J Comb Optim (2013) 26:223–236
DOI 10.1007/s10878-012-9457-z

Online algorithms for 1-space bounded multi dimensional bin packing and hypercube packing

Yong Zhang · Francis Y.L. Chin · Hing-Fung Ting · Xin Han

Published online: 17 February 2012

Abstract In this paper, we study 1-space bounded multi-dimensional bin packing and hypercube packing. A sequence of items arrive over time, each item is a d-dimensional hyperbox (in bin packing) or hypercube (in hypercube packing), and the length of each side is no more than 1. These items must be packed without overlapping into d-dimensional hypercubes with unit length on each side. In d-dimensional space, any two dimensions i and j define a space P_{ij}. When an item arrives, we must pack it into an active bin immediately without any knowledge of the future items, and 90°-rotation on any plane P_{ij} is allowed.

The objective is to minimize the total number of bins used for packing all these items in the sequence. In the 1-space bounded variant, there is only one active bin for packing the current item. If the active bin does not have enough space to pack the item, it must be closed and a new active bin is opened. For d-dimensional bin packing,

Y. Zhang's research supported by NSFC (11171086) and Shenzhen Internet Industry Development Fund under grant No. JC201005270342A.
F.Y.L. Chin's research supported by HK RGC grant HKU-7117/09E.
X. Han's research supported by NSFC(11101065).

Y. Zhang (✉) · F.Y.L. Chin · H.-F. Ting
Department of Computer Science, The University of Hong Kong, Hong Kong, China
e-mail: yzhang@cs.hku.hk

F.Y.L. Chin
e-mail: chin@cs.hku.hk

H.-F. Ting
e-mail: hfting@cs.hku.hk

Y. Zhang
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China

X. Han
School of Software, Dalian University of Technology, Dalian, China
e-mail: hanxin.mail@gmail.com

an online algorithm with competitive ratio 4^d is given. Moreover, we consider d-dimensional hypercube packing, and give a 2^{d+1}-competitive algorithm. These two results are the first study on 1-space bounded multi dimensional bin packing and hypercube packing.

Keywords Online algorithms · Bin packing · 1-space bounded · Multi dimensional

1 Introduction

Bin packing is a very fundamental problem in computer science, and has been well studied for more than thirty years. Given a sequence of items, we pack them into unit-size bins without overlapping. The objective is to minimize the number of bins for all items in the sequence.

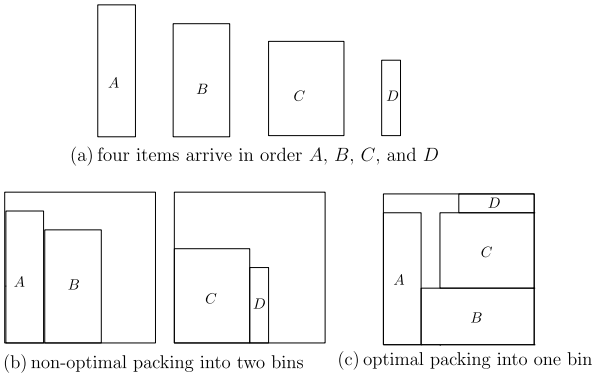
We focus on the online version of bin packing, where the items arrive over time, when packing the current item, we have no information of the future items. The positions of the packed items in the bin are fixed and cannot be repacked. To measure the performance of online bin packing, we study a general used method called *asymptotic competitive ratio*. Consider an online algorithm A and an optimal offline algorithm OPT. For any sequence S of items, let $A(S)$ be the cost (number of bins used) incurred by algorithm A and $OPT(S)$ be the corresponding optimal cost incurred by algorithm OPT. The *asymptotic competitive ratio* for algorithm A is:

$$R_A^{\infty} = \lim_{k\to\infty} \sup_S \left\{ \frac{A(S)}{OPT(S)} \middle| OPT(S) = k \right\}.$$

In the online bin packing, there are two models: *bounded space model* and *unbounded space model*. If we do not impose a limit on the number of bins available for packing the items (called *active bins*), we call it *unbounded space*. Otherwise, if the number of active bins is bounded by a *constant*, and each item can only be packed into one of the active bins, we call it *bounded space*, which is more realistic in many applications. If none of the active bins has enough space to pack the arrival item, one of the active bins must be closed and a new active bin will be opened to pack that item.

In this paper, we consider *1-space bounded* multi-dimensional bin packing and hypercube packing. In the 1-space bounded variant, the number of active bins is only *one*. If an item cannot be packed into the active bin, we have to close it and open a new bin to pack this item. In the 1-space bounded d-dimensional bin packing problem ($d \geq 2$), each item is a d-dimensional hyperbox such that the length on each side is no more than 1, while in the d-dimensional hypercube packing, each item is a d-dimensional hypercube with side length no more than 1. The items must be packed into d-dimensional hypercubes with side length 1. Any two dimensions i and j define a plane P_{ij}. 90°-rotation of the item in any plane P_{ij} is allowed in 1-space bounded bin packing, otherwise, the competitive ratio is unbounded (Fujita 2003).

To understand this problem clearly, we give an example for the 1-space bounded 2-dimensional bin packing. In Fig. 1(a), there are four items to be packed into unit square bins, and the arrival order is A, B, C and D. After the packing position of A



(a) four items arrive in order A, B, C, and D

(b) non-optimal packing into two bins

(c) optimal packing into one bin

Fig. 1 Example of optimal packing and non-optimal packing

is fixed, we have two choices to pack B: rotation and without rotation. If we pack B without rotation in the same bin with A as shown in Fig. 1(b), when item C arrives, we have to open a new bin since the current active bin does not have enough space for packing C. In the optimal solution, these four items can be packed into one bin (Fig. 1(c)), since item B, C and D can be rotated and the free space in the bin can accommodate all of them in their order of arrival.

1.1 Related works

Both the offline and online version of the bin packing problem have been well studied.

The offline bin packing is NP-Hard (Garey and Johnson 1979). For one-dimensional bin packing, Simchi-Levi gave a 1.5-approximation algorithm (Simchi-Levi 1994). Johnson and Garey (1985) gave an asymptotic 71/60-approximation algorithm. An AFPTAS was given by Karmarkar and Karp (1982). For two-dimensional bin packing. Chung et al. (1982) presented an approximation algorithm with an asymptotic performance ratio of 2.125. Caprara (2002) improved the upper bound to 1.69103. Bansal et al. (2006b) devised a randomized algorithm with an asymptotic performance ratio of at most 1.525. As for the offline lower bound of the approximation ratio, Bansal et al. (2006a) showed that the two-dimensional bin packing problem does not admit any asymptotic polynomial time approximation scheme.

The online bin packing has been studied for more than thirty years. For one-dimensional online bin packing, Johnson et al. (1974) showed that the First Fit algorithm (FF) has an asymptotic competitive ratio of 1.7. Yao (1980) improved the algorithm to obtain a better upper bound of 5/3. Lee and Lee (1985) introduced the class of Harmonic algorithms, and showed that an asymptotic competitive ratio of 1.63597 is achievable. Ramanan et al. (1989) further improved the upper bound to 1.61217. The best known upper bound is 1.58889, which was given by Seiden (2002). As for the lower bound of the competitive ratio of one dimensional bin packing, Yao (1980) showed that no online algorithm can have an asymptotic competitive ratio less than 1.5. The best known lower bound is 1.54014 (van Vliet 1992). For two-dimensional online bin packing, Coppersmith and Raghavan (1989) gave the first online algorithm with asymptotic competitive ratio 3.25. Csirik et al. (1993) improved the upper bound to 3.0625. Based on the techniques of the Improved Harmonic, Han et al. (2001) improved the upper bound to 2.7834. Seiden and van Stee

(2003) showed an upper bound of 2.66013 by implementing the Super Harmonic Algorithm. The best known upper bound of the competitive ratio for two dimensional bin packing is 2.5545, which was given by Han et al. (201). The best known lower bound is 1.907 (Blitz et al. 1996).

For bounded space online bin packing, Harmonic algorithm by Lee and Lee (1985) can be applied for one dimensional case, the competitive ratio is 1.69103 when the number of active bins goes to infinity. Csirik and Johnson (2001) presented an 1.7-competitive algorithm (K-Bounded Best Fit algorithms (BBF_K)) for one dimensional bin packing using K active bins, where $K \geq 2$. For multi-dimensional case, Epstein and van Stee (2005b) gave a 1.69103^d-competitive algorithm using $(2M-1)^d$ active bins, where $M \geq 10$ is an integer such that $M \geq 1/(1-(1-\varepsilon)^{1/(d+2)})-1$, $\varepsilon > 0$ and d is the dimension of the bin packing problem. For the *1-space bounded* variant, Fujita (2003) first gave an $O((\log\log m)^2)$-competitive algorithm, where m is the width of the square bin and the size of each item is $a \times b$ (a, b are integers and $a, b \leq m$). Chin et al. (2012) proposed an 8.84-competitive packing strategy, then they further improved the upper bound to 5.155 (Zhang et al. 2010), they also gave the lower bound 3 for 1-space bounded two dimensional bin packing.

For a special case where the items are squares (or hypercubes), there are also many results (Epstein and van Stee 2005a, 2007; Ferreira et al. 1999; Han et al. 2008; Januszewski and Lassak 1997; Leung et al. 1990; Kohayakawa et al. 2004; Meir and Moser 1968). For bounded space online square packing, Epstein and van Stee (2007) gave a 2.3692-competitive algorithm, they also proved that the lower bound of the competitive ratio is at least 2.36343. For bounded space d-dimensional hypercube packing, an $O(d/\log d)$-competitive algorithm was given (Epstein and van Stee 2007), however, to achieve this bound, the number of active bins is very large. Moreover, they proved that the asymptotic competitive ratio of bounded space hypercube packing is lower bounded by $\Omega(\log d)$. Januszewski and Lassak (1997) proved that any sequence of square items with a total area of at most $5/16$ can be packed into a unit bin. Han et al. (2008) studied a variant in which any packed item can be removed so as to guarantee a good competitive ratio and presented a packing algorithm that is 3-competitive. Note that in the above two studies, there is *only* one bin to pack the square items.

The remaining part of this paper is organized as follows. In Sect. 2, we show the 4^d-competitive algorithm for 1-space bounded d-dimensional bin packing. In Sect. 3, a 2^{d+1}-competitive algorithm is given for 1-space bounded d-dimensional hypercube packing.

2 1-space bounded d-dimensional bin packing

Let d be the highest dimension of the item and hypercube, each item a is associated with a vector $(a_1, a_2, \ldots, a_d)$, where a_i $(1 \leq i \leq d)$ is the length in the i-th dimension of item a.

In d-dimensional space, any two dimensions i and j define a space P_{ij}. For 1-space bounded multi-dimensional bin packing problem, rotation 90° in any plane P_{ij} is allowed. Otherwise, the performance ratio is unbounded. Consider an example of a sequence with $2n$ items: $\{A, B, A, B, \ldots\}$, where $A = (1/n, 1, 1, \ldots, 1)$ and

$B = (1, 1/n, 1, 1, \ldots, 1)$. If rotation is not allowed, any two adjacent items cannot be packed into the same bin by any online algorithm, thus, the number of used bins is $2n$. In the optimal packing, all A items can be packed into one bin, all B items can be packed into another bin, only two bins is enough to pack all these items. In this way, the performance ratio is n. If rotation is allowed, the first half part of items in the sequence can be packed into one bin by rotate B items 90° in the plane P_{12}. Similarly, the second half of items can be packed into another bin. Since rotation 90° in any plane P_{ij} is allowed, we may assume that the lengths in dimensions of any item a is non-increasing, i.e., $a_i \geq a_j$ $(i < j)$ for each item.

Denote the *size* of an item $a = (a_1, a_2, \ldots, a_d)$ to be $\prod_{i=1}^{d} a_i$. We say a $(k+1)$-dimensional hyperbox $b = (b_1, b_2, \ldots, b_{k+1})$ is a $(k+1, h)$-*hyperbox* if $b_1 = \cdots = b_{k-1} = 1$, $b_k = 1/2$ and $b_{k+1} = h$.

Let o_i and r_i $(2 \leq i < d)$ be the average occupancy ratio in the worst case and competitive ratio for packing i-dimensional items by our algorithm, respectively. Our target is to design an algorithm with competitive ratio as smaller as possible. Since any algorithm cannot pack items with total sizes more than 1 into one bin, we set $r_i = 1/o_i$. The target can be done by designing algorithm with the average occupancy ratio as larger as possible. According to the algorithm, o_i (r_i) until o_d (r_d) can be recursively computed. We say an item is *large w.r.t. its* $(i+1)$*-th dimension* if $\prod_{j=1}^{i} a_j \geq o_i$, and *small w.r.t. its* $(i+1)$*-th dimension* otherwise. For a small item a w.r.t. its $(k+1)$-th dimension, we have $a_{k+1} \leq a_k \leq o_k^{1/k} = r_k^{-1/k}$. From Table 1 and Lemma 1 in the later part of this paper, we have $r_k^{1/k} > 2$, thus, $a_{k+1} \leq a_k < 1/2$. A small items a w.r.t. its $(k+1)$-th dimension can be packed into a $(k+1, h)$-hyperbox such that $h/2 < a_{k+1} \leq h$ and $h = 2^{-j} \cdot o_k^{1/k}$ (for some $j = 0, 1, 2, \ldots$).

2.1 Packing strategy

Roughly speaking, items are recursively packed by the strategy from higher dimension to lower dimension. For an incoming item a, if it is large w.r.t. the d-th dimension, we pack it in a top-down order along the d-th dimension. Otherwise, it is small w.r.t. the d-th dimension, we first pack it into a (d, h)-hyperbox, then pack this (d, h)-hyperbox into the bin in a bottom-up manner. Since the length of $(d-1)$-th dimension of the (d, h)-hyperbox is $1/2$, the lower part along dimension d is partitioned into the "left" side and the "right" side: the "left" side is the area such that the $(d-1)$-th dimension is in the range $[0, 1/2]$, while the "right" side is in the range $[1/2, 1]$. When packing the (d, h)-hyperbox into the bin, we try to balance the heights of the left and right sides. When packing small item into (d, h)-hyperbox, we have no need to consider the length of the d-th dimension. In another word, packing small item into (d, h)-hyperbox can be regarded as packing $(d-1)$-dimensional item into a bin, therefore, the dimension of the packing problem is decreased by one. By implementing this idea, we can recursively pack items from higher dimension into lower dimension.

Note that in our algorithm, small items can be only packed into a $(k+1, h)$-hyperbox with $h < 2a_{k+1}$. Let o'_{k+1} be the average occupancy ratio for packing small items into $(k+1, h)$-hyper-box, we have the following fact.

Fact 1 $o'_{k+1} = o_k/2$.

Proof Since the length of the k-th dimension is no more than $1/2$, any small item a can be packed into the corresponding $(k+1, h)$-hyperbox such that, $h/2 < a_{k+1} \leq h$. Packing small items can be regarded as packing general items into k-dimensional bin by doubling the length in the k-th dimension of the small item. Thus, the average occupancy ratio is preserved in the first k dimensions. In the $(k+1)$-th dimension, the length is at least $h/2$. Thus, $o'_{k+1} = o_k/2$. □

Now we give our Algorithm for packing d-dimensional items with $d \geq 3$. In case of $d = 2$, we use our previous 5.15-competitive algorithm. Note that the occupation ratios o_i and competitive ratios r_i are all computed by analyzing the algorithm, thus, when packing an incoming item, these ratios are all known in advance.

Algorithm for Packing d-dimensional item a:

```
1: if a is large w.r.t. the d-th dimension. then
2:     Pack it by a top-down order such that a_d along the d-th dimension.
3:     if overlap happens then
4:         Close this bin then open a new bin for packing this item.
5:     end if
6: else if a is small w.r.t. the d-th dimension. then
7:     if there exists (d, h)-hyperbox with enough space for the item then
8:         Pack it into the (d, h)-hyperbox.
9:     else
10:        Open a new (d, h)-hyperbox for this item.
11:        Pack the (d, h)-hyperbox by a bottom-up order in the d-th dimension,
12:        such that the heights of the "left" part and the "right" part are balanced.
13:        if overlap happens then
14:            Close this bin then open a new bin for packing this (d, h)-hyperbox.
15:        end if
16:    end if
17: end if
```

To understand the algorithm clearly, we give an example to show how to pack an incoming item. In the current packing shown in Fig. 2, there are some large items packed in the upper part of the k-th dimension, some small items packed in the (k, h)-hyperboxes which located in the lower part of the k-th dimension such that the "left" and "right" part are balanced. When a large item comes, if it cannot be packed between the upper and lower part without overlapping, we will open a new bin. When a small item comes, we will first try to pack it into some existed (k, h)-hyperbox, if no such (k, h)-hyperbox, we open a new (k, h)-hyperbox then pack it into the right part.

The above algorithm recursively packing items from higher dimension to lower dimension, until dimension 3. We implement the algorithm in Zhang et al. (2010) for packing 2-dimensional items because that the performance ratio $r_2 = 5.15$ is better than implementing the above algorithm in dimension 2.

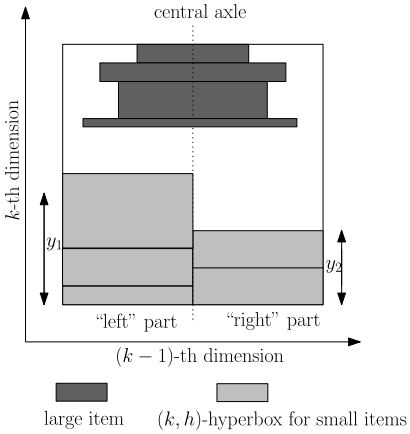


Fig. 2 Packing k-dimensional items into k-dimensional hypercube

2.2 Analysis of the strategy

When packing items which are small w.r.t. the k-dimension into (k,h)-hyperbox, we say the (k,h)-hyperboxes with the same height h are of the same type. From Fact 1, the average occupancy ratio $o'_k = o_{k-1}/2$. Thus, for each kind of (k,h)-hyperbox, except the last one, the average occupancy ratio is at least $o_{k-1}/2$.

Fact 2 *The total lengths in dimension k of the last hyperboxes of each type is at most* $2 \cdot o_{k-1}^{1/(k-1)}$.

Proof From previous definition $h_j = 2^{-j} \cdot o_{k-1}^{1/(k-1)}$ $(j = 0, 1, 2, \ldots)$, the length in dimension k of each type of (k, h_j)-hyperbox is fixed. Thus, the total length is at most $\sum_j h_j \leq 2 \cdot o_{k-1}^{1/(k-1)}$. □

Consider a packing configuration shown in Fig. 2, suppose the length in k-dimension of the upper part is y, the lengths in k-dimension of the "left" and "right" parts are y_1 and y_2 respectively. W.l.o.g., $y_1 \geq y_2$. The current occupancy in this bin is at least

$$y \cdot o_{k-1} + \frac{(y_1 + y_2 - 2 \cdot o_{k-1}^{1/(k-1)}) \cdot o'_k}{2}$$

The first term is the occupancy of large items, the second term is the occupancy of small items. Since in the lower part, the length in $(k-1)$-dimension of each hyperbox is $1/2$, we divide 2 in the second term. By Fact 1, we know the occupancy in this bin is at least

$$y \cdot o_{k-1} + \frac{(y_1 + y_2 - 2 \cdot o_{k-1}^{1/(k-1)}) \cdot o_{k-1}}{4}$$

If we only count the occupancy in one bin, the performance ratio is unbounded. For example, a bin contains a very small item, the next item is very large and cannot be packed together with the small one. We have to open a new bin for the later item.

In this case, the occupancy in the previous bin is very small. This example gives us an heuristic to amortize the occupancies of adjacent two bins: if we have to open a new bin due to an item, this items has contribution to two bins, one is the bin it packed and the other is the previous bin it cannot be packed.

To amortize the occupancy of adjacent two bins, item from the upper part compensate half occupancy to the previous bin; item from the lower part compensate the part which is larger than the ratio $o_{k-1}/4$ to the previous bin.

Now we study two cases of opening a new bin.

- a large item with length y' in dimension k cannot be packed into this bin.
  By amortized analysis, the occupancy in this bin is at least

$$\begin{aligned}
&\frac{y \cdot o_{k-1}}{2} + \frac{(y_1 + y_2 - 2 \cdot o_{k-1}^{1/(k-1)}) \cdot o_{k-1}}{4} + \frac{y' \cdot o_{k-1}}{2} \\
&= \frac{(y + y') \cdot o_{k-1}}{2} + \frac{(y_1 + y_2 - 2 \cdot o_{k-1}^{1/(k-1)}) \cdot o_{k-1}}{4} \\
&> \frac{(1 - y_1) \cdot o_{k-1}}{2} + \frac{(y_1 + y_2 - 2 \cdot o_{k-1}^{1/(k-1)}) \cdot o_{k-1}}{4} \\
&= \frac{o_{k-1}}{2} - \frac{o_{k-1}^{k/(k-1)}}{2} + \frac{(y_2 - y_1) \cdot o_{k-1}}{4} \\
&\geq \frac{o_{k-1}}{2} - \frac{o_{k-1}^{k/(k-1)}}{2} - \frac{o_{k-1}^{k/(k-1)}}{4} \\
&= \frac{o_{k-1}}{2} - \frac{3 \cdot o_{k-1}^{k/(k-1)}}{4}
\end{aligned}$$

- a small item with length y' in dimension k cannot be packed into this bin.
  - if $(y')^k \leq \frac{y' \cdot o_{k-1}}{4}$, this small item has no contribution to the previous bin. In this case, $y' \leq (o_{k-1}/4)^{1/(k-1)}$. The amortized occupancy is at least

$$\begin{aligned}
&\frac{y \cdot o_{k-1}}{2} + \frac{(y_1 + y_2 - 2(o_{k-1})^{1/(k-1)}) \cdot o_{k-1}}{4} \\
&\geq \frac{y \cdot o_{k-1}}{2} + \frac{(y_2 - (o_{k-1})^{1/(k-1)}) \cdot o_{k-1}}{2} \\
&= \frac{(y + y_2) \cdot o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} \\
&> \frac{(1 - y') \cdot o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} \\
&= \frac{o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} - \frac{y' \cdot o_{k-1}}{2} \\
&\geq \frac{o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} - \frac{(o_{k-1}/4)^{1/(k-1)} \cdot o_{k-1}}{2}
\end{aligned}$$

Table 1 The performance ratio for $k = 2$ to 6

k	2	3	4	5	6
r_k	5.15	30.86	127.969	518.156	2086.38
$r_k^{1/k}$	2.269	3.13	3.36	3.49	3.57

– if $(y')^k > \frac{y' \cdot o_{k-1}}{4}$, this small item has contribution $(y')^k - \frac{y' \cdot o_{k-1}}{4}$ to the previous bin. In this case, $y' > (o_{k-1}/4)^{1/(k-1)}$. The amortized occupancy is at least

$$\begin{aligned}
&\frac{y \cdot o_{k-1}}{2} + \frac{(y_1 + y_2 - 2(o_{k-1})^{1/(k-1)}) \cdot o_{k-1}}{4} + \left(y'\right)^k - \frac{y' \cdot o_{k-1}}{4} \\
&\geq \frac{y \cdot o_{k-1}}{2} + \frac{(y_2 - (o_{k-1})^{1/(k-1)}) \cdot o_{k-1}}{2} + \left(y'\right)^k - \frac{y' \cdot o_{k-1}}{4} \\
&= \frac{(y + y_2) \cdot o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} + \left(y'\right)^k - \frac{y' \cdot o_{k-1}}{4} \\
&\geq \frac{(1 - y') \cdot o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} + \left(y'\right)^k - \frac{y' \cdot o_{k-1}}{4} \\
&= \frac{o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} + \left(y'\right)^k - \frac{3 \cdot y' \cdot o_{k-1}}{4} \\
&\geq \frac{o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} - \frac{(o_{k-1}/4)^{1/(k-1)} \cdot o_{k-1}}{2}
\end{aligned}$$

the last inequality holds since $k \geq 3$ and $(o_{k-1}/4)^{1/(k-1)} < y' \leq o_{k-1}^{1/(k-1)} < 1/2$.

When $k = 3$, the above three formula are equivalent, when $k > 3$, the last two formulas are less. In this paper, the dimension we focused is at least 3, thus, we can say that the amortized occupancy ratio for packing k-dimensional items is at least

$$o_k = \frac{o_{k-1}}{2} - \frac{(o_{k-1})^{k/(k-1)}}{2} - \frac{(o_{k-1}/4)^{1/(k-1)} \cdot o_{k-1}}{2} \tag{1}$$

Since $r_k = 1/o_k$, we have

$$\frac{1}{r_k} = \frac{1}{2 \cdot r_{k-1}} - \frac{1}{2 \cdot r_{k-1}^{k/(k-1)}} - \frac{1}{2 \cdot r_{k-1} \cdot (4 \cdot r_{k-1})^{1/(k-1)}} \tag{2}$$

In Table 1, we give the performance ratio r_k for some lower dimensions ($k = 2$ to 6). We also compute $r_k^{1/k}$, which will help us to give the upper bound for the performance ratio.

We can see that the performance ratio is increased very fast, but $r_k^{1/k}$ is increased slowly.

Lemma 1 $3.5 < r_k^{1/k} < 4$ *if* $k > 6$ *and* $3.5 < r_{k-1}^{1/(k-1)} < 4$.

Proof Let $x_k = r_k^{1/k}$, from (2), we have

$$\frac{1}{x_k^k} = \frac{1}{2x_{k-1}^{k-1}} - \frac{1}{2x_{k-1}^k} - \frac{1}{2x_{k-1}^k 4^{1/(k-1)}} \geq \frac{1}{2x_{k-1}^{k-1}} - \frac{1}{x_{k-1}^k} \tag{3}$$

When $x_{k-1} \geq 3.5$, the above formula is larger than $1/4^k$. Thus, we can say that $x_k < 4$ if $x_{k-1} \geq 3.5$.

$$\begin{aligned}
\frac{1}{x_k^k} &= \frac{1}{2x_{k-1}^{k-1}} - \frac{1}{2x_{k-1}^k} - \frac{1}{2x_{k-1}^k 4^{1/(k-1)}} \\
&= \frac{1}{x_{k-1}^{k-1}}\left(\frac{1}{2} - \frac{1}{2x_{k-1}} - \frac{1}{2x_{k-1}4^{1/(k-1)}}\right) \\
&< \frac{1}{3.5^{k-1}}\left(\frac{1}{2} - \frac{1}{2x_{k-1}} - \frac{1}{2x_{k-1}4^{1/(k-1)}}\right) \\
&< \frac{1}{3.5^k}
\end{aligned}$$

The last inequality holds if $x_{k-1} < 4$. Thus, we can say that $x_k > 3.5$ if $x_{k-1} < 4$.

Combine the above two statements and $x_6 = 3.57$, this lemma can be proved by induction. □

From Lemma 1, we conclude that x_k is in the range $(3.5, 4)$ when $k \geq 6$. Therefore,

Theorem 2 *The competitive ratio of the algorithm for* 1*-space bounded* d*-dimensional bin packing is* 4^d.

3 1-Space bounded d-dimensional hypercube packing

In this section, we consider a special case of the multi-dimensional bin packing, where each item is a hypercube with side length no more than 1. Since the lengths of each side of a hypercube are same, this kind of items can be packed regularly inside a bin. We will first describe the packing strategy for hypercubes, then give the performance analysis to show the competitive ratio of this strategy is 2^{d+1}.

3.1 Packing strategy

Our packing strategy for hypercubes is based on the following two observations.

1. In d-dimensional hypercube packing, an item with side length $2^{-i-1} < x \leq 2^{-i}$ can be packed into a hypercube with side length 2^{-i}, and the occupation ratio in this hypercube is at least 2^{-d}.
2. d-dimensional hypercube can be regularly partitioned: a hypercube with side length 2^{-i} can be partitioned into 2^d smaller hypercubes with side length 2^{-i-1}.

In our packing strategy, we will find a hypercube with proper size for each incoming item. If a hypercube inside the bin is assigned to pack an item, this hypercube cannot be used for other item. According to the first observation, the occupation ratio in this hypercube is guaranteed. To pack an item into a hypercube with proper size, we may partition the bin regularly according to the method from the second observation.

By implementing the partition mentioned above, we can define the packing configuration of the active bin as follows. Define 2^{-i}*-hypercube* to be the hypercube with side length 2^{-i}. Let $B = (b_0, b_1, b_2, \ldots)$ denote the current packing configuration of an active bin, where b_i denotes the number of empty 2^{-i}-hypercubes. If $b_j = 0$ for all $j \geq i$, we will ignore b_j $(j \geq i)$ in B.

Initially, the whole bin is empty, in such configuration, $B = (1)$. Suppose an item with side length $1/3$ comes, to pack this item, the bin will be partitioned into 2^d $1/2$-hypercubes and use one $1/2$-hypercube to pack this item. After that, the configuration will be changed to $B = (0, 2^d - 1)$.

Next, we will describe the packing strategy based on the current configuration B and the coming item with side length x. W.l.o.g., suppose $2^{-i-1} < x \leq 2^{-i}$.

Procedure Hypercube Packing for an item with side length $2^{-i-1} < x \leq 2^{-i}$
  If $[b_i > 0]$
    Select one 2^{-i}-hypercube for packing this item.
    Modify B by decrease b_i by one.
  Otherwise, if [there exist $b_j > 0$ for some $j < i$.]
    Let j be the largest integer such that $b_j > 0$ and $j < i$.
    Let $k = j$.
    Repeat
      Partition one 2^{-k}-hypercube into 2^d 2^{-k-1}-hypercubes.
      $k = k + 1$.
    Until $k = i$.
    Using one 2^{-i}-hypercube to pack the coming item,
    Modify the configuration B by $b_j = b_j - 1$, $b_\ell = 2^d - 1$ for $j < \ell \leq i$.
  Otherwise
    This item cannot be packed into this active bin.
    Close this bin then open a new bin for packing this item.
    Let $k = 0$.
    Repeat
      Partition one 2^{-k}-hypercube into 2^d 2^{-k-1}-hypercubes.
      $k = k + 1$.
    Until $k = i$.
    Using one 2^{-i}-hypercube to pack the coming item.
    After packing, $b_0 = 0, b_1 = \cdots = b_i = 2^d - 1$.
end Hypercube Packing

3.2 Performance analysis

Lemma 3 *In a configuration* B, $0 \leq b_i \leq 2^d - 1$ *and all empty* 2^{-i}*-hypercubes are within a* 2^{-i+1}*-hypercube.*

Proof From the packing strategy, only when a 2^{-i}-hypercube will be used and $b_i = 0$, we partition a larger hypercube to create 2^d 2^{-i}-hypercubes. After the partition, one 2^{-i}-hypercube will be used, either for packing an item, or for partitioning to smaller hypercubes. Thus, b_i is at most $2^d - 1$.

When these 2^d 2^{-i}-hypercubes are created, they are within a 2^{-i+1}-hypercube. Only when these 2^d hypercubes are all used up, we will open another 2^d 2^{-i}-hypercubes. Therefore, at any time, all empty 2^{-i}-hypercubes are within a 2^{-i+1}-hypercube. □

Lemma 4 *In a configuration B, for any $i \geq 0$, the total size of empty 2^{-j}-hypercubes $(j > i)$ is no more than the size of a 2^{-i}-hypercube.*

Proof Suppose in the configuration B, $i < j_1 < j_2 < \cdots < j_\ell$ such that $b_{j_x} > 0$ $(1 \leq x \leq \ell)$. From Lemma 3, we have $b_{j_x} \leq 2^d - 1$. In d-dimensional space, the ratio between the sizes of a 2^{-j}-hypercube and a 2^{-j-1}-hypercube is 2^d. Thus, the total size of these 2^{-j_x}-hypercubes is no more than the size of a 2^{-i}-hypercube. □

Theorem 5 *The competitive ratio of the packing strategy for hypercubes is 2^{d+1}.*

Proof From the packing strategy, the occupation ratio of any used 2^{-i}-hypercube is at least 2^d. Thus, for an active bin with configuration $B = (b_0, b_1, b_2, \ldots)$, the total occupation in this bin is at least

$$\left(1 - \sum_{j \geq 0} b_j \cdot 2^{-jd}\right) \cdot 2^{-d}.$$

Suppose at this time, an item comes and we have to open a new bin according to the packing strategy. This happens when the item with side length $2^{-i-1} < x \leq 2^{-i}$ and $b_j = 0$ for all $j \leq i$ in the configuration B. The size of this item is $x^d > 2^{-id-d}$.

From Lemma 4, the total sizes of the empty hypercubes is no more than the size of a 2^{-i}-hypercube. The average occupation in these two bins is at least

$$\frac{(1 - \sum_{i>i} b_j \cdot 2^{-jd}) \cdot 2^{-d} + 2^{-id-d}}{2} \geq 2^{-d-1}$$

Suppose the packing strategy uses ℓ bins for a sequence of items, and the occupation in the i-th bin is c_i. Thus, the total occupation is

$$\sum_{i=1}^{\ell} c_i = \frac{c_1}{2} + \frac{c_\ell}{2} + \sum_{i=1}^{\ell-1} \left(\frac{c_i + c_{i+1}}{2}\right) > (\ell - 1) \cdot 2^{-d-1}$$

The above value is a lower bound of the used bins from the optimal offline algorithm. Thus, the competitive ratio of the packing strategy is at most

$$\frac{\ell}{(\ell - 1) \cdot 2^{-d-1}}$$

In the bin packing problem, we are interested in the asymptotic competitive ratio, according to the above analysis, this ratio is 2^{d+1}. □

References

Bansal N, Correa JR, Kenyon C, Sviridenko M (2006a) Bin packing in multiple dimensions: inapproximability results and approximation schemes. Math Oper Res 31(1):31–49

Bansal N, Caprara A, Sviridenko M (2006b) Improved approximation algorithm for multidimensional bin packing problems. In: FOCS 2006, pp 697–708

Blitz D, van Vliet A, Woeginger GJ (1996) Lower bounds on the asymptotic worst-case ratio of on-line bin packing algorithms. Unpublished manuscript

Caprara A (2002) Packing 2-dimensional bins in harmony. In FOCS 2002, pp 490–499

Chin FYL, Ting H-F, Zhang Y (2012) 1-space bounded algorithms for 2-dimensional bin packing. Int J Found Comput Sci (to appear)

Chung FRK, Garey MR, Johnson DS (1982) On packing two-dimensional bins. SIAM J Algebr Discrete Methods 3(1):66–76

Coppersmith D, Raghavan P (1989) Multidimensional on-line bin packing: algorithms and worst case analysis. Oper Res Lett 8:17–20

Csirik J, Johnson DS (2001) Bounded space on-line bin packing: best is better than first. Algorithmica 31:115–138

Csirik J, Frenk J, Labbe M (1993) Two-dimensional rectangle packing: on-line methods and results. Discrete Appl Math 45(3):197–204

Epstein L, van Stee R (2005a) Online square and cube packing. Acta Inform 41(9):595–606

Epstein L, van Stee R (2005b) Optimal online algorithms for multidimensional packing problems. SIAM J Comput 35(2):431–448

Epstein L, van Stee R (2007) Bounds for online bounded space hypercube packing. Discrete Optim 4:185–197

Ferreira CE, Miyazawa EK, Wakabayashi Y (1999) Packing squares into squares. Pesqui Oper 19:223–237

Fujita S (2003) On-line grid-packing with a single active grid. Inf Process Lett 85:199–204

Garey MR, Johnson DS (1979) Computers and intractability: a guide for the theory of NP-completeness. Freeman, San Francisco

Han X, Fujita S, Guo H (2001) A two-dimensional harmonic algorithm with performance ratio 2.7834. IPSJ SIG Not 93:43–50

Han X, Iwama K, Zhang G (2008) Online removable square packing. Theory Comput Syst 43(1):38–55

Han X, Chin F, Ting H-F, Zhang G, Zhang Y (2011) A new upper bound 2.5545 on 2D online bin packing. ACM Trans Algorithms 7(4):50

Januszewski J, Lassak M (1997) On-line packing sequences of cubes in the unit cube. Geom Dedic 67:285–293

Johnson DS, Garey MR (1985) A 71/60 theorem for bin-packing. J Complex 1:65–106

Johnson DS, Demers AJ, Ullman JD, Garey MR, Graham RL (1974) Worst-case performance bounds for simple one-dimensional packing algorithms. SIAM J Comput 3(4):299–325

Karmarkar N, Karp RM (1982) An efficient approximation scheme for the one-dimensional bin packing problem. In: Proc 23rd ann IEEE symp on foundations of comput sci. IEEE Computer Society, Los Alamitos, pp 312–320

Kohayakawa Y, Miyazawa FK, Raghavan P, Wakabayashi Y (2004) Multidimensionalcube packing. Algorithmica 40(3):173–187

Lee CC, Lee DT (1985) A simple on-line bin packing algorithm. J Assoc Comput Mach 32:562–572

Leung JY-T, Tam TW, Wong CS, Young GH, Chin FYL (1990) Packing squares into a square. J Parallel Distrib Comput 10:271–275

Meir A, Moser L (1968) On packing of squares and cubes. J Comb Theory 5:126–134

Ramanan PV, Brown DJ, Lee CC, Lee DT (1989) On-line bin packing in linear time. J Algorithms 10:305–326

Seiden SS (2002) On the online bin packing problem. J ACM 49:640–671

Seiden S, van Stee R (2003) New bounds for multi-dimensional packing. Algorithmica 36:261–293

Simchi-Levi D (1994) New worst-case results for the bin-packing problem. Nav Res Logist 41:579–585
van Vliet A (1992) An improved lower bound for on-line bin packing algorithms. Inf Process Lett 43:277–284
Yao AC-C (1980) New algorithms for bin packing. J ACM 27:207–227
Zhang Y, Chen J, Chin FYL, Han X, Ting H-F, Tsin YH (2010) Improved online algorithms for 1-space bounded 2-dimensional bin packing. In: Proc of the 21th annual international symposium on algorithms and computation (ISAAC 2010). LNCS, vol 6507. Springer, Berlin, pp 242–253