

Better Algorithms for Online Bin Stretching^{*}

Martin Böhm¹, Jiří Sgall¹, Rob van Stee² and Pavel Veselý¹

¹ Computer Science Institute of Charles University, Prague, Czech Republic.
`{bohm,sgall,vesely}@iuuk.mff.cuni.cz`.

² Department of Computer Science, University of Leicester, Leicester, UK.
`rob.vanstee@leicester.ac.uk`.

Abstract. ONLINE BIN STRETCHING is a semi-online variant of bin packing in which the algorithm has to use the same number of bins as the optimal packing, but is allowed to slightly overpack the bins. The goal is to minimize the amount of overpacking, i.e., the maximum size packed into any bin.

We give an algorithm for ONLINE BIN STRETCHING with a stretching factor of 1.5 for any number of bins. We also show a specialized algorithm for three bins with a stretching factor of $11/8 = 1.375$.

1 Introduction

The most famous algorithmic problem dealing with online assignment is arguably ONLINE BIN PACKING. In this problem, known since the 1970s, items of size between 0 and 1 arrive in a sequence and the goal is to pack these items into the least number of unit-sized bins, packing each item as soon as it arrives.

ONLINE BIN STRETCHING, which has been introduced by Azar and Regev in 1998 [3], deals with a similar online scenario. Again, items of size between 0 and 1 arrive in a sequence, and the algorithm needs to pack them as soon as each item arrives, but it has two advantages: (i) The packing algorithm knows m , the number of bins that an optimal offline algorithm would use, and must also use only at most m bins, and (ii) the packing algorithm can use bins of capacity R for some $R \geq 1$. The goal is to minimize the stretching factor R .

While formulated as a bin packing variant, ONLINE BIN STRETCHING can also be thought of as a semi-online scheduling problem, in which we schedule jobs in an online manner on exactly m machines, before any execution starts. We have a guarantee that the optimum offline algorithm could schedule all jobs with makespan 1. Our task is to present an online algorithm with makespan of the schedule being at most R .

^{*} Partially supported by the project 14-10003S of GA ĀR and by the GAUK project 548214.

Motivation. We give two of applications of ONLINE BIN STRETCHING.

Server upgrade. This application has first appeared in [3]. In this setting, an older server (or a server cluster) is streaming a large number of files to the newer server without any guarantee on file order. Both servers have m disk drives, but the newer server has a larger capacity of each drive. The goal is to present an algorithm that stores all incoming files from the old server as they arrive.

Shipment checking. A number m of containers arrive at a shipping center. It is noted that all containers are at most $p \leq 100$ percent full. The items in the containers are too numerous to be individually labeled, yet all items must be unpacked and scanned for illicit and dangerous material. After the scanning, the items must be speedily repackaged into the containers for further shipping. In this scenario, an algorithm with stretching factor $100/p$ can be used to repack the objects into containers in an online manner.

History. ONLINE BIN STRETCHING itself has been proposed by Azar and Regev [3]. The original lower bound of $4/3$ for three bins has appeared even before that, in [12], for two bins together with a matching algorithm. Azar and Regev extended the same lower bound to any number of bins.

The problem has been revisited recently, with both lower bound improvements and new efficient algorithms. On the algorithmic side, Kellerer and Kotov [11] have achieved a stretching factor $11/7 \approx 1.57$ and Gabay et al. [8] have achieved $26/17 \approx 1.53$. In the case with only three bins, the previously best algorithm was due to Azar and Regev [3], with a stretching factor of 1.4.

On the lower bound side, the lower bound $4/3$ of [3] was surpassed only for the case of three bins by Gabay et al. [7], who show a lower bound of $19/14$, using an extensive computer search.

Our contributions. In Section 2, we present a new algorithm for ONLINE BIN STRETCHING with a stretching factor of 1.5. We build on the techniques of [11,8] who designed two-phase algorithms where the first phase tries to fill some bins close to $R - 1$ and achieve a fixed ratio between these bins and empty bins, while the second phase uses the bins in blocks of fixed size and analyzes each block separately. This technique, with some case analysis, seemed to be able to lead to improved results approaching 1.5. To actually reach 1.5, we needed to significantly improve the analysis using amortization techniques (represented by a weight function in our presentation) to amortize among blocks and bins of different types.

In Section 3, we focus on the case of three bins. For this case, there is a recent lower bound of $19/14 \approx 1.357$. We present an algorithm for three bins of capacity $11/8 = 1.375$. This is the first improvement of the stretching factor 1.4 of Azar and Regev [3] for three bins and significantly decreases the remaining gap.

Related work. The NP-hard problem BIN PACKING was originally proposed by Ullman [13] and Johnson [10] in the 1970s. Since then it has seen major interest and progress, see the survey of Coffman et al. [5] for many results on classical Bin Packing and its variants. While our problem can be seen as a variant of BIN PACKING, note that the algorithms cannot open more bins than the optimum and thus general results for BIN PACKING do not translate to our setting.

As noted, ONLINE BIN STRETCHING can be formulated as the online scheduling on m identical machines with known optimal makespan. Such algorithms were studied and are important in designing constant-competitive algorithms without the additional knowledge, e.g., for scheduling in the more general model of uniformly related machines [2,4,6].

For scheduling, also other types of semi-online algorithms are studied. Historically first is the study of ordered sequences with non-decreasing processing times [9]. Most closely related is the variant with known sum of all processing times studied in [12] and the currently best results are a lower bound of 1.585 and an algorithm with ratio 1.6, both from [1]. Note that this shows, somewhat surprisingly, that knowing the actual optimum gives a significantly bigger advantage to the online algorithm over knowing just the sum of the processing times (which, divided by m , is a lower bound on the optimum).

Definitions and notation. Our main problem, ONLINE BIN STRETCHING, can be described as follows:

Input: an integer m , a stretching factor R , and a sequence of items $I = i_1, i_2, \dots$ given online one by one. Each item has a *size* $s(i) \in [0, 1]$.

Output: Partitioning (packing) of I into bins B_1, \dots, B_m so that $\sum_{i \in B_j} s(i) \leq R$ for all $j = 1, \dots, m$.

Guarantee: there exists a packing of all items of the sequence I into m bins of capacity 1.

Goal: Design an online algorithm with the stretching factor R as small as possible which packs all input sequences satisfying the guarantee.

For a bin B , we define the *size of the bin* $s(B) = \sum_{i \in B} s(i)$. Unlike $s(i)$, $s(B)$ can change during the course of the algorithm, as we pack more and more items into the bin. To easily differentiate between items, bins

and lists of bins, we use lowercase letters for items (i, b, x) , uppercase letters for bins (A, B, X) , and calligraphic letters for lists of bins $(\mathcal{A}, \mathcal{C}, \mathcal{L})$.

In both sections of our paper, we rescale the item sizes and bin capacities for simplicity. Therefore, in our setting, each item has an associated size $s(i) \in [0, k]$, where $k \in \mathbb{N}$ is also the capacity of the bins which the optimal offline algorithm uses. The online algorithm for ONLINE BIN STRETCHING uses bins of capacity $t \in \mathbb{N}$, $t \geq k$. The resulting stretching factor is thus t/k .

2 Upper bound for an arbitrary number of bins

Theorem 2.1. *There exists an algorithm for ONLINE BIN STRETCHING with a stretching factor of 1.5 for an arbitrary number of bins with.*

We rescale the bin sizes so that the optimal bins have size 12 and the bins of the algorithm have size 18. We take an instance with an optimal packing into m bins of size 12 and, assuming that our algorithm fails, we derive a contradiction. One way to get a contradiction is to show that the size of all items is larger than $12m$. We also use two other bounds in the spirit of weight functions: weight $w(i)$ and value $v(i)$. The weight $w(i)$ is a slightly modified size to account for items of size larger than 6. The value $v(i)$ only counts the number of items with relatively large sizes. Since the number of bins is fixed, it is convenient to normalize the functions so that they are at most 0 for optimal bins.

We classify the items and define their value $v(i)$ as follows.

Type	huge	large	medium	regular
$s(i)$	$(9, 12]$	$(6, 9]$	$(3, 4]$	$(0, 3] \cup (4, 6]$
$v(i)$	3	2	1	0

Definition 2.2. *For a set of items A , we define the value of A as $v(A) = (\sum_{i \in A} v(i)) - 3$ and we define weight $w(A)$ as follows. Let $k(A)$ be the number of large and huge items in A . Then $w(A) = s(A) + k(A) - 13$. For a set of bins \mathcal{A} we define $v(\mathcal{A}) = \sum_{A \in \mathcal{A}} v(A)$, $w(\mathcal{A}) = \sum_{A \in \mathcal{A}} w(A)$ and $k(\mathcal{A}) = \sum_{A \in \mathcal{A}} k(A)$.*

Lemma 2.3. *For any packing of a valid instance into m bins \mathcal{A} of any size, we have $w(\mathcal{A}) \leq 0$ and $v(\mathcal{A}) \leq 0$.*

Proof. For the value $v(\mathcal{A})$, no optimal bin can contain items with the sum of their values larger than 3. The bound follows by summing over all bins

and the fact that the number of bins is the same for the optimum and the algorithm.

For $w(\mathcal{A})$, we have $s(\mathcal{A}) \leq 12m$ and $k(\mathcal{A}) \leq m$, as the optimum packs all items in m bins of volume 12 and no bin can contain two items larger than 6. Thus $w(\mathcal{A}) = s(\mathcal{A}) + k(\mathcal{A}) - 13m \leq 12m + m - 13m = 0$. \square

First phase.

During the first phase, our algorithm maintains the invariant that only bins of the following types exist.

E Empty bins: bins that have no item.

G Complete bins: all bins that have $w(A) \geq 0$ and $s(A) \geq 12$;

H Huge-item bins: all bins that contain a huge item (plus possibly some other items) and have $s(A) < 12$;

L One large-item bin: a bin containing only a single large item;

M One medium-item bin: a bin A with $s(A) \leq 13$ and only medium items;

T One tiny bin: a bin with $s(A) \leq 3$;

R Regular bins: all other bins with $s(A) \in (3, 6]$;

First-phase algorithm:

Let e be the number of empty bins and r the number of regular bins. If $r \geq 3e$, stop the first phase.

Assign the current item i according to the following rules, always using the first possible option. If there are two types listed, the new bin type depends on the new size and weight of the bin. When packing regular items, the item is packed in a regular or tiny bin only if the total size packed into this bin does not exceed 6 afterwards.

Item type	huge	large	medium	regular
Option 1	R \rightarrow G	L \rightarrow G	M \rightarrow G/M	H \rightarrow G/H
Option 2	T \rightarrow H	E \rightarrow L	E \rightarrow M	R \rightarrow R
Option 3	E \rightarrow H			T \rightarrow T/R
Option 4				E \rightarrow T/R

First we observe that the algorithm described in the box above is properly defined. For every type of item, packing it into an empty bin is an option, and the stopping criterion guarantees that the algorithm stops when no empty bin is available. We now state properties of the algorithm; all are simple invariants that follow from the description of the algorithm.

Lemma 2.4. *At any time during the first phase the following holds:*

- (i) *All bins used by the algorithm are of type **E**, **G**, **H**, **L**, **M**, **T**, or **R**.*

- (ii) All complete bins B have $s(B) \geq 12$, $v(B) \geq 0$, and $w(B) \geq 0$.
- (iii) If there is a huge-item bin, there is no regular and no tiny bin.
- (iv) There is at most one large-item bin and at most one medium-item bin.
- (v) There is at most one tiny bin T . If T exists, then for any regular bin, $s(T) + s(R) > 6$. There is at most one regular bin R with $s(R) \leq 4$.
- (vi) At the end of the first phase $3e \leq r \leq 3e + 3$.

Proof. (i): This follows from the table and in particular from the fact that each regular and tiny bin is assigned a total size of at most 6, unless a huge item is assigned to it.

(ii): If a complete bin is created from a medium-item bin M , we know that $w(M) \geq 0$ and $k(M) = 0$, thus $s(M) \geq 13$ and M contains at least 4 medium items. It follows that $v(M) \geq 1$. Otherwise a complete bin has either a huge item and size at least 12 or two large items and the bounds follow trivially.

(iii): If a huge-item bin exists, any regular item is packed there, so no new tiny or regular bin can be created. If a regular or tiny bin exists, any huge item is packed there, so no new huge-item bin can be created.

(iv): We always pack a large resp. medium item in an existing large-item resp. medium-item bin if such a bin exists. If a medium bin M did not become complete, we have $s(M) < 13$, which implies that any medium item fits there. Similarly, any large item fits into a large-item bin, which can contain only a single large item.

(v): A regular item is packed in a new bin only if it does not fit into existing regular and tiny bins. It follows that the sum of the sizes of any two of them is more than 6 and at most one tiny bin exists.

Suppose now that we have a regular bin R with $s(R) \leq 4$ and a second regular bin with size in $(3, 4]$ is created by packing an item i . It cannot be created from an empty bin, as the single item in it would be a medium item. So it has to be created from a tiny bin T . However, then $s(T) + s(R) > 6$ and $s(R) + s(i) > 6$; it follows that $s(T) > 2$ and $s(i) > 2$, which is a contradiction with the assumption that putting i into T creates a bin of size at most 4.

(vi): Consider the quantity $r - 3e$. In each step, both r and e change at most by 1, so overall $r - 3e$ can increase by at most 4. Thus in the first step when $r - 3e \geq 0$ we have $r - 3e \leq 3$. \square

If the algorithm packs all items in the first phase, it stops. Otherwise according to Lemma 2.4(iii) we split the algorithm in two very different branches. If there is no regular bin, follow the second phase with huge-

item bins below. If there is at least one regular bin, follow the second phase with regular bins.

Let \mathcal{G} be the set of all complete bins; we do not use these bins in the second phase. In addition to \mathcal{G} and either huge-item bins, or the regular and empty bins, there may exist at most three *special bins* denoted and ordered as follows: the large-item bin L , the medium item bin M , and the tiny bin T .

Second phase with huge-item bins.

Let the list of bins \mathcal{L} contain first all the huge-item bins, followed by the special bins L , M , in this order, if they exist. There are no other non-empty bins by Lemma 2.4 and no empty bins because we have $3e \leq r = 0$. We use First Fit on \mathcal{L} , without allowing new bins to be opened. Suppose that we have an instance that has a packing into bins of capacity 12 and on which our algorithm fails. We may assume that the algorithm fails on the last item f . By considering the total volume, there always exists a bin with size at most 12. Thus $s(f) > 6$ and $v(f) \geq 2$.

If during the second phase an item n with $s(n) \leq 6$ is packed into the last bin in \mathcal{L} , we know that all other bins have size more than 12, thus all the remaining items fit into the last bin. Otherwise we consider $v(\mathcal{L})$. Any bin $B \in \mathcal{G}$ has $v(B) \geq 0$ by Lemma 2.4(ii) and each huge-item bin gets nonnegative value too. Also $v(L) \geq -1$ if L exists. This shows that M must exist, since otherwise $v(\mathcal{L}) + v(f) \geq -1 + 2 \geq 1$, a contradiction.

M is the last bin of \mathcal{L} and thus in this last case M contains only medium items from the first phase and possibly large and/or huge items from the second phase. We claim that $v(M) + v(f) \geq 2$ using the fact that f does not fit into M and M contains no item a with $v(a) = 0$: If f is huge we have $s(M) > 6$, thus M must contain either two medium items or one medium item and one large or huge item and $v(M) \geq -1$. If f is large, we have $s(M) > 9$; thus M contains either three medium items or one medium and one large or huge item and $v(M) \geq 0$. Thus we always have $v(\mathcal{L}) \geq -1 + v(M) + v(f) \geq 1$, a contradiction.

Second phase with regular bins.

Let \mathcal{E} resp. \mathcal{R} be the set of empty resp. regular bins at the beginning of the second phase, and let $e = |\mathcal{E}|$. Let $\lambda \in \{0, 1, 2, 3\}$ be such that $|\mathcal{R}| = 3e + \lambda$; Lemma 2.6(vi) implies that it exists.

We order the bins that are not complete into a list \mathcal{L} as follows. We group the bins in $\mathcal{E} \cup \mathcal{R}$ into blocks of typically one empty and three regular bins as follows. Denote the empty bins E_1, E_2, \dots, E_e . The regular bins are denoted by $R_{i,j}$, $i = 1, \dots, e + 1$, $j = 1, 2, 3$. The i th block \mathcal{B}_i consists

of bins $R_{i,1}, R_{i,2}, R_{i,3}, E_i$ in this order. There are two exceptions: The last block \mathcal{B}_{e+1} has no empty bin, only exactly 3 regular bins. The first block contains only λ regular bins instead of 3 and an empty bin. As the first regular bin we choose the one with size less than 4, if there is such a bin. By Lemma 2.4(v) there exists at most one such bin and all the remaining $R_{i,j}$ have size at least 4. Denote the first regular bin by \bar{R} if $\mathcal{R} \neq \emptyset$; note that \bar{R} is either the first bin in \mathcal{B}_1 or the first bin in \mathcal{B}_2 if $\lambda = 0$.

The list of bins \mathcal{L} we use in the second phase contains first the special bins and then all the blocks $\mathcal{B}_1, \dots, \mathcal{B}_{e+1}$. Thus the list \mathcal{L} is (some or all of the first six bins may not exist):

$$L, M, T, R_{1,1}, R_{1,2}, R_{1,3}, E_1, R_{2,1}, R_{2,2}, R_{2,3}, E_2, \dots, \\ R_{e,1}, R_{e,2}, R_{e,3}, E_e, R_{e+1,1}, R_{e+1,2}, R_{e+1,3}.$$

Whenever we refer to the ordering of the bins, we mean the ordering in \mathcal{L} .

We use First Fit on the reversed list \mathcal{L} for huge items (that is, we pack each huge item to the last bin in \mathcal{L} where it fits) and we use First Fit on \mathcal{L} for all other items.

Suppose that we have an instance that has a packing into bins of capacity 12 and on which our algorithm fails. We may assume that the algorithm fails on the last item. Let us denote this item by f . Call the items that arrived in the second phase *new* (including f), the items from the first phase are *old*.

Our overall strategy is to obtain a contradiction by showing that

$$w(\mathcal{L}) + w(f) > 0.$$

In some cases, we instead argue that $v(\mathcal{L}) + v(f) > 0$ or $s(\mathcal{L}) + s(f) > 12|\mathcal{L}|$. Any of these is sufficient for a contradiction, as all bins in \mathcal{G} have both volume and weight nonnegative and size larger than 12. Note also that $s(f) > 6$ since by considering the total volume, there always exists a bin with size at most 12.

Let \mathcal{H} denote all the bins from \mathcal{L} with a huge item, and let $h = |\mathcal{H}| \bmod 4$. First we show that the average size of bins in \mathcal{H} is large and exclude some degenerate cases.

Lemma 2.5. *Let ρ be the total size of old items in \bar{R} if $\bar{R} \in \mathcal{H}$, otherwise set $\rho = 4$.*

- (i) *The bins \mathcal{H} are a final segment of the list and $\mathcal{H} \subsetneq \mathcal{E} \cup \mathcal{R}$.*
- (ii) *We have $s(\mathcal{H}) \geq 12|\mathcal{H}| + h + \rho - 4$.*

- (iii) If \mathcal{H} does not include \bar{R} , then $s(\mathcal{H}) \geq 12|\mathcal{H}| + h \geq 12|\mathcal{H}|$.
- (iv) If \mathcal{H} includes \bar{R} , then $s(\mathcal{H}) \geq 12|\mathcal{H}| + h - 1 \geq 12|\mathcal{H}| - 1$.

Proof. Let $\mathcal{H}' = \mathcal{H} \cap (\mathcal{E} \cup \mathcal{R})$. We first prove the lemma for \mathcal{H}' . That is, we ignore the special bins at this stage.

If a huge item does not fit into a bin B in $\mathcal{E} \cup \mathcal{R}$ with no huge item, then B contains a new item of size at most 9 which did not fit into any previous bin. Thus the huge item also does not fit into any previous bin, and it is the item f . This shows that \mathcal{H}' is a final segment of the list, and also that if \mathcal{H}' is not the entire $\mathcal{E} \cup \mathcal{R}$, then no special bin has a huge item and $\mathcal{H}' = \mathcal{H}$.

All bins $R_{i,j}$ with a huge item have size at least $4 + 9 = 13$, with a possible exception of the first such bin which has size at least $\rho + 9 = 13 + \rho - 4$, by the definition of ρ . Each E_i with a huge item has size at least 9. Thus for each i with $E_i \in \mathcal{H}'$, $s(E_i) + s(R_{i+1,1}) + s(R_{i+1,2}) + s(R_{i+1,3}) \geq 4 \cdot 12$, with a possible exception of $i = 1$ in the case $\lambda = 0$. Summing over all i with $E_i \in \mathcal{H}'$ and the h bins in \mathcal{R} from the first block intersecting \mathcal{H}' , and adjusting for \bar{R} if $\bar{R} \in \mathcal{H}'$, (ii) for \mathcal{H}' follows. The claims (iii) and (iv) for \mathcal{H}' are an immediate consequence as $\rho > 3$ if $\bar{R} \in \mathcal{H}'$ and $\rho = 4$ otherwise.

It remains to show that $\mathcal{H}' \neq \mathcal{E} \cup \mathcal{R}$, implying $\mathcal{H}' = \mathcal{H}$. Suppose for a contradiction that $\mathcal{H}' = \mathcal{E} \cup \mathcal{R}$, thus

$$s(\mathcal{H}') \geq 12|\mathcal{H}'| - 1.$$

We distinguish four cases.

Case 1: There are no special bins. Then each bin B is a huge-item bin and has $v(B) \geq 0$ and in addition f is not packed and it has $v(f) \geq 2$. So, overall $v(\mathcal{L}) + v(f) > 0$, a contradiction.

Case 2: There is only one special bin. Then its size together with f is more than 18, thus $s(\mathcal{L}) > 18 + 12|\mathcal{H}'| - 1 > 12|\mathcal{L}|$, a contradiction.

Case 3: The last special bin $B \neq L$ has a new item n ; this is always the case if T exists. We first claim that

$$s(B) - s(n) + s(\mathcal{H}') > 12|\mathcal{H}'|. \tag{1}$$

If $B = M$ then there is at least one old item in M of size more than 3 and (1) follows. If $B = T$ and it has old items of total size o , we have $o + \rho > 6$ by Lemma 2.4(v) and (1) also follows, as from (ii) we get $s(B) - s(n) + s(\mathcal{H}') = o + s(\mathcal{H}') \geq o + 12|\mathcal{H}'| + \rho - 4 > 12|\mathcal{H}'| + 2$.

If there are two special bins, then the size of the first one plus $s(n)$ is more than 18 because n did not fit there, $s(f) > 6$ and thus summing with (1) we get $s(\mathcal{L}) + s(f) > 18 + 6 + 12|\mathcal{H}'| = 12|\mathcal{L}|$, a contradiction.

If there are three special bins (necessarily L , M , and $B = T$), we have $s(L) + s(n) > 18$ and $s(M) + s(f) > 18$, as the items do not fit in the respective bins, and summing with (1) we get $s(\mathcal{L}) + s(f) > 18 + 18 + 12|\mathcal{H}'| = 12|\mathcal{L}|$, a contradiction.

Case 4: There are two special bins L and M , moreover M has no new item (otherwise Case 3 applies). Then either f is huge and M has at least two medium items, or f is large and M has at least three items. Thus $v(M) + v(f) \geq 2$, $v(L) \geq -1$ since L has a large item, and $v(A) \geq 0$ for any other bin A since either it is a complete bin from the first phase or it has a huge item. Summing over all bins we get that the total value $v(\mathcal{L}) > 0$, a contradiction. \square

Let F , the *final bin* be the last bin in \mathcal{L} before \mathcal{H} , or the last bin if $\mathcal{H} = \emptyset$; by Lemma 2.5 we have $F \in \mathcal{E} \cup \mathcal{R}$. Now modify the packing so that f is put into F , f is also considered a new item. Thus $s(F) > 18$ and f as well as all the new items packed in F or a bin before F satisfy the property that they do not fit into any previous bin. Let C , the *critical bin*, be the first bin in \mathcal{L} of size at most 12; such a bin exists, as otherwise the total size is more than $12m$.

We start by some easy observations. Only items of size at most 9 are packed in bins before F , in F itself only the item f can be larger. All the new items in the bins after C are large; f can be also huge. Each bin, possibly with the exception of L and M , contains a new item, as it enters the phase with size at most 6, and the algorithm failed. Each bin in \mathcal{E} before F contains two new items. The bin F always has two new items, one that did fit into it and f . More observations are given in the next two lemmata.

Lemma 2.6. (i) *Let B be any bin before F . Then $s(B) > 9$.*

(ii) *Let B, B', B'' be any three bins in this order before or equal to F and let B'' contain two new items. Then $s(B) + s(B') + s(B'') > 36 + o$, where o is the size of old items in B'' .*

(iii) *Let B be arbitrary and $B' \in \mathcal{R}$ after both B and C .*

If $B' \neq \bar{R}$ then $s(B) + s(B') > 22$, in particular $s(B) > 11$ or $s(B') > 11$.

If $B' = \bar{R}$ then $s(B) + s(B') > 21$.

Proof. F contains a new item n different from f . To prove (i), note that $s(n) \leq 9$, and n does not fit into B .

To prove (ii), let n, n' be two new items in B'' and note that $s(B) + s(n) > 18$ and $s(B') + s(n') > 18$.

To prove (iii), let B' be after B . The bin B' has a new item of size larger than $18 - s(B)$, and it also has old items of size at least 3 or even 4 if $B' \neq \bar{R}$. \square

Lemma 2.7. *The critical bin C is before F , there are at least two bins between C and F and C is not in the same block as F .*

Proof. All bins before C have size larger than 12. Using Lemma 2.5 we have

$$s(F) + s(\mathcal{H}) > 18 + 12|\mathcal{H}| - 1 = 12(|\mathcal{H}| + 1) + 5.$$

It remains to bound the sizes of the other bins. Note that $F \neq C$ as $s(F) > 18$. If C is after F , all bins before F have size more than 12, so all together $s(\mathcal{L}) > 12|\mathcal{L}| + 5$, a contradiction. If C is just before F , then by Lemma 2.6(i), $s(C) > 9 = 12 - 3$ and the total size of bins in $s(\mathcal{L}) > 12|\mathcal{L}| + 5 - 3 > 12|\mathcal{L}|$, a contradiction.

If there is a single bin B between C and F , then $s(C) + s(B)$ plus the size of two new items in F is more than 36 by Lemma 2.6(ii). If $F \in \mathcal{E}$ then \mathcal{H} starts with three bins in \mathcal{R} and $h = 3$, thus $s(\mathcal{H}) \geq 12|\mathcal{H}| + 2$ by Lemma 2.5, and we get a contradiction. If $F \in \mathcal{R}$ then $\bar{R} \notin \mathcal{H}$, thus Lemma 2.5 gives $s(\mathcal{H}) \geq 12|\mathcal{H}|$, and we get a contradiction as well.

The last case when C and F are in the same block is when $F \in \mathcal{E}$, so $h = 3$, and C is the first bin of the three other bins from the same block, so $\bar{R} \notin \mathcal{H}$. Then $s(C) > 9$, the remaining two bins together with F have size more than 36 by Lemma 2.6(ii) and we use $s(\mathcal{H}) > 12|\mathcal{H}| + 3$ from Lemma 2.5 to get a contradiction. \square

Now we partition \mathcal{L} into several parts. Let $\mathcal{F} = \mathcal{B}_i \cup \mathcal{H}$, where $F \in \mathcal{B}_i$. Let \mathcal{D} be the set of all bins after C and before \mathcal{F} . Let \mathcal{C} be the set of all bins before and including C . Lemma 2.7 shows that the parts are non-overlapping. We analyze the weight of the parts separately, essentially block by block. The proof is relatively straightforward if C is not special (and thus also $\mathcal{F} \notin \mathcal{B}_1$), which is the most important case driving our choices for w . A typical block has nonnegative weight, we gain more weight in the block of F which exactly compensates the loss of weight in \mathcal{C} , which occurs mainly in C itself.

Lemma 2.8. *If F is not in the first block then $w(\mathcal{F}) > 5$, else $w(\mathcal{F}) > 4$.*

Proof. All the new items in bins of \mathcal{F} are large or huge. Each bin has a new item and the bin F has two new items. Thus $k(\mathcal{F}) \geq |\mathcal{F}| + 1$. All

that remains is to show that $s(\mathcal{F}) > 12|\mathcal{F}| + 3$, and $s(\mathcal{F}) > 12|\mathcal{F}| + 4$ if F is not in the first block.

If F is the first bin in a block, the lemma follows as $s(F) > 18$ and $s(\mathcal{H}) \geq 12|\mathcal{H}| - 1$, thus $s(\mathcal{F}) = s(F) + s(\mathcal{H}) > 12|\mathcal{F}| + 5$.

In the remaining cases there is a bin in $\mathcal{R} \cap \mathcal{F}$ before F . Lemma 2.5 gives $s(\mathcal{H}) \geq 12|\mathcal{H}|$; moreover, if $F \in \mathcal{E}$, then $s(\mathcal{H}) \geq 12|\mathcal{H}| + 3$.

If F is preceded by three bins from $\mathcal{F} \cap \mathcal{R}$, then $F \in \mathcal{E}$ and thus $s(\mathcal{H}) \geq 12|\mathcal{H}| + 3$. Using Lemma 2.6(iii) twice, two of the bins in \mathcal{R} before F have size at least 11 and using Lemma 2.6(i) the remaining one has size 9. Thus the size of these four bins is more than $11 + 11 + 9 + 18 = 4 \cdot 12 + 1$, summing with the bound for \mathcal{H} we get $s(\mathcal{F}) > 12|\mathcal{F}| + 4$.

If F is preceded by two bins from $\mathcal{F} \cap \mathcal{R}$, then by Lemma 2.6(i) the total size of these two bins and two new items in F is more than 36. If $F \in \mathcal{R}$, the size of old items in F is at least 4 and with $s(\mathcal{H}) \geq 12|\mathcal{H}|$ we get $s(\mathcal{F}) > 12|\mathcal{F}| + 4$. If $F \in \mathcal{E}$, which also implies that F is in the first block, then $s(\mathcal{H}) \geq 12|\mathcal{H}| + 3$, thus $s(\mathcal{F}) > 12|\mathcal{F}| + 3$.

If F is preceded by one bin R from $\mathcal{F} \cap \mathcal{R}$, then let n be a new item in F different from f . We have $s(R) + s(n) > 18$ and $s(f) > 6$. We conclude the proof as in the previous case. \square

Lemma 2.9. *If $C \in \mathcal{R}$ then $w(C) \geq -6$. If $C \in \mathcal{E}$ then $w(C) \geq -5$. If C is a special bin then $w(C) \geq -4$.*

Proof. For every bin B before C , $s(B) > 12$ and thus $w(B) > -1$ by the definition of C . Let \mathcal{C}' be the set of bins before C with $w(B) \leq 0$. This implies that for $B \in \mathcal{C}'$, $s(B) \in (12, 13]$ and B has no large item. Any new item in any bin after the first bin in \mathcal{C}' has size more than 5. We have

$$w(C) \geq w(\mathcal{C}') + w(C) \geq -|\mathcal{C}'| + w(C). \quad (2)$$

First we argue that either $|\mathcal{C}'| \leq 1$ or $\mathcal{C}' = \{M, T\}$. Suppose that $|\mathcal{C}'| > 1$, choose $B, B' \in \mathcal{C}'$ so that B is before B' . If $B' \in \mathcal{E}$, either B' has at most two (new) non-large items and $s(B') \leq 6 + 6 = 12$, or it has at least three items and $s(B') > 5 + 5 + 5 = 15$; both options are impossible for $B' \in \mathcal{C}'$. If $B' \in \mathcal{R}$, it has old items of total size in $[3, 6]$. Either B' has a single new item and $s(B') \leq 6 + 6 = 12$, or it has at least two new items and $s(B') > 3 + 5 + 5 = 13$; both options are impossible for $B' \in \mathcal{C}'$. The only remaining option is that B' is a special bin. Since L has a large item, $L \notin \mathcal{C}'$ and $\mathcal{C}' = \{M, T\}$.

By Lemma 2.6(i), we have $w(C) \geq -4$ and the lemma follows by (2) if $C \in \mathcal{R}$ or $\mathcal{C}' = \emptyset$ and also if $C \in \mathcal{E}$ and $|\mathcal{C}'| = 1$. For the remaining cases, we need to show $w(C) \geq -3$. If $C \in \mathcal{E}$ and $\mathcal{C}' = \{M, T\}$ then C

contains two new items of size at least 5, thus $w(C) \geq -3$. If $C = T$ and $C' = \{M\}$ then C either has a large item, or it has two new items: otherwise it would have size at most 3 of old items plus at most 6 from a single new item, total of at most 9, contradicting Lemma 2.6(i). Thus $w(C) \geq -3$ in this case as well. \square

Lemma 2.10. (i) *For every block $\mathcal{B}_i \subseteq \mathcal{D}$ we have $w(\mathcal{B}_i) \geq 0$.*
(ii) *If there is no special bin in \mathcal{D} , then $w(\mathcal{D}) \geq 0$. If also $C \in \mathcal{R}$ then $w(\mathcal{D}) \geq 1$.*

Proof. First we claim that for each block $\mathcal{B}_i \subseteq \mathcal{D}$ with three bins in \mathcal{R} , we have

$$w(\mathcal{B}_i) \geq 0. \quad (3)$$

By Lemma 2.6(iii), one of the bins in $\mathcal{R} \cap \mathcal{B}_i$ has size at least 11. By Lemma 2.6(ii), the remaining three bins have size at least 36. We get (3) by observing that $k(\mathcal{B}_i) \geq 5$, as all the new items are large after C , each bin contains a new item and E_i contains two new items.

Next, we consider an incomplete block, that is, a set of bins \mathcal{B} with at most two bins from $\mathcal{R} \cap \mathcal{D}$ followed by a bin $E \in \mathcal{E} \cap \mathcal{D}$. We claim

$$w(\mathcal{B}) \geq 1. \quad (4)$$

The bin E contains two large items, since it is after C . In particular, $w(E) \geq 1$ and (4) follows if $|\mathcal{B}| = 1$. If $|\mathcal{B}| = 2$, the size of one item from E plus the previous bin is more than 18, the size of the other item is more than 6, thus $s(\mathcal{B}) \geq 24$; since $k(\mathcal{B}) \geq 3$, (4) follows. If $|\mathcal{B}| = 3$, by Lemma 2.6(ii) we have $s(\mathcal{B}) \geq 36$; $k(\mathcal{B}) \geq 4$ and (4) follows as well.

By definition, \mathcal{D} ends by a bin in \mathcal{E} (if nonempty). Thus the lemma follows by using (4) for the block of $C \in \mathcal{R}$ or for \mathcal{B}_1 if it does not have three bins in \mathcal{R} and adding (3) for all the remaining blocks. Note that $C \in \mathcal{R}$ implies $\mathcal{D} \neq \emptyset$. \square

We are now ready to derive the final contradiction. If \mathcal{D} does not contain a special bin, we add the appropriate bounds from Lemmata 2.9, 2.10 and 2.8. If $C \in \mathcal{R}$ then F is not in the first block and $w(\mathcal{L}) = w(C) + w(\mathcal{D}) + w(\mathcal{F}) > -6 + 1 + 5 = 0$. If $C \in \mathcal{E}$ then F is not in the first block and $w(\mathcal{L}) = w(C) + w(\mathcal{D}) + w(\mathcal{F}) > -5 + 0 + 5 = 0$. If C is the last special bin then $w(\mathcal{L}) = w(C) + w(\mathcal{D}) + w(\mathcal{F}) > -4 + 0 + 4 = 0$. In all subcases $w(\mathcal{L}) > 0$, a contradiction.

Suppose \mathcal{D} does contain a special bin, thus $C = L$ or $C = M$. We analyze the special bins together with the first block, up to F if F belongs

to it. First observe that the only bin possibly before C is L and in that case $w(L) \geq 0$, so $w(C) \geq w(C)$. Let A denote F if $F \in \mathcal{B}_1$ or E_1 if $F \notin \mathcal{B}_1$. Let \mathcal{A} be the set containing C and all bins between C and A , not including A . Thus \mathcal{A} contains two or three special bins followed by at most three bins from \mathcal{R} . We have $k(\mathcal{A}) \geq |\mathcal{A}| - 1$ as each bin in \mathcal{A} contains a large item, with a possible exception of M . Furthermore $k(A) \geq 2$. Thus

$$w(\mathcal{A}) + w(A) \geq s(\mathcal{A}) + s(A) - 12|\mathcal{A}| - 12$$

and it is sufficient to bound $s(\mathcal{A}) + s(A)$. The precise bound we need depends on what bin A is. Typically, we bound the size by creating pairs of bins of size 21 or 22 by Lemma 2.6(iii). We also use that $S(B) > 9$ for any $B \in \mathcal{A}$ by Lemma 2.6(i) and that n, n' together with any two bins in \mathcal{A} have size at least 36 by Lemma 2.6(ii). Here n and n' are two new items in A .

If $A \neq F$, then $F \notin \mathcal{B}_1$. It is sufficient to prove that $s(\mathcal{A}) + s(A) \geq 12|\mathcal{A}| + 7$ and sum it with all the other bounds, in particular $w(\mathcal{F}) > 5$ from Lemma 2.8 and $w(\mathcal{B}_i) \geq 0$ for whole blocks $\mathcal{B}_i \in \mathcal{D}$ from Lemma 2.10. The items n and n' from A together with the first two special bins in \mathcal{A} have size more than 36. Let \mathcal{A}' be the set of the remaining bins; it contains possibly T and at most three bins from \mathcal{R} . It remains to show $s(\mathcal{A}') \geq 12|\mathcal{A}'| - 5$. For $|\mathcal{A}'| = 0$ it holds trivially. If $|\mathcal{A}'| = 1$, the only remaining bin has size more than 9 and this is sufficient. For $|\mathcal{A}'| > 1$ we apply Lemma 2.6(iii) and pair as many bins from \mathcal{A}' as possible. If $|\mathcal{A}'| = 2$, then $s(\mathcal{A}') > 21 = 2 \cdot 12 - 3$. For $|\mathcal{A}'| = 3$ we get $s(\mathcal{A}') > 22 + 9 = 3 \cdot 12 - 5$, since we can create a pair without \bar{R} . Finally, if $|\mathcal{A}'| = 4$ then $s(\mathcal{A}') > 22 + 21 = 4 \cdot 12 - 5$.

Otherwise $A = F$. If $F = E_1$ then it is sufficient to prove $s(\mathcal{A}) + s(F) > 12|\mathcal{A}| + 9$ and use $w(\mathcal{H}) \geq 3$ from Lemma 2.5(iii) whenever \mathcal{A} contains a bin from \mathcal{R} , i.e., $\bar{R} \notin \mathcal{H}$; otherwise it suffices to prove $s(\mathcal{A}) + s(F) > 12|\mathcal{A}| + 10$ and use $w(\mathcal{H}) \geq 2$ from Lemma 2.5(iv). If $F \in \mathcal{R}$ then it is sufficient to show $s(\mathcal{A}) + s(F) > 12|\mathcal{A}| + 12$ and use $s(\mathcal{H}) \geq 12|\mathcal{H}|$ from Lemma 2.5, as $\bar{R} \notin \mathcal{H}$; furthermore, we know that F also contains old items of size at least 3 or even 4 (if $F \neq \bar{R}$), thus it is sufficient to prove $s(\mathcal{A}) + s(n) + s(n') > 12|\mathcal{A}| + 9$ if $F = \bar{R}$, or $s(\mathcal{A}) + s(n) + s(n') > 12|\mathcal{A}| + 8$ otherwise, i.e., when $\bar{R} \in \mathcal{A}$.

Summarizing and using $s(F) \geq s(n) + s(n')$, for $A = F$ it is sufficient to prove that

$$s(\mathcal{A}) + s(n) + s(n') > 12|\mathcal{A}| + \begin{cases} 8 & \text{if } F \in \mathcal{R} \text{ and } \bar{R} \in \mathcal{A}, \\ 9 & \text{in all cases except if both} \\ & F = E_1 \text{ and } \bar{R} \notin \mathcal{A}, \\ 10 & \text{in all cases.} \end{cases}$$

We now distinguish cases depending on $|\mathcal{A}|$. Note that $\bar{R} \in \mathcal{A}$ whenever $|\mathcal{A}| \geq 4$.

Case $|\mathcal{A}| = 2$: The two bins together with n and n' have size more than 36. Thus $s(\mathcal{A}) + s(n) + s(n') > 36 = 12 \cdot 2 + 12$.

Case $|\mathcal{A}| = 3$: We have $s(C) > 9$ and the remaining two bins together with n and n' have size more than 36. Thus $s(\mathcal{A}) + s(n) + s(n') > 12|\mathcal{A}| + 9$, which is sufficient in all cases except if $F = E_1$ and $\bar{R} \notin \mathcal{A}$. In this remaining case, $C = L$ and $\mathcal{A} = \{L, M, T\}$.

Let o be the size of old items in T . We apply Lemma 2.5(ii), using the fact that $o + \rho > 6$ by Lemma 2.4(v), where ρ is the total size of old items in $\bar{R} \in \mathcal{H}$, and $h = 3$, as \mathcal{H} starts with 3 bins in \mathcal{R} . We get $o + s(\mathcal{H}) \geq o + 12|\mathcal{H}| + h + \rho - 4 > 12|\mathcal{H}| + 5$.

Let n'' be a new item in T . Since n'' does not fit into M , $s(M) + s(n'') > 18$; also $s(C) > 9$ and $s(F) > 18$. Summing all the bounds, we have $s(\mathcal{L}) = o + s(\mathcal{H}) + s(M) + s(n'') + s(C) + s(F) > 12|\mathcal{H}| + 5 + 18 + 9 + 18 = 12|\mathcal{L}| + 2$, a contradiction.

Case $|\mathcal{A}| = 4$: The last bin $R \in \mathcal{A}$ is in \mathcal{R} . Together with any previous bin it has size more than 21, the remaining two bins together with n and n' have size more than 36 by Lemma 2.6(ii). Thus $s(\mathcal{A}) + s(n) + s(n') > 21 + 36 = 4 \cdot 12 + 9$ which suffices, since $\bar{R} \in \mathcal{A}$.

Case $|\mathcal{A}| = 5$: The last two bins of \mathcal{A} are in \mathcal{R} , we pair them with two previous bins to form pairs of size more than $21 + 22$. If $F = E_1$ then the remaining bin has size at least 9, since n does not fit into it and $s(n) < 9$, and $s(F) > 18$, thus $s(\mathcal{A}) + s(A) > 21 + 22 + 9 + 18 = 5 \cdot 12 + 10$.

If $F \in \mathcal{R}$ then one of the last two bins of \mathcal{A} has size more than 11 and the other forms a pair of size more than 21 with one special bin. The remaining two bins together with n and n' have size more than 36 by Lemma 2.6(ii). Thus $s(\mathcal{A}) + s(n) + s(n') > 11 + 21 + 36 = 5 \cdot 12 + 8$ which is sufficient, because $\bar{R} \in \mathcal{A}$.

Case $|\mathcal{A}| = 6$: Then \mathcal{A} contains all three special bins and three bins from \mathcal{R} , therefore also $F = E_1$. We form three pairs of a special bin with a bin

from \mathcal{R} of total size more than $21 + 22 + 22$. Since $s(F) > 18$, we have $s(\mathcal{A}) + s(F) > 21 + 22 + 22 + 18 = 6 \cdot 12 + 11$. Since in this case $A = F = E_1$, we have $s(\mathcal{H}) \geq 12|\mathcal{H}| + 3$ and $s(\mathcal{L}) > 12|\mathcal{L}|$, a contradiction.

In all of the cases we have derived a contradiction, which implies that our algorithm cannot fail. This concludes the proof of Theorem 2.1.

3 Bin stretching on three bins

Theorem 3.1. *There exists an algorithm that solves ONLINE BIN STRETCHING for three bins with stretching factor $1 + 3/8 = 1.375$.*

We scale the input sizes by 16. The stretched bins in our setting therefore have capacity 22 and the optimal offline algorithm can pack all items into three bins of capacity 16 each. The three bins of our setting are named A , B , and C .

Before the algorithm proper, we list several *good situations* (GS). These are configurations of the three bins which allow us to complete the packing regardless of the following input.

Good Situation 1. *Given a partial packing such that $s(A) + s(B) > 26$, there exists an online algorithm that packs all remaining items into three bins of capacity 22.*

Proof. Since the optimum can pack into three bins of size 16, the total size of items in the instance is at most $3 \cdot 16 = 48$. If two bins have size $s(A) + s(B) \geq 26$, all the remaining items (including the ones already placed on C) have size at most 22. Thus we can pack them all into bin C . \square

Good Situation 2. *Given a partial packing such that $s(A) \in [4, 6]$, there exists an online algorithm that packs all remaining items into three bins of capacity 22.*

Proof. Let A be the bin with size between 4 and 6 and B be one of the other bins (choose arbitrarily). Put all the items greedily on B . When an item does not fit, put it on A , where it fits, as originally A is below 6. Now A and B have at least 4 originally on A and 22 together on B and the new item on A . In total we have $s(A) + s(B) \geq 26$, allowing us to apply GS1. \square

Good Situation 3. *Given a partial packing such that $s(A) \in [15, 22]$ and either (i) $s(C) \leq 6$ or (ii) $s(B) + s(C) \geq 22$, there exists an online algorithm that packs all remaining items into three bins of capacity 22.*

Proof. If $s(B) + s(C) \geq 22$, then $\max(s(B), s(C)) \geq 11$, so we are in GS1 on bins A and B or on bins A and C .

Else, if $s(C) \leq 6$, we pack arriving items into B . If $s(B) \geq 11$, we apply GS1 on bins A and B . Thus we can assume $s(B) < 11$ and we cannot continue packing into B any further. This implies that an item i arrives such that $s(i) > 11$. As $s(C) \leq 6$, we pack i into it and apply GS1 on bins A and C . \square

Good Situation 4. *Given a partial packing such that $s(A) + s(B) \geq 15 + \frac{1}{2}s(C)$, $s(B) < 4$, and $s(C) < 4$, there exists an online algorithm that packs all remaining items into three bins of capacity 22.*

Proof. Any item that fits on A will be placed there. If this brings the size of A to 15 or more, we reach GS3.

Any item of size at least $11 - \frac{1}{2}s(C)$ will be placed on B (where it fits). Then $s(A) + s(B) > 26$ and we reach GS1. with the bins A , and B .

The only case left is where $15 + \frac{1}{2}s(C) - s(B) \leq s(A) < 15$ and one or more items with size in the interval $I := (22 - s(A), 11 - \frac{1}{2}s(C))$ arrive. Note that $I \subseteq (7, 11)$. The items with size in I will be placed on C (where two such items fit together), while all other items are handled as above.

If two such items arrive, a total size of at least $s(A) + 2(22 - s(A)) = 44 - s(A) > 29$ has arrived, so we reach GS1 with the bins A and C .

If only one such item $i \in I$ arrives, we either reach GS1 on A , B or we wait until an item x arrives which is supposed to go into A but cannot fit; we place x on C . The total size of A and C is now $s(A) + s(x) + s(i) \geq 22 + 7$, triggering GS1. \square

Good Situation 5. *Given a partial packing such that an item a with $s(a) > 6$ is packed into bin A , $s(B) \in [3, 6]$, and C is empty, there exists an algorithm that packs all remaining items into three bins of capacity 22.*

Proof. We have $s(B) \in [3, 4)$, otherwise we reach GS2. Pack all incoming items into A using FIRST FIT as long as it possible. If $s(A) \geq 12$, we can apply GS4, and so we assume the contrary. Therefore, $s(A) < 12$ and an item i arrives which cannot be packed into A .

Place i into B . If $s(i) \geq 12$, we can apply GS3. We thus have $s(i) \in (10, 12)$ and $s(A) \geq 22 - s(i) > 10$. Continue with FIRST FIT on bins B , A , and C in this order.

We claim that GS3 is reached at the latest after FIRST FIT has packed two items, x and y , on bins other than B . If one of them (say x) is packed into bin A , this holds because $s(x) + s(B) > 22$ and $s(A) > 10$ already before this assignment – enough for GS1. If both items do not fit in A ,

they are both larger than 10, since $s(A) < 12$. We will show that this cannot happen.

As $s(A) < 12$ from our previous analysis, we note that $s(x), s(y) > 10$. We therefore have three items i, x, y with $s(i), s(x), s(y) > 10$ and an item $s(a) > 6$ from our initial conditions. These four items cannot be packed together by any offline algorithm into three bins of capacity 16, and so we have a contradiction with $s(x), s(y) > 10$. \square

Good Situation 6. *If $s(C) \leq 6 \leq s(B)$ and $s(A) \geq s(B) + 4 - s(C)$, there exists an algorithm that packs all remaining items into three bins of capacity 22.*

Proof. If any bin has load in $[4, 6]$, we are in GS2. Otherwise $s(C) < 4$ and $s(A) \geq s(B) > 6$. Pack all items into A , until an item x does not fit. At this point $s(A) + x > 22$. If x fits on B , we put it there and reach GS1 because $s(B) > 6$. Otherwise, x definitely fits on C because $s(C) < 4$. By the condition on $s(A)$, we have $s(A) + s(C) - 4 + x \geq s(B) + x > 22$, so $s(A) + s(C) + x \geq s(B) + 4 + x > 26$, and we are in GS1 again. \square

Corollary 3.2. *Suppose $s(C) \leq s(B) < 6 < s(A)$. If an item of size at least $s(A) - s(C) + 4 - s(B)$ arrives, we reach GS6 by placing it on B or C . If an item of size at most $s(A) - s(C) - (4 - s(B))$ arrives, it can be placed on C ; we reach a good situation if $s(C) \geq 4$ as a result.*

Good Situation 7. *Suppose $s(C) \leq s(B) < 6 < s(A)$. If $s(A) \leq 9 + \frac{1}{2}(s(C) + s(B))$ and for some item x we have $s(A) + x > 22$, there exists an online algorithm that packs all remaining items into three bins of capacity 22.*

Proof. We have $x > 22 - s(A) \geq 13 - \frac{1}{2}(s(C) + s(B)) \geq s(A) + 4 - s(C) - s(B)$. We reach GS6 by Corollary 3.2. \square

We now proceed to describe the bin packing algorithm itself. The algorithm will often use a special variant of FIRST FIT, described as follows:

Definition 3.3. *Let $\mathcal{L} = (X|_k, Y|_l, \dots)$ be a list of bins X, Y, \dots where each bin X has an associated integral capacity k satisfying $s(X) \leq k$.*

$\text{GSFF}(\mathcal{L})$ (*Good Situation First Fit*) is an online algorithm for bin stretching that works as follows:

Algorithm GSFF(\mathcal{L}): For each item i :

If it is possible to pack i into any bin (including bins not in \mathcal{L} , and using capacities of 22 for all bins) such that a good situation is reached, do so and continue with the algorithm of the relevant good situation.

Otherwise, traverse the list \mathcal{L} in order and pack i into the first bin X such that $X|_k \in \mathcal{L}$ and $s(X) + s(i) \leq k$.

For example, $\text{GSFF}(A|_4, B|_{22})$ checks whether either $(A \cup \{j\}, B, C)$, $(A, B \cup \{j\}, C)$ or $(A, B, C \cup \{j\})$ is a partial packing of any good situation. If this is not the case, the algorithm packs j into bin A provided that $s(A) + s(j) \leq 4$. If $s(A) + s(j) > 4$, the algorithm packs j into bin B with capacity 22. If j cannot be placed into B , $\text{GSFF}(A|_4, B|_{22})$ reports failure and another online algorithm must be applied.

3.1 First phase

In the first phase, we pack items into two bins so that either an item of size 6 arrives relatively early in the input sequence, or we can reach a good situation.

Algorithm FIRST PHASE:

- (1) $\text{GSFF}(A|_4, B|_4)$. Rename the bins so that $s(A) \geq s(B)$.
- (2) Let the item on which Step (1) failed be j . If $s(j) > 6$ place j into A , and go to the SECOND PHASE.
- (3) Place j into B and rename the bins so that $s(A) > s(B) \geq s(C) = 0$.
- (4) $\text{GSFF}(B|_4, A|_q, C|_4)$ where $q := 9 + \frac{1}{2}(s(B) + s(C))$. As soon as C receives its first item, rename the bins B and C so that $s(B) \geq s(C)$.

For the rest of this section, we assume that we have reached Step (4). Thus $s(j) \leq 6$. In fact, assuming that no good situation is reached before Step (4), we observe the following:

Observation 3.4. After Step (3), we have $s(B) \in (3, 4)$, $s(A) \leq s(B) + 4$, and $s(A) \in (6, 8)$.

Proof. If $s(j) \in [4, 6]$, we reach GS2 by assigning j to C . So $s(j) < 4$ and j is assigned to the least loaded bin, which must have load more than 2 already, else we reach GS2 or $\text{GSFF}(A|_4, B|_4)$ does not fail on j . This implies that both A and B received items in Step (1), so $s(A) + s(B) - s(j) > 6$, else a good situation would have been reached before j arrived.

We therefore have $s(A) < s(B) + 4$ and $s(B) > 3$ after renaming the bins. It follows that $s(B) \in (3, 4)$ and $s(A) \in (6, 8)$. \square

Observation 3.5. *In Step (4), as long as C is empty, packing any item of size at least 4 leads to a good Situation. Thus while C is empty, all items that arrive in Step (4) and are not put on B have size in $(6 - s(B), 4)$.*

Proof. Any item with size in $[4 - s(B), 6 - s(B)] \cup [4, 6]$ leads to GS2. Any item with size more than 6 is assigned to A if it fits there, reaching GS5, and else to B or C , reaching GS7 since $s(A) < 9 + \frac{1}{2}(s(B) + s(C))$. The only remaining possible sizes of items that are not packed into B are $(6 - s(B), 4)$. \square

Corollary 3.6. *C receives at least one item in Step (4). Afterwards, $s(B) + s(C) > 6$.* \square

Lemma 3.7. *In Step (4), the algorithm reaches a good situation.*

Proof. The lemma is clear if an item in $[4 - s(B), 6 - s(C)]$ arrives after C gets its first item. (Note that $s(B) - s(C) \in [0, 2)$ in this case.) Items with size less than $4 - s(B)$ are just put on B , maintaining $s(B) \geq s(C)$. We need to show that an item x with $s(x) > 6 - s(C)$ fits on A below q or immediately gives a good situation. For that, we will make use of the interval

$$(L, U) := (s(A) - s(C) - (4 - s(B)), s(A) - s(C) + 4 - s(B)).$$

This is an interval formed by the bounds from Corollary 3.2.

If $s(x) \notin (L, U)$, then x can be placed on C , since $s(C) < 6$, thus we reach GS6 by Corollary 3.2 and because $s(x) > 6 - s(C)$.

If $s(x) \in (L, U)$, then x can be assigned to A , where it fits. This follows because

$$s(A) + U \leq 2 \left(9 + \frac{s(B) + s(C)}{2} \right) - s(C) + (4 - s(B)) = 22.$$

After this assignment, the new load of bin A is at least $s(A) + L = 2s(A) - s(C) - (4 - s(B))$. If this is more than q , then we find $2s(A) - s(C) - (4 - s(B)) > 9 + \frac{1}{2}(s(B) + s(C))$, so $2s(A) > 13 + \frac{3}{2}(s(B) + s(C))$, implying $s(A) > 11$ by Observation 3.6. In this case, we have $2s(A) - s(C) - (4 - s(B)) > 15 - s(B) + \frac{1}{2}s(C)$, reaching GS4. The inequality holds because $2s(A) > 15 + \frac{3}{2}s(C) - 2s(B)$, which follows from $s(A) > 11$ and $s(C) \leq s(B)$. \square

3.2 Second phase

We start the algorithm SECOND PHASE only after Step (2) of FIRST PHASE fails when an item j of size $s(j) > 6$ arrives. We have not entered GS5 before placing j on A , and so we have $s(A \setminus \{j\}) \leq 3$.

Algorithm SECOND PHASE:

- (1) GSFF($A|_q, B|_4$), where $q := 6 + s(j)$.
- (2) If the next item x fits into A , apply GSFF($A|_{22}, B|_{22}, C|_{22}$).
- (3) Else: Place x into B . Let j' be the smallest item of $\{j, x\}$.
- (4) Reorder the bins A and B so that $j' \in A$.
- (5) GSFF($A|_q, B|_{22}$), where $q := 6 + s(j')$.
- (6) Place next item y into C . Let j'' be the smallest item of $\{j', y\}$.
- (7) Reorder the bins A and C so that $j'' \in A$.
- (8) GSFF($A|_q, B|_{22}, C|_{22}$), where $q := 6 + s(j'')$.

We now present the analysis of the algorithm SECOND PHASE.

By choosing q to be $j + 6$ in Step (1), we have ensured that any item i larger than 6 that cannot be placed into A must satisfy $s(i) + s(j) > 16$ and so i cannot be with j in the same bin in the offline optimum packing.

Let us define A_S as the set of items on A of size less than 6 (packed before or after j). We note the following:

Observation 3.8. *During Step (1), if B contains any item, it is true that $s(A_S) + s(B) > 6$. Furthermore, when Step (1) fails, it must fail on an item x , $s(x) > 6$.*

Proof. We first pack items greedily into A , and so if there is no space for an item i in A during Step (1), the load of A would be over $6 + s(j)$, which implies $s(A_S) + s(i) > 6$. Notice that at this point, if i can be packed into B (as we assume), $s(A) + s(B) > 12$.

As for the second part of the observation, assume $s(x) \in [0, 6]$. GS2 excludes the range $[4, 6]$.

If it were true that $s(x) < 3$, then $s(B) > 3$ (x does not fit into B under 6) and we were in GS5 even before x arrived, because $s(j) > 6$. Therefore, $s(x) \in [3, 4)$. But in this case, we have $s(A) + s(B) > 12$, and packing x into A (over q) triggers GS3. This packing is possible because $s(A) < 15$ (if not, we were already in GS3). \square

We now consider two cases based on x :

Case 1: Item x fits into bin A ; we pack it there.

We first note that $s(A)+s(B) < 15$, else we are in GS4. This inequality also implies that $s(B) = 0$, otherwise we have $s(A)+s(B) = s(x)+s(A_S)+s(j) + s(B) > 18$ via Observation 3.8.

We continue with Step (2) until the end of Case 1. Consider the first two items w, y that do not fit into A . The item w can fit into B and will be placed there. Hence, the items w, y have to satisfy $s(w), s(y) > 7$, because we would be in GS3 already before the arrival of y otherwise. On the other hand, y cannot fit into B , or we would have $s(A) + s(B) \geq s(j) + s(x) + s(w) + s(y) > 26$, enough for GS1.

Hence, both B and C now contain a single item. The fact that $s(w) + s(y) > 22$ gives us that either $s(w) > 11$ or $s(y) > 11$. Without loss of generality, assume that it is w .

Let us consider a third item z that does not fit into A . At this point, if $s(A) > 15$, we would be in GS1 with bins A, B , and so $s(z) > 7$. Therefore, packing z into any bin would grant us GS1, and the algorithm only fails if z cannot fit anywhere. Similarly to w , we learn that as $s(z) + s(y) > 22$, one of the items y, z is over 11. However, such a configuration can never arrive: we have five items over 6 (j, x, w, y, z) and two of those would be over 11. There is no way to pack those five items into three bins of capacity 16.

Case 2: Item x does not fit into bin A . Note that as $s(A_S) < 6$, we get $s(j) + s(x) > 16$. Item x is placed on B ; before this assignment, $s(B) < 4$. Hence, after the renaming in Step (4), indeed $s(A) < 6 + \min(s(x), s(j)) = 6 + s(j')$.

Assume that Step (5) fails on an item y . First let us disprove that $s(y) < 7$. If $s(y) < 7$, then $s(B) > 15$. If $s(A) < 15$, we pack y into A and note that $s(C) = 0 \leq 6$; when $s(A) \geq 15$, we pack y into C and get $s(B) + s(C) = s(B) + s(y) > 22$. Either way, we end up in GS3. Hence, our assumption on $s(y)$ was false, we must have $s(y) > 7$.

If y fits on A , we reach GS1 as $s(B) + s(y) + s(A) > 22 + 6$. Otherwise, we have no choice but to pack y into C in Step (6). We see now that no two items of $\{y, x, j\}$ can be packed together in the offline optimum, since y does not fit with j' on bin A ; this will be useful for our final contradiction.

We reorder the bins in Step (7) again; we may also need to recalculate q . What is important for us is that the smallest item of $\{j, x, y\}$, denoted as j'' , is in A and that $s(A) \leq q = s(j'') + 6$. (Note that C was empty before it got y .)

We resume the analysis when Step (8) fails with an item z . Not reaching GS1 implies $s(A) + s(B) < 26$, and so $\min(s(A), s(B)) < 13$, which in turn gives us $s(z) > 9$.

If z fits into A , we can reach GS3 using $s(C) + s(B) > 22$; the last inequality holds because the large item that arrived later in the input sequence did not fit with the large item that arrived earlier. So z does not fit into A , and as $A_S < 6$, we have that $z + j'' > 16$.

The item z not fitting into A is all we need; the large item $j'' \in A$ was the smallest in the set $\{j, x, y\}$, which means that $\min\{s(j), s(x), s(y)\} + s(z) > 16$.

The offline optimum has no chance of packing z with any $\{j, x, y\}$ together into bins of capacity 16, and we have already proved that no two items of $\{j, x, y\}$ can be packed together either. This is a contradiction with the guarantee of ONLINE BIN STRETCHING.

We have shown that all items are packed in the second phase, which concludes the proof of Theorem 3.1.

Conclusions. With our algorithm for $m = 3$, the remaining gap is small. For arbitrary m , we feel that significantly new approach would be needed for an algorithm with a better stretching factor than $3/2$. It is quite surprising that there are no lower bounds for $m > 3$ larger than the easy bound of $4/3$.

Acknowledgment. The authors thank Emese Bittner for useful discussions during her visit of Charles University.

References

1. S. Albers and M. Hellwig. Semi-online scheduling revisited. *Theor. Comput. Sci.*, 443:1–9, 2012.
2. J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line load balancing with applications to machine scheduling and virtual circuit routing. *J. ACM*, 44:486–504, 1997.
3. Y. Azar and O. Regev. On-line bin-stretching. In *Randomization and Approximation Techniques in Computer Science*, pages 71–81. Springer, 1998.
4. P. Berman, M. Charikar, and M. Karpinski. On-line load balancing for related machines. *J. Algorithms*, 35:108–121, 2000.
5. E. Coffman Jr., J. Csirik, G. Galambos, S. Martello, and D. Vigo. Bin Packing Approximation Algorithms: Survey and Classification, In P. M. Pardalos, D.-Z. Du, and R. L. Graham, editors, *Handbook of Combinatorial Optimization*, pages 455–531. Springer New York, 2013.
6. T. Ebenlendr, W. Jawor, and J. Sgall. Preemptive online scheduling: Optimal algorithms for all speeds. *Algorithmica*, 53:504–522, 2009.
7. M. Gabay, N. Brauner, V. Kotov. Computing lower bounds for semi-online optimization problems: Application to the bin stretching problem. HAL preprint hal-00921663, 2013.
8. M. Gabay, V. Kotov, N. Brauner. Semi-online bin stretching with bunch techniques. HAL preprint hal-00869858, 2013.

9. R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM J. Appl. Math.*, 17:263–269, 1969.
10. D. Johnson. *Near-optimal Bin Packing Algorithms*. Massachusetts Institute of Technology, project MAC. Massachusetts Institute of Technology, 1973.
11. H. Kellerer and V. Kotov. An efficient algorithm for bin stretching. *Operations Research Letters*, 41(4):343–346, 2013.
12. H. Kellerer, V. Kotov, M. G. Speranza, and Z. Tuza. Semi on-line algorithms for the partition problem. *Oper. Res. Lett.*, 21:235–242, 1997.
13. J. Ullman. The Performance of a Memory Allocation Algorithm. *Technical Report 100*, 1971.