use the total resource it can use but a small part of it. Similar applications can be found in packing problems.

We consider (parametric versions of) the following problems:

1. *Two-dimensional Bin Packing Problem (2BP)*: Given a list L of rectangles, 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 (TPP)*: Given a list L of boxes, 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 (3BP)*: Given a list L of boxes, and boxes of unit dimensions $B = (1, 1, 1)$, called containers, pack the boxes of L into a minimum number of containers.

These problems are well known to be NP-hard. For a survey on approximation algorithms for packing problems the reader is referred to [3, 4]. The approach used here to tackle these problems is to develop approximation algorithms: polynomial time fast 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 $\text{OPT}(L)$ 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 $\mathcal{A}(L) \leq \alpha \cdot \text{OPT}(L) + \beta$.

We mention here results on the parametric versions of these problems. Off-line algorithms for these problems have been more investigated than on-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, $m \geq 5$. In [5] Coffman *et al.* presented an algorithm with asymptotic performance bound $(m+2)/(m+1)$ for the *Strip Packing Problem (SPP)*. In [15] Li and Cheng presented an algorithm for the TPP 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 $(m + 1)/m$, $m \geq 2$.

In [9], Galambos presents lower bounds for the on-line bin packing problem. For $m = 2$ and $m = 3$ he shows that the Harmonic algorithm H_M have bounds 1.423 and 1.302, respectively, for sufficiently large values of M .

For the *two-dimensional bin packing problem*, and the *three-dimensional packing problem* we present algorithms with asymptotic performance bound close to $((m + 2)/(m + 1))^2 + 2/(m(m + 1))$. For the *container packing problem* we describe an algorithm with asymptotic performance bound close to $(m^4 + 5m^3 + 10m^2 + 7m + 2)/(m^2(m + 1)^2)$. These results extend the results of Coppersmith and Raghavan [6] for the on-line parametric two-dimensional bin packing and the container packing problem.

For the general case, the best bounds known for (off-line) algorithms for these problems are the following. For SPP, an algorithm of Baker *et al.* [1] with bound 1.25; for the 2BP, an algorithm of Chung *et al.* [2] with bound 2.125; for TPP, an algorithm of Miyazawa and Wakabayashi [18] with bound 2.67; and for 3BP, 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 and notation, and in the subsequent sections we present the algorithms.

2 Definitions and Notation

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 axis is not considered, and for the one-dimensional bin packing problem only one of these coordinates is considered.

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 $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_v$ are packings of disjoint lists L_1, L_2, \dots, L_v , respectively, we denote the *concatenation* of these packings by $\mathcal{P}_1 \parallel \mathcal{P}_2 \parallel \dots \parallel \mathcal{P}_v$.

When considering the three-dimensional packing problem we assume that there is no box with height greater than a constant Z .

Given a function $f : C \rightarrow \mathbb{R}$ 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 an input list.

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

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

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

We call the value s in inequalities of the form $\#(\mathcal{P}) \leq \frac{1}{s} \cdot S(L) + C$, an *area guarantee* of the packing \mathcal{P} and the value v in inequalities of the form $\#(\mathcal{P}) \leq \frac{1}{v} \cdot V(L) + C$ a *volume guarantee* of the packing \mathcal{P} . Note that there is no loss of generality, to consider (as we did in the definition of the problems) that the limited dimensions of the recipients are 1.

2.1 One-dimensional Bin Packing Problem

In the description of an algorithm we may use some known on-line algorithms. Two of these algorithms are the NF (Next Fit) and the FF (First Fit) algorithms, both for the on-line bin packing problem. Their asymptotic performance bounds are 2 and 1.7, respectively [10, 11].

Moreover, when using an on-line algorithm as a subroutine, for simplicity, we may describe the new algorithm as being off-line, assuming implicitly that a transformation into an on-line algorithm can be done. The next example for the one-dimensional bin packing problem illustrates one such transformation.

The One-dimensional Bin Packing Problem (1BP) can be formulated as follows: given a list of items $L = (e_1, \dots, e_n)$, each with length $x(e_i)$, and bins B with length 1, find a packing of the items of L into bins B that uses the smallest possible number of bins B .

Let us first describe the algorithm NF (Next Fit). Given a list of items $L = (e_1, \dots, e_n)$, 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, \dots, B_m) , where B_m is the last generated bin.

Now consider the algorithm MNF (Modified Next Fit), also for 1BP, that uses algorithm NF as a subroutine. The input of this algorithm is a list with items whose dimensions are not greater than $\frac{1}{m}$.

Algorithm MNF

// For the One-dimensional Bin Packing Problem (1BP).
 // Input: An input list $L \subset \mathcal{X}[0, \frac{1}{m}]$.
 // Output: A packing of L into bins B .

- 1 (Partition L into sublists) Let $L_i \leftarrow L \cap \mathcal{T}_i$, $i = 1, 2, 3$,
 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 $\mathcal{P} \leftarrow \text{NF}(L_1) \parallel \text{NF}(L_2) \parallel \text{NF}(L_3)$.
- 3 Return \mathcal{P} .

End Algorithm.

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 (1BP).

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

// 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 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 makes it the current bin in \mathcal{P}_i .

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

End algorithm.

As mentioned previously, we shall present the forthcoming algorithms as off-line algorithms. The idea of how to transform them into corresponding on-line algorithms is usually straightforward. However, the direct description of on-line algorithms may hide the ideas behind them, and makes the analysis more complicated. So, we leave to the reader the design of the on-line versions, but refer to them in the statement of the theorems.

Recall that 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. Algorithm FF can be seen as an improvement of algorithm NF, as it tries to pack an item in the previously generated bins. It can be described as follows. Let $\{B_1, B_2, \dots, B_k\}$ 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 , such that e_i can be packed into the bin B_j . If there is no such 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.

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 bound that can be made as close to 2.86 as desired. First, we describe the family of algorithms $\text{FF}_p^{(2)}$, with parameter $0 < p < 1$, presented by Li and Cheng which we use as subroutine for our algorithm. This algorithm use a rounding set $\mathcal{S}_p = \{1, p, p^2, \dots, p^i, \dots\}$. In these algorithms, 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 $\text{FF}_p^{(2)}$

// For the Two-dimensional Bin Packing Problem (2BP).

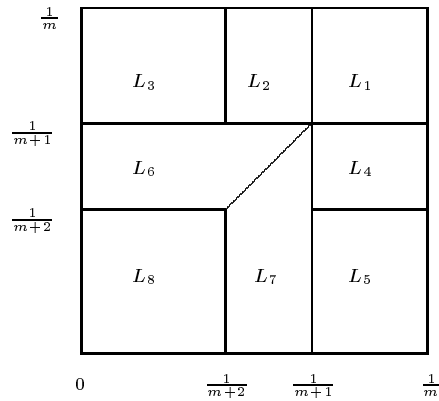
// Input: A list of rectangles $L = (r_1, \dots, r_n)$.// Output: A packing of L into bins $R = (1, 1)$.**1** For $j \leftarrow 1$ to n do**1.1** Let i be an integer such that r_j is an i -rectangle.**1.2** Consider the rectangle r_j as an item of width $x(r_j)$ and the i -levels generated until this step as one-dimensional bins of length 1. Use 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 algorithm FF to pack this i -level inside the 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. We also denote this algorithm as $\text{FF}_p^{(x2)}$. The analogous version of algorithm $\text{FF}_p^{(2)}$ that generates a packing with strips in the y direction is denoted by $\text{FF}_p^{(y2)}$.

Li and Cheng proved that algorithm $\text{FF}_p^{(2)}$ 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 .

Now, we describe an algorithm to pack small rectangles. The algorithm is described as an off-line algorithm but it can be easily transformed into an on-line one, therefore, we assume the presentation of the on-line version. The basic idea is to guarantee a minimum area utilization in each bin, for rectangles of relatively big size (using only list subdivision). For smaller rectangles we use algorithm $\text{FF}_p^{(2)}$.

In this algorithm the list L is subdivided into sublists L_1, \dots, L_8 , as indicated in Figure 1 and a specialized algorithm is executed for each of these lists.

Figure 1: Partition of the list L performed by algorithm O2D_m .

Algorithm O2D_m

// For the Two-dimensional Bin Packing Problem (2BP).
 // Input: A list of rectangles $L = (r_1, \dots, r_n) \subset \mathcal{C}_m$.
 // Output: A packing of L into bins $R = (1, 1)$.

1 Subdivide the list L into sublists L_1, \dots, L_6 in the following way:

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

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

3 $\mathcal{P}_i \leftarrow \text{FF}_{1/m}^{(y^2)}(L_i), \quad i = 4, 5.$

4 $\mathcal{P}_6 \leftarrow \text{FF}_{1/(m+1)}^{(x^2)}(L_6).$

5 $\mathcal{P}_7 \leftarrow \text{FF}_{1/(m+1)}^{(y^2)}(L_7).$

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

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

8 Return \mathcal{P} .

End Algorithm.

The following theorem shows that algorithm O2D_m has an asymptotic performance bound $\left(\frac{m+2}{m+1}\right)^2 + \frac{2}{m(m+1)}$. 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, 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 because of the rounding also turns out to be constant.

Theorem 3.1 For any list L of rectangles with dimensions at most $\frac{1}{m}$,

$$\text{O2D}_m(L) \leq \alpha_m \cdot \text{OPT}(L) + \beta_m,$$

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

Proof. Denote by \mathcal{P}_{2-8} the packing $\mathcal{P}_2 \cup \dots \cup \mathcal{P}_8$. For the list L_1 the algorithm 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 $(m/(m+1))^2$. Therefore, the following inequalities can be obtained:

$$\text{OPT}(L_1) = \#(\mathcal{P}_1), \quad (1)$$

$$\#(\mathcal{P}_1) \leq \left(\frac{m+1}{m}\right)^2 S(L_1) + 1. \quad (2)$$

For the lists L_2, \dots, L_7 , the algorithm applies a version of algorithm $\text{FF}_p^{(2)}$ for each sublist obtaining an area guarantee of at least $m/(m+2)$. That is,

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

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

$$\#(\mathcal{P}_8) \leq \left(\frac{m+2}{m}\right) S(L_8) + (m+1)^2. \quad (4)$$

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 until the length $1 - \frac{1}{m+2}$,

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

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 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:

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

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

$$S(L \setminus L_1) \geq \left(\frac{m}{m+2}\right) \#\mathcal{P}_2 \cup \dots \cup \mathcal{P}_8 - (m+1)^2 - 6. \quad (5)$$

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 $n_1 := \#\mathcal{P}_1 - 1$ and $n_2 := \#\mathcal{P}_2 \cup \dots \cup \mathcal{P}_8 - (m+1)^2 - 6$. From inequalities (1) and (9), we have

$$\text{OPT}(L) \geq \max \left\{ n_1, \left(\frac{m}{m+1}\right)^2 n_1 + \left(\frac{m}{m+2}\right) n_2 \right\}. \quad (6)$$

Since $\#\mathcal{P} = \#\mathcal{P}_1 + \#\mathcal{P}_2 \cup \dots \cup \mathcal{P}_8 = n_1 + n_2 + (m+1)^2 + 7$, we have

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

where $\alpha_m = (n_1 + n_2) / \max\{n_1, \left(\frac{m}{m+1}\right)^2 n_1 + \left(\frac{m}{m+2}\right) n_2\}$ and $\beta_m = (m+1)^2 + 7$.

Considering the two cases where the denominator attains the maximum value, we obtain $\alpha_m \leq \left(\frac{m+2}{m+1}\right)^2 + \frac{2}{m(m+1)}$. \square

4 Three Dimensional Packing Problem

Li and Cheng [15] were the first authors to develop approximation algorithms for this 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 $\text{OTP}_{m,p}$, for packing boxes with bottom dimensions not greater than $\frac{1}{m}$, whose asymptotic performance bound can be made as close to $((m+2)(m+1))^2 + 2/(m(m+1))$ as desired.

Algorithm $\text{OTP}_{m,p}$ uses a rounding strategy in the same way as used by Li and Cheng in [17]. That is, 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 algorithm O2D_m (for 2BP).

Algorithm $\text{OTP}_{m,p}$

// For the Three-dimensional Packing Problem (TPP).

// Input: A list of boxes $L = (b_1, \dots, b_n)$, $b_i \in \mathcal{C}_m$.// 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 O2D_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 of $\text{OTP}_{m,p}$ is constituted by 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 algorithm O2D_m . The following result can be proved about algorithm $\text{OTP}_{m,p}$.

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 , $0 < p < 1$, the following holds:*

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

where $\lim_{p \rightarrow 1} \alpha_{m,p} = \alpha(\text{OTP}_{m,p}) \leq \left(\frac{m+2}{m+1} \right)^2 + \frac{2}{m(m+1)}$ 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 $i = 1, \dots, 8$ let L_i^j be a partition of L^j as defined in step 1 of algorithm O2D_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 ; $\mathcal{P}_i = \mathcal{P}_i^0 \parallel \mathcal{P}_i^1 \parallel \dots$, and $\#(\mathcal{P}_1^j)$ be the number of levels in packing \mathcal{P}_1^j . Denote by \mathcal{P}_{2-8} the packing $\mathcal{P}_2 \parallel \dots \parallel \mathcal{P}_8$. Since we cannot pack more than m^2 boxes of $\cup_j \mathcal{P}_1^j$ side by side in a same level and at most one j -level of \mathcal{P}_1^j can have less than m^2 boxes, we have

$$\begin{aligned} \text{OPT}(L) &\geq \text{OPT}(L_1) \\ &\geq \sum_j p^{j+1} Z (\#(\mathcal{P}_1^j) - 1) \\ &= p \cdot \left(H(\mathcal{P}_1) - \frac{Z}{1-p} \right). \end{aligned} \tag{7}$$

Considering volume inequalities, we have

$$\begin{aligned}
 V(L_1) &= \sum_j V(L_1^j) \\
 &\geq \sum_j p^{j+1} \cdot Z \cdot S(L_1^j) \\
 &\geq \sum_j p^{j+1} \cdot Z \cdot (\#\mathcal{P}_1^j - 1) \cdot \left(\frac{m}{m+1}\right)^2 \quad (\text{from inequality (2)}) \\
 &\geq \left(\frac{m}{m+1}\right)^2 \cdot p \left(\sum_j p^j \cdot Z \cdot \#\mathcal{P}_1^j - Z \sum_j p^j \right) \\
 &\geq \left(\frac{m}{m+1}\right)^2 \cdot p \left(H(\mathcal{P}_1) - \frac{Z}{1-p} \right). \tag{8}
 \end{aligned}$$

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

$$V(L_2 \cup \dots \cup L_8) \geq \left(\frac{m}{m+2}\right) \cdot p \left(H(\mathcal{P}_2 \parallel \dots \parallel \mathcal{P}_6) - \frac{(m+1)^2 + 6}{1-p} Z \right). \tag{9}$$

Setting $h_1 := H(\mathcal{P}_1) - \frac{Z}{1-p}$ and $h_2 := H(\mathcal{P}_{2-8}) - \frac{(m+1)^2 + 6}{1-p} Z$, the following inequality can be derived from (8) and (9):

$$V(L) \geq \left(\frac{m}{m+1}\right)^2 \cdot p \cdot h_1 + \left(\frac{m}{m+2}\right) \cdot p \cdot h_2. \tag{10}$$

Since $\text{OPT}(L) \geq V(L)$, we have from (7) and (10),

$$\text{OPT}(L) \geq \max \left\{ p \cdot h_1, \left(\frac{m}{m+1}\right)^2 \cdot p \cdot h_1 + \left(\frac{m}{m+2}\right) \cdot p \cdot h_2 \right\}.$$

Proceeding analogously as in the previous section, we have

$$H(\mathcal{P}) \leq \alpha_{p,m} \cdot \text{OPT}(L) + \beta_{m,p} \cdot Z, \tag{11}$$

where $\lim_{p \rightarrow 1} \alpha_{p,m} = \left(\frac{m+2}{m+1}\right)^2 + \frac{2}{m(m+1)}$ and $\beta_{p,m} = \frac{(m+1)^2 + 7}{1-p}$. \square

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 $(m^4 + 5m^3 + 10m^2 + 7m + 2)/(m^2(m + 1)^2)$.

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

Algorithm H3D

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

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

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

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

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

3 Apply algorithm \mathcal{A}_{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}_{H3D} be the packing generated with this procedure.

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

End algorithm.

Note that if the algorithms \mathcal{A}_{UNI} and \mathcal{A}_{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 O2D_m and FF. The input list L is partitioned into five sublists, L_1, \dots, 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 , $i = 1, \dots, 5$. Denote by \bar{b} a box defined as the box b with height rounded in the following way: the height of the boxes for list L_i , $i = 1, \dots, 4$ 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 have its height rounded, it is packed in a level of height \bar{b} with algorithm O2D_m , considering all levels of this height. If a new level of height \bar{b} is generated, this level is packed into bins of the same type, using algorithm FF. In order not to loose a constant factor because of the roundings, as considered for the parameter p in algorithm O2D_m , q_m is taken as a function of m .

Algorithm O3D_m

// For the Three-dimensional Bin Packing Problem (3BP).

// 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, \dots, L_5 in the following way

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

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

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

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

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

2 $\mathcal{P}_i \leftarrow \text{H3D}(\text{OTP}_{m,q_i}, L_i, \text{FF})$, with $q_i = 1/(m+i-1)$, for $i = 1, \dots, 4$ and $q_5 = \frac{(m+1)(m+3)}{(m+2)^2}$;

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

6 Return \mathcal{P} .

End algorithm.

Again, we note that this algorithm 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 the following holds:*

$$\text{O3D}_m(L) \leq \alpha_m \cdot \text{OPT}(L) + \mathcal{O}(m^4),$$

where $\alpha_m \leq (m^4 + 5m^3 + 10m^2 + 7m + 2)/(m^2(m+1)^2)$.

Proof. Denote by \mathcal{P}_{2-5} the packing $\mathcal{P}_2 \cup \dots \cup \mathcal{P}_5$. 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:

$$\text{OPT}(L_1) = \#(\mathcal{P}_1), \tag{12}$$

$$\#(\mathcal{P}_1) \leq \left(\frac{m+1}{m} \right)^3 V(L_1) + 1. \tag{13}$$

Let N_2 be the number of levels generated by algorithm $\text{OTP}_{m,p}$ in the packing of L_2 . Since algorithm OTP uses algorithm O2D_m to pack boxes of L_2 into levels of height $\frac{1}{m}$, we have

$$\begin{aligned} V(L_2) &\geq \left(\frac{1}{m+1} \right) \cdot S(L_2) \\ &\geq \left(\frac{1}{m+1} \right) \left[\left(\frac{m}{m+2} \right) \cdot N_2 - \beta_m \right] \quad (\text{from inequality 5}) \end{aligned}$$

$$\begin{aligned}
&\geq \left(\frac{1}{m+1}\right) \left(\frac{m}{m+2}\right) \cdot (m \cdot (\#\mathcal{P}_2) - 1) - \beta_m \\
&= \left(\frac{m^2}{(m+1)(m+2)}\right) \#\mathcal{P}_2 - \gamma'_m,
\end{aligned}$$

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,

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

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 , $i \geq 0$. These levels are generated by algorithm $\text{OTP}_{m,q}$ (with $q := q_5$) which uses algorithm O2D_m to pack the boxes into levels. Therefore, by Lemma 3.1 we have

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

Therefore,

$$\begin{aligned}
V(L_5) &= \sum_{i \geq 0} V(L_5^i) \\
&\geq \sum_{i \geq 0} q^{i+1} \left[\left(\frac{m}{m+1}\right)^2 N_5^i - \beta_m \right] \\
&= q \left[\left(\frac{m}{m+1}\right)^2 \sum_{i \geq 0} q^i N_5^i - \beta_m / (1 - q) \right] \\
&\geq q \left[\left(\frac{m}{m+1}\right)^2 \left(1 - \frac{1}{m+3}\right) (\#\mathcal{P}_5) - \beta_m / (1 - q) \right].
\end{aligned}$$

Substituting 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 $(m/(m+1))^3$, and the other part we have a packing with volume guarantee $m^2/((m+1)(m+2))$.

Let $n_1 := \#\mathcal{P}_1 - 1$ and $n_2 := \#\mathcal{P}_2 \cup \dots \cup \mathcal{P}_5 - (3\gamma'_m + \gamma''_m)$. From the above inequalities, we have

$$\text{OPT}(L) \geq \max \left\{ n_1, \left(\frac{m}{m+1}\right)^3 n_1 + \frac{m^2}{(m+1)(m+2)} n_2 \right\}. \quad (14)$$

Since $\#(\mathcal{P}) = \#(\mathcal{P}_1) + \#(\mathcal{P}_2 \cup \dots \cup \mathcal{P}_5) = n_1 + n_2 + (\gamma_m)$, we have

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

where $\alpha_m \leq (m^4 + 5m^3 + 10m^2 + 7m + 2)/(m^2(m + 1)^2)$ and $\gamma_m := 3\gamma'_m + \gamma''_m + 1$. \square

6 Conclusion

The asymptotic performance bounds of the algorithms we presented here are summarized in the following table. We note that for $m > 1$ these are the best bounds known for on-line algorithms for the problems considered here. In the first column we indicate the best bounds known (to our knowledge) for on-line algorithms for the corresponding general problems.

Problem	Best Known	$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$
2D bin packing	2.86 [2]	3.25	2.112	1.73	1.54	1.428	1.354	1.302
3D Packing	2.89 [18]	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 of the on-line algorithms mentioned.

References

- [1] B. S. Baker, D. J. Brown, and H. P. Katseff. A $\frac{5}{4}$ algorithm for two-dimensional packing. *J. of Algorithms*, 2:348–368, 1981.
- [2] F. R. K. Chung, M. R. Garey, and D. S. Johnson. On packing two-dimensional bins. *SIAM J. Algebraic Discrete Methods*, 3:66–76, 1982.
- [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. Springer-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 J. Comput.*, 9:808–826, 1980.
- [6] D. Coppersmith and P. Raghavan. Multidimensional on-line bin packing: algorithms and worst-case analysis. *Oper. Res. Lett.*, 8(1):17–20, 1989.
- [7] J. Csirik. The parametric behavior of the first-fit decreasing bin packing algorithm. *J. of Algorithms*, 15:1–28, 1993.

- [8] J. Csirik and A. van Vliet. An on-line algorithm for multidimensional bin packing. *Oper. Res. Lett.*, 13:149–158, 1993.
- [9] G. Galambos. Parametric lower bound for on-line bin-packing. *SIAM J. Algebraic Discrete Methods*, 7:362–367, 1986.
- [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. *J. Comput. Syst. Sci.*, 8:272–314, 1974.
- [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 J. Comput.*, 3:299–325, 1974.
- [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 J. Comput.*, 19:847–867, 1990.
- [16] K. Li and K-H. Cheng. Static job scheduling in partitionable mesh connected systems. *J. Parallel and Distributed Computing*, 10:152–159, 1990.
- [17] K. Li and K-H. Cheng. Heuristic algorithms for on-line packing in three dimensions. *J. of Algorithms*, 13:589–605, 1992.
- [18] F. K. Miyazawa and Y. Wakabayashi. An algorithm for the three-dimensional packing problem with asymptotic performance analysis. *Algorithmica*, 18(1):122–144, 1997.
- [19] F. K. Miyazawa and Y. Wakabayashi. Approximation algorithms for the orthogonal z -oriented 3-D packing problem. *SIAM J. Comput.*, 29(3):1008–1029, 2000.