APPROXIMATION ALGORITHMS FOR THE ORTHOGONAL Z-ORIENTED THREE-DIMENSIONAL PACKING PROBLEM*

F. K. MIYAZAWA† AND Y. WAKABAYASHI‡

Abstract. We present approximation algorithms for the *orthogonal z-oriented three-dimensional packing problem* (TPP^z) and analyse their asymptotic performance bound. This problem consists in packing a list of rectangular boxes $L = (b_1, b_2, \ldots, b_n)$ into a rectangular box $B = (l, w, \infty)$, orthogonally and oriented in the z-axis, in such a way that the height of the packing is minimized. We say that a packing is oriented in the z-axis when the boxes in L are allowed to be rotated (by ninety degree) around the z-axis. This problem has some nice applications but has been less investigated than the well-known variant of it—denoted by TPP—in which rotations of the boxes are not allowed. The problem TPP can be reduced to TPP^z. Given an algorithm for TPP^z we can obtain an algorithm for TPP with the same asymptotic bound. We present an algorithm for TPP^z, called R; and three other algorithms, called LS, BS and SS, for special cases of this problem in which the instances are more restricted. The algorithm LS is for the case in which all boxes in L have square bottom, BS is for the case the box B has square bottom, and SS is for the case the box B and all boxes in L have square bottom. For an algorithm $\mathcal{A}$, we denote by $r(\mathcal{A})$ the asymptotic performance bound of $\mathcal{A}$. We show that $2.5 \leq r(R) < 2.67$, $2.5 \leq r(LS) \leq 2.528$, $2.5 \leq r(BS) \leq 2.543$ and $2.333 \leq r(SS) \leq 2.361$. The algorithms presented here have the same complexity $\mathcal{O}(n \log n)$ as the other known algorithms for these problems, but they have better asymptotic performance bounds.

Key words. approximation algorithms, three-dimensional packing, asymptotic performance bound.

AMS subject classifications. 68Q25, 52C17

1. Introduction. We present approximation algorithms for the orthogonal z-oriented three-dimensional packing problem and show results concerning their asymptotic performance bound. All algorithms described here have time complexity $\mathcal{O}(n \log n)$, where n is the number of boxes in the input list.

Let $L = (b_1, b_2, \ldots, b_n)$ be a list of rectangular boxes $b_i = (x_i, y_i, z_i)$, where x_i, y_i and z_i is the *length*, *width* and *height* of b_i, respectively. The *Orthogonal z-Oriented Three-dimensional Packing Problem* (TPP^z), can be defined as follows. Given a box $B = (l, w, \infty)$ and a list of boxes $L = (b_1, b_2, \ldots, b_n)$, find an orthogonal z-oriented packing of L into B that minimizes the total height. In the next section we define the concept of *orthogonal z-oriented packing*. For the moment, let us informally say that it is an orthogonal packing in which the boxes may be rotated around the z-axis (but may not be turned down).

A variant of TPP^z, in which the boxes may not be rotated around the z-axis, has been more investigated and is known as the *Three-Dimensional Orthogonal Packing Problem* [3, 5, 6]. We denote it by TPP. Since all packings to be mentioned here are orthogonal we omit this term. Here we show that TPP can be reduced to TPP^z. Since the uni-dimensional packing problem [2] can be reduced to TPP, it follows that both TPP and TPP^z are $\mathcal{NP}$-hard.

*This research is part of the Project ProComb (CNPq, Proc. 680065/94-6) and PCE (CNPq, Proc. 304527/89-0). It is also supported by FAPESP (Proc. 96/4505–2) and ProNEx (MCT/FINEP Proc. 107/97).

†Partially supported by CNPq individual research grant (Proc. 300301/98-7) Instituto de Computação — Universidade Estadual de Campinas — Caixa Postal 6176 — 13083-970 — Campinas, SP — Brazil (`fkm@dcc.unicamp.br`).

‡Partially supported by CNPq individual research grant (Proc. 304527/89-0). Instituto de Matemática e Estatística — Universidade de São Paulo — Rua do Matão, 1010 – Cidade Universitária — 05508–900 — São Paulo, SP — Brazil (`yw@ime.usp.br`).

If $\mathcal{A}$ is an algorithm for TPP^z or TPP and L is a list of boxes, then $\mathcal{A}(L)$ denotes the height of the packing generated by algorithm $\mathcal{A}$ when applied to a list L; and $\mathrm{OPT}(L)$ denotes the height of an optimal packing of L. We say that α is *an asymptotic performance bound* of an algorithm $\mathcal{A}$ if there exists a constant β such that for all lists L, in which all boxes have height bounded by a constant Z, the following holds: $\mathcal{A}(L) \leq \alpha \cdot \mathrm{OPT}(L) + \beta \cdot Z$. Furthermore, if for any small ϵ and any large M, both positive, there is an instance L such that $\mathcal{A}(L) > (\alpha - \epsilon)\mathrm{OPT}(L)$ and $\mathrm{OPT}(L) > M$, then we say that α is *the asymptotic performance bound* of algorithm $\mathcal{A}$. We denote by $r(\mathcal{A})$ the asymptotic performance bound of $\mathcal{A}$.

In 1990, Li and Cheng [4] presented TPP^z as a model for a *job scheduling problem in partitionable mesh connected systems*. In this problem a set of jobs $J_1, J_2, \ldots, J_n$ is to be processed in a partitionable mesh connected system that consists of $l \times w$ processing elements connected as a rectangular mesh. Each job J_i is specified by a triplet $J_i = (x_i, y_i, t_i)$ indicating that a submesh of size either (x_i, y_i) or (y_i, x_i) is required by job J_i, and t_i is its processing time. The objective is to assign the jobs to the submeshes so as to minimize the total processing time. The algorithm for TPP^z described in [4] has asymptotic performance bound $4\frac{4}{7}$.

In [3] Li and Cheng describe several algorithms for TPP: for the general case, an algorithm whose asymptotic performance bound is 3.25; and for the special case in which all boxes have square bottom, an algorithm whose asymptotic performance bound is 2.6875. In 1992, these authors [5] also presented an on-line algorithm with asymptotic performance bound that can be made as close to 2.89 as desired.

In [6] we present an algorithm for TPP whose asymptotic performance bound is less than 2.67. In this paper we describe an algorithm for TPP^z that has a similar asymptotic performance bound. We also describe an algorithm for the special case of TPP^z in which the box B has square bottom and show that its asymptotic performance bound is less than 2.528. For the case in which all boxes of L have square bottom, we present an algorithm with asymptotic performance bound less than 2.543. Moreover, for the case in which all boxes have square bottom, we present an algorithm whose asymptotic performance is less than 2.361. The algorithms we describe here for special instances of TPP^z are not straightforward simplifications of the algorithm for the general case. Each one resulted from a careful analysis of the instances under consideration.

There is a fundamental aspect in which the algorithms we have developed differ from those of Li and Cheng. Their strategy is to divide the input list into sublists and apply appropriate algorithms for each sublist, returning a packing that is a concatenation of these individual packings. The strategy we use also makes subdivisions (different ones) of the input list, but generates not only packings of each sublist but also those that are obtained by appropriate combinations of different sublists. In fact, we may say that the key idea behind our algorithms is to consider sublists which can be combined to generate better packings.

This paper is organized as follows. In Section 2 we define some basic concepts, establish the notation and discuss relations between TPP and TPP^z. In Section 3 we describe the main algorithm (for TPP^z) and analyse its asymptotic performance bound. In each of the next three sections we describe an algorithm for a special instance of TPP^z and prove results on its asymptotic performance bound.

2. Notation and basic results. Given a list of boxes $L = (b_1, \ldots, b_n)$ to be packed into a box $B = (l, w, \infty)$, we assume that each box b_i is of the form $b_i = (x_i, y_i, z_i)$, with $x_i \leq l$ and $y_i \leq w$ or $x_i \leq w$ and $y_i \leq l$ (that is, each box b_i can be packed into B in some orientation). We also assume throughout this paper that the list L consists of boxes with height bounded by a constant Z. In all algorithms mentioned here—unless otherwise stated—the input box B is assumed to be of the form $B = (l, w, \infty)$.

Given a triplet $t = (a, b, c)$, we also refer to each of its elements a, b and c as $x(t)$, $y(t)$ and $z(t)$, respectively. For each box $b_i = (x_i, y_i, z_i)$, we denote by $\rho(b_i)$ the box consisting of the triplet (y_i, x_i, z_i) and we set $\Gamma(L) = \{(c_1, c_2, \ldots, c_n) : c_i \in \{b_i, \rho(b_i)\}\}$. Given a real function $f : C \to \mathbb{R}$, and a subset $C' \subseteq C$, we denote by $f(C')$ the sum $\sum_{e \in C'} f(e)$.

Although a list is given as an ordered n-tuple of boxes, when the order of the boxes is irrelevant the corresponding list may be viewed as a set.

Note that, by using a three-dimensional coordinate system, the box $B = (l, w, \infty)$ can be seen as the region $[0, l) \times [0, w) \times [0, \infty)$ and we may define a *z-oriented packing* $\mathcal{P}$ of a list of boxes L into B as a mapping $\mathcal{P} : L' = (b_1, \ldots, b_n) \to [0, l) \times [0, w) \times [0, \infty)$, such that

$$L' \in \Gamma(L), \quad \mathcal{P}^x(b_i) + x_i \leq l \text{ and } \mathcal{P}^y(b_i) + y_i \leq w ,$$

where $\mathcal{P}(b_i) = (\mathcal{P}^x(b_i), \mathcal{P}^y(b_i), \mathcal{P}^z(b_i))$, $i = 1, \ldots, n$.

And furthermore, if $\mathcal{R}(b_i)$ is defined as

$$\mathcal{R}(b_i) = [\mathcal{P}^x(b_i), \mathcal{P}^x(b_i) + x_i) \times [\mathcal{P}^y(b_i), \mathcal{P}^y(b_i) + y_i) \times [\mathcal{P}^z(b_i), \mathcal{P}^z(b_i) + z_i),$$

then the following must hold

$$\mathcal{R}(b_i) \cap \mathcal{R}(b_j) = \emptyset \quad \forall i, j, \ 1 \leq i \neq j \leq n .$$

If in the above definition we replace $L' \in \Gamma(L)$ by $L' = L$ then we have the concept of *oriented packing* (note that the condition $L' = L$ means that the boxes in L may not be rotated around the z-axis).

In what follows, we may use the term *packing* to refer to both the z-oriented or to the oriented packing. To be precise, sometimes we should refer to a *z-oriented packing* (when some boxes are being rotated), but we simply say *packing* as this will be clear from the context. When this may cause a confusion we specify which packing we are referring to.

Given a packing $\mathcal{P}$ of L, we denote by $H(\mathcal{P})$ the *height* of the packing $\mathcal{P}$, i.e., $H(\mathcal{P}) := \max\{\mathcal{P}^z(b) + z(b) : \ b \in L\}$.

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 define the *concatenation* of these packings as a packing $\mathcal{P} = \mathcal{P}_1 \| \mathcal{P}_2 \| \ldots \| \mathcal{P}_v$ of $L = L_1 \cup L_2 \cup \ldots \cup L_v$, where $\mathcal{P}(b) = (\mathcal{P}_i^x(b), \mathcal{P}_i^y(b), \sum_{j=1}^{i-1} H(\mathcal{P}_j) + \mathcal{P}_i^z(b))$, for all $b \in L_i$, $1 \leq i \leq v$. If each list $L_i = (b_1^i, b_2^i, \ldots, b_{n_i}^i)$, $i = 1, \ldots, v$, then the **concatenation** of these lists, denoted by $L_1 \| L_2 \| \ldots \| L_v$, is the list $(b_1^1, \ldots, b_{n_1}^1, b_1^2, \ldots, b_{n_2}^2, \ldots, b_1^v, \ldots, b_{n_v}^v)$.

The following notation is used to consider sublists of the list L.

- $\mathcal{C} := \{b_i = (x_i, y_i, z_i) : 0 \leq x_i \leq l, \ \ 0 \leq y_i \leq w, \ \ z_i > 0\}$;
- $\mathcal{C}\,[p'', p' \ ; \ q'', q'] := \{b_i = (x_i, y_i, z_i) : \ p'' \cdot l < x_i \leq p' \cdot l, \ q'' \cdot w < y_i \leq q' \cdot w\}$, for $0 \leq p'' < p' \leq 1, \ \ 0 \leq q'' < q' \leq 1$;
- $\mathcal{Q}\,[p'', p' \ ; \ q'', q'] := \{b_i = (x_i, y_i, z_i) : \ b_i \in \mathcal{C}\,[p'', p' \ ; \ q'', q'] \text{ and } x_i = y_i\}$;
- $\mathcal{X} := \{b_i = (x_i, y_i, z_i) : \ y_i < x_i\}$, $\mathcal{Y} := \{b_i = (x_i, y_i, z_i) : \ y_i \geq x_i\}$;

- $\mathcal{C}_m := \mathcal{C}\left[0, \frac{1}{m}\ ;\ 0, \frac{1}{m}\right]$, $\quad \mathcal{Q}_m := \mathcal{Q}\left[0, \frac{1}{m}\ ;\ 0, \frac{1}{m}\right]$, for $m > 0$;
- $\mathcal{R}_1 := \mathcal{C}\left[0, \frac{1}{2}\ ;\ 0, \frac{1}{2}\right]$, $\mathcal{R}_2 := \mathcal{C}\left[0, \frac{1}{2}\ ;\ \frac{1}{2}, 1\right]$, $\mathcal{R}_3 := \mathcal{C}\left[\frac{1}{2}, 1\ ;\ 0, \frac{1}{2}\right]$, $\mathcal{R}_4 := \mathcal{C}\left[\frac{1}{2}, 1\ ;\ \frac{1}{2}, 1\right]$.

If $\mathcal{R}$ is a set of boxes, then we say that a box b is of *type* $\mathcal{R}$ if $b \in \mathcal{R}$ or $\rho(b) \in \mathcal{R}$.

We denote by $S(b)$ and $V(b)$ the *bottom area* (i.e., $S(b) := x(b)y(b)$) and the *volume* of the box b, respectively.

A *level* N in a packing $\mathcal{P}$ is a region $[0, l) \times [0, w) \times [Z_1, Z_2)$ in which there is a set L' of boxes such that for all $b \in L'$, $\mathcal{P}^z(b) = Z_1$ and $Z_2 - Z_1 = \max\{z(b) :\ b \in L'\}$. Sometimes we shall consider the level N as a packing of the list L'; we denote by $S(N)$ the sum $\sum_{b \in L'} S(b)$.

A *layer* (in the x-axis direction) in a level is a region $[0, l) \times [Y_1, Y_2) \times [Z_1, Z_2)$ in which there is a set L' of boxes such that, for all $b \in L'$, $\mathcal{P}^y(b) = Y_1$ and $\mathcal{P}^z(b) = Z_1$; and moreover, $Y_2 - Y_1 = \max\{y(b) :\ b \in L'\}$ and $Z_2 - Z_1 = \max\{z(b) :\ b \in L'\}$.

Relations between TPP **and** TPP^z

One way to solve TPP^z is to adapt algorithms for TPP. A simple approach is to generate for each instance $L = (b_1, b_2, \ldots, b_n)$ a new instance $\phi(L) \in \Gamma(L)$, such that $\phi(L) = (d_1, d_2, \ldots, d_n)$, where

$$d_i = \begin{cases} b_i, & \text{if } \ x_i \leq l \ \text{ and } \ y_i \leq w, \\ \rho(b_i), & \text{otherwise}, \end{cases}$$

and then apply an algorithm for TPP on the list $\phi(L)$.

For each algorithm $\mathcal{A}$ for TPP, let us denote by $\widehat{\mathcal{A}}$ the corresponding algorithm for TPP^z, as described above. That is, for every instance L of TPP^z, algorithm $\widehat{\mathcal{A}}$ applies algorithm $\mathcal{A}$ on the list $\phi(L)$. It is easy to see that the algorithm $\widehat{\mathcal{A}}$ may not preserve the asymptotic performance of the original algorithm $\mathcal{A}$.

The next result shows that there is no algorithm $\widehat{\mathcal{A}}$ for TPP^z, obtained from an algorithm $\mathcal{A}$ for TPP, as described previously, that has an asymptotic performance bound less than 3.

Proposition 2.1. *If $\widehat{\mathcal{A}}$ is an algorithm for* TPP^z *obtained from an algorithm $\mathcal{A}$ for* TPP*, as described above, then the asymptotic performance bound of $\widehat{\mathcal{A}}$ is at least 3.*

Proof. Let $L = (b_1, b_2, \ldots, b_{3k})$ and $B = (3 + 3\epsilon, 2, \infty)$ be an instance of of TPP^z, where $b_1 = b_2 = \cdots = b_{3k} = (2, 1 + \epsilon, 1)$, k is a positive integer, and ϵ is a positive small number.

First, observe that it is possible to pack L in k levels, that is, $\text{OPT}(L) \leq k$. For that, initially rotate each box in L and generate a packing putting three boxes per level. Now it suffices to note that, since $L = \phi(L)$, any algorithm $\mathcal{A}$ for TPP is such that $\widehat{\mathcal{A}}(L) \geq 3k$ (as the algorithm $\widehat{\mathcal{A}}$ packs only one box per level). ∎

Now suppose we have an algorithm $\mathcal{A}$ for TPP^z. There is a natural way to adapt it to an algorithm, say $\mathcal{A}'$, for TPP. The question is what can we say about the performance of $\mathcal{A}'$. The next result gives the answer.

Theorem 2.2. *There is a polynomial reduction of* TPP *to* TPP^z *that preserves the approximability. Moreover, this reduction also preserves the additive constant β. That is, if $\mathcal{A}$ is a polynomial algorithm for* TPP^z*, such that $\mathcal{A}(L) \leq \alpha \cdot \text{OPT}(L) + \beta \cdot Z$ then there exists a polynomial algorithm $\mathcal{A}'$ for* TPP *such that $\mathcal{A}'(L) \leq \alpha \cdot \text{OPT}'(L) + \beta \cdot Z$, where $\text{OPT}'(L)$ is the height of an optimum oriented packing of L.*

Proof. Let L, as described, and $B = (l, w, \infty)$ be an instance of TPP. Consider the following algorithm $\mathcal{A}'$. First scale B to $B' = (l', w', \infty)$ and L to L' in the same proportion, in such a way that $\min\{x(b) : b \in L'\} > w$; then apply algorithm $\mathcal{A}$ to pack L' into the box B', obtaining a packing $\mathcal{P}'$. Finally, rescale $\mathcal{P}'$ back, obtaining a packing of the original list L (into B). It is clear that $\mathcal{A}'(L) \leq \alpha \cdot \mathrm{OPT}'(L) + \beta Z$. $\square$

For all algorithms presented in the next sections, we consider, without loss of generality, that $L = \phi(L)$. That is, we may assume that the boxes in L need not be rotated to fit in the box B.

Before we present the algorithms for TPP^z, let us mention some algorithms used as subroutines and also the related results that are needed.

We denote by NFDH, the next fit decreasing height algorithm for TPP, presented by Li and Cheng in [3]. For the description of this algorithm the reader may refer to [3] or [6]. This algorithm has two variants: NFDH^x and NFDH^y. The notation NFDH is used to refer to any of these variants.

Li and Cheng [3] proved the following result.

LEMMA 2.3. *If* $L \subset \mathcal{C}\left[\frac{1}{m+1}, \frac{1}{m}\ ;\ 0, \frac{1}{m}\right]$, *then* $\mathrm{NFDH}^y(L) \leq \left(\frac{m+1}{m-1}\right)\frac{V(L)}{l \cdot w} + Z$. *The same result also holds for the algorithm* NFDH^x *when applied to a list* $L \subset \mathcal{C}\left[0, \frac{1}{m}\ ;\ \frac{1}{m+1}, \frac{1}{m}\right]$.

The following result is more general and gives as a corollary the result above [6].

LEMMA 2.4. *Let L be an instance of* TPP *and $\mathcal{P}$ be a packing of L consisting of levels $N_1, \ldots, N_v$ such that* $\min\{z(b) : b \in N_i\} \geq \max\{z(b) : b \in N_{i+1}\}$, *and* $S(N_i) \geq s \cdot l \cdot w$ *for a given constant* $s > 0$, $i = 1, \ldots, v-1$. *Then* $H(\mathcal{P}) \leq \frac{1}{s}\frac{V(L)}{l \cdot w} + Z$.

The constant s mentioned in the above lemma is called an *area guarantee* of the packing $\mathcal{P}$.

Li and Cheng presented in [4] an algorithm called LL, for instances $L \subset \mathcal{C}_m$, $m \geq 3$. We write $\mathrm{LL}(L, m)$ to indicate that we are applying the algorithm LL to a list $L \subset \mathcal{C}_m$. They proved that the following result holds for this algorithm.

LEMMA 2.5. *Let* $L \subset \mathcal{C}\left[0, \frac{1}{m}\ ;\ 0, \frac{1}{m}\right]$ *be an instance of* TPP *and $\mathcal{P}$ be a packing of L obtained by applying the algorithm* LL. *Then* $H(\mathcal{P}) \leq \left(\frac{m}{m-2}\right)\frac{V(L)}{l \cdot w} + Z$.

We give an idea of the algorithm $\mathrm{LL}(L, m)$, as we need to refer to it in the proof of Lemma 2.6. Initially, it sorts the boxes in L in non-increasing order of their height. Then it divides L into sublists $L_1, \ldots, L_v$, such that $L = L_1 \| L_2 \| \cdots \| L_v$, each sublist preserving the (non-increasing) order of the boxes and

$$
\begin{aligned}
&S(L_i) \leq \left[\left(\tfrac{m-2}{m}\right) + \left(\tfrac{1}{m}\right)^2\right] lw && \text{for } \ i = 1, \ldots, v, \\
&S(L_i) + S(first(L_{i+1})) > \left[\left(\tfrac{m-2}{m}\right) + \left(\tfrac{1}{m}\right)^2\right] lw && \text{for } \ i = 1, \ldots, v-1,
\end{aligned}
$$

where $first(L')$ is the first box in L'. Then, the algorithm LL uses a two-dimensional packing algorithm to pack each list L_i in only one level, say, N_i. The final packing is the concatenation of each of these levels.

The next lemma is used to prove lower bounds for the asymptotic performance bound of some algorithms shown here.

LEMMA 2.6. *Let $\mathcal{A}$ be an algorithm for* TPP (TPP^z) *that partitions the input list L into two sublists $L_1 \subset \mathcal{R}_4$ and $L_2 \subset \mathcal{Q}_m$, $m \geq 3$, and generates a packing $\mathcal{P} = \mathcal{P}_1 || \mathcal{P}_2$, where $\mathcal{P}_1$ is any packing of L_1 and $\mathcal{P}_2$ is a packing of L_2 using the algorithm* LL. *Then the asymptotic performance bound $r(\mathcal{A})$ of $\mathcal{A}$ is such that* $r(\mathcal{A}) \geq \frac{7m-8}{4m-8}$.

Proof. Consider a box $B = (1, 1, \infty)$. Let L be a list of boxes, $L = L_1 \cup L_2$, with $L_1 \subset \mathcal{R}_4$ and $L_2 \subset \mathcal{Q}_m$, $m \geq 3$. Let $L_1 = (b'_1, \ldots, b'_{N'})$ and $L_2 = (b''_1, \ldots, b''_{M \cdot N''})$, where

$$
b'_i = \left(\frac{1}{2} + \frac{1}{k}, \frac{1}{2} + \frac{1}{k}, 1\right), \quad \text{and} \quad b''_i = \begin{cases} \left(\frac{1}{m}, \frac{1}{m}, 1 - (i-1)\xi\right), & \text{if } i \bmod M = 1; \\ \left(\frac{1}{k}, \frac{1}{k}, 1 - (i-1)\xi\right), & \text{otherwise.} \end{cases}
$$

Recall that the algorithm LL groups the first boxes with total bottom area no greater than $\left(\frac{m-2}{m}\right)+\left(\frac{1}{m}\right)^2$. This instance was chosen in such a way that, in the packing generated by the algorithm LL, each level has M boxes whose bottom area is $\left(\frac{m-2}{m}\right)+\left(\frac{1}{k}\right)^2$.

Note that the algorithm LL divides the list L_2 into N'' sublists, each sublist consisting of one box of the form $\left(\frac{1}{m},\frac{1}{m},1-(i-1)\xi\right)$ and $M-1$ boxes of the form $\left(\frac{1}{k},\frac{1}{k},1-(i-1)\xi\right)$.

The strategy is to take the instance $L=L_1\cup L_2$ in such a way that the optimum packing consists of N' levels, each level containing one box of L_1 and the remaining space (in each level) is almost filled with boxes of L_2. Taking N' and k as very large integers, with $N''=\left\lceil\frac{3}{4}N'\frac{m}{m-2}\right\rceil$ and k a multiple of $2m$, we may choose M appropriately so that $r(\mathcal{A})$ can be made as close to $\frac{7m-8}{4m-8}$ as desired. □

Another algorithm that plays an important role for the algorithms presented here is the algorithm COMBINE. This algorithm is a slightly modified version of the algorithm COLUMN presented in [6]. This algorithm generates a partial packing of a list L. The packing consists of several stacks of boxes, referred to as *columns*. Each column is built by putting one box on top of the other, and each column consists only of boxes of type either $\mathcal{T}^1$ or $\mathcal{T}^2$.

The algorithm COMBINE is called with the parameters $(L,\mathcal{T}^1,p^1,\mathcal{T}^2,p^2)$, where $p^1=[p^1_1,p^1_2,\ldots,p^1_{n_1}]$ consists of the positions in the bottom of box B where the columns of boxes of type $\mathcal{T}^1$ should start and $p^2=[p^2_1,p^2_2,\ldots,p^2_{n_2}]$ consists of the positions in the bottom of box B where the columns of boxes of type $\mathcal{T}^2$ should start. Each point $p^i_j=(x^i_j,y^i_j)$ represents the x-axis and the y-axis coordinates where the first box (if any) of each column of the respective type must be packed. Note that the z-axis coordinate need not be specified since it may always be assumed to be 0 (corresponding to the bottom of box B). Here we are assuming that the positions p^1, p^2 and the types $\mathcal{T}^1$, $\mathcal{T}^2$ are chosen in such a way that the defined packing can always be performed.

We call *height of a column* the sum of the height of all boxes in that column.

Initially, all n_1+n_2 columns are empty, starting at the bottom of box B. At each iteration, the algorithm chooses a column with the smallest height, say a column given by the position p^i_j, and packs the next box b of type $\mathcal{T}^i$, updating the list L after each iteration. If there is no such box b, then the algorithm halts returning the partial packing $\mathcal{P}$ of L. We also say that $\mathcal{P}$ *combines* the lists of types $\mathcal{T}^1$ and $\mathcal{T}^2$.

If each box of type $\mathcal{T}^i$ has bottom area at least $s_i\cdot l\cdot w$, then $(n_1s_1+n_2s_2)\cdot l\cdot w$ is called the *combined area* of the packing generated by the algorithm COMBINE.

The following result about this algorithm holds. The proof is analogous to the one given in [6] for the algorithm COLUMN.

LEMMA 2.7. *Let $\mathcal{P}$ be the packing of $L'\subseteq L$ generated by the algorithm* COMBINE *when applied to lists of types $\mathcal{T}^1$ and $\mathcal{T}^2$ and list of positions $p^i_1,p^i_2,\ldots,p^i_{n_i}$, $i=1,2$. If $S(b)\geq s_i\cdot l\cdot w$ for all boxes b in $\mathcal{T}^i$ $(i=1,2)$, then $H(\mathcal{P})\leq\frac{1}{s_1n_1+s_2n_2}\frac{V(L')}{l\cdot w}+Z$.*

To simplify the notation, given two lists L_1 and L_2, we denote by COLUMN(L_1, p_1,L_2,p_2) the algorithm COMBINE called with parameters $(L_1||L_2,L_1,p_1,L_2,p_2)$ and assume that it returns a pair $(\mathcal{P}',L')$ where $\mathcal{P}'$ is the partial packing of $L_1||L_2$ and L' is the set of boxes packed in $\mathcal{P}'$.

Another simple algorithm that we use is the algorithm OC (one column). Given a list of boxes, say $L=(b_1,\ldots,b_n)$, this algorithm packs each box b_{i+1} on top of the box b_i, for $i=1,\ldots,n-1$. It is easy to verify the following results.

LEMMA 2.8. *If $\mathcal{P}$ is the packing generated by the algorithm* OC *when applied to a list L and s is a constant such that $S(b) \geq s \cdot l \cdot w$ for all boxes b in L, then $H(\mathcal{P}) \leq \frac{V(L)}{s \cdot l \cdot w}$.*

LEMMA 2.9. *If $\mathcal{P}$ is the packing generated by the algorithm* OC *when applied to a list L of boxes b such that $b \in \mathcal{R}_4$ and ($\rho(b) \in \mathcal{R}_4$ or $\rho(b) \notin \mathcal{C}$) then $H(\mathcal{P}) = \mathrm{OPT}(L)$.*

We use two other algorithms, UD^x and UD^y described in [6]. These algorithms are based on the algorithm UD, developed by Baker, Brown and Kattseff [1] for the strip packing problem. The following results hold for these algorithms [6].

LEMMA 2.10. *Let L be an instance for* TPP *such that $b \in \mathcal{C}\left[\frac{1}{2}, 1 \;;\; 0, 1\right]$ (resp., $b \in \mathcal{C}\left[0, 1 \;;\; \frac{1}{2}, 1\right]$) for all boxes b in L. Then the packing $\mathcal{P}$ generated by the algorithm UD^x (resp., UD^y) is such that $H(\mathcal{P}) \leq \frac{5}{4}\mathrm{OPT}(L) + \frac{53}{8}Z$.*

LEMMA 2.11. *Let L be an instance for TPP^z consisting of boxes b such that $b \in \mathcal{C}\left[\frac{1}{2}, 1 \;;\; 0, 1\right]$ (resp., $b \in \mathcal{C}\left[0, 1 \;;\; \frac{1}{2}, 1\right]$), and whenever $x(b) > y(b)$ (resp., $y(b) > x(b)$) then $\rho(b) \notin \mathcal{C}$. That is, no two boxes of L can be packed side by side in the x-direction (resp., y-direction). Then the packing $\mathcal{P}$ generated by the algorithm UD^x (resp., UD^y) is such that $H(\mathcal{P}) \leq \frac{5}{4}\mathrm{OPT}(L) + \frac{53}{8}Z$.*

Proof. This result follows directly from the previous lemma and the fact that no two boxes can be packed side by side in the x-direction (resp., y-direction), even if rotations are allowed. ◻

3. The Algorithm R_k. In [6] we presented an algorithm for TPP, called $\mathcal{A}_k$, that has an asymptotic performance bound less than 2.67. In this section we present an algorithm for TPP^z, called R_k, that is based on the algorithm $\mathcal{A}_k$. The algorithm depends on a parameter k, an integer that is assumed to be greater than 5.

Before we give the description of the algorithm we define some numbers which are used to define sublists, called critical sets.

DEFINITION 3.1. *Let $r_1^{(k)}, r_2^{(k)}, \ldots, r_{k+15}^{(k)}$ and $s_1^{(k)}, s_2^{(k)}, \ldots, s_{k+14}^{(k)}$ be real numbers defined as follows:*

- *$r_1^{(k)}, r_2^{(k)}, \ldots, r_k^{(k)}$ are such that*
  $r_1^{(k)}\frac{1}{2} = r_2^{(k)}(1 - r_1^{(k)}) = r_3^{(k)}(1 - r_2^{(k)}) = \ldots = r_k^{(k)}(1 - r_{k-1}^{(k)}) = \frac{1}{3}(1 - r_k^{(k)})$
  and $r_1^{(k)} < \frac{4}{9}$;
- *$r_{k+1}^{(k)} = \frac{1}{3},\ r_{k+2}^{(k)} = \frac{1}{4},\ \ldots,\ r_{k+15}^{(k)} = \frac{1}{17}$;*
- *$s_i^{(k)} = 1 - r_i^{(k)}$ for $i = 1, \ldots, k$;*
- *$s_{k+i}^{(k)} = 1 - \left(\frac{2i+4-\lfloor\frac{i+2}{3}\rfloor}{4i+10}\right)$ for $i = 1, \ldots, 14$;*

The following result can be proved using a continuity argument.

CLAIM 3.1. *The numbers $r_1^{(k)}, r_2^{(k)}, \ldots, r_k^{(k)}$ are such that $r_1^{(k)} > r_2^{(k)} > \cdots > r_k^{(k)} > \frac{1}{3}$ and $r_1^{(k)} \to \frac{4}{9}$ as $k \to \infty$.*

For simplicity we omit the superscripts $^{(k)}$ of the notation $r_i^{(k)}, s_i^{(k)}$ when k is clear from the context.

Using the numbers in Definition 3.1, we define the following critical sets.

$$\mathcal{C}_i^A = \mathcal{C}\left[r_{i+1}, r_i \;;\; \frac{1}{2}, s_i\right], \quad \mathcal{C}_i^B = \mathcal{C}\left[\frac{1}{2}, s_i \;;\; r_{i+1}, r_i\right],$$

$$\mathcal{C}^A = \bigcup_{i=1}^{k+14} \mathcal{C}_i^A, \quad \mathcal{C}^B = \bigcup_{i=1}^{k+14} \mathcal{C}_i^B, \quad \mathcal{C}_{[1-k]}^A = \bigcup_{i=1}^{k} \mathcal{C}_i^A, \quad \mathcal{C}_{[1-k]}^B = \bigcup_{i=1}^{k} \mathcal{C}_i^B.$$

The next result refers to a list of positions $p_{i,j}, q_{i,j}, p'_j, q'_j, p''_j$ and q''_j to be considered when applying the algorithm COMBINE. In [6] we give such a list of positions, defined for a box $B = (1, 1, \infty)$. To use in this context, we have to consider a proportional reparameterization for a box $B = (l, w, \infty)$. For completeness, we define here these positions (only for $i < j$, since the case $i > j$ is symmetric). See figure 3.1(a).

Positions to combine sublists of $\mathcal{C}_i^A$ and $\mathcal{C}_j^B$.

For simplicity, we denote by A_i the list of boxes of type $\mathcal{C}_i^A$, and by B_j the list of boxes of type $\mathcal{C}_j^B$.

- To combine the lists A_i $(1 \le i \le k)$ and B_j $(i \le j \le k)$, take
  $p_{i,j} = \left[(0,0), (\frac{1}{2}, 0)\right]$ and $q_{i,j} = [(0, s_i)]$.
  In this case we have an area guarantee of at least $\frac{1}{2}$.
- To combine the list $A_{[1-k]} = A_1 \cup \ldots \cup A_k$ with B_j $(k+1 \le j \le k+14)$, we consider two phases. We divide $A_{[1-k]}$ into A' and A'' taking $A' = \{b \in A_{[1-k]} : x(b) \le 1 - s_j\}$ and $A'' = A_{[1-k]} \setminus A'$.
  - ⋆ To combine A' with B_j take
    $$p'_j = [(s_j, 0)] \text{ and}$$
    $$q'_j = \left[(0,0), \left(0, \tfrac{1}{j-k+2}\right), \left(0, \tfrac{2}{j-k+2}\right), \ldots, \left(0, \tfrac{j-k+1}{j-k+2}\right)\right] .$$
    In this case we have an area guarantee of at least $\frac{13}{24}$. This minimum is attained when $j = k+1$.
  - ⋆ To combine A'' with B_j take
    $$p''_j = \left[(0,0), (\tfrac{1}{2}, 0)\right] \text{ and}$$
    $$q''_j = \left[\left(0, \tfrac{2}{3}\right), \left(0, \tfrac{2}{3} + \tfrac{1}{j-k+2}\right), \left(0, \tfrac{2}{3} + \tfrac{2}{j-k+2}\right), \ldots, \left(0, \tfrac{2}{3} + \left(\lfloor \tfrac{j-k+2}{3} \rfloor - 1\right) \tfrac{1}{j-k+2}\right)\right] .$$
    Here we obtain an area guarantee of at least $\frac{27}{56}$.
- To combine the lists A_i $(k+1 \le i \le k+14)$ and B_j $(i \le j \le k+14)$, take
  $$p_{i,j} = \left[(s_j, 0), \left(s_j + \tfrac{1}{i-k+2}, 0\right), \left(s_j + \tfrac{2}{i-k+2}, 0\right), \ldots, \left(s_j + (\lfloor (1-s_j)\cdot(i-k+2) \rfloor - 1) \tfrac{1}{i-k+2}, 0\right)\right] \text{ and}$$
  $$q_{i,j} = \left[(0,0), \left(0, \tfrac{1}{j-k+2}\right), \left(0, \tfrac{2}{j-k+2}\right), \ldots, \left(0, \tfrac{j-k+1}{j-k+2}\right)\right] .$$
  In this case we also obtain an area guarantee of at least $\frac{27}{56}$.

LEMMA 3.2. *The following statements are valid for the list of positions $p_{i,j}$, $q_{i,j}$, p'_j, q'_j, p''_j and q''_j:*

(a) If $\mathcal{P}$ is a packing generated by the algorithm COMBINE with parameters $(L, \mathcal{C}_i^A, p_{i,j}, \mathcal{C}_j^B, q_{i,j})$, $1 \le i, j \le k$ or $k+1 \le i, j \le k+14$ then we have that $H(\mathcal{P}) \le \frac{56}{27} \frac{V(\mathcal{P})}{l \cdot w} + Z$.

(b) There is a partition of $\mathcal{C}_{[1-k]}^A$ into sets $\mathcal{C}'_{A,j}$ and $\mathcal{C}''_{A,j}$ such that a packing $\mathcal{P}'$ generated by the algorithm COMBINE *with parameters $(L, \mathcal{C}'_{A,j}, p'_j, \mathcal{C}_j^B, q'_j)$, $k+1 \le j \le k+14$, is such that $H(\mathcal{P}') \le \frac{56}{27} \frac{V(\mathcal{P}'')}{l \cdot w} + Z$ and a packing $\mathcal{P}''$ generated by the algorithm* COMBINE *with parameters $(L, \mathcal{C}''_{A,j}, p''_j, \mathcal{C}_j^B, q''_j)$, $k+1 \le j \le k+14$, is such that $H(\mathcal{P}'') \le \frac{56}{27} \frac{V(\mathcal{P}'')}{l \cdot w} + Z$.*

(c) Defining positions symmetric to $p_{i,j}, q_{i,j}, p'_j, q'_j, p''_j$ and q''_j, analogous results hold when the letter A and B are exchanged in the items above.

The algorithm R_k is inspired on the algorithm $\mathcal{A}_k$ presented in [6]. The reader may compare both algorithms to see where they differ; it should be noted that now there are steps where rotations are performed. This is done, since otherwise we may not obtain valid inequalities with respect to the optimum packing.

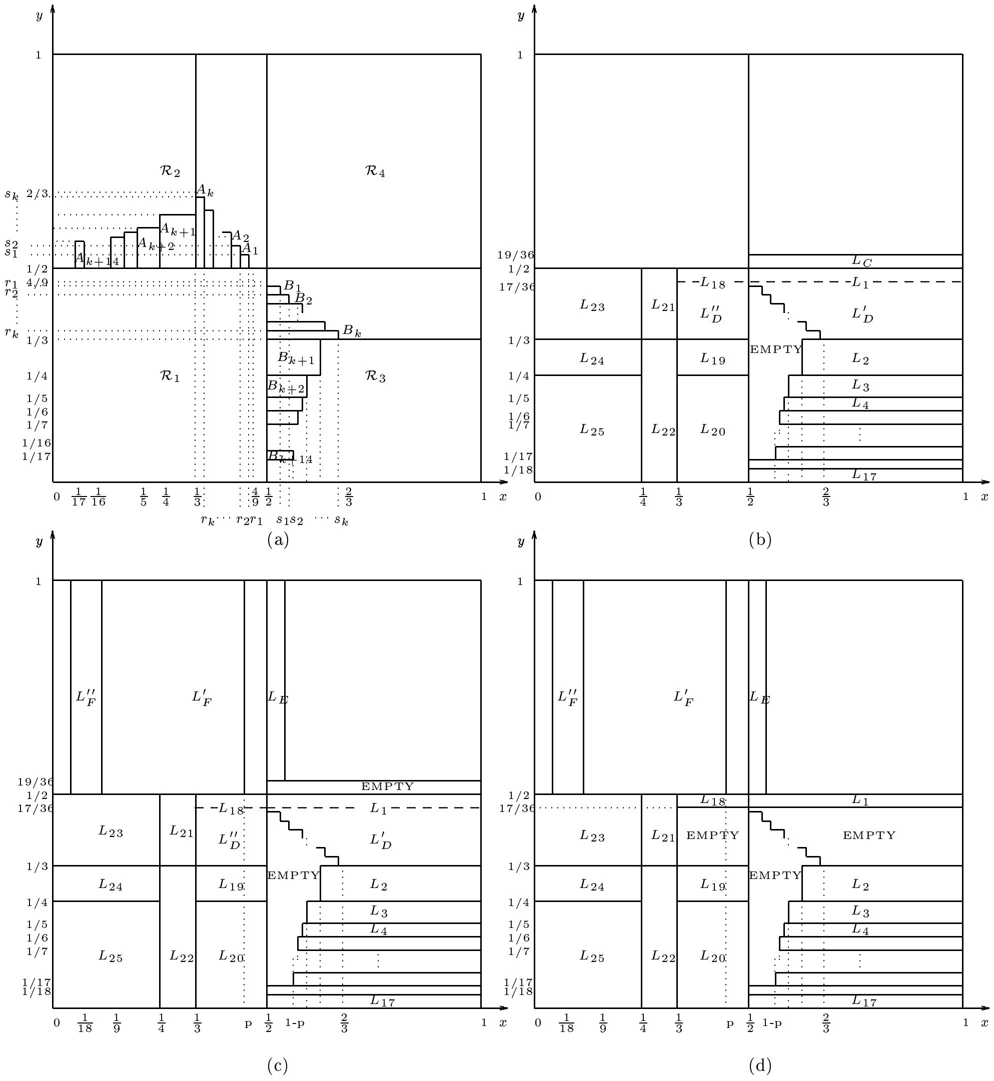


FIG. 3.1. *Partition of list L for algorithm R_k. The sets A_i and B_i in (a) correspond to the sets $\mathcal{C}_i^A$ and $\mathcal{C}_i^B$, respectively.*

Algorithm R_k

Input: List of boxes L.

Output: Packing $\mathcal{P}$ of L into $B = (l, w, \infty)$.

1 Rotate all boxes b that are in $\mathcal{R}_4$ such that $\rho(b) \in \mathcal{R}_2 \cup \mathcal{R}_3$.
/* I.e., Let $T \leftarrow \{b \in L \cap \mathcal{R}_4 : \rho(b) \in \mathcal{R}_2 \cup \mathcal{R}_3\}$. $L \leftarrow (L \setminus T) \bigcup \rho(T)$. */

2 Rotate all boxes b of L that are in $\mathcal{R}_2 \cup \mathcal{R}_3$ such that $\rho(b) \in \mathcal{R}_1$.

3 Let $p_{i,j}, q_{i,j}$, $1 \le i, j \le k + 14$, and p'_j, p''_j, q'_j, q''_j, $k + 1 \le j \le k + 14$, be as defined above.

4 Combine boxes of types $\mathcal{C}^A$ and $\mathcal{C}^B$ of L as follows (see figure 3.1(a)).

4.1 $i \leftarrow 1$; $j \leftarrow 1$; $\mathcal{P}_{AB} \leftarrow \emptyset$.

4.2 While ($i \le k$ and $j \le k$) do

$\mathcal{P}_{i,j} \leftarrow \text{COMBINE}(L, \mathcal{C}_i^A, p_{i,j}, \mathcal{C}_j^B, q_{i,j})$.
$\mathcal{P}_{AB} \leftarrow \mathcal{P}_{AB} \| \mathcal{P}_{i,j}$.
Update the list L removing the packed boxes.
If all boxes of type $\mathcal{C}_i^A$ have been packed then increment i; else increment j.

4.3 If all boxes of type $\mathcal{C}_{[1-k]}^B$ have been packed
4.3.1 Then
While ($j \leq k+14$ and there is a box of type $\mathcal{C}_{[}^A 1-k]$) do
Let $\mathcal{C}'_{A,j}$ and $\mathcal{C}''_{A,j}$ be a partition of $\mathcal{C}_{[1-k]}^A$, as in Lemma 3.2.
$\tilde{\mathcal{P}}'_j \leftarrow \text{COMBINE}(L, \mathcal{C}'_{A,j}, p'_j, \mathcal{C}_j^B, q'_j)$. Update L removing the packed boxes.
$\tilde{\mathcal{P}}''_j \leftarrow \text{COMBINE}(L, \mathcal{C}'_{A,j}, p'_j, \mathcal{C}_j^B, q'_j)$. Update L removing the packed boxes.
$\mathcal{P}_{AB} \leftarrow \mathcal{P}_{AB} \| \tilde{\mathcal{P}}'_j \| \tilde{\mathcal{P}}''_j$.
if $B_j = \emptyset$ then $j \leftarrow j+1$.
$i \leftarrow k+1$
4.3.2 Else /* All boxes of types $\mathcal{C}_{[1-k]}^A$ have been packed */
Perform steps symmetric to the ones given in the case 4.3.1.

4.4 While ($i \leq k+14$ and $j \leq k+14$) do
$\mathcal{P}_{i,j} \leftarrow \text{COMBINE}(L, \mathcal{C}_i^A, p_{i,j}, \mathcal{C}_j^B, q_{i,j})$. Update L removing the packed boxes.
$\mathcal{P}_{AB} \leftarrow \mathcal{P}_{AB} \| \mathcal{P}_{i,j}$.
If all boxes of type $\mathcal{C}_i^A$ have being packed then increment i; else increment j.

5 If all boxes of type $\mathcal{C}^B$ have been packed then
5.1 Rotate the boxes of $L \cap \mathcal{R}_2$ that fit in $\mathcal{R}_3$.
5.2 Rotate the boxes of $L \cap (\mathcal{R}_2 \cup \mathcal{R}_4)$ such that if $b \in L \cap (\mathcal{R}_2 \cup \mathcal{R}_4)$ then $x(b) \leq y(b)$ or $\rho(b) \notin \mathcal{C}$.
5.3 Subdivide the list L into sublists $L_1, \ldots, L_{25}$ as follows (see figure 3.1(b)).

$$L_i = L \bigcap \mathcal{C}\left[\tfrac{1}{2}, 1\ ;\ \tfrac{1}{i+2}, \tfrac{1}{i+1}\right], \text{ for } i = 1, \ldots, 16 \qquad L_{17} = L \bigcap \mathcal{C}\left[\tfrac{1}{2}, 1\ ;\ 0, \tfrac{1}{18}\right],$$
$$L_{18} = L \bigcap \mathcal{C}\left[\tfrac{1}{3}, \tfrac{1}{2}\ ;\ \tfrac{1}{3}, \tfrac{1}{2}\right], \qquad L_{19} = L \bigcap \mathcal{C}\left[\tfrac{1}{3}, \tfrac{1}{2}\ ;\ \tfrac{1}{4}, \tfrac{1}{3}\right],$$
$$L_{20} = L \bigcap \mathcal{C}\left[\tfrac{1}{3}, \tfrac{1}{2}\ ;\ 0, \tfrac{1}{4}\right], \qquad L_{21} = L \bigcap \mathcal{C}\left[\tfrac{1}{4}, \tfrac{1}{3}\ ;\ \tfrac{1}{3}, \tfrac{1}{2}\right],$$
$$L_{22} = L \bigcap \mathcal{C}\left[\tfrac{1}{4}, \tfrac{1}{3}\ ;\ 0, \tfrac{1}{3}\right], \qquad L_{23} = L \bigcap \mathcal{C}\left[0, \tfrac{1}{4}\ ;\ \tfrac{1}{3}, \tfrac{1}{2}\right],$$
$$L_{24} = L \bigcap \mathcal{C}\left[0, \tfrac{1}{4}\ ;\ \tfrac{1}{4}, \tfrac{1}{3}\right], \qquad L_{25} = L \bigcap \mathcal{C}_4,$$
$$L_C = L \bigcap \mathcal{C}\left[\tfrac{1}{2}, 1\ ;\ \tfrac{1}{2}, \tfrac{19}{36}\right] \qquad L'_D = L_1 \bigcap \mathcal{C}\left[0, \tfrac{17}{36}\ ;\ 0, 1\right],$$
$$L''_D = L_{18} \bigcap \mathcal{C}\left[0, \tfrac{17}{36}\ ;\ 0, 1\right] \qquad L_D = L'_D \bigcup L''_D.$$

5.4 Generate packing $\mathcal{P}_{CD}$ as follows.
$(\mathcal{P}_{CD'}, L_{CD'}) \leftarrow \text{COLUMN}(L_C, [(0,0)], L'_D, [(0, \frac{19}{36})])$.
$(\mathcal{P}_{CD''}, L_{CD''}) \leftarrow \text{COLUMN}(L_C \setminus L_{CD'}, [(0,0)], L''_D, [(0, \frac{19}{36}), (\frac{1}{2}, \frac{19}{36})])$.
$\mathcal{P}_{CD} \leftarrow \mathcal{P}_{CD'} \| \mathcal{P}_{CD''}$.
$L_{CD} \leftarrow L_{CD'} \bigcup L_{CD''}$. $L_1 \leftarrow L_1 \setminus L_{CD}$. $L_{18} \leftarrow L_{18} \setminus L_{CD}$.
5.5 Generate packings $\mathcal{P}_1, \ldots, \mathcal{P}_{25}$ as follows.
$\mathcal{P}_i \leftarrow \text{NFDH}^y(L_i)$ for $i = 1, \ldots, 22$.
$\mathcal{P}_i \leftarrow \text{NFDH}^x(L_i)$ for $i = 23, 24$.
$\mathcal{P}_{25} \leftarrow \text{LL}(L_{25}, 4)$.
5.6 Update L removing the packed boxes. Note that $L \subseteq \mathcal{R}_2 \cup \mathcal{R}_4$.

5.7 If $L_C \subseteq L_{CD}$
then /* (Case 1) */
$p \leftarrow \frac{\sqrt{199145}-195}{570} = 0.440\ldots$. /* L_C is totally packed, see figure 3.1(c) */
else /* (Case 2) $L_D \subseteq L_{CD}$ */
$p \leftarrow \frac{\sqrt{23401}-71}{180} = 0.455\ldots$. /* L_D is totally packed see figure 3.1(d) */
5.8 $L_E \leftarrow L \cap \mathcal{C}\left[\frac{1}{2}, 1-p\ ;\ \frac{1}{2}, 1\right]$. $L'_F \leftarrow L \cap \mathcal{C}\left[\frac{1}{9}, p\ ;\ \frac{1}{2}, 1\right]$.
$L''_F \leftarrow L \cap \mathcal{C}\left[\frac{1}{18}, \frac{1}{9}\ ;\ \frac{1}{2}, 1\right]$. $L_F \leftarrow L'_F \cup L''_F$.
5.9 $(\mathcal{P}_{EF'}, L_{EF'}) \leftarrow \text{COLUMN}(L_E, [(0,0)], L'_F, [(1-p, 0)])$.
$(\mathcal{P}_{EF''}, L_{EF''}) \leftarrow \text{COLUMN}(L_E \setminus L_{EF'}, [(0,0)], L''_F, [(0, 1-p), (0, 1-p+\frac{1}{9}), \ldots, (0, 1-p+(\lfloor 9p \rfloor - 1)\frac{1}{9})])$.
$\mathcal{P}_{EF} \leftarrow \mathcal{P}_{EF'} \| \mathcal{P}_{EF''}$.
$L_{EF} \leftarrow L_{EF'} \cup L_{EF''}$.
5.10 If $L_E \subseteq L_{EF}$ /* (Subcase 1) L_E is totally packed */
then
$\mathcal{P}_{\text{UD}} \leftarrow \text{UD}^x(L)$.
$\mathcal{P}_{OC} \leftarrow \text{OC}((L \setminus L_{EF}) \cap \mathcal{R}_4)$.
$\mathcal{P}_{2e} \leftarrow \text{NFDH}^x((L \setminus L_{EF}) \cap \mathcal{C}\left[0, \frac{1}{3}\ ;\ 0, 1\right])$.
$\mathcal{P}_{2d} \leftarrow \text{NFDH}^x((L \setminus L_{EF}) \cap \mathcal{C}\left[p, \frac{1}{2}\ ;\ 0, 1\right])$.
$\mathcal{P}' \leftarrow \mathcal{P}_{OC} \| \mathcal{P}_{2e} \| \mathcal{P}_{2d} \| \mathcal{P}_{EF}$.
$\mathcal{P}'' \leftarrow \{\mathcal{P} \in \{\mathcal{P}_{\text{UD}}, \mathcal{P}'\} :\ H(\mathcal{P}) \text{ is minimum }\}$.
$\mathcal{P}_{aux} \leftarrow \mathcal{P}_{AB} \| \mathcal{P}_{CD} \| \mathcal{P}_1 \| \ldots \| \mathcal{P}_{25}$.
Let L'' and L_{aux} be the lists of boxes packed in $\mathcal{P}''$ and $\mathcal{P}_{aux}$, resp.
$\mathcal{P} \leftarrow \mathcal{P}_{aux} \| \mathcal{P}''$.
5.11 If $L_F \subseteq L_{EF}$ /* (Subcase 2) L_F is totally packed */
then
$\mathcal{P}_{OC} \leftarrow \text{OC}((L \setminus L_{EF}) \cap \mathcal{R}_4)$.
$\mathcal{P}_{2e} \leftarrow \text{NFDH}^x((L \setminus L_{EF}) \cap \mathcal{C}\left[0, \frac{1}{18}\ ;\ \frac{1}{2}, 1\right])$.
$\mathcal{P}_{2d} \leftarrow \text{NFDH}^x((L \setminus L_{EF}) \cap \mathcal{C}\left[p, 1\ ;\ \frac{1}{2}, 1\right])$.
$\mathcal{P}' \leftarrow \mathcal{P}_{OC} \| \mathcal{P}_{EF}$.
$\mathcal{P}_{aux} \leftarrow \mathcal{P}_{AB} \| \mathcal{P}_{CD} \| \mathcal{P}_{2e} \| \mathcal{P}_{2d} \| \mathcal{P}_1 \| \ldots \| \mathcal{P}_{25}$.
Let L' and L_{aux} be the lists of boxes packed in $\mathcal{P}'$ and $\mathcal{P}_{aux}$, resp.
$\mathcal{P} \leftarrow \mathcal{P}_{aux} \| \mathcal{P}'$.
6 If all boxes of type $\mathcal{C}^A$ have been packed then generate a packing $\mathcal{P}$ of L as in step 5 (in a symmetric way).
7 Return $\mathcal{P}$.
end algorithm.

The next theorem gives an asymptotic performance bound of the algorithm R_k when $k \to \infty$.

THEOREM 3.3. *For any instance L of* TPP^z *we have*

$$R_k(L) \leq \alpha_k \cdot \text{OPT}(L) + \left(2k + \frac{597}{8}\right) Z\ ,$$

where $\alpha_k \to \frac{579+\sqrt{199145}}{384} = 2.669\ldots$ *as* $k \to \infty$.

Proof. We present the proof for the case all boxes of type $\mathcal{C}^{\mathcal{B}}$ have been packed (see step 5). The proof of the other case (step 6) is analogous. We consider 4 cases, according to step 5.7 ($L_C \subseteq L_{CD}$), step 5.10 ($L_E \subseteq L_{EF}$) and step 5.11 ($L_F \subseteq L_{EF}$).

As many steps of the algorithm R_k are similar to the ones of the algorithm $\mathcal{A}_k$ for TPP, many of the inequalities obtained in the analysis of $\mathcal{A}_k$ are valid in these cases. We only mention them in the four claims below (see [6]).

Case 1.1 ($L_C \subseteq L_{CD}$) and ($L_E \subseteq L_{EF}$).

Claim 1.1:

$$H(\mathcal{P}'') \leq \frac{1}{(1-p)\frac{19}{36}} \frac{V(L'')}{l \cdot w} + 4Z \quad \text{and} \quad H(\mathcal{P}_{aux}) \leq \frac{1}{r_1} \frac{V(L_{aux})}{l \cdot w} + (2k+68)Z.$$

Let $\mathcal{H}_1 := H(\mathcal{P}'') - \frac{53}{8}Z$ and $\mathcal{H}_2 := H(\mathcal{P}_{aux}) - (2k+68)Z$.

Using the definition of $\mathcal{H}_1$ and $\mathcal{H}_2$ in the two inequalities above we obtain

$$\text{OPT}(L) \geq \frac{V(L)}{l \cdot w} = \frac{V(L'')}{l \cdot w} + \frac{V(L_{aux})}{l \cdot w} \geq \frac{(1-p)19}{36}\mathcal{H}_1 + r_1\mathcal{H}_2 \ ,$$

that is,

$$\text{OPT}(L) \geq \frac{(1-p)19}{36}\mathcal{H}_1 + r_1\mathcal{H}_2 \ . \tag{3.1}$$

Note that from steps 1, 2, 4, 5.1 and 5.2 the list L'' satisfies the condition of Lemma 2.11. Hence, we have

$$H(\mathcal{P}'') \leq \text{UD}^x(L'') \leq \frac{5}{4}\text{OPT}(L'') + \frac{53}{8}Z \leq \frac{5}{4}\text{OPT}(L) + \frac{53}{8}Z \ ,$$

that is,

$$\text{OPT}(L) \geq \frac{4}{5}\mathcal{H}_1 \ . \tag{3.2}$$

Combining inequalities (3.1) and (3.2) we have

$$\text{OPT}(L) \geq \max\{\frac{4}{5}\mathcal{H}_1, (1-p)\frac{19}{36}\mathcal{H}_1 + r_1\mathcal{H}_2\}.$$

From the definition of $\mathcal{H}_1$ and $\mathcal{H}_2$, we obtain

$$H(\mathcal{P}) = H(\mathcal{P}'') + H(\mathcal{P}_{aux}) = \mathcal{H}_1 + \mathcal{H}_2 + \left(2k + \frac{597}{8}\right) Z.$$

Using the last inequality in the above equation we have

$$H(\mathcal{P}) \leq \left(\frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\frac{4}{5}\mathcal{H}_1, (1-p)\frac{19}{36}\mathcal{H}_1 + r_1\mathcal{H}_2\}}\right) \text{OPT}(L) + \left(2k + \frac{597}{8}\right) Z \ .$$

Analysing the two possibilities for the maximum, we can prove (see [6]) that

$$\alpha'_k(r_1) := \frac{49 + 95p + 180r_1}{144r_1} \geq \frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\frac{4}{5}\mathcal{H}_1, (1-p)\frac{19}{36}\mathcal{H}_1 + r_1\mathcal{H}_2\}.}$$

Thus,

$$H(\mathcal{P}) \leq \alpha'_k(r_1) \cdot \text{OPT}(L) + \left(2k + \frac{597}{8}\right) Z \ .$$

Since $r_1 \to \frac{4}{9}$ as $k \to \infty$, we can conclude that $\alpha'_k(r_1) \to \frac{579+\sqrt{199145}}{384}$ as $k \to \infty$.
Case 1.2 ($L_C \subseteq L_{CD}$) and ($L_F \subseteq L_{EF}$).
Claim 1.2:

$$H(\mathcal{P}'') \leq \frac{72}{19}\frac{V(P')}{l \cdot w} + 2Z, \quad \text{and} \quad H(\mathcal{P}_{aux}) \leq \frac{1}{p}\frac{V(L_{aux})}{l \cdot w} + (2k+70)Z.$$

Let $\mathcal{H}_1 := H(\mathcal{P}') - 2Z$ and $\mathcal{H}_2 := H(\mathcal{P}_{aux}) - (2k+70)Z$.
Then we have

$$\mathrm{OPT}(L) \geq \frac{V(L')}{l \cdot w} + \frac{V(L_{aux})}{l \cdot w} \geq \frac{19}{72}\mathcal{H}_1 + p\mathcal{H}_2. \tag{3.3}$$

Note that each box in $L' \cap \mathcal{R}_4$ considered in step 5.11 cannot be rotated, or if it can be rotated, then it fits in $\mathcal{R}_4$ again. So we can conclude that

$$\mathrm{OPT}(L) \geq \mathrm{OPT}(L') \geq \frac{19}{72}\mathcal{H}_1. \tag{3.4}$$

Proceeding as in Case 1.1, using inequalities (3.3) and (3.4) we have

$$H(\mathcal{P}) \leq \alpha''_k \cdot \mathrm{OPT}(L) + (2k+72)Z,$$

where $\alpha''_k = \frac{53+72p}{72p}$.
Thus, from the analysis of subcases 1.1 and 1.2 we can conclude that

$$\mathcal{A}_k(L) \leq \alpha_k \cdot \mathrm{OPT}(L) + \left(2k + \frac{597}{8}\right) Z \ ,$$

where $\alpha_k \to \alpha'_k(\frac{4}{9}) = \alpha''_k = \frac{\sqrt{199145}+579}{384} = 2.669\ldots$ as $k \to \infty$.

Case 2.1 ($L_D \subseteq L_{CD}$) and ($L_E \subseteq L_{EF}$).
Claim 2.1:

$$H(\mathcal{P}'') \leq \frac{1}{(1-p)\frac{1}{2}}\frac{V(L'')}{l \cdot w} + \frac{53}{8}Z \quad \text{and} \quad H(\mathcal{P}_{aux}) \leq \frac{1}{\frac{1}{4}+\frac{r_1}{2}}\frac{V(L_{aux})}{l \cdot w} + (2k+68)Z.$$

Let $\mathcal{H}_1 := H(\mathcal{P}'') - \frac{53}{8}Z$ and $\mathcal{H}_2 := H(\mathcal{P}_{aux}) - (2k+68)Z$.
Then we have $\mathrm{OPT}(L) \geq \frac{V(L'')}{l \cdot w} + \frac{V(L_{aux})}{l \cdot w} \geq \frac{1-p}{2}\mathcal{H}_1 + \left(\frac{1}{4} + \frac{r_1}{2}\right)\mathcal{H}_2$.
Using the same idea used in Case 1.1 we have $\mathrm{OPT}(L) \geq \mathrm{OPT}(L'') \geq \frac{4}{5}\mathcal{H}_1$, and so, we obtain $H(\mathcal{P}) \leq \beta'_k(r_1)\mathrm{OPT}(L) + \left(2k + \frac{597}{8}\right)Z$, where $\beta'_k(r_1) = \frac{11+10p+10r_1}{4+8r_1}$.
Case 2.2 ($L_D \subseteq L_{CD}$) and ($L_F \subseteq L_{EF}$).
Claim 2.2:

$$H(\mathcal{P}') \leq 4\frac{V(P')}{l \cdot w} + 2Z, \quad \text{and} \quad H(\mathcal{P}_{aux}) \leq \frac{1}{p}\frac{V(L_{aux})}{l \cdot w} + (2k+70)Z.$$

Let $\mathcal{H}_1 := H(\mathcal{P}') - 2Z$ and $\mathcal{H}_2 := H(\mathcal{P}_{aux}) - (2k+70)Z$.
In this case we have $\mathrm{OPT}(L) \geq \frac{V(L')}{l \cdot w} + \frac{V(L_{aux})}{l \cdot w} \geq \frac{1}{4}\mathcal{H}_1 + p\mathcal{H}_2$ and $\mathrm{OPT}(L) \geq \mathcal{H}_1$. So, $H(\mathcal{P}) \leq \beta''_k \cdot \mathrm{OPT}(L) + (2k+72)Z$, where $\beta''_k = \frac{3+4p}{4p}$.
Thus, for the given value of p, as in the previous cases, we can conclude that

$$H(\mathcal{P}) \leq \beta_k \cdot \mathrm{OPT}(L) + \left(2k + \frac{597}{8}\right) Z,$$

where $\beta_k \to \beta'_k(\frac{4}{9}) = \beta''_k = \frac{\sqrt{23401}+207}{136} = 2.64\ldots$ as $k \to \infty$.

The theorem follows from the conclusions obtained in all cases we have analysed. $\square$

The following result proved in [6] is also valid for this algorithm and can be proved analogously. It shows that for relatively small value of k ($k = 13$) the algorithm R_k has already an asymptotic performance bound that is very close to the value shown for $k \to \infty$.

COROLLARY 3.4. *For any instance L of* TPP^z *and $k \geq 13$ we have*

$$R_k(L) \leq \gamma_k \cdot \mathrm{OPT}(L) + \left(2k + \frac{597}{8}\right) Z \ ,$$

where $\gamma_k = \frac{99+1080r_1^{(k)}+\sqrt{199145}}{864r_1^{(k)}} < 2.67$.

PROPOSITION 3.5. *The asymptotic performance bound of the algorithm R_k, $k \geq 13$, is between* 2.5 *and* 2.67.

Proof. Follows directly from Corollary 3.4 and Lemma 2.6 (using $m = 4$). $\square$

4. The Algorithm LS: boxes in L have square bottom. In this section and in the next ones we apply the idea, used in algorithm R_k, to generate algorithms for particular instances of TPP^z. Here we consider the case the list L consists of boxes with square bottom.

Without loss of generality, we consider that the box B has dimensions $(1, w, \infty)$, $w \geq 1$.

Given a list of boxes $L = (b_1, \ldots, b_n)$, consider the list of points given by the set $\{(x_1, y_1), \ldots, (x_n, y_n)\}$. Note that all points lay down in a line on the xy-plane that goes through $(0, 0)$ and $(1, 1)$. We call it a *box-line*. The algorithm consider two cases, according to the position in the x-axis, where the box-line crosses the line $y = \frac{1}{2}$ (that is, in the position $xw = \left(\frac{1}{2w}\right) w$).

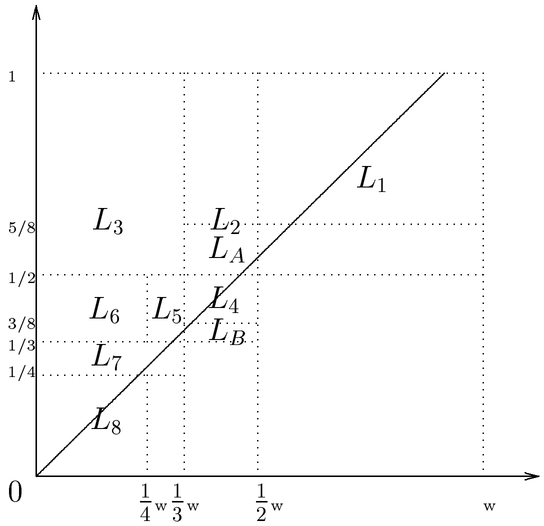
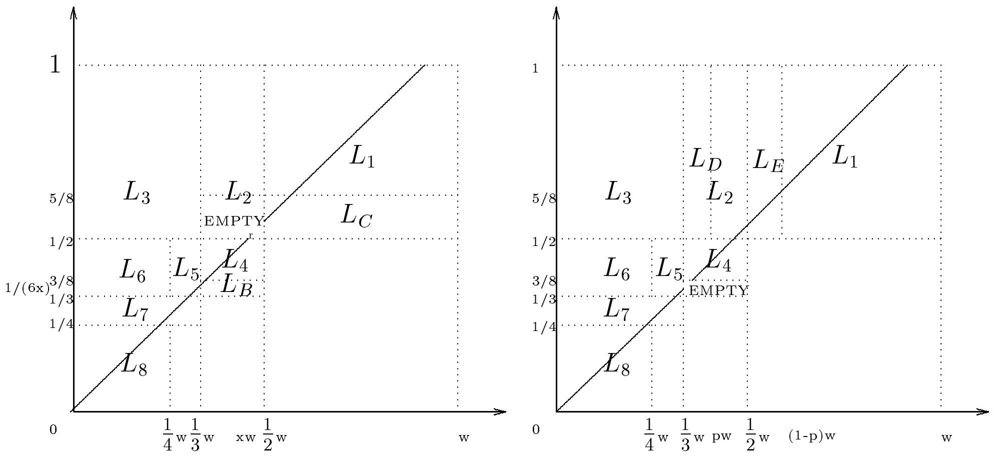


FIG. 4.1. *Partition of L into sublists.*



CASE 1: (L_A IS TOTALLY PACKED) CASE 2: (L_B IS TOTALLY PACKED)

FIG. 4.2. *Combination of sublists L_A and L_B.*

Algorithm LS

Input: List of boxes $L \subset \mathcal{Q}[0,1\ ;\ 0,1]$.

Output: Packing $\mathcal{P}$ of L into $B = (1, w, \infty)$.

1 Take $p := 0.4791964$ and subdivide the list L into sublists $L_1, \ldots, L_7, L_A, L_B, L_C$ as follows (see Figure 4.1 and Figure 4.2).

$$\begin{array}{lll} L_1 = L \bigcap \mathcal{C}\left[\frac{1}{2}, 1\ ;\ \frac{1}{2}, 1\right], & L_2 = L \bigcap \mathcal{C}\left[\frac{1}{3}, \frac{1}{2}\ ;\ \frac{1}{2}, 1\right], & L_3 = L \bigcap \mathcal{C}\left[0, \frac{1}{3}\ ;\ \frac{1}{2}, 1\right], \\ L_4 = L \bigcap \mathcal{C}\left[\frac{1}{3}, \frac{1}{2}\ ;\ \frac{1}{3}, \frac{1}{2}\right], & L_5 = L \bigcap \mathcal{C}\left[\frac{1}{4}, \frac{1}{3}\ ;\ \frac{1}{3}, \frac{1}{2}\right], & L_6 = L \bigcap \mathcal{C}\left[0, \frac{1}{4}\ ;\ \frac{1}{3}, \frac{1}{2}\right], \\ L_7 = L \bigcap \mathcal{C}\left[0, \frac{1}{3}\ ;\ \frac{1}{4}, \frac{1}{3}\right], & L_8 = L \bigcap \mathcal{C}\left[0, \frac{1}{4}\ ;\ 0, \frac{1}{4}\right], & L_A = L_2 \bigcap \mathcal{C}\left[0, 1\ ;\ 0, \frac{5}{8}\right], \\ L_B = L_3 \bigcap \mathcal{C}\left[0, 1\ ;\ 0, \frac{3}{8}\right], & L_C = L_1 \bigcap \mathcal{C}\left[0, 1\ ;\ 0, \frac{5}{8}\right]. & \end{array}$$

2 Let $x \leftarrow \frac{1}{2w}$.

3 if $x \leq \frac{2}{5}$

then /* **(Case 1)** this means that there is no box in L_C */

$\mathcal{P}'_{1,2,3} \leftarrow \mathrm{OC}(L_1) \| \mathrm{NFDH}^x(L_2) \| \mathrm{NFDH}^x(L_3)$.

$\mathcal{P}''_{1,2,3} \leftarrow \mathrm{UD}(L_1 \cup L_2 \cup L_3)$.

$\mathcal{P}'' \leftarrow \big(\mathcal{P} \in \{\mathcal{P}'_{1,2,3}, \mathcal{P}''_{1,2,3}\} :\ H(\mathcal{P}) \text{ is minimum}\big)$.

$\mathcal{P}_{aux} \leftarrow \mathrm{NFDH}^x(L_4) \| \ldots \| \mathrm{NFDH}^x(L_7) \| LL(L_8, 4)$.

$\mathcal{P} \leftarrow \mathcal{P}'' \| \mathcal{P}_{aux}$.

return $\mathcal{P}$.

else /* **(Case 2)** this means that there is no box in $L_2 \setminus L_A$ */

4 $(\mathcal{P}_{AB}, L_{AB}) \leftarrow \mathrm{COLUMN}(L_A, [(0,0), (\frac{1}{2}, 0)], L_B, [(0, \frac{5}{8}), (\frac{1}{2}, \frac{5}{8})])$.

5 $L_4 \leftarrow L_4 \setminus L_{AB}$. $\quad L_B \leftarrow L_B \setminus L_{AB}$.

6 **(Case 2.1) if** $L_A \subseteq L_{AB}$ /* L_A is totally packed. */

$(\mathcal{P}_{BC}, L_{BC}) \leftarrow \mathrm{COLUMN}(L_C, [(0,0)], L_B, [(0, \frac{5}{8}), (\frac{1}{2}, \frac{5}{8})]$.

$L_1 \leftarrow L_1 \setminus L_{BC}$. $L_4 \leftarrow L_4 \setminus L_{BC}$.

(Subcase 2.1.1) $L_B \subseteq L_{BC}$.

$\mathcal{P}' \leftarrow \mathrm{OC}(L_1) \| \mathcal{P}_{BC}$.

$\mathcal{P}_{aux} \leftarrow \mathcal{P}_{AB} \| \mathrm{NFDH}^x(L_4) \| \ldots \| \mathrm{NFDH}^x(L_7) \| \mathrm{LL}(L_8, 4)$.

(Subcase 2.1.2) $L_C \subseteq L_{BC}$.

$\mathcal{P}' \leftarrow \mathrm{OC}(L_1) \| \mathcal{P}_{BC}$.

$\mathcal{P}_{aux} \leftarrow \mathcal{P}_{BC} \| \mathrm{NFDH}^x(L_2) \| \ldots \| \mathrm{NFDH}^x(L_7) \| \mathrm{LL}(L_8, 4)$.

$\mathcal{P} \leftarrow \mathcal{P}' \| \mathcal{P}_{aux}$.
Let L' and L_{aux} be the lists of boxes packed in $\mathcal{P}'$ and $\mathcal{P}_{aux}$, respectively.

7 (Case 2.2) if $L_B \subseteq L_{AB}$ /* L_B is totally packed. */
/* Define two new sublists (L_D and L_E) as follows. */
$L_D = L_2 \bigcap \mathcal{C}\,[0, p \; ; \; 0, 1]$ and $L_E = L_1 \bigcap \mathcal{C}\,[0, 1-p \; ; \; 0, 1]$.
$(\mathcal{P}_{DE}, L_{DE}) \leftarrow \text{COLUMN}(L_D, [(0,0)], L_E, [(p,0)])$.
$L_1 \leftarrow L_1 \setminus L_{DE}$. $L_2 \leftarrow L_2 \setminus L_{DE}$.
/* We have two subcases considering the result of this packing. */
(Subcase 2.2.1) if $L_D \subseteq L_{DE}$ or $x \geq p$, $x = \frac{1}{2w} \in \left(p, \frac{1}{2}\right]$ then
/* Note that when $x \geq p$, $L_D = \emptyset$. */
$\mathcal{P}' \leftarrow \text{OC}(L_1) \| \mathcal{P}_{DE}$.
$\mathcal{P}_{aux} \leftarrow \mathcal{P}_{AB} \| \text{NFDH}^x(L_2) \| \ldots \| \text{NFDH}^x(L_7) \| \text{LL}(L_8, 4)$.
Let L' and L_{aux} be the lists of boxes packed in $\mathcal{P}'$ and $\mathcal{P}_{aux}$, respectively.
$\mathcal{P} \leftarrow \mathcal{P}' \| \mathcal{P}_{aux}$.
(Subcase 2.2.2) if $L_E \subseteq L_{DE}$ then
$\mathcal{P}'_{1,2} \leftarrow \text{OC}(L_1) \| \text{NFDH}^x(L_2) \| \mathcal{P}_{DE}$.
$\mathcal{P}''_{1,2} \leftarrow \text{UD}(L_1 \cup L_2 \cup L_{DE})$.
$\mathcal{P}'' \leftarrow \left(\mathcal{P} \in \{\mathcal{P}'_{1,2}, \mathcal{P}''_{1,2}\} : \; H(\mathcal{P}) \text{ is minimum}\right)$.
$\mathcal{P}_{aux} \leftarrow \mathcal{P}_{AB} \| \text{NFDH}^x(L_4) \| \ldots \| \text{NFDH}^x(L_7) \| \text{LL}(L_8, 4)$.
Let L'' and L_{aux} be the lists of boxes packed in $\mathcal{P}''$ and $\mathcal{P}_{aux}$, respectively.
$\mathcal{P} \leftarrow \mathcal{P}'' \| \mathcal{P}_{aux}$.

8 Return $\mathcal{P}$.
end algorithm.

THEOREM 4.1. *For any instance of* TPP^z *consisting of a list L of boxes with square bottom we have*

$$\text{LS}(L) \leq 2.543 \cdot \text{OPT}(L) + \frac{101}{8} Z.$$

Proof. As the proof technique is analogous to the previous one, we only give the inequalities that are valid in each case. We suggest the reader to follow the analysis of each case, together with the corresponding case in the description of the algorithm. Throughout this proof $l = 1$, as we are considering that $B = (1, w, \infty)$.

Case 1: In this case we obtain the followings inequalities

$$H(\mathcal{P}'') \leq \frac{16}{5} \frac{V(L'')}{l \cdot w} + \frac{53}{8} Z,$$
$$H(\mathcal{P}'') \leq \frac{5}{4} \text{OPT}(L) + \frac{53}{8},$$
$$H(\mathcal{P}_{aux}) \leq 2 \frac{V(L_{aux})}{l \cdot w} + 5Z.$$

Defining $\mathcal{H}_1 := H(\mathcal{P}'') - \frac{53}{8} Z$ and $\mathcal{H}_2 := H(\mathcal{P}_{aux}) - 4Z$ we have $\text{OPT}(L) \geq \max\{\frac{4}{5}\mathcal{H}_1, \frac{5}{16}\mathcal{H}_1 + \frac{1}{2}\mathcal{H}_2\}$ and therefore, proceeding as before, we obtain

$$H(\mathcal{P}) \leq \alpha_1 \cdot \text{OPT}(L) + \frac{53}{8} Z,$$

where $\alpha_1 \leq \frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\frac{4}{5}\mathcal{H}_1, \frac{5}{16}\mathcal{H}_1 + \frac{1}{2}\mathcal{H}_2\}} \leq 2.5$.

Note that for the remaing cases, the lists L_3 and $L_2 \setminus L_A$ are empty and the box-line crosses the region L_C.

Subcase 2.1.1: Note that in this case, $L_A \cup L_B$ is totally packed in $\mathcal{P}_{AB} \| \mathcal{P}_{BC}$. Note also that the boxes of L_C in $\mathcal{P}_{BC}$ are of type $\mathcal{R}_4$ and therefore we obtain the following inequality.

$$H(\mathcal{P}') \leq 4\frac{V(L')}{l \cdot w} + Z.$$

Since,

$$\begin{aligned} H(\mathcal{P}_{aux}) &\leq 2\frac{V(L_{aux})}{l \cdot w} + 7Z, \\ H(\mathcal{P}') &\leq \mathrm{OPT}(L) + Z, \end{aligned}$$

using the above inequalities and defining $\mathcal{H}_1 := H(\mathcal{P}') - Z$ and $\mathcal{H}_2 := H(\mathcal{P}_{aux}) - 7Z$ we have

$$H(\mathcal{P}) \leq \alpha_{2,1,1} \cdot \mathrm{OPT}(L) + 8Z,$$

where $\alpha_{2,1,1} \leq \frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\mathcal{H}_1, \frac{1}{4}\mathcal{H}_1 + \frac{1}{2}\mathcal{H}_2\}} \leq 2.5$.

Subcase 2.1.2: In this case the boxes of $L_A \cup L_C$ are totally packed in $\mathcal{P}_{AB} \| \mathcal{P}_{BC}$. Furthermore, we have $x = \frac{1}{2w}$ and $x \in (\frac{2}{5}, \frac{1}{2}]$ (note that $x \cdot w$ is the position in the x-axis where the box-line crosses the line $y = \frac{1}{2}$). In this case we have the following inequalities with respect to x.

$$\begin{aligned} H(\mathcal{P}') &\leq \frac{32}{25x}\frac{V(L')}{l \cdot w} + Z, \\ H(\mathcal{P}_{aux}) &\leq \frac{1}{\min\{\frac{2}{9x}, \frac{1}{2}\}}\frac{V(L_{aux})}{l \cdot w} + 8Z, \\ H(\mathcal{P}') &\leq \mathrm{OPT}(L) + Z, \end{aligned}$$

and therefore,

$$H(\mathcal{P}) \leq \alpha_{2,1,2} \cdot \mathrm{OPT}(L) + 9Z,$$

where $\alpha_{2,1,2} \leq \frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\mathcal{H}_1, \frac{25x}{32}\mathcal{H}_1 + \min\{\frac{2}{9x}, \frac{1}{2}\}\mathcal{H}_2\}}$. Evaluating the value of $\alpha_{2,1,2}$, when $x \geq \frac{4}{9}$ and when $x < \frac{4}{9}$, we obtain that $\alpha_{2,1,2} \leq 2.5$.

Subcase 2.2.1: In this case $L_B \cup L_D$ is totally packed in $\mathcal{P}_{AB} \| \mathcal{P}_{DE}$. Recall that $x = \frac{1}{2w}$. We divide the analysis in two cases. We consider first $x \in (p, \frac{1}{2}]$. Here we have

$$\begin{aligned} H(\mathcal{P}') &\leq 8x\frac{V(L')}{l \cdot w} + Z, \\ H(\mathcal{P}_{aux}) &\leq \frac{1}{x}\frac{V(L_{aux})}{l \cdot w} + 7Z, \\ H(\mathcal{P}') &\leq \mathrm{OPT}(L) + Z, \end{aligned}$$

and therefore,

$$H(\mathcal{P}) \leq \alpha_{2,2,1} \cdot \mathrm{OPT}(L) + 8Z,$$

where $\alpha_{2,2,1} \leq \frac{\mathcal{H}_1+\mathcal{H}_2}{\max\{\mathcal{H}_1,\frac{1}{8x}\mathcal{H}_1+x\mathcal{H}_2\}} \leq \frac{\mathcal{H}_1+\mathcal{H}_2}{\max\{\mathcal{H}_1,\frac{1}{8p}\mathcal{H}_1+p\mathcal{H}_2\}} \leq 2.543$.

If $x \in (\frac{2}{5}, p]$, the analysis is similar and is omitted.

Subcase 2.2.2: Here $L_B \cup L_E$ is totally packed in $\mathcal{P}_{AB} \| \mathcal{P}_{DE}$. Let $x = \frac{1}{2w}$, $x \in (\frac{2}{5}, p]$. In this case,

$$\begin{aligned}
H(\mathcal{P}'') &\leq \frac{2x}{(1-p)^2}\frac{V(L')}{l \cdot w} + \frac{53}{8}Z, \\
H(\mathcal{P}_{aux}) &\leq 2\frac{V(L_{aux})}{l \cdot w} + 5Z, \\
H(\mathcal{P}'') &\leq \frac{7}{5}4\mathrm{OPT}(L) + \frac{53}{8}Z,
\end{aligned}$$

and so,

$$H(\mathcal{P}) \leq \alpha_{2,2,2} \cdot \mathrm{OPT}(L) + \frac{93}{8}Z,$$

where $\alpha_{2,2,2} \leq \frac{\mathcal{H}_1+\mathcal{H}_2}{\max\{\frac{4}{5}\mathcal{H}_1,\frac{(1-p)^2}{2x}\mathcal{H}_1+\frac{1}{2}\mathcal{H}_2\}} \leq \frac{\mathcal{H}_1+\mathcal{H}_2}{\max\{\frac{4}{5}\mathcal{H}_1,\frac{(1-p)^2}{2p}\mathcal{H}_1+\frac{1}{2}\mathcal{H}_2\}} \leq 2.543$.

In fact, the value of p was taken in such a manner that the two subcases above (2.2.1 and 2.2.2) lead to the same bound.

The theorem follows considering the cases analysed above. □

PROPOSITION 4.2. *The asymptotic performance bound of the algorithm LS is between* 2.5 *and* 2.5425.

Proof. Follows directly from Theorem 4.1 and Lemma 2.6 (using $m = 4$). □

5. The Algorithm BS: box B has square bottom. We consider now the special case of TPP^z where B has square bottom. Without loss of generality, we consider $B = (1, 1, \infty)$.

First, we present an algorithm called NFDH$_p^{xy}$, $0 < p < 1$, that is used as a subroutine. This algorithm packs the boxes of a list L in the following way. Initially, it sorts L in a non-increasing order of height, then generates a packing divided into levels. Each level is divided into two parts; first the boxes are packed in the region $[0,1) \times [0,p)$ and then in the region $[0,1) \times [1-p,1)$. The boxes are packed in the region $[0,1) \times [0,p)$ using the algorithm NFDH^x, until a box b_i cannot be packed in the same level, then NFDH$_p^{xy}$ uses the algorithm NFDH^y to pack boxes $\rho(b_i), \rho(b_{i+1}), \ldots$ in the region $[0,1) \times [1-p,1)$ until a box b_k cannot be packed in the same level. At this point, the algorithm NFDH$_p^{xy}$ considers the two parts as only one level and continue to pack the box b_k in a new level. The process continues until all boxes in L have been packed.

Another variant of the above algorithm is called NFDH$_p^{yx}$. This algorithm is similar to the NFDH$_p^{xy}$ algorithm, except that NFDH$_p^{yx}$ first packs boxes b with $x(b) \leq p$, in the y-axis direction, and then packs the next boxes in the x-axis direction.

The following notation is used in the description of the algorithm.

$$\mathcal{X}' := \{b_i = (x_i, y_i, z_i) : \ y_i \leq 1 - x_i\}.$$

Algorithm BS

Input: List of boxes L.

Output: Packing $\mathcal{P}$ of L into $B = (1, 1, \infty)$.

1 Rotate the boxes of L in such a way that for each box b, $x(b) \leq y(b)$.
Take $p = 0.43322958$ and $q = 1 - p$.

2 Divide L into sublists $L_1', L_2', L_3', L_A, L_B, L_C, L_4, \ldots, L_{14}$, as follows (see figure 5.1).

$L_1' = L \bigcap \mathcal{C}\,[q, 1\ ;\ q, 1]$, $\quad L_A = L \bigcap \mathcal{C}\,[\frac{1}{2}, q\ ;\ \frac{1}{2}, 1]$, $\quad L_2' = L \bigcap \mathcal{C}\,[p, \frac{1}{2}\ ;\ \frac{1}{2}, 1] \setminus \mathcal{X}'$,
$L_B = L \bigcap \mathcal{C}\,[\frac{1}{3}, p\ ;\ \frac{1}{2}, 1] \setminus \mathcal{X}'$, $\quad L_3' = L \bigcap \mathcal{C}\,[p, \frac{1}{2}\ ;\ \frac{1}{3}, \frac{1}{2}]$, $\quad L_C = L \bigcap \mathcal{C}\,[\frac{1}{3}, p\ ;\ \frac{1}{3}, \frac{1}{2}]$,
$L_4 = L \bigcap \mathcal{C}\,[\frac{1}{4}, \frac{1}{3}\ ;\ \frac{2}{3}, 1]$, $\quad L_5 = L \bigcap \mathcal{C}\,[\frac{1}{5}, \frac{1}{4}\ ;\ \frac{2}{3}, 1]$, $\quad L_6 = L \bigcap \mathcal{C}\,[0, \frac{1}{5}\ ;\ \frac{8}{13}, 1]$,
$L_7 = L \bigcap \mathcal{C}\,[\frac{1}{3}, \frac{1}{2}\ ;\ \frac{1}{2}, \frac{2}{3}] \bigcap \mathcal{X}'$, $\quad L_8 = L \bigcap \mathcal{C}\,[\frac{1}{4}, \frac{1}{3}\ ;\ \frac{1}{2}, \frac{2}{3}]$, $\quad L_9 = L \bigcap \mathcal{C}\,[\frac{1}{5}, \frac{1}{4}\ ;\ \frac{1}{2}, \frac{2}{3}]$,
$L_{10} = L \bigcap \mathcal{C}\,[0, \frac{1}{5}\ ;\ \frac{1}{2}, \frac{8}{13}]$, $\quad L_{11} = L \bigcap \mathcal{C}\,[\frac{1}{4}, \frac{1}{3}\ ;\ \frac{1}{3}, \frac{1}{2}]$, $\quad L_{12} = L \bigcap \mathcal{C}\,[0, \frac{1}{4}\ ;\ \frac{1}{3}, \frac{1}{2}]$,
$L_{13} = L \bigcap \mathcal{C}\,[0, \frac{1}{3}\ ;\ \frac{1}{4}, \frac{1}{3}]$, $\quad L_{14} = L \bigcap \mathcal{C}\,[0, \frac{1}{4}\ ;\ 0, \frac{1}{4}]$.

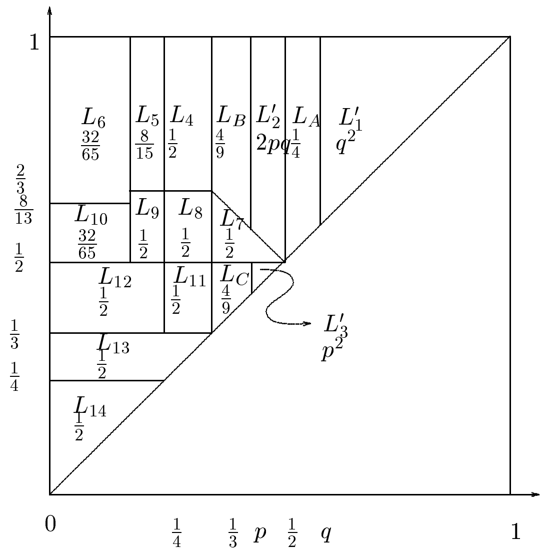


FIG. 5.1. *Partition of list L done by algorithm* BS.

3 $(\mathcal{P}_{AB}, L_{AB}) \leftarrow \text{COLUMN}(L_A, [(0,0)], L_B, [(q,0)])$.

4 $(\mathcal{P}_{AC}, L_{AC}) \leftarrow \text{COLUMN}(L_A \setminus L_{AB}, [(0,0)], L_C, [(q,0), (q, \frac{1}{2})])$.

5 $L_1 \leftarrow (L_1' \cup L_A) \setminus (L_{AB} \cup L_{AC})$. $L_2 \leftarrow (L_2' \cup L_B) \setminus L_{AB}$. $L_3 \leftarrow (L_3' \cup L_C) \setminus L_{AC}$.

6 Let $\mathcal{P}_7$ be a packing of L_7 obtained as follows.

6.1 Sort L_7 in a non-increasing order of height.

6.2 Construct a partition of L_7 given by $L_7^1, L_7^2, \ldots, L_7^{n_7}$ such that

$$\begin{cases} L_7 & = L_7^1 \| L_7^2 \| \ldots \| L_7^{n_7}, \\ |L_7^i| & = 3, \quad i = 1, \ldots, n_7 - 1, \\ |L_7^{n_7}| & \leq 3. \end{cases}$$

6.3 Generate a packing $\mathcal{P}_7^i$ of L_7^i, $i = 1, \ldots, n_7$ as follows

6.3.1 Choose $b \in L_7^i$, such that $x(b)$ is minimum.
6.3.2 Pack $\rho(b)$ in the position $(0, 1 - x(b))$ and the boxes in $L_7^i \setminus \{b\}$ in the positions $(0,0)$ and $(\frac{1}{2}, 0)$.
6.4 $\mathcal{P} \leftarrow \mathcal{P}_7^1 \| \dots \| \mathcal{P}_7^{n_7}$.
7 $\mathcal{P}' \leftarrow \mathrm{OC}(L_1) \| \mathcal{P}_{AB} \| \mathcal{P}_{AC}$.
8 $\mathcal{P}_{aux} \leftarrow \mathrm{NFDH}^x(L_2) \| \dots \| \mathrm{NFDH}^x(L_6) \| \mathcal{P}_7 \| \mathrm{NFDH}^{xy}_{\frac{2}{3}}(L_8) \| \mathrm{NFDH}^{xy}_{\frac{2}{3}}(L_9) \|$ $\mathrm{NFDH}^{xy}_{\frac{8}{13}}(L_{10}) \| \mathrm{NFDH}^x(L_{11}) \| \dots \| \mathrm{NFDH}^x(L_{13}) \| \mathrm{LL}(L_{14})$.
9 $\mathcal{P} \leftarrow \mathcal{P}' \| \mathcal{P}_{aux}$.
10 Return $\mathcal{P}$.
end algorithm.

THEOREM 5.1. *For any list L for the* TPP^z, *where B has square bottom,*

$$\mathrm{BS}(L) \leq 2.528 \cdot \mathrm{OPT}(L) + 15Z \ .$$

Proof. From the steps 3 and 4, we can conclude that either L_A is totally packed or $L_B \cup L_C$ is totally packed in $\mathcal{P}_{AB} \| \mathcal{P}_{BC}$. So we divide the proof in these two subcases.

Subcase 1: L_A is totally packed in $\mathcal{P}_{AB} \| \mathcal{P}_{BC}$.

Here we have

$$\begin{aligned} H(\mathcal{P}') &\leq \frac{1}{q^2} V(L') + 2Z, \\ H(\mathcal{P}_{aux}) &\leq \frac{9}{4} V(L_{aux}) + 13Z, \\ H(\mathcal{P}') &\leq \mathrm{OPT}(L) + 2Z. \end{aligned}$$

As before, we have $H(\mathcal{P}) \leq \alpha_1 \cdot \mathrm{OPT}(L) + 15Z$, where $\alpha_1 \leq \frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\mathcal{H}_1, q^2\mathcal{H}_1 + \frac{4}{9}\mathcal{H}_2\}} \leq 2.528$.

Subcase 2: $(L_B \cup L_C)$ is totally packed in $\mathcal{P}_{AB} \| \mathcal{P}_{BC}$.

Here we have

$$\begin{aligned} H(\mathcal{P}') &\leq 4V(L') + 2Z, \\ H(\mathcal{P}_{aux}) &\leq \frac{1}{2pq} V(L_{aux}) + 13Z, \\ H(\mathcal{P}') &\leq \mathrm{OPT}(L) + 2Z. \end{aligned}$$

Analogously, we have $H(\mathcal{P}) \leq \alpha_2 \cdot \mathrm{OPT}(L) + 15Z$, where $\alpha_2 \leq \frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\mathcal{H}_1, \frac{1}{4}\mathcal{H}_1 + 2pq\mathcal{H}_2\}} \leq 2.528$.

From the two cases above, the theorem follows. ☐

PROPOSITION 5.2. *The asymptotic performance bound of the algorithm* BS *is between* 2.5 *and* 2.528.

Proof. It follows directly from Theorem 5.1 and Lemma 2.6 (using $m = 4$). ☐

6. The Algorithm SS: boxes in L and box B have square bottom. Now we consider the special case of TPP^z where all boxes in L and box B have square bottom. Without loss of generality, we take $B = (1, 1, \infty)$.

In 1990, Li and Cheng [3] presented an algorithm for this problem with asymptotic performance bound 2.6875. The algorithm we present here, called SS, has an asymptotic performance bound of 2.361.

Here we need an algorithm, called GQ_m, described in [3], to pack boxes in $\mathcal{Q}_m$.

The algorithm GQ_m works in the same way as algorithm LL. It sorts the boxes in $L \subset \mathcal{Q}_m$ in non-increasing order of their height, divides L into sublists $L_1, \dots, L_v$ and uses the same two-dimensional packing algorithm to pack each sublist L_i in a level. The only place where algorithm GQ_m differs from LL is the bottom area size used to subdivide L into sublists L_i. This is because the two-dimensional packing algorithm used by algorithm LL can guarantee a better area if all boxes have square bottom. In this case the sublists L_i satisfy the following inequalities:

$$\begin{aligned} &S(L_i) \le \left[\left(\tfrac{m-1}{m}\right)^2 + \left(\tfrac{1}{m}\right)^2\right] lw && \text{for} \quad i = 1, \dots, v, \\ &S(L_i) + S(first(L_{i+1})) > \left[\left(\tfrac{m-1}{m}\right)^2 + \left(\tfrac{1}{m}\right)^2\right] lw && \text{for} \quad i = 1, \dots, v-1. \end{aligned}$$

The following result is proved in [3].

LEMMA 6.1. *Let $L \subset \mathcal{Q}_m$, $m \ge 2$. Then,* $GQ_m(L) \le \left(\frac{m}{m-1}\right)^2 \frac{V(L)}{lw} + Z$.

Analogously to the Lemma 2.6 for algorithm LL, we can show tha the following result holds for algorithm GQ_m.

LEMMA 6.2. *Let $\mathcal{A}$ be an algorithm for TPP^z to pack a list $L \subset \mathcal{Q}[0,1\ ;\ 0,1]$ into a box $B = (1,1,\infty)$. If $\mathcal{A}$ subdivides the input list L into two sublists $L_1 \subset \mathcal{R}_4$ and $L_2 \subset \mathcal{Q}_m$, $m \ge 2$, and applies algorithm GQ_m to pack L_2, then the asymptotic performance bound of $\mathcal{A}$ is at least $\frac{4(m-1)^2+3m^2}{4(m-1)^2}$.*

Proof. The proof is similar to the proof of Lemma 2.6, now using the value $\left(\frac{m-1}{m}\right)^2$ instead of $\left(\frac{m-2}{m}\right)$ □

Algorithm SS

Input: List of boxes $L \subset \mathcal{Q}[0,1\ ;\ 0,1]$.

Output: Packing $\mathcal{P}$ of L into $B = (1,1,\infty)$.

1 Take $p = 0.37123918$ and $q = 1 - p$.

Divide L into sublists, $L'_1, L_A, L'_2, L_B, L_3$ and L_4 (see Figure 6.1),

$$\begin{aligned} L'_1 &= L \bigcap \mathcal{Q}\left[\tfrac{1}{2}, 1\ ;\ \tfrac{1}{2}, 1\right], & L_A &= L \bigcap \mathcal{Q}\left[\tfrac{1}{2}, q\ ;\ \tfrac{1}{2}, q\right], \\ L'_2 &= L \bigcap \mathcal{Q}\left[\tfrac{1}{3}, \tfrac{1}{2}\ ;\ \tfrac{1}{3}, \tfrac{1}{2}\right], & L_B &= L \bigcap \mathcal{Q}\left[\tfrac{1}{3}, p\ ;\ \tfrac{1}{3}, p\right], \\ L_3 &= L \bigcap \mathcal{Q}\left[\tfrac{1}{4}, \tfrac{1}{3}\ ;\ \tfrac{1}{4}, \tfrac{1}{3}\right], & L_4 &= L \bigcap \mathcal{Q}\left[0, \tfrac{1}{4}\ ;\ 0, \tfrac{1}{4}\right]. \end{aligned}$$

2 $(\mathcal{P}_{AB}, L_{AB}) \leftarrow \mathrm{COLUMN}(L_A, [(0,0)], L_B, [(0,q),(q,q),(q,0)])$.
3 $L_i \leftarrow L'_i \setminus L_{AB}$ for $i = 1, 2$.
4 $\mathcal{P}_1 \leftarrow \mathrm{OC}(L_1) \| \mathcal{P}_{AB}$.
5 $\mathcal{P}_{aux} \leftarrow \mathrm{NFDH}^x(L_2) \| \mathrm{NFDH}^x(L_3) \| GQ_4(L_4)$.
6 $\mathcal{P} = \mathcal{P}_1 \| \mathcal{P}_{aux}$.
7 Return $\mathcal{P}$.
end algorithm.

THEOREM 6.3. *For any list L for TPP^z, where all boxes have square bottom,*

$$\mathrm{SS}(L) \le 2.361 \cdot \mathrm{OPT}^z(L) + 4Z.$$

Proof. Again we analyse two cases, considering the packing generated in step 2.

Case 1: L_A is totally packed in $\mathcal{P}_{AB}$.

$$H(\mathcal{P}_1) \le \frac{1}{q^2} V(L') + Z,$$

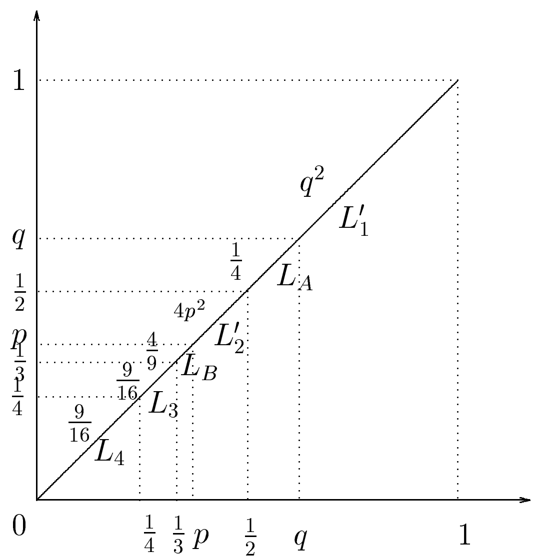


FIG. 6.1. *Partition of list L done by algorithm SS.*

$$H(\mathcal{P}_{aux}) \leq \frac{9}{4}V(L_{aux}) + 3Z,$$
$$H(\mathcal{P}_1) \leq \mathrm{OPT}(L) + Z.$$

Proceeding as before, we have $H(\mathcal{P}) \leq \alpha_1 \cdot \mathrm{OPT}(L) + 4Z$, where

$$\alpha_1 \leq \frac{\mathcal{H}_1 + \mathcal{H}_2}{\max\{\mathcal{H}_1, q^2\mathcal{H}_1 + \frac{4}{9}\mathcal{H}_2\}} \leq 2.361 .$$

Case 2: L_B is totally packed in $\mathcal{P}_{AB}$.

Here we have

$$H(\mathcal{P}_1) \leq 4V(L') + Z,$$
$$H(\mathcal{P}_{aux}) \leq \frac{1}{4p^2}V(L_{aux}) + 3Z,$$
$$H(\mathcal{P}_1) \leq \mathrm{OPT}(L) + Z.$$

Thus, $H(\mathcal{P}) \leq \alpha_2 \cdot \mathrm{OPT}(L) + 4Z$, where $\alpha_2 \leq \frac{\mathcal{H}_1+\mathcal{H}_2}{\max\{\mathcal{H}_1, \frac{1}{4}\mathcal{H}_1 + 4p^2\mathcal{H}_2\}} \leq 2.361$.

From the two cases above, the theorem follows. ☐

PROPOSITION 6.4. *The asymptotic performance bound of the algorithm SS is between* 2.333 *and* 2.361.

Proof. It follows directly from Theorem 6.3 and Lemma 6.2 (with $m = 4$). ☐

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 (1981), pp. 348–368.

[2] M. R. GAREY AND D. S. JOHNSON, *Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness*, Freeman, San Francisco, 1979.

[3] K. LI AND K.-H. CHENG, *On three-dimensional packing*, SIAM J. Comput., 19 (1990), pp. 847–867.

[4] ———, *Static job scheduling in partitionable mesh connected systems*, J. Parallel and Distributed Computing, 10 (1990), pp. 152–159.

[5] ———, *Heuristic algorithms for on-line packing in three dimensions*, J. of Algorithms, 13 (1992), pp. 589–605.

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