# Resource augmented semi-online bounded space bin packing

Leah Epstein [*]
Department of Mathematics, University of Haifa, 31905 Haifa, Israel.

Elena Kleiman [†]
Department of Mathematics, University of Haifa, 31905 Haifa, Israel.

**Abstract**

We study on-line bounded space bin-packing in the resource augmentation model of competitive analysis. In this model, the on-line bounded space packing algorithm has to pack a list $L$ of items with sizes in $(0, 1]$, into a minimum number of bins of size $b$, $b \geq 1$. A bounded space algorithm has the property that it only has a constant number of active bins available to accept items at any point during processing. The performance of the algorithm is measured by comparing the produced packing with an optimal offline packing of the list $L$ into bins of size 1. The competitive ratio then becomes a function of the on-line bin size $b$. Csirik and Woeginger studied this problem in [3] and proved that no on-line bounded space algorithm can perform better than a certain bound $\rho(b)$ in the worst case. We relax the on-line condition by allowing a complete repacking within the active bins, and show that the same lower bound holds for this problem as well, and repacking may only allow to obtain the exact best possible competitive ratio of $\rho(b)$ having constant number of active bins, instead of achieving this bound in the limit. We design a polynomial time on-line algorithm that uses three active bins and achieves the *exact* best possible competitive ratio $\rho(b)$ for the given problem.

## 1 Introduction

**Problem definition.** *Bin-Packing* (BP) is one of the basic problems in theoretical computer science and combinatorial optimization. It was first introduced and investigated by Ullman in [25]. In the classical one-dimensional problem we are given a finite list $L = \{a_1,a_2,\ldots,a_n\}$ of $n$ elements, called *items*, each element $a_i$ has a fixed size in (0,1]. In a slight abuse of notation, we use $a_i$ to indicate both the $i$-th element and its size. We have a potentially infinite supply of unit-capacity containers, called *bins*. The problem consists of assigning (packing) each item to an unique bin such that the sum of sizes of elements in a bin does not exceed the bin capacity and such that the total number of bins used is as small as possible. Research of bin-packing and its many variants was motivated by the fact this abstract problem models a large variety of real world problems, such as cutting stock problems (cutting pieces of variable sizes from standard paper sheets, from standard textile cloth measures, etc.), machine scheduling problems (minimizing the number of machines necessary for completing all tasks by a given deadline) and storage allocation problems (allocating spaces on a disc or in a computer memory). The problem is known to be NP-hard [12], thus research has concentrated on the study and development of *approximation* algorithms, that is, algorithms which do not guarantee to find an optimal solution for every instance, but attempt to find *near-optimal* packings in polynomial time. A particular class consists of the so-called on-line algorithms.

A bin-packing algorithm is called *on-line*, if it is given the items from $L$ one at a time, and must assign each item $a_i$ to a bin immediately upon arrival. The assignment must be based solely on its size and the packing of the previous items $a_1,a_2,\ldots,a_{i-1}$, without having any information on neither the sizes of the subsequent items, nor their number. The decisions of the algorithm are irrevocable; An item assigned to a bin must not be repacked during the execution of the algorithm at later times. Since on-line algorithms do not need to know the

[*]Email: lea@math.haifa.ac.il.

[†]Email: elena.kleiman@gmail.com. This work was submitted as the M.Sc. thesis of the second author.

future, they also work in real-time environments where decisions must be made very fast, without delay, and without complete information. We usually do not put any restrictions on the computational complexity of on-line algorithms, although efficient algorithms are preferred.

As opposed to on-line algorithms, *offline* algorithms for bin-packing have complete knowledge about the list of items and can thus apply more advanced strategies for packing.

As an intermediate class between these two classes, one can define semi-on-line algorithms. In contrast to pure on-line algorithms, these algorithms slightly relax the on-line restriction by allowing to execute certain types of additional operations in each step, as repacking some finite number of already packed items [9, 10, 16, 17], preprocessing the items by ordering them with regard to sizes [7] or buffering some items before packing them [14, 15]. Each bin becomes active (or *open*), when it receives its first item, but once it is declared inactive (or *closed*), the algorithm no longer considers it for packing and it can never become active again. A bin is *empty* if no item is assigned to it. Some on-line algorithms keep all bins into which items have been placed open, while others allow only a restricted finite number of bins available to accept items at any point during processing. These latter algorithms are the so-called *bounded space* algorithms. An on-line bin-packing algorithm is said to use $k$*-bounded space* if, for each new item $a_i$, the number of active bins in which it may be packed is at most $k$ ($k \geq 1$). The bounded space restriction is quite natural, especially so in on-line bin-packing. It models situations in which bins are exported once they are packed. Thus, the bounded space restriction guarantees the constant flow of the output bins, and that the packer does not accumulate an enormous amount of bins which are only given as output at the end of processing.

These latter restrictions (on-line and bounded space) arise in many applications, as in packing trucks at a loading dock that has positions for a limited number of trucks or in communicating via channels with bounded buffer size in which information moves in fixed-size blocks that are filled with smaller packets with various sizes.

In this paper, we study algorithms that are on-line, bounded space, and allow full repacking within the current $k$ active bins. That means, that in addition to the standard actions of bounded space bin-packing, where we are allowed to: (1) Open a new bin (if the number of active bins is less or equal to $k-1$),
(2) Close some active bin (and never open it again),
(3) Pack a new item into some active bin (if the contents of the bin remains below one).

We are also allowed to: (4) Repack the set of active bins as the new item arrives, using the information on the size of the newly arrived item, i.e. if $B_1, ..., B_k$ denote the active bins (we identify the contents of a bin with the bin), to form a new partition $B_1^{'}, ..., B_k^{'}$ of the items inside the active bins such that $\bigcup B_i = \bigcup B_i^{'}$ holds, and such that the items in each part of the new partition have overall size less than or equal to 1.

To allow action (4) is a natural assumption. As long as an item is in an active bin, the item is available for the packer to change its position. In the loading dock example, trucks will be partially repacked and items will be moved from one truck at the loading dock to another in order to increase the number of items packed.

**Performance evaluation of on-line bin packing algorithms.** Since it is impossible (in general) to produce the best possible solution when computation occurs on-line, we consider approximation algorithms. Approximation algorithms have been analyzed from different points of view. In this paper we restrict ourselves to worst-case analysis. When we discuss the performance of on-line algorithms, we use the term *competitive* instead of *approximation* which is used for offline algorithms. The quality of on-line algorithms is usually evaluated using competitive analysis. Competitive analysis tries to find the maximum 'distance' between the optimal packing and the packing constructed by the considered algorithm. In the case of bin-packing, the standard metric for the worst-case performance is the asymptotic worst-case competitive ratio, or simply *asymptotic competitive ratio (ACR)*. In particular, we want to find an algorithm that incurs cost within a constant factor of the minimum possible cost (which is denoted by OPT) no matter what the input list is. This constant factor is the ACR.

We define the asymptotic competitive ratio more formally. For a list $L = \{a_1, a_2, \ldots, a_n\}$ of items with sizes in $(0, 1]$ and an on-line algorithm $A$: If $A(L)$ denotes the number of unit-capacity bins used by algorithm $A$ to pack the input-list $L$, and $OPT(L)$ denotes the number of bins used in an optimal packing, then the ACR of

$A$, denoted by $R_A^\infty$, is given by: $R_A^\infty := \limsup_{k\to\infty}\left\{\sup_L\left\{\frac{A(L)}{k}\,\middle|\, OPT(L) = k\right\}\right\}$. A more instructive sufficient condition is the following definition, which states that $R_A^\infty$ is the smallest constant such that there exists a constant $0 \le K < \infty$ for which $A(L) \le R_A^\infty \cdot OPT(L) + K$ for every list $L$; the asymptotic ratio, a multiplicative constant, hides the additive constant $K$. This ratio is of most interest in those applications where $K$ is small relative to $A(L)$. It is apparent that the smaller the value $R_A^\infty$ is, the better the heuristic algorithm $A$ performs in terms of the worst-case scenario. In other words, the smaller the $R_A^\infty$'s value is, the closer the heuristic solution is to the optimal one. Hence, we want to minimize $R_A^\infty$ as much as possible when we design a heuristic algorithm. Experience shows that the ACR is the more reasonable measure of performance for a quality of a bin-packing algorithm as it is robust against anomalies with a small number of bins in the optimum packing, and it also allows the packing algorithm more freedom while packing the first few bins.
In this paper we will often drop the term "asymptotic" when we mention the asymptotic competitive ratio.
**Weighting functions.** In our paper we use the *weighting functions* technique, which is a major tool in the analysis of algorithms for bin packing. This technique was introduced in [11, 19, 25] and subsequently applied in many other papers (see, for example, [8, 18, 21, 27]). The idea of such weighting functions is simple. An item is assigned a weight according to its size and its packing in some fixed solution. The weights are assigned in a way that the cost of an algorithm is close to the total sum of weights. The weight of a bin is the weight of all items in it. In order to complete the analysis, it is usually necessary to consider the total weight that can be packed into a single bin of an optimal solution. But, as there is no systematic way to find weighting functions, the main difficulty in using the approach is finding the appropriate weighting function.
**Resource augmentation.** This technique for analyzing on-line algorithms was introduced in 1995 by Kalyanasundaram and Pruhs in [20]. The resource augmentation model was introduced due to the following drawback of standard competitive analysis. Competitive analysis compares the performance of an on-line algorithm, which must pack each item upon arrival, to that of the omniscient and all-powerful optimal offline algorithm that gets the entire input as a set. The main idea behind the resource augmentation technique is to give the on-line algorithm additional power so it would have fairer chance in competing against the powerful offline adversary advantage, by giving him better resources than the optimal offline algorithm to which it is compared. In bin-packing applications, the extra resource we give the online algorithm is additional bin space. In order to illustrate this idea for packing applications, we use again the loading dock example: as we need to pack the trucks online, and know that most probably the packing will not be optimal, we can consider taking a larger truck, hoping that the extra space would cover up for the waste, and allow us to shift more items. The motivation is to preclude pathological examples that may drive the worst-case competitive ratio; with resource augmentation we derive improved, more realistic and meaningful competitive ratios. During the last few years the resource augmentation technique has become a very popular tool, and it has been applied to many problems in scheduling (see [4],[22, 2]), in paging (see [1], [5]), and in combinatorial optimization(see [20]). Let $L = \{a_1, a_2, \ldots, a_n\}$ be a list of items in $(0, 1]$. The offline optimum $OPT_1(L)$ is the minimum number of unit-sized bins into which the items in $L$ can fit. We investigate the behavior of on-line bounded space bin-packing algorithms that pack the list $L$ into bins of size $b \ge 1$. This larger bin size $b$ is the augmented resource of the on-line algorithm; the offline algorithm has to work with bins of size 1. For an on-line algorithm $A$ and a bin size $b$, we denote by $A_b(L)$ the number of bins of size $b$ that algorithm $A$ uses in packing the items in $L$. The asymptotic worst-case competitive ratio of algorithm $A$ for bin size $b$, denoted by $R_A^\infty(b)$, is defined as $R_A^\infty(b) := \limsup_{k\to\infty}\left\{\sup_L\left\{\frac{A_b(L)}{k}\,\middle|\, OPT_1(L) = k\right\}\right\}$. The competitive ratio then becomes a function of the on-line bin size $b$.
**Preliminaries.** In this section we define the sequence $\{t_i^b\}_{i=1}^\infty$, that was originally introduced by Csirik and Woeginger in [3], and will be essential in the definition and in the analysis of our algorithm.

Given $b \ge 1$, we associate with it an infinite sequence $T(b) = \{t_1^b, t_2^b, \ldots\}$ of positive integers, defined as follows:

$$t_1^b = \lfloor 1 + b \rfloor \qquad and \qquad r_1^b = \frac{1}{b} - \frac{1}{t_1^b}, \tag{1}$$

$$t^b_{i+1} = \left\lfloor 1 + \frac{1}{r^b_i} \right\rfloor \qquad and \qquad r^b_{i+1} = r^b_i - \frac{1}{t^b_{i+1}}, \quad i = 1, 2, \ldots. \tag{2}$$

The intuition behind $T(b)$ is to find a sequence of positive integers, such that the next integer at each point is picked greedily to be minimal, and the sum of their reciprocals is less than $\frac{1}{b}$. In this interpretation, the value $r^b_i$ represents the difference between $\frac{1}{b}$ and the sum accumulated so far, after adding the reciprocal value of the integer $t^b_i$ to the sum. This means that $r^b_i > 0$, and

$$r^b_i = \frac{1}{b} - \sum_{j=1}^{i} \frac{1}{t^b_j}. \tag{3}$$

Note that the next inequality directly follows from the definitions in (1) and (2):

$$r^b_{i-1} \le \frac{1}{t^b_i - 1}. \tag{4}$$

We now mention several facts on the sequence $T(b)$ that will be used later. First, we observe that for every $b \ge 1$ the corresponding sequence $T(b) = \{t^b_1, t^b_2, \ldots\}$ is growing rapidly.

**Lemma 1.** *For every $b \ge 1$ :*
*(i) The values of $t^b_i$ are strictly increasing as a function of $i$.*
*(ii) The values of $t^b_i$ satisfy $t^b_i > b$.*

**Lemma 2.** *For every $b \ge 1$ the elements of the sequence $\{t^b_1, t^b_2, \ \ldots\}$ satisfy:*

$$t^b_{i+1} \ge t^b_i(t^b_i - 1) + 1 \qquad for\ all\ i \ge 1. \tag{5}$$

Csirik and Woeginger [3] used this sequence to define the function

$$\rho(b) = \sum_{i=1}^{\infty} \frac{1}{t^b_i - 1}. \tag{6}$$

$\rho(b)$ is a strictly decreasing function of $b$ (see Figure 1 in [3]). Note that $\rho(1) = h_\infty \approx 1.69103$.

**Lemma 3.** *The infinite sum in the righthand side of (6) converges for every value of $b \ge 1$.*

**Previous work.** The bin-packing problem holds a special place, both in the history of approximation algorithms and in the history of on-line problems and has been studied extensively since the early 1970's. Various heuristic algorithms with guaranteed bounds on their performance were proposed for the classical problem. The demands for on-line and bounded space algorithms arise in a wide variety of real-world applications, consequently, the problem was analyzed thoroughly in the 1980's and the 1990's. Among bounded space algorithms, those of *harmonic-type* play an important role. First such algorithm was formulated by Lee and Lee in [21]. Their sequences of algorithms Harmonic(k) are based on a special nonuniform partition of the interval $(0, 1]$ into $k$ subintervals. To each of these subintervals there corresponds a single active bin and only items belonging to this subinterval are packed into this bin. If some item does not fit into its assigned bin, this bin is closed and a new bin is used. In [27] Woeginger presents a sequence of algorithms Simplified Harmonic(k), that work very similarly to the Harmonic(k) algorithms, but use more complicated partition of the interval $(0, 1]$, one based on a Golomb sequences studied in [13]. (Note that it is the same sequence introduced in the above section, for the case $b = 1$). The asymptotic worst-case ratios of these algorithms approach $h_\infty$ as the number of active bins tends to infinity; but none of the known bounded space algorithms reaches this bound while using a finite number of active bins. On the negative side, Lee and Lee proved in [21] that no on-line approximation algorithm can have a competitive ratio less than the constant $h_\infty \approx 1.69103$ using bounded space. The best on-line algorithm so far was developed by Seiden and has ACR 1.58889 [24]. This algorithm, called Harmonic++ belongs to the class of *Super Harmonic Algorithms*, defined in [24]. The best on-line algorithms [21, 23] known prior to it, belong to this class as well. The best known lower bound 1.5401 for the ACR of on-line algorithms has been given by van Vliet [26], while the lower bound of any Super Harmonic type algorithm is 1.58333 [23]. From this lower bound we can conclude that in order to get a solution which is very close to optimal, the algorithm cannot be online in the usual sense, and we should consider a semi-online model which allows a small amount of modifications to the solution produced by the algorithm (and thus partly lose its online quality). The first semi-online algorithm was given by Galambos [7] for the bounded space version of the classical bin-packing problem where only a bounded number of bins are open while packing. This algorithm uses two "buffer-bins" for temporary storing of items. The idea was further developed by Galambos and Woeginger in [9], where they present an on-line algorithm REP$_3$ and demonstrate that the bound $h_\infty$ can be reached with three active bins, if the algorithm is allowed to repack the items within the three active bins (i.e. to move items from one active bin to another).

Bin-packing with resource augmentation was first studied by Csirik and Woeginger in [3]. They gave on-line bounded space bin-packing algorithms for every $b \geq 1$, whose worst case ratio in this model comes arbitrary close to the $\rho(b)$ bound. Moreover, they proved that for every $b \geq 1$ no on-line bounded space algorithm can perform better than $\rho(b)$ in the worst case, thus showing that the optimal asymptotic competitive ratio for the on-line bounded space algorithms with resource augmentation $b$ is a strictly decreasing function $\rho(b)$ of $b$. Unbounded space resource augmented bin-packing was studied in [6].

**Our results and organization of the paper.** In this paper we study the on-line bounded space bin-packing problem with limited repacking allowed, in the resource augmentation model of competitive analysis. Unfortunately, the $\rho(b)$ lower bound of Csirik and Woeginger carries over to this problem, too, as we prove in Section 4. We extend the ideas in [9] for the resource augmented environment; We design, for every $b \geq 1$, an on-line algorithm for resource augmented bounded space bin-packing problem called RESOURCE AUGMENTED REP$_3$(b) (or shortly RAR$_3$(b)), which is allowed to repack a constant number of active bins (three active bins to be precise) and has exact worst-case competitive ratio of $\rho(b)$, and so show that in a resource augmented environment, allowing the repacking of finite number of bins, allows us to reach the exact optimal worst-case ratio instead of achieving it in the limit.

The main tool we apply is weighting functions technique, which we introduced above. The weighting function we use in a generalized version of the one used in [27]. In order to adapt it to fit our purposes, we use an alternative definition to the resource augmented bin-packing problem; We compare an on-line algorithm which uses bins of size 1 to an optimal offline algorithm whose bins are of size $\frac{1}{b}$. We assume that all item sizes are bounded by $\frac{1}{b}$ (i.e. the sizes of the items are scaled into the interval $(0, \frac{1}{b}]$). This definition is equivalent to the one mentioned above.

The paper is organized as follows. In Section 2 we present the weighting function we use and define our algorithm RAR$_3$(b), Section 3 is dedicated to a proof of its correctness. In Section 4 we analyze its worst-case asymptotic behavior.

## 2 The algorithm RAR$_3$(b)

**Classification of the items.** Recall that all the items have sizes in the interval $(0, \frac{1}{b}]$. We classify the items according to the following partition of the interval $(0, \frac{1}{b}]$. The partition of $(0, \frac{1}{b}]$ changes according to different values of $b$.

For $b \in [1, 1.2)$ the partition of $\left(0\,, \frac{1}{b}\right]$ is as follows: $B_1 = \left(\frac{1}{t_1^b}, \frac{1}{b}\right]$, and for $i \geq 2$, $B_i = \left(\frac{1}{t_i^b}, \frac{1}{t_i^b - 1}\right]$, $C_i = \left(\frac{1}{t_i^b + 1}, \frac{1}{t_i^b}\right]$ and $D_i = \left(\frac{1}{t_{i+1}^b - 1}, \frac{1}{t_i^b + 1}\right]$. Note that in this case, by the definitions of the sequence in (2), $t_1^b = 2$, $t_2^b = 3$ and $t_3^b \geq 7$. Thus, the results from [9] for $b = 1$ hold, with few modifications, for this case as well, and we only need to consider the case $b \geq 1.2$.

For $b \geq 1.2$ the partition of $(0\,, \frac{1}{b}]$ is as follows: $B_1 = \left(\frac{1}{t_1^b}, \frac{1}{b}\right]$, for $i \geq 2$, $B_i = \left(\frac{1}{t_i^b}, \frac{1}{t_i^b - 1}\right]$ and for $i \geq 1$,

$C_i = \left(\frac{1}{t_i^b+1}, \frac{1}{t_i^b}\right]$ and $D_i = \left(\frac{1}{t_{i+1}^b-1}, \frac{1}{t_i^b+1}\right]$.

From definition of the sequence $t_i^b$ and by inequality (5), we can see that the above partition of the interval $(0, \frac{1}{b}]$ is well defined for any $b \geq 1$. The B$_i$, C$_i$ and D$_i$-type intervals do not overlap, and cover the entire interval.

**Definition of the weights.** Let us define our weighting function $W(x) : (0, \frac{1}{b}] \rightarrow \mathbb{R}^+$.

$$W(x) = \begin{cases} x + \frac{1}{t_1^b(t_1^b-1)} & \text{for} \quad \frac{1}{t_1^b} < x \leq \frac{1}{b} \quad \text{and} \quad i = 1 \quad (t_1^b - 1 \leq b < t_1^b) \\ x + \frac{1}{t_i^b(t_i^b-1)} & \text{for} \quad \frac{1}{t_i^b} < x \leq \frac{1}{t_i^b-1} \quad \text{and} \quad i \geq 2 \\ \frac{t_i^b+1}{t_i^b} \cdot x & \text{for} \quad \frac{1}{t_{i+1}^b-1} < x \leq \frac{1}{t_i^b} \quad \text{and} \quad i \geq 1 \end{cases}$$

This weighting function is similar to the weighting function in [27].
Itis not difficult to prove the following properties of our weighting function.

**Observation 4.** *For any $b \geq 1$:*

*(i) $W(x)$ is nondecreasing in $(0, \frac{1}{b}]$.*

*(ii) For $i \geq 1$ and $x \leq \frac{1}{t_i^b}$, $\frac{W(x)}{x} \leq \frac{t_i^b+1}{t_i^b}$ holds.*

*(iii) For $i \geq 1$ and $x > \frac{1}{t_{i+1}^b-1}$, $\frac{W(x)}{x} \geq \frac{t_i^b+1}{t_i^b}$ holds.*

**Lemma 5.** *For any $b \geq 1$: given an item of size $x \leq \frac{1}{t_2^b-1}$, let $\ell$ be an integer such that $x \in \left(\frac{1}{\ell+1}, \frac{1}{\ell}\right]$. Then, for any item of size $y > \frac{1}{\ell+1}$, $\frac{W(y)}{y} \geq \frac{\ell+2}{\ell+1}$ holds.*

**Proof.** Item $x$ can be a B$_i$-, C$_i$- or D$_i$- item. We discuss these cases separately.
If $x \in$ B$_j$, for some $j \geq 2$, then $\ell = t_j^b - 1 \Rightarrow t_j^b = \ell + 1$. For any $y > \frac{1}{t_j^b-1}$, by Observation 4(iii), $\frac{W(y)}{y} \geq \frac{t_{j-1}^b+1}{t_{j-1}^b} = 1 + \frac{1}{t_{j-1}^b} > 1 + \frac{1}{t_j^b} = \frac{t_j^b+1}{t_j^b} = \frac{\ell+2}{\ell+1}$, as $t_{j-1}^b < t_j^b$ for any $j \geq 2$. For any $y \in$ B$_j$, again by Observation 4(iii), $\frac{W(y)}{y} \geq \frac{t_j^b+1}{t_j^b} = \frac{\ell+2}{\ell+1}$.
If $x \in$ C$_j$, for some $j \geq 2$, then $\ell = t_j^b$. If $y > \frac{1}{t_j^b-1}$, by Observation 4(iii), $\frac{W(y)}{y} \geq \frac{t_{j-1}^b+1}{t_{j-1}^b} = 1 + \frac{1}{t_{j-1}^b} > 1 + \frac{1}{t_j^b} > 1 + \frac{1}{t_j^b+1} = \frac{t_j^b+2}{t_j^b+1} = \frac{\ell+2}{\ell+1}$, as $t_{j-1}^b < t_j^b$ for any $j \geq 2$.
If $y \in$ B$_j \cup$C$_j$, again by Observation 4(iii), $\frac{W(y)}{y} \geq \frac{t_j^b+1}{t_j^b} = 1 + \frac{1}{t_j^b} > 1 + \frac{1}{t_j^b+1} = \frac{t_j^b+2}{t_j^b+1} = \frac{\ell+2}{\ell+1}$, as $t_j^b > 0$ for any $j \geq 2$.
If $x \in$ D$_j$, for some $j \geq 2$, then $\ell \geq t_j^b + 1 \Rightarrow \ell + 1 \geq t_j^b + 2$ and $\ell + 1 \leq t_{j+1}^b - 1$. From this, we get $\frac{\ell+2}{\ell+1} = 1 + \frac{1}{\ell+1} \leq 1 + \frac{1}{t_j^b+2}$.
If $y > \frac{1}{\ell+1} \geq \frac{1}{t_{j+1}^b-1}$, by Observation 4(iii) $\frac{W(y)}{y} \geq \frac{t_j^b+1}{t_j^b} = 1 + \frac{1}{t_j^b} > 1 + \frac{1}{t_j^b+2} \geq \frac{\ell+2}{\ell+1}$, as $t_j^b > 0$ for any $j \geq 2$.
□

**The resource augmented repacking algorithm.** In this section we define the RESOURCE AUGMENTED REP$_3$(b) algorithm (RAR$_3$(b) for short), for $b \geq 1$. Our algorithm is a generalization of the algorithm REP$_3$ introduced in [9] for $b = 1$. It uses a well known *First-Fit Decreasing* heuristic (FFD) with a small modification. Given a list of items, the original FFD first sorts the items in non-increasing order according to sizes, and then applies *First-Fit* (FF) algorithm that goes through the sorted list and places each item in turn into the lowest indexed bin where it fits. A new bin is opened only in the case an item does not fit into any non-empty bin. Our algorithm always keeps three active bins that we call BIN$_1$, BIN$_2$ and BIN$_3$. RAR$_3$(b) proceeds as follows:

(1) Get a new item $x$ and put $x$ into an empty active bin.

(2) Remove all items from the three active bins.

(3) Sort the items by a non-increasing order of their sizes.

(4) Scan the sorted list.

  (4.1) If $x$ is a $B_i$-item for some $i \geq 1$ and there is a set of $(t_i^b - 1)$ $B_i$-items (including $x$), remove these items from the list and pack them using one active bin.

  (4.2) If $x$ is a $C_i$-item for some $i \geq 1$ and there is a set of $t_i^b$ $C_i$-items (including $x$), remove these items from the list and pack them using one active bin.

  (4.3) Apply FF on the items in the list.

(5) Compute the weight of each active bin, close any bins having a total weight at least 1 and open new bins instead of them. Go to Step (1).

## 3 Proof of correctness

The crucial step of the algorithm RAR$_3$(b) is Step (4). The main part of this section is devoted to establishing the fact that Step (4) is always possible, and that after Step (4) either at least one bin is empty or at least one bin has weight greater or equal to one, thus being closed after Step (5) and an empty bin is open instead of it. Thus, Step (1) is well defined. To do this, we prove the following theorems.
Similarly to [9], we call a *good packing* to a packing with an empty bin, and a *good subset* of items is a subset of weight greater or equal one and of size at most one. The following holds:

**Lemma 6.** *For any $b \geq 1$ and $i \geq 1$:*

*(i) $(t_i^b - 1)$ $B_i$-items compose a good subset.*

*(ii) $t_i^b$ $C_i$-items compose a good subset.*

**Theorem 7.** *Let BIN$_1$, BIN$_2$ and BIN$_3$ be three active bins. Then we can either repack the bins by FFD to produce a packing with an empty bin or we can find a subset of items with weight at least one and size at most one. Such a subset is either the contents of a bin in the FFD-packing, or found in Step (4.1) or Step (4.2).*

**Proof.** We assume that it is neither possible to produce a good packing (a packing with an empty bin), nor to find a good subset of items (a subset of weight greater or equal one and of size at most one), and we derive a contradiction from this. Since we assumed that there neither exists a good packing nor a good subset, in the produced FFD-packing neither BIN$_3$ will be empty, nor will there be a bin with weight greater or equal one.
We will prove a number of combinatorial properties of the FFD-packing. Our purpose is to show that by assuming the above, we get that BIN$_3$ can not possibly contain any of the $B_i$-, $C_i$- or $D_i$- items, and thus FFD-packing is a good packing as it leaves an empty bin.

First, we consider the case $b \geq 1.2$, and the corresponding partition of $(0, \frac{1}{b}]$. The proof is split into several claims.
**Claim 1a.**

(i) *For $b \in [1.2, 2)$ : In the FFD-packing, no bin contains a $B_1$-item $x$.*

(ii) *For other values of b $(b \geq 2)$: In the FFD-packing, neither BIN$_2$ nor BIN$_3$ contains any $B_1$-item $x$.*

**Proof.** (i) If such an item $x$ exists, then $W(x) = x + \frac{1}{t_1^b(t_1^b-1)} > \frac{1}{2} + \frac{1}{2\cdot 1} = 1$ since $t_1^b = 2$. So every $B_1$-item has weight of at least 1, and any subset of items containing this item would form a good subset. We derive a contradiction.
(ii) Assume by contradiction that a $B_1$-item $x$ was put by the packing into BIN$_2$ or BIN$_3$. Since $x$ was not put into BIN$_1$ and we apply FFD, BIN$_1$ must contain $\lfloor b \rfloor = t_1^b - 1$ $B_1$-items, which form a good subset by Lemma 6(i) for $i = 1$. Again, the contents of BIN$_1$ will form a good subset, in contradiction to our assumption. $\square$
**Claim 2a.** *For any value of $b \geq 1.2$: In the FFD-packing, neither BIN$_2$ nor BIN$_3$ contains any $D_i$-item $x$, $i \geq 1$.*
**Proof.** Assume the opposite: that either BIN$_2$ or BIN$_3$ contains a $D_i$-item $x$. Since $x$ was not put into BIN$_1$, BIN$_1$ is at least $\frac{t_i^b}{t_i^b+1}$ full with items of size greater than $\frac{1}{t_{i+1}^b-1}$ (since $x \in (\frac{1}{t_{i+1}^b-1}, \frac{1}{t_i^b+1}]$, and we apply FFD). By Observation 4(iii), for $y > \frac{1}{t_{i+1}^b-1}$, $\frac{W(y)}{y} \geq \frac{t_i^b+1}{t_i^b}$ holds. But now, the total weight of BIN$_1$ is at least: $\frac{t_i^b}{t_i^b+1} \cdot \frac{t_i^b+1}{t_i^b} = 1$, and the contents of BIN$_1$ would be a good subset, in contradiction to the assumption. $\square$
**Claim 3a.**
$\diamond$ *For $b \geq 3$ ($t_1^b \geq 4$):*
*In the FFD-packing, neither BIN$_2$ nor BIN$_3$ contains any $B_i$-item $x$, $i \geq 2$.*
$\diamond$ *For $b \in [2, 3)$ ($t_1^b = 3, t_2^b \geq 7$):*
*In the FFD-packing, neither BIN$_2$ nor BIN$_3$ contains any $B_i$-item $x$, $i \geq 3$.*
*For values of b that satisfy $t_2^b \geq 9$, neither BIN$_2$ nor BIN$_3$ contains $B_2$-item.*
*For values of b that satisfy $t_2^b \in \{7, 8\}$, BIN$_2$ and BIN$_3$ together contain at most one $B_2$-item.*
$\diamond$ *For $b \in [1.2, 2)$ ($t_1^b = 2, t_2^b \geq 4$):*
*In the FFD-packing, neither BIN$_2$ nor BIN$_3$ contains any $B_i$-item $x$, $i \geq 3$.*
*For values of b that satisfy $t_2^b \geq 6$, neither BIN$_2$ nor BIN$_3$ contains any $B_2$-item.*
*For values of b that satisfy $t_2^b \in \{4, 5\}$, BIN$_2$ and BIN$_3$ both contain at most one $B_2$-item.* **Proof.** Assume, by contradiction, that for some $i$ there is a $B_i$-item $x$ in BIN$_2$ (without loss of generality). We denote by $X$ the overall size of all items in BIN$_1$ that are larger than the $B_i$-items. By Observation 4(iii), the total weight of these items is at least $\frac{t_{i-1}^b+1}{t_{i-1}^b}X$. Let $\beta$ be the number of $B_i$-items in BIN$_1$, and let $B$ denote their overall size. So W(BIN$_1$)$\geq \frac{t_{i-1}^b+1}{t_{i-1}^b}X + B + \frac{\beta}{t_i^b(t_i^b-1)}$. Using the fact that by the assumption $W$(BIN$_1$)$< 1$ must hold, as the contents of BIN$_1$ is not a good subset, we get:

$$\frac{t_{i-1}^b + 1}{t_{i-1}^b}X + B + \frac{\beta}{t_i^b(t_i^b - 1)} < 1. \tag{7}$$

As the $B_i$-item $x \leq \frac{1}{t_i^b-1}$ did not fit into BIN$_1$, we know that $X + B > \frac{t_i^b-2}{t_i^b-1}$. We subtract the last inequality multiplied by $(t_{i-1}^b + 1)$ from (7) multiplied by $t_{i-1}^b$ to get:

$$\frac{\beta t_{i-1}^b}{t_i^b(t_i^b - 1)} < t_{i-1}^b - \frac{t_i^b - 2}{t_i^b - 1}(t_{i-1}^b + 1) + B \tag{8}$$

In addition, we know that every $B_i$-item in BIN$_1$ has size at most $\frac{1}{t_i^b-1}$ and there are exactly $\beta$ such items, this implies $B \leq \frac{\beta}{t_i^b-1}$. Plugging this into inequality (8) and simplifying the resulting inequality yields:

$$\beta > \frac{t_i^b - t_{i-1}^b - 2}{t_i^b - t_{i-1}^b} \cdot t_i^b = t_i^b - \frac{2t_i^b}{t_i^b - t_{i-1}^b} \tag{9}$$

If the righthand side of (9) is at least $t_i^b - 3$, that means that $\beta$ is at least $t_i^b - 2$. Together with the item $x$ in BIN$_2$, there are at least ($t_i^b - 1$) $B_i$-items. By Lemma 6(i) those ($t_i^b - 1$) $B_i$-items form a good subset. Thus we derive a contradiction.

So, we have to check for which values of $i$ the righthand side of (9) is at least $t_i^b - 3$, and the above holds:$t_i^b - \frac{2t_i^b}{t_i^b - t_{i-1}^b} \geq t_i^b - 3 \Rightarrow t_i^b - 3t_{i-1}^b \geq 0$ If we apply (5) to the last inequality, we get $t_i^b - 3t_{i-1}^b \geq t_{i-1}^b(t_{i-1}^b - 1) + 1 - 3t_{i-1}^b = (t_{i-1}^b)^2 - 4t_{i-1}^b + 1 \geq 0$. So, $t_{i-1}^b \geq 3.73$ must hold, i.e. $t_{i-1}^b \geq 4$ (as $t_i^b$ is a sequence of integers). We would like to find the value of $i$ starting from which this holds for the different values of $b$. We split the proof to several subcases, in accordance to the value of $b$, and treat them separately. There are three cases we need to consider: $b \in [1.2, 2)$, $b \in [2, 3)$ and $b \geq 3$. We now treat these cases one by one.

$\diamond$ For $b \geq 3$: In the case $b \geq 3$, $t_1^b \geq 4$, and $\{t_i^b\}$ is an increasing sequence, so for $i - 1 \geq 1 \Rightarrow i \geq 2$ there are no $B_i$-items in BIN$_2$ or in BIN$_3$.

$\diamond$ For $b \in [1.2, 2)$: By the definition of the $\{t_i^b\}$ sequence in (2), for values of $b$ in this interval: $t_1^b = 2$, $r_1^b = \frac{1}{b} - \frac{1}{2}$ and $t_2^b = \lfloor 1 + \frac{1}{r_1^b} \rfloor = \lfloor 1 + \frac{1}{\frac{1}{b} - \frac{1}{2}} \rfloor = \lfloor 1 + \frac{1}{\frac{2-b}{2b}} \rfloor = \lfloor 1 + \frac{2b}{2-b} \rfloor = \lfloor \frac{2+b}{2-b} \rfloor$. Since $\lfloor \frac{2+b}{2-b} \rfloor$ is non-decreasing for $b \in [1.2, 2)$, $t_2^b \geq 4$ holds. Also $\{t_i^b\}$ is an increasing sequence. So for $i - 1 \geq 2 \Rightarrow i \geq 3$, according to previous considerations, there are no $B_i$-items in BIN$_2$ or in BIN$_3$.
It is left to consider the $B_2 \in (\frac{1}{t_2^b}, \frac{1}{t_2^b - 1}]$ items. As to the case $i = 2$: According to (9), $\beta > t_2^b - \frac{2t_2^b}{t_2^b - t_1^b} = t_2^b - \frac{2t_2^b}{t_2^b - 2}$ holds. We check which values of $b$ satisfy $t_2^b - \frac{2t_2^b}{t_2^b - 2} \geq t_2^b - 3$. This is equivalent to $t_2^b \geq 6$. So, for values of $b$ that satisfy $t_2^b \geq 6$, BIN$_2$ and BIN$_3$ can not contain a $B_2$-item.
But we showed that for $b \in [1.2, 2)$ $t_2^b \geq 4$ holds. So it is left for us to check what happens for values of $b$ in this interval that satisfy $t_2^b = 4$ or $t_2^b = 5$.

In the case $t_2^b = 4$: the righthand side of (9) is: $t_2^b - \frac{2t_2^b}{t_2^b - 2} = 0$, and this yields $\beta \geq 1$. So there is at least one $B_2$-item in BIN$_1$. If there are two or more $B_2$-items in both BIN$_2$ and BIN$_3$, it means that in total there are at least three $B_2$-items in the set, and according to Lemma 6(i) for $i = 2$, three of those items form a good subset, and we get a contradiction. So BIN$_2$ and BIN$_3$ contain together at most one such item.

In the case $t_2^b = 5$: the righthand side of (9) is: $t_2^b - \frac{2t_2^b}{t_2^b - 2} \sim 1.66$, and this yields $\beta \geq 2$. So there are at least two $B_2$-items in BIN$_1$. If there are two or more $B_2$-items in BIN$_2$ and BIN$_3$ together, it means that in total there are at least four $B_2$-items in the set, and according to Lemma 6(i) for $i = 2$, four of those items form a good subset, and we get a contradiction. So BIN$_2$ and BIN$_3$ contain together at most one such item.

$\diamond$ For $b \in [2, 3)$: According to the definition of the $\{t_i^b\}$ sequence in (2), for values of $b$ in this interval: $t_1^b = 3$, $r_1^b = \frac{1}{b} - \frac{1}{3}$ and $t_2^b = \lfloor 1 + \frac{1}{r_1^b} \rfloor = \lfloor 1 + \frac{1}{\frac{1}{b} - \frac{1}{3}} \rfloor = \lfloor 1 + \frac{1}{\frac{3-b}{3b}} \rfloor = \lfloor \frac{3+2b}{3-b} \rfloor$. Since $\lfloor \frac{3+2b}{3-b} \rfloor$ is non-decreasing for $b \in [2, 3)$, $t_2^b \geq 7$ holds, and $\{t_i^b\}$ is an increasing sequence. So for $i - 1 \geq 2 \Rightarrow i \geq 3$, according to previous considerations, there are no $B_i$-items in BIN$_2$ or in BIN$_3$. It is left for us to check the case $i = 2$: the righthand side of (9) is: $t_2^b - \frac{2t_2^b}{t_2^b - t_1^b} = t_2^b - \frac{2t_2^b}{t_2^b - 3}$. We would like to know for which values of $b$, $\beta \geq t_2^b - 3$, that is $t_2^b - \frac{2t_2^b}{t_2^b - 3} \geq t_2^b - 3$, and for these values we get that there are not any $B_2$-items in BIN$_2$ or in BIN$_3$. So, we get that for values of $b$ in $[2, 3)$ that satisfy $t_2^b \geq 9$, the above holds.
We showed that for $b \in [2, 3)$ $t_2^b \geq 7$ holds. We also showed that for values of $b$ that satisfy $t_2^b \geq 9$ there are no $B_i$-items in BIN$_2$ or in BIN$_3$.

So, it is left for us to examine what happens for values of $b$ in this interval that satisfy $t_2^b = 7$ or $t_2^b = 8$.

In the case $t_2^b = 7$: the righthand side of (9) is: $t_2^b - \frac{2t_2^b}{t_2^b - 3} = 3.5$, and this yields $\beta \geq 4$. So there are at least four $B_2$-items in BIN$_1$. If there are two or more $B_2$-items in both BIN$_2$ and BIN$_3$, it means that in total we have at least six $B_2$-items in the set, and according to Lemma 6(i) for $i = 2$, six of those items form a good subset, and we get a contradiction. So BIN$_2$ and BIN$_3$ together contain at most one such item.

In the case $t_2^b = 8$: the righthand side of (9) is: $t_2^b - \frac{2t_2^b}{t_2^b - 3} = 4.8$, and this yields $\beta \geq 5$. So there are at least five $B_2$-items in BIN$_1$. If there are two or more $B_2$-items in BIN$_2$ and BIN$_3$ together, it means that altogether we have at least seven $B_2$-items in the set, and according to Lemma 6(i) for $i = 2$, seven of those items form a good subset, and we get a contradiction. So BIN$_2$ and BIN$_3$ contain together at most one such item. $\square$

**Claim 4a.** *For $b \geq 1.2$: In the FFD-packing, neither BIN$_2$ nor BIN$_3$ contains any $C_1$-item $x$.*
**Proof.** We will split the proof to two subcases.

For $b \in [1.2, 2)$: Assume the opposite, that there is a $C_1$-item in BIN$_2$ (without loss of generality). We have proved in Claim 1a that in this case no bin contains any $B_1$-item. Consequently, the $C_1$-item $x$ is the largest possible item in these three bins. As $x$ did not fit into BIN$_1$, BIN$_1$ must contain two $C_1$-items (since $C_1 \in (\frac{1}{t_1^b+1}, \frac{1}{t_1^b}]$, and for $b \in [1.2, 2)$ $t_1^b = 2$ holds). According to Lemma 6(ii) for $i = 1$, these two $C_1$-items in BIN$_1$ define a good subset, so BIN$_1$ contains a good subset, an existence of which contradicts our assumption. So the contrary holds.

For $b \geq 2$: Assume that there is a $C_1$-item in BIN$_3$. That means that there are $t_1^b$ $C_1$-items in BIN$_2$ (since we have proved in Claim 1a that for $b \geq 2$ there are not any $B_1$-items in BIN$_2$, we do FFD and $x$ did not fit into BIN$_2$). According to Lemma 6(ii) for $i = 1$, these $t_1^b$ $C_1$- items in BIN$_2$ form a good subset, so BIN$_2$ contains a good subset-again we derive a contradiction. So, there are no $C_1$-items in BIN$_3$. Regarding BIN$_2$; since $b \geq 2$, BIN$_1$ may contain a $B_1$-item. Assume that there is a $C_1$-item $x$ in BIN$_2$. If BIN$_1$ contains no $B_1$-items, we can argue analogously to above (only now it is the contents of BIN$_1$ that will form a good subset). Hence, we may assume that BIN$_1$ does contain some $B_1$-item. We have assumed that BIN$_3$ is not empty (otherwise we produce a packing with an empty bin, in contrary to the assumption). Let $\alpha$ be an item that arrived at BIN$_3$. We discussed the case where $\alpha$ is a $C_1$-item above. So assume $\alpha$ is not a $C_1$-item. Then any item that arrives at BIN$_3$ can be at most a $B_2$-item (because we proved that there can not be any $B_1$ and $D_1$ items in BIN$_3$), and thus has size at most $\frac{1}{t_2^b-1}$. Since we showed in Claim 2a that there are no $D_i$ items in BIN$_3$ for $i \geq 1$, BIN$_3$ can contain only $B_i$ and $C_i$,$i \geq 2$ items. So $\alpha$ has to be one of those. There is a positive integer $\ell$, for which $\alpha \in (\frac{1}{\ell+1}, \frac{1}{\ell}]$, $\ell \geq t_2^b - 1$. Since $\alpha$ did not fit into BIN$_1$, BIN$_1$ is full by more than $1 - \frac{1}{\ell}$ with items of size more than $\frac{1}{\ell+1}$. Let these items be $q_1, q_2, ..., q_m$, $q_1 \in B_1$. Because we used FFD, $q_1 \geq q_2 \geq ... \geq q_m \geq \alpha$ holds. By Theorem 5, we may claim that for a integer $\ell$ and an item of size greater than $\frac{1}{\ell+1}$, the ratio between its weight and its size is at least $\frac{\ell+2}{\ell+1}$. So, the total weight of the items in BIN$_1$ is at least

$$\begin{aligned}\sum_{i=1}^{m} W(q_i) &= q_1 + \frac{1}{t_1^b(t_1^b-1)} + \sum_{i=2}^{m} W(q_i) \geq q_1 + \frac{1}{t_1^b(t_1^b-1)} + \left(1 - \frac{1}{\ell} - q_1\right)\frac{\ell+2}{\ell+1} \\ &= 1 - \frac{2}{\ell(\ell+1)} - \frac{1}{\ell+1}q_1 + \frac{1}{t_1^b(t_1^b-1)}\end{aligned}$$

But $\ell \geq t_2^b - 1 \geq t_1^b(t_1^b - 1)$ holds (by Lemma 2). Also $q_1 \leq \frac{1}{t_1^b-1}$ (since $q_1$ is a $B_1$ item), and $q_1 \leq \frac{1}{b} \leq \frac{1}{2}$ since $t_1^b - 1 \leq b$ and $b \geq 2$, so $W(BIN_1) \geq 1 - \frac{2}{\ell(\ell+1)} - \frac{1}{2(\ell+1)} + \frac{1}{\ell} = 1 - \frac{4+\ell-2\ell-2}{2\ell(\ell+1)} = 1 - \frac{2-\ell}{2\ell(\ell+1)}$. We want to know when W(BIN$_1$)$\geq 1$, i.e. when $\frac{2-\ell}{2\ell(\ell+1)} \leq 0$. $2\ell(\ell+1) \geq 0$ for any positive integer $\ell$, and $2 - \ell \leq 0$ for $\ell \geq 2$. Since $\ell \geq t_2^b - 1$, and $t_2^b \geq 6$ for any $b \geq 2$, $\ell \geq 5$, and the above holds. So, W(BIN$_1$)$\geq 1$, thus the contents of BIN$_1$ form a good subset in contradiction to our assumption. We conclude that BIN$_2$ does not contain any $C_1$-item, either. □

**Claim 5a.** *For $b \in [1.2, 2)$ that satisfy $t_2^b \in \{4, 5\}$, and for $b \in [2, 3)$ that satisfy $t_2^b \in \{7, 8\}$:*
*In the FFD-packing the bin BIN$_3$ does not contain any $B_2$-item $y$.*
**Proof.** We have proved for these values of $b$ (among others) that BIN$_2$ and BIN$_3$ do not contain any $B_1$, $C_1$ or $D_1$ item. Consequently, the $B_2$-item $y$ is the largest possible item in these two bins. We also proved in Claim 3a, that for the mentioned values of $b$, the bins BIN$_2$ and BIN$_3$ together contain at most one $B_2$-item. So FFD puts this single $B_2$-item into BIN$_2$. □

**Claim 6a.**
*For any $b \in [1.2, 3)$ : In the FFD-packing, BIN$_3$ cannot contain any $C_i$-item $x$ with $i \geq 3$.*
*For $b \geq 3$ : In the FFD-packing BIN$_3$ cannot contain any $C_i$-item $x$ with $i \geq 2$.*
**Proof.** Assume by contradiction that BIN$_3$ contains some $C_i$-item $x$. We denote by $Y$ the overall size of items in BIN$_2$ that are larger than the $C_i$-items. All of these items are at least $C_{i-1}$-items. That is true starting with

$i \geq 3$, since there are no $D_{i-1}$ and $B_i$, $i \geq 3$ items in BIN$_2$. That does not always hold for $i = 2$ though, as for $b \in [1.2, 3)$ there may be a $B_2$-item in BIN$_2$. By Observation 4(iii), the total weight of these items is at least $\frac{(t^b_{i-1}+1)}{t^b_{i-1}} Y$. Let $\gamma$ be the number of $C_i$-items in BIN$_2$, and denote their overall size by $C$. So $W$(BIN$_2$)$\geq$ $\frac{(t^b_{i-1}+1)}{t^b_{i-1}} Y + \frac{t^b_i+1}{t^b_i} C$. Since by the assumption the contents of BIN$_2$ is not a good subset, $W$(BIN$_2$)$< 1$ holds, and this yields:

$$\frac{(t^b_{i-1}+1)}{t^b_{i-1}} Y + \frac{t^b_i+1}{t^b_i} C < 1. \tag{10}$$

Since the $C_i$-item $x \leq \frac{1}{t^b_i}$ did not fit into BIN$_2$, we have $Y + C > \frac{t^b_i - 1}{t^b_i}$. We multiply the last inequality by $(t^b_{i-1} + 1)$ and the inequality (10) by the factor $t^b_{i-1}$. Subtracting one inequality from the other gives:

$$\left(\frac{t^b_{i-1} - t^b_i}{t^b_i}\right) C < \frac{t^b_{i-1} - t^b_i + 1}{t^b_i}.$$

We divide both sides of the inequality by $(t^b_{i-1} - t^b_i)$. Note that $t^b_{i-1} < t^b_i$ (by Lemma 1).

$$C > \frac{t^b_{i-1} - t^b_i + 1}{t^b_{i-1} - t^b_i} = 1 + \frac{1}{t^b_{i-1} - t^b_i} = 1 - \frac{1}{t^b_i - t^b_{i-1}}. \tag{11}$$

Moreover, $C \leq \frac{\gamma}{t^b_i}$ holds (since size of each $C_i$-item is at most $\frac{1}{t^b_i}$ and there are $\gamma$ of those). Combining this with the inequality (11), we derive:

$$\gamma > t^b_i - \frac{t^b_i}{t^b_i - t^b_{i-1}}. \tag{12}$$

If the righthand side of (12) is larger than $t^b_i - 2$, that means we have at least $(t^b_i - 1)$ $C_i$-items in BIN$_2$, and at least one $C_i$-item in BIN$_3$. Together that gives $t^b_i$ $C_i$-items in the set. By Lemma 6(ii) those $t^b_i$ $C_i$-items construct a good subset, and thus we derive a contradiction as wanted.
So, we want to check for which values of $i$ the inequality $t^b_i - \frac{t^b_i}{t^b_i - t^b_{i-1}} > t^b_i - 2 \Rightarrow t^b_i - 2t^b_{i-1} > 0$ holds. If we apply Lemma 2 to the above inequality, we get: $t^b_i - 2t^b_{i-1} \geq t^b_{i-1}(t^b_{i-1} - 1) + 1 - 2t^b_{i-1} = (t^b_{i-1})^2 - t^b_{i-1} + 1 - 2t^b_{i-1} = (t^b_{i-1})^2 - 3t^b_{i-1} + 1 > 0$. So we get that $t^b_{i-1} > 2.61$ must hold, i.e. $t^b_{i-1} \geq 3$.

But for $b \geq 1.2$, $t^b_2 \geq 4$ holds, and $\{t^b_i\}$ is an increasing sequence. So this holds starting with $i - 1 \geq 2 \Rightarrow i \geq 3$. Thus for $b \in [1.2, 3)$ there are no $C_i$-items with $i \geq 3$ in BIN$_3$.

For $b \geq 3$, $t^b_1 \geq 4$ already, and there are no $B_2$-items in BIN$_2$. So the above holds starting with $i = 2$, and BIN$_3$ cannot contain any $C_2$-item. □

**Claim 7a.** *For $b \in [1.2, 3)$: In the FFD-packing, BIN$_3$ cannot contain any $C_2$-item $x$.*
**Proof.** Recall that in this case there can be a $B_2$-item in BIN$_2$.
Assume, by contradiction, that BIN$_3$ contains some $C_2$-item $x$. Note that in Claim 5a we showed that BIN$_3$ does not contain $B_2$-items. If BIN$_2$ does not contain a $B_2$-item, it must contain $t^b_2$ $C_2$-items. This is because the $C_2$-items are the largest items that can be in BIN$_2$, since we have proved that BIN$_2$ cannot contain any $B_1$, $C_1$ or $D_1$ item, and since $x$ did not fit in it. By Lemma 6(ii) for $i = 2$, those $t^b_2$ $C_2$-items form a good subset, and thus we derive a contradiction. So BIN$_3$ cannot contain a $C_2$-item
Hence, we may assume that BIN$_2$ does contain some $B_2$-item $y$ (and by Claims 3a and 5a it is the only $B_2$-item in BIN$_2$). In Claim 3a we showed that this is possible only for those values of $b$ in interval $[1.2, 2)$ which satisfy $t^b_2 = 4$ or $t^b_2 = 5$, or for values of $b$ in interval $[2, 3)$ which satisfy $t^b_2 = 7$ or $t^b_2 = 8$ (in the other cases we proved there cannot be $B_2$-item in BIN$_2$). We split the proof to consider each one of these options separately.

(A)(1) *For values of $b$ in $[1.2, 2)$ that satisfy $t^b_2 = 4$:* $B_2 = (\frac{1}{4}, \frac{1}{3}]$, $C_2 = (\frac{1}{5}, \frac{1}{4}]$
Similarly as in the proof of Claim 3a, we denote by $X$ the overall size of all items in BIN$_1$ that are larger than

the $B_2$-items (i.e. larger than $\frac{1}{3}$). By Observation 4(iii), the total weight of these items is at least $\frac{t_1^b+1}{t_1^b}X = \frac{3}{2}X$, $t_1^b = 2$.
Let the number of $B_2$-items in $\mathrm{BIN}_1$ be $\beta$, and denote their overall size by $B$. Then $W(\mathrm{BIN}_1) < 1$ (that holds by the assumption) implies:

$$\frac{3}{2}X + B + \frac{\beta}{t_2^b(t_2^b-1)} = \frac{3}{2}X + B + \frac{\beta}{12} < 1. \tag{13}$$

As the $C_2$-item $x \leq \frac{1}{4}$ did not fit into $\mathrm{BIN}_1$, it is at least $\frac{3}{4}$ full, so we have $X + B > \frac{3}{4}$. We subtract the last inequality multiplied by $\frac{3}{2}$ from (13). This yields $B > \frac{1}{4} + \frac{\beta}{6}$. Finally, we plug in $B \leq \frac{\beta}{3}$ (there are $\beta$ $B_2-$items, each of size smaller than $\frac{1}{3}$), and derive: $\frac{\beta}{3} > \frac{1}{4} + \frac{\beta}{6} \Longrightarrow \beta > \frac{3}{2}$, i.e $\beta \geq 2$. Altogether (both in $\mathrm{BIN}_1$ and $\mathrm{BIN}_2$), there are at least three $B_2$-items, and according to Lemma 6(i) for $i = 2$, those three items form a good subset. So, again we detected a good subset of items in the set.

(A)(2) *For values of $b$ in* $[1.2, 2)$ *that satisfy* $t_2^b = 5$*:* $B_2 = (\frac{1}{5}, \frac{1}{4}]$, $C_2 = (\frac{1}{6}, \frac{1}{5}]$
Once again, we denote by $X$ the overall size of all items in $\mathrm{BIN}_1$ that are larger than the $B_2$-items (i.e. larger than $\frac{1}{4}$ in this case). By Observation 4(iii), the total weight of these items is at least $\frac{t_1^b+1}{t_1^b}X = \frac{3}{2}X$, $t_1^b = 2$.
Let the number of $B_2$-items in $\mathrm{BIN}_1$ be $\beta$, denote their overall size by $B$. Then $W(\mathrm{BIN}_1) < 1$ implies:

$$\frac{3}{2}X + B + \frac{\beta}{t_2^b(t_2^b-1)} = \frac{3}{2}X + B + \frac{\beta}{20} < 1. \tag{14}$$

As $C_2$-item $x \leq \frac{1}{5}$ did not fit into $\mathrm{BIN}_1$, it is at least $\frac{4}{5}$ full, so we have: $X + B > \frac{4}{5}$. We subtract the last inequality multiplied by $\frac{3}{2}$ from (14). This yields $B > \frac{2}{5} + \frac{\beta}{10}$. We plug in $B \leq \frac{\beta}{4}$ (there are $\beta$ $B_2$-items, each of size smaller than $\frac{1}{4}$), and derive: $\frac{\beta}{4} > \frac{2}{5} + \frac{\beta}{10} \Longrightarrow \beta > \frac{8}{3}$, i.e $\beta \geq 3$. Altogether, in $\mathrm{BIN}_1$ and $\mathrm{BIN}_2$ there are at least four $B_2$-items, and according to lemma 6(i) for $i = 2$, those four items form a good subset.

B)(1) *For values of $b$ in* $[2, 3)$ *that satisfy* $t_2^b = 7$*:* $B_2 = (\frac{1}{7}, \frac{1}{6}]$, $C_2 = (\frac{1}{8}, \frac{1}{7}]$
Again, we denote by $X$ the overall size of all items in $\mathrm{BIN}_1$ that are larger than the $B_2$-items (i.e. larger than $\frac{1}{6}$). By Observation 4(iii), the total weight of these items is at least $\frac{t_1^b+1}{t_1^b}X = \frac{4}{3}X$,$t_1^b = 3$.
Let the number of $B_2$-items in $\mathrm{BIN}_1$ be $\beta$, and denote by $B$ their overall size. Then $W(\mathrm{BIN}_1) < 1$ implies:

$$\frac{4}{3}X + B + \frac{\beta}{t_2^b(t_2^b-1)} = \frac{4}{3}X + B + \frac{\beta}{42} < 1. \tag{15}$$

As the $C_2$-item $x \leq \frac{1}{7}$ did not fit into $\mathrm{BIN}_1$, we have: $X + B > \frac{6}{7}$. We subtract the last inequality multiplied by $\frac{4}{3}$ from (15). This yields $B > \frac{3}{7} + \frac{\beta}{14}$. Finally, we plug in $B \leq \frac{\beta}{6}$ (there are $\beta$ $B_2$-items, each of size smaller than $\frac{1}{6}$), and derive: $\frac{\beta}{6} > \frac{3}{7} + \frac{\beta}{14} \Longrightarrow \beta > \frac{9}{2}$, i.e $\beta \geq 5$. Altogether, there are at least six $B_2$-items (in $\mathrm{BIN}_1$ and $\mathrm{BIN}_2$ together), and according to Lemma 6(i) for $i = 2$, those six items form a good subset.

B)(2) *For values of $b$ in* $[2, 3)$ *that satisfy* $t_2^b = 8$*:* $B_2 = (\frac{1}{8}, \frac{1}{7}]$, $C_2 = (\frac{1}{9}, \frac{1}{8}]$
As before, we denote by $X$ the overall size of all items in $\mathrm{BIN}_1$ that are larger than the $B_2$-items (i.e. larger than $\frac{1}{7}$). By Observation 4(iii), the total weight of these items is at least $\frac{t_1^b+1}{t_1^b}X = \frac{4}{3}X$, $t_1^b = 3$.
Let the number of $B_2$-items in $\mathrm{BIN}_1$ be $\beta$, and denote their overall size by $B$. Then $W(\mathrm{BIN}_1) < 1$ implies:

$$\frac{4}{3}X + B + \frac{\beta}{t_2^b(t_2^b-1)} = \frac{4}{3}X + B + \frac{\beta}{56} < 1. \tag{16}$$

As the $C_2$-item $x \leq \frac{1}{8}$ did not fit into $\mathrm{BIN}_1$, we have: $X + B > \frac{7}{8}$. We subtract the last inequality multiplied by $\frac{4}{3}$ from (16). This yields $B > \frac{1}{2} + \frac{3\beta}{56}$. Finally, we plug in $B \leq \frac{\beta}{7}$ (there are $\beta$ $B_2$-items, each of size smaller than $\frac{1}{7}$), and derive: $\frac{\beta}{7} > \frac{1}{2} + \frac{3\beta}{56} \Longrightarrow \beta > \frac{28}{5}$, i.e $\beta \geq 6$. Together with the single $B_2$-item in $\mathrm{BIN}_2$, there are at least seven $B_2$-items (in $\mathrm{BIN}_1$ and $\mathrm{BIN}_2$ together), and according to Lemma 6(i) for $i = 2$, those seven items form a good subset. In all the cases we derived a contradiction to our assumptions by showing the existence of a good subset. So, the opposite holds. □

Summarizing, in Claims 1a through 7a we have shown that for $b \geq 1.2$ in the FFD-packing BIN$_3$ can neither contain a $B_i$-, nor a $C_i$-, nor a $D_i$-item, under our assumptions. Thus, BIN$_3$ is empty, and the FFD-packing is a good packing. This is the final contradiction, which completes the proof of Theorem 7 for the case $b \geq 1.2$.

As to the case $b \in [1, 1.2)$ and the suitable partition of $(0, \frac{1}{b}]$. Note that in this case, by the definitions of the sequence in (2), $t_1^b = 2$,$t_2^b = 3$ and $t_3^b \geq 7$. Thus, the proof from [9] for $b = 1$ that in the FFD-packing BIN$_3$ can neither contain a $B_i$-, nor a $C_i$-, nor a $D_i$-item holds, using a small number of modifications.

Recall that our main assumption was a non-existence of a good subset and of a good packing, we saw that for any $b \geq 1$ the opposite holds, and we can always obtain a good packing or a good subset. □

Now, let us show that Step (4) is always executable and well-defined.

**Theorem 8.** *For any $b \geq 1$: The algorithm RAR$_3$(b) can be implemented so that it never gets stuck in Step (4).*

**Proof.** The proof is by induction on the number of packed items. We keep the following invariant: As RAR$_3$(b) receives a new item to pack, one of the three active bins is empty. Without loss of generality, let this bin be BIN$_3$ and the rest of the items are packed in the remaining two active bins. Obviously, this invariant holds as RAR$_3$(b) receives the first item. Assume it holds after the packing of item $a_j$, and consider the moment RAR$_3$(b) receives $a_{j+1}$. By inductive assumption, there is a set of items that were packed in the previous step into two of the three active bins by RAR$_3$(b) and BIN$_3$ is empty at this point. When item $a_{j+1}$ arrives, it is put into BIN$_3$ in Step (1). Then we remove all the items from the bins and sort them in non-increasing order by their size. There are two possible cases:

(i) The algorithm finds a good subset of $(t_i^b - 1)$ $B_i$-items or $t_i^b$ $C_i$-items, and after we remove these $B_i$-items in Step (4.1) or $C_i$-items in Step (4.2) from the set, RAR$_3$(b) packs the remaining items in the set (which are still sorted in non-increasing order) by FF using two bins in Step (4.3).

(ii) RAR$_3$(b) packs the entire set of items by FF using three bins in Step (4.3).

We discuss these two cases separately. We claim that all items can be packed. (i) If we perform Step (4.1), it means that the new item that has arrived is a $B_i$-item, thus completes the amount of $B_i$-items in the list to $t_i^b - 1$ (otherwise, if the new item is not a $B_i$-item and there are $(t_i^b - 1)$ $B_i$-items present in the set we would have combined them as a good subset in the previous step). If we perform Step (4.2), it means that the new item that has arrived is a $C_i$-item, thus completes the amount of $C_i$-items in the list to $t_i^b$ (by similar considerations). In any case, by Lemma 6 we know that these items form a good subset and fit into a single bin. We put them in BIN$_3$. Then we go to Step (4.3) and pack the rest of the items in BIN$_1$ and BIN$_2$ by FF. Assume by contradiction that some item $z$ does not fit in any of those bins. First, we consider the case when the good subset we remove consists of $B_i$-items. For $t_i^b = 2$: $z \in \left(\frac{1}{2}, \frac{1}{b}\right]$ ($t_i^b = 2$ for $i = 1$, as $t_2^b \geq 3$ for any $b \geq 1$ and $t_i^b$ for $i > 2$ is even greater, since $t_i^b$ is a strictly increasing sequence). Then, $z$ is a $B_1$-item, but by Lemma 6(i) for $i = 1$ such $z$ forms a good subset by itself, so there can not be such $z$ since the algorithm removes it as a good subset in Step (4.1) as soon as it arrives. For $t_i^b \geq 3$: we remove $(t_i^b - 2)$ $B_i$-items (not including the new item) from the set. The total size of the removed items is at least $\frac{1}{t_i^b} \cdot (t_i^b - 2) = 1 - \frac{2}{t_i^b} \geq \frac{1}{3}$, as $t_i^b \geq 3$.

In the case the good subset we remove consists of $C_i$-items; For $t_i^b \geq 2$: we remove $(t_i^b - 1)$ $C_i$-items (not including the new item) from the set. The total size of the removed items is at least $\frac{1}{t_i^b+1} \cdot (t_i^b - 1) = 1 - \frac{2}{t_i^b+1} \geq \frac{1}{3}$, as $t_i^b \geq 2$. So, as $z$ did not fit in any of BIN$_1$ or BIN$_2$, and we assumed the entire set of items (without the new item) fits in two bins, the sum of the sizes of all the items that remain in the set after we remove the good subset (which has to contain $a_{j+1}$) either in Step (4.1) or Step (4.2), is upper bounded by $\frac{5}{3}$. As $z$ did not fit in in any of the two bins, each of these bins is more than $1 - z$ full. We get $2(1 - z) + z < \frac{5}{3} \Rightarrow z > \frac{1}{3}$. So, we are looking at values of $z$ in $\left(\frac{1}{3}, \frac{1}{2}\right]$. There are few possible cases:

$\diamond$ $t_1^b = 2$ and $t_2^b = 3$ ($b \in [1, 1.2)$). Then $z$ is a $B_2$-item, as $B_2 = \left(\frac{1}{3}, \frac{1}{2}\right]$.

In this case, the size of any of the items in BIN$_1$ and BIN$_2$ is greater or equal to $z$ as they are packed by FFD heuristic, but less than $\frac{1}{2}$ (otherwise they are removed as a good subset in earlier stage), hence all these items have sizes in the interval $\left(\frac{1}{3}, \frac{1}{2}\right]$. Thus, they are also $B_2$-items. So, as there was no room for $z$, each of the bins $\mathrm{BIN}_1$ and $\mathrm{BIN}_2$ contains exactly two such items, who by Lemma 6(i) for $i = 2$ form a good subset of $B_2$-items. But that is not possible, as according to our algorithm such subset is removed from the sorted list as soon as the second $B_2$-item arrives. So we get a contradiction.

$\diamond$ $t_1^b = 2$ and $t_2^b > 3$ ($b \in [1.2, 2)$). Then $z$ is a $C_1$-item, as $C_1 = \left(\frac{1}{3}, \frac{1}{2}\right]$.
In this case, a very similar consideration as in the case above combined with Lemma 6(ii) for $i = 1$ shows that if there are two $C_1$-items in $\mathrm{BIN}_1$ or $\mathrm{BIN}_2$ we get a contradiction.

$\diamond$ $t_1^b = 3$ ($b \in [2, 3)$). Then $z$ is a $B_1$-item, as $B_1 = \left(\frac{1}{3}, \frac{1}{b}\right]$.
In this case, a very similar consideration as in the case above combined with Lemma 6(i) for $i = 1$ shows that if there are two $B_1$-items in $\mathrm{BIN}_1$ or $\mathrm{BIN}_2$ we get a contradiction.

$\diamond$ $t_1^b > 3$ ($\lfloor 1 + b \rfloor > 3 \Rightarrow \lfloor b \rfloor > 2 \Rightarrow b \geq 3$). In this case there can not be an item from interval $\left(\frac{1}{3}, \frac{1}{2}\right]$ in the list, as the size of any item is bounded from above by $\frac{1}{b}$, and here $b \geq 3$.
So, as we showed that any of the cases above is not possible according to our algorithm, we can conclude there is no such item $z$, and hence after we remove the good subset of $B_i$ or $C_i$-items from the set and put them into $\mathrm{BIN}_3$, all the remaining items being packed by FF fit in two active bins. Then we just move on to Step (5) where $\mathrm{BIN}_3$ is closed and replaced by a empty bin. Thus, the invariant holds after we pack the $a_{j+1}$ item.

(ii) In this case, there is no good subset of $(t_i^b - 1)$ $B_i$-items or $t_i^b$ $C_i$-items in the set. So, the new item $a_{j+1}$ has size in $\left(0, \frac{1}{2}\right]$ (otherwise it would form a good subset of $B_1$-items all by itself). We pack the set of items by FFD heuristic using the three active bins. Assume by contradiction that some item $z$ does not fit in any of those bins. As $z$ did not fit in any of $\mathrm{BIN}_1$ or $\mathrm{BIN}_2$, and we assumed the entire set of items (without the new item $a_{j+1}$) fits in two bins, the sum of the sizes of all the items in the set after we receive the new item is upper bounded by 2.5. As $z$ did not fit in in any of these bins, each bin is more than $1 - z$ full. We get $3(1 - z) + z < 2.5 \Rightarrow z > \frac{1}{4}$. So, we are looking at values of $z$ in $\left(\frac{1}{4}, \frac{1}{b}\right]$.

This $z$ can not be from interval $\left(\frac{1}{2}, \frac{1}{b}\right]$, because such $z$ forms a good subset of $B_1$-items by itself, and the algorithm removes it in Step (4.1) as soon as it arrives. So $z \in \left(\frac{1}{4}, \frac{1}{2}\right]$. We distinguish between two subcases.

$\diamond$ $z \in \left(\frac{1}{3}, \frac{1}{2}\right]$. In this case, the size of any of the items in $\mathrm{BIN}_1$, $\mathrm{BIN}_2$ and $\mathrm{BIN}_3$ is greater or equal to $z$ as they are packed by FFD heuristic, but at most $\frac{1}{2}$, hence all these items have sizes in the interval $\left(\frac{1}{3}, \frac{1}{2}\right]$. So, as $z$ did not fit in any bin, each of the bins $\mathrm{BIN}_1$, $\mathrm{BIN}_2$ and $\mathrm{BIN}_3$ contains exactly two such items, hence, together with $z$ there are 7 items with sizes in $\left(\frac{1}{3}, \frac{1}{2}\right]$ in the set. Before $a_{j+1}$ arrived there were at least 6 of those items in the set, and this contradicts the invariant that holds according to our assumption, because these items could not possibly be packed in two active bins, as their overall size is greater than 2.

$\diamond$ $z \in \left(\frac{1}{4}, \frac{1}{3}\right]$. In this case, the size of any of the items in $\mathrm{BIN}_1$, $\mathrm{BIN}_2$ and $\mathrm{BIN}_3$ is greater or equal to $z$ as they are packed by FFD heuristic, but at most $\frac{1}{2}$, hence these items have sizes in interval $\left(\frac{1}{3}, \frac{1}{2}\right]$ or in $\left(\frac{1}{4}, \frac{1}{3}\right]$. Assume that there are $a$ items from $\left(\frac{1}{3}, \frac{1}{2}\right]$ and $b$ items from $\left(\frac{1}{4}, \frac{1}{3}\right]$. In total, there can be at most 7 such items in the set, as we assumed the entire set of items (without the new item) fits in two bins, so $a + b \leq 7$. On the other hand, $a \leq 1$ must hold, since we saw in the case discussed earlier that two items with size in $\left(\frac{1}{3}, \frac{1}{2}\right]$ form a good subset, in contradiction to our assumption that there is no good subset of $B_i$ or $C_i$-items in the set. If $a = 0$, as there was no room for $z$, each of the bins $\mathrm{BIN}_1$, $\mathrm{BIN}_2$ and $\mathrm{BIN}_3$ contains exactly three items with sizes in $\left(\frac{1}{4}, \frac{1}{3}\right]$, hence, together with $z$ there are 10 such items in the set-a contradiction to $a + b \leq 7$. If $a = 1$, as there was no room for $z$, $\mathrm{BIN}_1$ contains one item with size in $\left(\frac{1}{3}, \frac{1}{2}\right]$ and either one or two items with sizes in $\left(\frac{1}{4}, \frac{1}{3}\right]$, $\mathrm{BIN}_2$ and $\mathrm{BIN}_3$ contains three items with sizes in $\left(\frac{1}{4}, \frac{1}{3}\right]$ each. Hence, together with $z$ there are 8 or 9 such items in the set-a contradiction to $a + b \leq 7$.

So, as we showed that any of the cases above is not possible according to our algorithm, we can conclude there is no such item $z$, and hence all the items (including the new item) being packed by FFD heuristic fit in three active bins. According to Theorem 7, either these items are packed into two bins such that $\mathrm{BIN}_3$ is empty-in this case we go to Step (5) and the invariant is fulfilled, or we detect a good subset in $\mathrm{BIN}_1$ or $\mathrm{BIN}_2$- in this case we go to Step (5) where the bin that contains the good subset is closed, and a new bin is opened instead. So in both cases the invariant holds as we finish packing the $a_{j+1}$ item. $\square$

## 4 The asymptotic competitive ratio of RAR$_3$(b)

To establish the upper bound, we use the weighting function technique.

**Theorem 9.** *For any $b \geq 1$: In any packing of a list $L$ into bins of size $\frac{1}{b}$, the weight of any bin is at most $\rho(b)$. Hence, $W(L) \leq \rho(b)$ OPT$_{\frac{1}{b}}(L)$ holds.*

**Proof.** Let us consider some fixed bin $\mathcal{B}$ (of size $\frac{1}{b}$) that contains items $q_1 \geq q_2 \geq \ldots \geq q_m$. Obviously $\sum_{i=1}^{m} q_i \leq \frac{1}{b}$ holds. We distinguish between two cases.

(a) $q_i \in \left(\frac{1}{t_i^b}, \frac{1}{t_i^b-1}\right]$ for $i = 1, \ldots, m$. Then, by the definition of the weighting function $W(\mathcal{B}) = \sum_{i=1}^{m} W(q_i) = \sum_{i=1}^{m}\left(q_i + \frac{1}{t_i^b(t_i^b-1)}\right) = \sum_{i=1}^{m} q_i + \sum_{i=1}^{m}\frac{1}{t_i^b-1} - \sum_{i=1}^{m}\frac{1}{t_i^b} \leq \frac{1}{b} - \sum_{i=1}^{m}\frac{1}{t_i^b} + \sum_{i=1}^{m}\frac{1}{t_i^b-1}$, as $\sum_{i=1}^{m} q_i \leq \frac{1}{b}$ holds. Note that from (3) we can see that $r_m^b = \frac{1}{b} - \sum_{i=1}^{m}\frac{1}{t_i^b}$, and from (4) for $i = m+1$ we get $r_m^b \leq \frac{1}{t_{m+1}^b-1}$. Plugging this into the above inequality we get $W(\mathcal{B}) \leq \sum_{i=1}^{m}\frac{1}{t_i^b-1} + \frac{1}{t_{m+1}^b-1} = \sum_{i=1}^{m+1}\frac{1}{t_i^b-1} < \sum_{i=1}^{\infty}\frac{1}{t_i^b-1} = \rho(b)$. So the weight of bin $\mathcal{B}$ is upper bounded by $\rho(b)$.

(b) Now assume $s \leq m$ is the least $i$ such that $q_i \notin \left(\frac{1}{t_i^b}, \frac{1}{t_i^b-1}\right]$, and hence $q_s \leq \frac{1}{t_s^b}$ (as any of the items $q_1, \ldots, q_{s-1}$ is contained in an interval $\left(\frac{1}{t_i^b}, \frac{1}{t_i^b-1}\right]$ for the corresponding $i$, $\frac{1}{b} \geq \sum_{i=1}^{s} q_i > \frac{1}{t_1^b} + \frac{1}{t_2^b} + \ldots + \frac{1}{t_{s-1}^b} + q_s$ holds, combining this with (3) and (4) for $i = s$, we get $q_s < \dfrac{1}{b} - \dfrac{1}{t_1^b} + \dfrac{1}{t_2^b} - \ldots - \dfrac{1}{t_{s-1}^b} = r_{s-1}^b \leq \dfrac{1}{t_s^b - 1}$ and together with the fact $q_s \notin \left(\frac{1}{t_s^b}, \frac{1}{t_s^b-1}\right]$, this implies the inequality stated above).

We denote by $Q$ the sum of the sizes of the remaining items $q_i$ with $i \geq s$. $Q = \sum_{i=s}^{m} q_i$. Since $\sum_{i=1}^{m} q_i = \sum_{i=1}^{s-1} q_i + \sum_{i=s}^{m} q_i \leq \frac{1}{b}$, $Q + \sum_{i=1}^{s-1} q_i \leq \frac{1}{b}$ holds. As any of the items $q_1, \ldots, q_{s-1}$ is contained in an interval $\left(\frac{1}{t_i^b}, \frac{1}{t_i^b-1}\right]$, it has size of at least $\frac{1}{t_i^b}$, for the corresponding $i$. Combining this with (3) and (4) for $i = s$, we get that $Q \leq \frac{1}{b} - \sum_{i=1}^{s-1} q_i < \frac{1}{b} - \sum_{i=1}^{s-1}\frac{1}{t_i^b} = r_{s-1}^b \leq \frac{1}{(t_s^b-1)}$. Since the largest one of the $q_s, \ldots, q_m$ items, $q_s$ has size in $\left(0, \frac{1}{t_s^b}\right]$, we conclude that the size of every one of these items is at most $\frac{1}{t_s^b}$. Then, by Observation 4(ii), their overall weight is at most $\frac{t_s^b+1}{t_s^b} \cdot Q$. As $\sum_{i=1}^{s-1} q_i \leq \frac{1}{b} - Q$ we get $W(\mathcal{B}) \leq \sum_{i=1}^{s-1} q_i + \sum_{i=1}^{s-1}\frac{1}{t_i^b(t_i^b-1)} + \frac{t_s^b+1}{t_s^b} \cdot Q \leq \frac{1}{b} - Q + \sum_{i=1}^{s-1}\frac{1}{t_i^b(t_i^b-1)} + \frac{t_s^b+1}{t_s^b} \cdot Q = \frac{1}{b} - Q + \sum_{i=1}^{s-1}\frac{1}{t_i^b(t_i^b-1)} + Q + \frac{1}{t_s^b} \cdot Q = \frac{1}{b} + \sum_{i=1}^{s-1}\frac{1}{t_i^b(t_i^b-1)} + \frac{1}{t_s^b} \cdot Q$. Using $Q \leq \frac{1}{(t_s^b-1)}$ (proved above), we derive $W(\mathcal{B}) \leq \frac{1}{b} + \sum_{i=1}^{s-1}\frac{1}{t_i^b(t_i^b-1)} + \frac{1}{t_s^b} \cdot \frac{1}{(t_s^b-1)} = \frac{1}{b} + \sum_{i=1}^{s-1}\frac{1}{t_i^b-1} - \sum_{i=1}^{s-1}\frac{1}{t_i^b} + \frac{1}{t_s^b-1} - \frac{1}{t_s^b} = \frac{1}{b} - \sum_{i=1}^{s}\frac{1}{t_i^b} + \sum_{i=1}^{s}\frac{1}{t_i^b-1}$.

Note that from (3) we can see that $r_s^b = \frac{1}{b} - \sum_{i=1}^{s}\frac{1}{t_i^b}$, and from (4) for $i = s+1$ we get $r_s^b \leq \frac{1}{t_{s+1}^b-1}$. Plugging this into the above inequality we get $W(\mathcal{B}) \leq \frac{1}{t_{s+1}^b-1} + \sum_{i=1}^{s}\frac{1}{t_i^b-1} = \sum_{i=1}^{s+1}\frac{1}{t_i^b-1} < \sum_{i=1}^{\infty}\frac{1}{t_i^b-1} = \rho(b)$. Thus the weight of the bin $\mathcal{B}$ is at most $\rho(b)$. The number of such bins is OPT$_{\frac{1}{b}}(L)$, and the proof of Theorem (9) is complete. □

As the repacking in our algorithm is done within the active bins, the lower bound from [3] carries over to our problem as well.

**Theorem 10.** *For any $b \geq 1$ and for any on-line bounded space bin-packing algorithm $A$ that allows repacking within $k$ active bins, we have $R_A^{\infty}(b) \geq \rho(b)$ .*

**Theorem 11.** *For any $b \geq 1$: The algorithm RAR$_3$(b) has the best possible worst case competitive ratio $\rho(b)$.*

**Proof.** To prove that RAR$_3$(b) has the best possible worst-case competitive ratio $\rho(b)$, note that for any list $L$ of items, RAR$_3$(b)$\leq W(L) + 3$ holds (the algorithm closes only bins of weight at least 1 and the last three active bins are added). Combining this with Theorem 9, we get RAR$_3$(b) – $3 \leq W(L) \leq \rho(b)$ OPT$_{\frac{1}{b}}(L)$. Together with the lower bound of Theorem 10 we get that the competitive ratio of RAR$_3$(b) is $\rho(b)$. □

## References

[1] S. Albers, S. Arora, and S. Khanna. Page replacement for general caching problems. In *Proc. 10th Symp. on Discrete Algorithms (SODA)*, pages 31–40. ACM/SIAM, 1999.

[2] Y. Azar, L. Epstein, and R. van Stee. Resource augmentation in load balancing. *J. of Scheduling*, 3(5):249–258, 2000.

[3] J. Csirik and G. J. Woeginger. Resource augmentation for online bounded space bin packing. *Journal of Algorithms*, 44(2):308–320, 2002.

[4] J. Edmonds. Scheduling in the dark. In *Proc. 31st Symp. Theory of Computing (STOC)*, pages 179–188. ACM, 1999.

[5] L. Epstein, Y. Kleiman, J. Sgall, and R. van Stee. Paging with connections: Fifo strikes again. *Theor. Comput. Sci.*, 377(1-3):55–64, 2007.

[6] L. Epstein and R. van Stee. Online bin packing with resource augmentation. In *Proc. of the Approximation and Online Algorithms, Second International Workshop, WAOA*, pages 23–35, 2004.

[7] G. Galambos. A new heuristic for the classical bin packing problem. Technical Report 82, Institut für Mathematik, Universität Augsburg, Germany, 1985.

[8] G. Galambos. Parametric lower bounds for online bin packing. *SIAM J. on Alg. and Disc. Methods*, 7:362–367, 1986.

[9] G. Galambos and G. J. Woeginger. Repacking helps in bounded space online bin packing. *Computing*, 49:329–338, 1993.

[10] G. Gambosi, A. Postiglione, and M. Talamo. Algorithms for the relaxed online bin-packing model. *SIAM J. Comput.*, 30:1532–1551, 2000.

[11] M. R. Garey, R. L. Graham, and J. D. Ullman. Worst-case analysis of memory allocation algorithms. In *Proc of the 4th Symp. Theory of Computing (STOC'72)*, pages 143–150, 1972.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the theory of of NP-Completeness*. Freeman and Company, San Francisco, 1979.

[13] S. W. Golomb. On certain nonlinear recurring sequences. *AMM*, 70(4):403–405, 1963.

[14] E. F. Grove. Online binpacking with lookahead. In *Proc. 6th Symp. on Discrete Algorithms (SODA)*, pages 430–436. ACM/SIAM, 1995.

[15] G. Gutin, T. Jensen, and A. Yeo. Batched bin packing. *Discrete Optimization.*, 2:71–82, 2005.

[16] Z. Ivkovic and E. Lloyd. A fundamental restriction on fully dynamic maintenance of bin packing. *Inform. Process. Lett.*, 59:229–232, 1996.

[17] Z. Ivkovic and E. L. Lloyd. Fully dynamic algorithms for bin packing: Being (mostly) myopic helps. *SIAM J. Comput.*, 28(2):574–611, 1998.

[18] D. S. Johnson. Fast algorithms for bin packing. *J. Comput. Systems Sci.*, 8:272–314, 1974.

[19] D. S. Johnson, A. Demers, J. D. Ullman, M. R. Garey, and R. L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Comput.*, 3:256–278, 1974.

[20] B. Kalyanasundaram and K. Pruhs. Speed is as powerful as clairvoyance. *J. ACM*, 47(4):617–643, 2000.

[21] C. C. Lee and D. T. Lee. A simple online bin packing algorithm. *J. ACM*, 32:562–572, 1985.

[22] C. A. Phillips, C. Stein, E. Torng, and J. Wein. Optimal time-critical scheduling via resource augmentation. *Algorithmica*, 32(2):163–200, 2002.

[23] P. Ramanan, D. J. Brown, C. C. Lee, and D. T. Lee. Online bin packing in linear time. *J. Algorithms*, 10:305–326, 1989.

[24] S. S. Seiden. On the online bin packing problem. *Journal of the ACM*, 49(5):640–671, 2002.

[25] J. D. Ullman. The performance of a memory allocation algorithm. Technical Report 100, Princeton University, Princeton, NJ, 1971.

[26] A. van Vliet. An improved lower bound for online bin packing algorithms. *Inform. Process. Lett.*, 43:277–284, 1992.

[27] G. J. Woeginger. Improved space for bounded-space online bin packing. *SIAM J. Discrete Math.*, 6:575–581, 1993.