Comparing Online Algorithms for Bin Packing Problems

Leah Epstein[1], Lene M. Favrholdt[2], and Jens S. Kohrt[2]

[1] Department of Mathematics, University of Haifa, Israel, lea@math.haifa.ac.il
[2] Department of Mathematics and Computer Science, University of Southern Denmark, Odense, Denmark, {lenem,svalle}@imada.sdu.dk

Abstract The relative worst-order ratio is a measure of the quality of online algorithms. In contrast to the competitive ratio, this measure compares two online algorithms directly instead of using an intermediate comparison with an optimal offline algorithm.
In this paper, we apply the relative worst-order ratio to online algorithms for several common variants of the bin packing problem. We mainly consider pairs of algorithms that are not distinguished by the competitive ratio and show that the relative worst-order ratio prefers the intuitively better algorithm of each pair.
Keywords: Online algorithms, Relative Worst-Order Ratio, Bin Packing, Bin Covering, Bin Coloring, Class-Constrained Bin Packing, Open-End Bin Packing.

1 Introduction

The relative worst-order ratio is a quality measure for online algorithms inspired by [2,19], defined in [4], and refined in [5]. Contrary to the competitive ratio [14,18], the relative worst-order ratio compares two online algorithms directly instead of using an intermediate comparison with an optimal offline algorithm. The definitions of the competitive ratio and the relative worst-order ratio are given in Section 2.
The relative worst-order ratio has previously been used to analyze several different online problems [3,4,5,6,12,13], and proved useful in identification of good algorithms for many different online problems. In this paper, we apply the relative worst-order ratio to several common bin packing problems.

The problems. In *standard bin packing* [7,16,17] the input is a sequence σ of items of sizes $0 < \sigma[i] \leq 1$, which are to be packed into unit capacity bins, without exceeding the capacity. The goal is to minimize the number of bins used. We consider online packing problems. Thus, the algorithms receive the items one at a time, and have to irrevocably decide which open bin to place it in or whether to open a new bin for the item.
Online bin packing has various applications in storage, scheduling and resource allocation. Applications in scheduling come from systems with multiple processors which can be used for limited durations.
In *parametric bin packing* [16,17], the sizes of items are bounded from above by $\frac{1}{t}$ for some positive integer t. Thus, for $t = 1$, the problem is equivalent to standard bin packing.
We further consider a maximization variant called *bin covering* [1,9,11,8,15]. Here, the goal is to maximize the number of bins with a total size of items of at least 1.
In *open-end bin packing* [24] a bin is allowed to contain items of total size larger than 1, as long the items assigned to the bin, except for the last one, have a total size strictly smaller than 1. The goal is to minimize the number of bins used.
Finally, we consider two variants where all items have the same size (so all bins can contain the same number of items), but each item has a color associated with it. In the first model, called *bin coloring* [20], the algorithm is allowed to have only a constant number q of open bins at any time. A bin is open, if it is non-empty but not completely filled. The goal is to minimize the maximum number of different colors in any bin. In the second problem, called *class-constrained*

bin packing [22,23], bins have a fixed number of compartments, and the number of different colors of items in a given bin cannot exceed this number. The goal is to minimize the number of bins used.
All these bin packing variants have applications which can be found in the literature [7,9,10]. In particular, parametric bin packing takes into account the fact that typically items, which are packed into containers, are much smaller than the containers.

Algorithms. A bin packing algorithm is called an Any-Fit algorithm if it opens a new bin only when the new item does not fit in any open (i.e., non-empty) bin. We study three specific Any-Fit algorithms, First-Fit (FF), Worst-Fit (WF), and Almost-Worst-Fit (AWF). FF considers the bins according to the order in which they were opened and places each item in the first possible bin. WF packs each item in the least filled (non-empty) bin. AWF is a close variant of WF. If the current item fits in more than one open bin, AWF chooses the second least filled bin. Otherwise, the algorithm works like WF.
The algorithm Next-Fit (NF) uses a single active bin, and opens a new bin when a new item cannot be packed in the current active bin. Thus NF is not an Any-Fit algorithm.
We also study the family of Harmonic algorithms defined by Lee and Lee [21]. The Harmonic algorithms partition the item sizes into intervals, and the items of each interval are packed separately, using NF.

Previous work. Boyar and Favrholdt [4] studied standard bin packing in terms of the relative worst-order ratio. They showed that any Any-Fit algorithm is no better than FF and no worse than WF. They further showed that $\mathrm{WR}_{\mathrm{WF,FF}} = 2$ and $\mathrm{WR}_{\mathrm{NF,WF}} = 2$. Thus, FF is strictly better than WF, which is strictly better than NF. Note that, surprisingly, the competitive ratios of FF and AWF are both equal to 1.7, whereas the competitive ratios of WF and NF are both 2 [16,17].

Our results. We extend the results of [4] to parametric bin packing. In addition, we consider the algorithm AWF and show that it is strictly worse than FF and strictly better than WF, for standard bin packing as well as for parametric bin packing.
We further discuss (adaptations of) the Harmonic algorithms for three variants: standard bin packing, bin covering, and open-end bin packing. For all three problems, we show that all pairs of algorithms in the class are comparable and their relative worst-order ratios are found. In all cases, the algorithms that seem to be more clever indeed yield better results, even though the competitive ratios of the algorithms are equal or not very different.
For bin coloring, two algorithms were suggested by Krumke et al. [20], a natural greedy algorithm and an algorithm using the NF strategy. The authors show that the greedy algorithm has the worse competitive ratio, though intuitively it should be the better algorithm. We show that the greedy algorithm is better in terms of the relative worst-order ratio.
For class-constrained packing, two algorithms of competitive ratio 2 were suggested by Shachnai and Tamir [22], and each one of them seems to have advantages over the other one. We show that indeed, these two algorithms are not comparable.

2 Quality Measures

In this section we introduce the quality measures which we use for the comparison of algorithms. We give simplified definitions that, for the problems studied here, are equivalent to the original definitions. We first define the competitive ratio.

Definition 1. *For any algorithm* Alg *and any input sequence* σ*, let* $\mathrm{Alg}(\sigma)$ *be the value (i.e., the cost or the profit) of the solution obtained when running* Alg *on* σ*. In particular, let* Opt *be an optimal offline algorithm and let* $\mathrm{Opt}(\sigma)$ *be the value of the best solution for the input.*

Definition 2. *Let* ALG *be an online algorithm defined for an optimization problem.*
If the considered problem is a minimization problem, then the competitive ratio *of* ALG *is*

$$\mathcal{R}\left(\textsc{Alg}\right) = \lim_{N\to\infty}\left(\sup_{\sigma:\textsc{Opt}(\sigma)\geq N}\frac{\textsc{Alg}(\sigma)}{\textsc{Opt}(\sigma)}\right).$$

If the considered problem is a maximization problem, then the competitive ratio *of* ALG *is*

$$\mathcal{R}\left(\textsc{Alg}\right) = \lim_{N\to\infty}\left(\sup_{\sigma:\textsc{Opt}(\sigma)\geq N}\frac{\textsc{Opt}(\sigma)}{\textsc{Alg}(\sigma)}\right).$$

Definition 3. *Let* ALG *be an online algorithm defined for an optimization problem.*
If the considered problem is a minimization problem, then the strict competitive ratio *of* ALG *is*

$$\mathcal{SR}\left(\textsc{Alg}\right) = \sup_{\sigma}\frac{\textsc{Alg}(\sigma)}{\textsc{Opt}(\sigma)}.$$

If the considered problem is a maximization problem, then the strict competitive ratio *of* ALG *is*

$$\mathcal{SR}\left(\textsc{Alg}\right) = \sup_{\sigma}\frac{\textsc{Opt}(\sigma)}{\textsc{Alg}(\sigma)}.$$

Now we turn to the relative worst-order ratio.

Definition 4. *For minimization problems, any algorithm* ALG*, and any input sequence σ, let* $\textsc{Alg}_{\mathrm{W}}(\sigma)$ *be the cost of* ALG *on its worst permutation of σ, i.e.,* $\textsc{Alg}_{\mathrm{W}}(\sigma) = \max_p \textsc{Alg}(p(\sigma))$*, where p is a permutation on $|\sigma|$ elements. Similarly, for a maximization problem, we define* $\textsc{Alg}_{\mathrm{W}}(\sigma) = \min_p \textsc{Alg}(p(\sigma))$.

Definition 5. *Let* A *and* B *be two algorithms defined for an optimization problem. If*

$$\lim_{N\to\infty}\left(\inf_{\sigma:\mathrm{B}_{\mathrm{W}}(\sigma)\geq N}\frac{\mathrm{A}_{\mathrm{W}}(\sigma)}{\mathrm{B}_{\mathrm{W}}(\sigma)}\right) \geq 1,$$

or if

$$\lim_{N\to\infty}\left(\inf_{\sigma:\mathrm{A}_{\mathrm{W}}(\sigma)\geq N}\frac{\mathrm{B}_{\mathrm{W}}(\sigma)}{\mathrm{A}_{\mathrm{W}}(\sigma)}\right) \geq 1,$$

we say that A *and* B *are comparable.*
If A *and* B *are comparable, the* relative worst-order ratio *of* A *to* B *is defined as*

$$\mathrm{WR}_{\mathrm{A,B}} = \lim_{N\to\infty}\left(\sup_{\sigma:\mathrm{B}_{\mathrm{W}}(\sigma)\geq N}\frac{\mathrm{A}_{\mathrm{W}}(\sigma)}{\mathrm{B}_{\mathrm{W}}(\sigma)}\right).$$

Note that both limits exist since the sequences are monotone. It can be the case, however, that the limit is infinite.
For a maximization problem, if $\mathrm{WR}_{\mathrm{A,B}} > 1$*,* A *and* B *are* comparable in A's favor*, and if* $\mathrm{WR}_{\mathrm{A,B}} < 1$*, the algorithms are* comparable in B's favor*. For a minimization problem, it is the other way around: if* $\mathrm{WR}_{\mathrm{A,B}} > 1$*,* A *and* B *are* comparable in B's favor*, and if* $\mathrm{WR}_{\mathrm{A,B}} < 1$*, the algorithms are* comparable in A's favor*.*

In Section 5, we analyze a pair of bin coloring algorithms that have previously been analyzed with the competitive ratio. We show that, according to the competitive ratio, one algorithm is the better algorithm, and according to the relative worst-order ratio, the other algorithm is the better one. Such a disagreement between the two measures is only possible, when the optimal offline solution may depend on the permutation of the input sequence, which is not the case for standard bin packing and many other variants of bin packing.

Definition 6. *If* $\mathrm{A_W}(\sigma) \geq \mathrm{B_W}(\sigma)$ *for any input* σ*, or if* $\mathrm{B_W}(\sigma) \geq \mathrm{A_W}(\sigma)$ *for any input* σ*, then we say that* A *and* B *are* strictly comparable.
If A *and* B *are strictly comparable, the* strict relative worst-order ratio *of* A *to* B *is defined as*

$$\mathrm{SWR_{A,B}} = \sup_{\sigma} \frac{\mathrm{A_W}(\sigma)}{\mathrm{B_W}(\sigma)}.$$

For a maximization problem, if $\mathrm{SWR_{A,B}} > 1$*,* A *and* B *are* strictly comparable in A's favor*, and if* $\mathrm{SWR_{A,B}} < 1$*, the algorithms are* strictly comparable in B's favor*. For a minimization problem, it is the other way around: if* $\mathrm{SWR_{A,B}} > 1$*,* A *and* B *are* strictly comparable in B's favor*, and if* $\mathrm{SWR_{A,B}} < 1$*, the algorithms are* strictly comparable in A's favor*.*

Definition 5 is equivalent to the one given in [5]. The strict version in Definition 6 appears in [4,13].
By definition, the relation of one algorithm being comparable to another one is transitive [5], and the relative worst-order ratio between two algorithms A and B can never exceed the competitive ratio of A. For the case where the algorithms are compared using the strict relative worst order ratio, the same results hold (with respect to the strict competitive ratio).
Since packing problems are often studied using asymptotic measures (as in Definition 2), in our analysis we mostly use Definition 5, that is, compare the outputs for inputs with large enough costs or profits. The competitive ratios discussed in such cases are according to Definition 2. However, in Section 5 the bin coloring problem [20] is studied with respect to Definition 6. The study of bin coloring in [20] uses the strict competitive ratio, as in Definition 3, and we adopt that approach.

3 Parametric bin packing

In this section we study the standard bin packing problem with the restriction that all item sizes are at most $\frac{1}{t}$, for some integer $t \geq 1$. We first show that, for all $t \geq 1$, AWF is strictly better than WF and strictly worse than FF (Theorems 1 and 2). This contrasts with the competitive ratio which is the same for AWF and FF. Then we extend results of [4] comparing WF to FF and NF to the class of Any-Fit algorithms.
As stated in the introduction, it was shown in [4] that, for standard bin packing, all Any-Fit algorithms are comparable to FF, WF, and NF. Such a result clearly carries over to parametric bin packing, since for $t \geq 2$, parameterized bin packing is a special case of standard bin packing.
We use $\mathrm{WR_{A,B}}(t)$ to denote the relative worst-order ratio of the two algorithms A and B for the parameter t. Throughout the paper, we use the notation $\langle s_1, s_2, \ldots, s_k \rangle$, $k \geq 1$, to denote a sequence of k items with sizes $s_1, s_2, \ldots, s_k$. For any sequence σ, we let σ^n denote σ repeated n times.

Theorem 1. $\mathrm{WR_{WF,AWF}}(t) = \begin{cases} 2, & \text{for } t = 1 \\ \frac{t}{t-1}, & \text{for } t \geq 2 \end{cases}$

Proof. The results of [4] imply that WF is comparable to AWF in AWF's favor.
Since the relative worst-order ratio cannot exceed the competitive ratio, the upper bound follows directly from the competitive ratio of WF, shown by Johnson [16].
For the lower bound, we use an input similar to the one of [16]. We prove the bound for $t \geq 2$; the lower bound for $t = 1$ is identical to the bound for $t = 2$. For some large integer n, let $\varepsilon = \frac{1}{t^2 n}$, and let $\sigma = (\langle \frac{1}{t} \rangle^{t-1} \langle \varepsilon \rangle)^{tn}$. WF packs these tn sets of items using tn bins. No matter how σ is ordered, AWF maintains the invariant that all ε-items are packed in the first bin and at most one additional bin is open. Thus, in total, AWF uses at most $(t-1)n + 1$ bins. For increasingly large n, this gives a ratio approaching the competitive ratio of WF [16]. □

Theorem 2. $\mathrm{WR_{AWF,FF}}(t) \geq \begin{cases} \frac{4}{3}, & \text{for } t = 1 \\ \frac{2t}{2t-1}, & \text{for } t \geq 2 \end{cases}$

Proof. The results of [4] imply that AWF is comparable to FF in FF's favor. Hence, we only need to find a family of input sequences where FF is strictly better than AWF.

We prove the lower bounds for $t \geq 2$; the lower bound for $t = 1$ is identical to the lower bound for $t = 2$. For some large integer n, let $\varepsilon = \frac{1}{2t^2 n}$, and consider the input sequence $\langle \frac{1}{t} \rangle^{t-1} (\langle \frac{1}{2t} \rangle \langle \frac{1}{t} \rangle^{t-1} \langle \varepsilon \rangle)^{tn}$.

Given in this order, for each of the later tn sets of items, AWF puts the item of size $\frac{1}{2t}$ in the latest opened bin, opens a new bin for the next $t-1$ items, and puts the ε-item in the previous bin. In total, AWF uses $nt+1$ bins. For FF, no matter how the items are ordered, all the ε-items are placed in the same bin. In total, at most one bin is not completely filled, and FF uses at most $n(t-\frac{1}{2})+1$ bins. For increasingly large n, the ratio approaches $\frac{2t}{2t-1}$ from below. □

Finally, we extend the results of [4], which are given for $t = 1$, and prove the following two theorems:

Theorem 3. $\mathrm{WR}_{\mathrm{WF,FF}}(t) = \frac{t}{t-1}$ *for* $t \geq 2$.

Proof. For the upper bound, the stated ratio matches the competitive ratio of WF [17]. For the lower bound, by the previous theorems, we have $\mathrm{WR}_{\mathrm{WF,AWF}}(t) = \frac{t}{t-1}$ and $\mathrm{WR}_{\mathrm{AWF,FF}}(t) \geq \frac{2t}{2t-1}$. Hence, by Theorem 11 in [5], we have $\mathrm{WR}_{\mathrm{WF,FF}}(t) \geq \max\{\frac{t}{t-1}, \frac{2t}{2t-1}\} = \frac{t}{t-1}$. □

Theorem 4. *For any Any-Fit algorithm* A, $\frac{2t+2}{2t+1} \leq \mathrm{WR}_{\mathrm{NF,A}}(t) \leq \frac{t}{t-1}$ *for* $t \geq 2$.

Proof. The results of [4] imply that NF is comparable to every Any-Fit algorithm A in A's favor.

For the upper bound, the stated ratio matches the competitive ratio of NF [17].

For the lower bound, consider the input sequence $\sigma = (\langle \frac{1}{t+1} \rangle^t \langle \frac{1}{t} \rangle \langle \frac{1}{t+1} \rangle^{t-1})^n$. NF will use $2n$ bins.

Now, consider the output of any Any-Fit algorithm A on an arbitrary permutation of the items. All output bins will contain a total size of $\frac{\ell}{t(t+1)}$, for some integer ℓ such that $t \leq \ell \leq t(t+1) = t^2 + t$. Let ℓ_i be the value of ℓ for the ith output bin. Since the items are of size at most $\frac{t+1}{t(t+1)}$, all bins except possibly the last have $\ell_i \geq t^2$.

Using number theory, it can be shown that, for every number $t^2 + c$ where c is an integer such that $0 \leq c \leq t-1$, there is a unique combination (x, y), such that $xt + y(t+1) = t^2 + c$, namely $(t-c, c)$. For $c = t$, there are two combinations: $(t+1, 0)$ and $(0, t)$.

Hence, for $c = 0$, the combination is $(t, 0)$, and all bins of this type contain items of size $\frac{1}{t+1}$ only. If there were two such bins, then the first item of the second would fit in the first bin. Consequently, there can be at most one bin with $\ell_i = t^2$.

Now, let n_c be the number of bins with $\ell_i = t^2 + c$, i.e., with c $\frac{1}{t}$-items and $t - c$ $\frac{1}{t+1}$-items. Since there are n items of size $\frac{1}{t} = \frac{t+1}{t(t+1)}$, we have $\sum_{c=1}^{t-1} c n_c \leq n$.

The total number of bins used by the Any-Fit algorithm is the sum of the number of bins with $\ell_i \leq t^2$, the bins with $t^2 < \ell_i < t^2 + t$, and the bins which are completely filled with either only

$\frac{1}{t}$ or $\frac{1}{t+1}$ items. Thus,

$$\begin{aligned}
\mathrm{A}(\sigma) \leq \quad & 2+\sum_{c=1}^{t-1} n_c + \left\lceil \frac{n-\sum_{c=1}^{t-1} cn_c}{t} \right\rceil + \left\lceil \frac{(2t-1)n-\sum_{c=1}^{t-1}(t-c)n_c}{t+1} \right\rceil \\
& \leq 4+\sum_{c=1}^{t-1} n_c + \frac{n-\sum_{c=1}^{t-1} cn_c}{t} + \frac{(2t-1)n-\sum_{c=1}^{t-1}(t-c)n_c}{t+1} \\
& = 4+\frac{(2t^2+1)n+t\sum_{c=1}^{t-1} n_c - \sum_{c=1}^{t-1} cn_c}{t^2+t} \\
& \leq 4+\frac{(2t^2+1)n+t\sum_{c=1}^{t-1} cn_c - \sum_{c=1}^{t-1} cn_c}{t^2+t} \\
& \leq 4+\frac{(2t^2+1)n+(t-1)n}{t^2+t} = 4+\frac{2t^2+t}{t^2+t}n = 4+\frac{2t+1}{t+1}n
\end{aligned}$$

In the last two inequalities, we use $\sum_{c=1}^{t-1} n_c \leq \sum_{c=1}^{t-1} cn_c$ and $\sum_{c=1}^{t-1} cn_c \leq n$. Comparing to the cost of NF which is $2n$ we get the required bound. □

4 Harmonic algorithms

In this section, we study the family of Harmonic algorithms, Harmonic$_i$ (H_i), for three variants of bin packing. For all three variants, we find that H_j is better than H_i for $i < j$. For bin covering and open-end bin packing, this contrasts with the competitive ratio, which is 2 for all Harmonic algorithms. For standard bin packing, the relative worst-order ratio agrees with the competitive ratio in the sense that the competitive ratio also prefers H_j over H_i when $1 < i < j$. However, with the relative worst-order ratio, the difference between H_j and H_i is more pronounced, especially for H_1 and H_2 which are not distinguished by the competitive ratio.

4.1 Bin packing

For standard bin packing the family of algorithms H_i, designed by Lee and Lee [21], is defined as follows. For $i \geq 1$, H_i divides the items into i classes. For $1 \leq j < i$, class j consists of all the items $\sigma[k]$ of size $\frac{1}{j+1} < \sigma[k] \leq \frac{1}{j}$. Class i consists of items of size at most $\frac{1}{i}$.
When receiving items, H_i does not mix items from different classes. It keeps exactly one open bin for each class, and assigns an item to the open bin of its class, if possible. If the item is too large, a new bin is opened for the class, and the previous one is closed. This means that every closed bin for class $j < i$ contains j items, and the items of class i are packed using NF.
H_1 corresponds to NF, all items are in the same class, and there is a single active bin at any time. It was shown in [21] that as i tends to infinity, the competitive ratio of these algorithms tends to 1.691. The competitive ratios of H_1 and H_2 are 2, but the competitive ratios of all other algorithms in the class are no larger than 1.75. Thus, it seems that using an algorithm H_i of relatively large i does not give much advantage over using H_3.
In the following, we use the relative worst-order ratio to compare pairs of harmonic algorithms, and show that the algorithm with the larger index is the better algorithm.

Lemma 1. *For any input sequence σ and $i < j$, $\mathrm{H}_{i\mathrm{W}}(\sigma) \geq \mathrm{H}_{j\mathrm{W}}(\sigma) - j + i - 1$, and thus H_i is comparable to H_j.*

Proof. Consider any input sequence σ and the packing of σ done by H_j on its worst ordering of the sequence. We construct a permutation on the items by reordering the bins as follows: First, all closed bins occur in decreasing order of the class indexes. Next, all the open bins follow in some order. Now the items are given to H_i in the order they appear in the bins.
Note that each closed class k bin has an empty space of less than $\frac{1}{k+1}$. Hence, for $\ell \leq k$, no class ℓ item fits in a closed class k bin. It follows that H_i packs all closed bins identically to H_j. Moreover, H_i packs all open bins of class k, $1 \leq k \leq i-1$ identically to H_j. Thus, H_i uses at least as many bins as H_j, except for at most $j-i+1$ open bins of classes $i, i+1, \ldots, j$. □

It turns out that the relative worst-order ratio of $\textsc{Harmonic}_i$ to $\textsc{Harmonic}_j$, $j > i$, is determined by sequences that consist solely of items that fall in class i when handled by $\textsc{Harmonic}_i$, and using just one more class, the bins would be filled completely. Hence, the ratio depends only on i:

Theorem 5. *For any $i < j$,*

$$\mathrm{WR}_{\mathrm{H}_i,\mathrm{H}_j} = \begin{cases} 2, & \text{for } i = 1 \\ \frac{i}{i-1}, & \text{for } i \geq 2 \end{cases}$$

Proof. For the upper bound, consider any input sequence σ. The number of bins containing items of any class $k < i$ is the same for the two algorithms, regardless of the ordering of the input. Such bins may only decrease the relative worst-order ratio, and we can therefore assume that there are no such items. In the case $i = 1$, the upper bound follows from the competitive ratio of NF. Consider the case $i \geq 2$. Every closed bin packed by H_i containing items of class i, contains a total size of items of at least $\frac{i-1}{i}$. This implies the upper bound.
To prove a lower bound for $i = 1$, we use the sequence of [4], $\sigma = \langle 1, \frac{1}{n} \rangle^n$, for some large integer n. For this ordering, H_1 packs the items in $2n$ bins, whereas H_j, for any $j \geq 2$, uses $n+1$ bins. If $i \geq 2$, consider the sequence $\sigma = (\langle \frac{1}{i} \rangle^{i-1} \langle \frac{1}{n} \rangle)^{in}$, for some large integer n. Using this ordering, H_i uses in bins, whereas H_j, for any $j \geq i+1$, packs the $\frac{1}{n}$-items separately from the $\frac{1}{i}$-items and thus uses only $n(i-1)+i$ bins. □

4.2 Bin covering

The bin covering problem [1,11,8,15] is in some sense the opposite of the bin packing problem as we now want to maximize the number of bins containing items of a total size of at least one. The Harmonic algorithms for this problem are the same as for standard bin packing, except that a bin is closed only when the total size of the items in the bin reaches at least one.
Assmann et al [1] studied the algorithm Next-Cover (NC) which assigns items to one bin until it is covered, and only then moves to the next bin. This algorithm has a competitive ratio of 2. Csirik and Totik [11] showed that this is the best possible competitive ratio for the problem. The same competitive ratio can be achieved by the Harmonic algorithms.
H_1 corresponds to NC: All items are in the same class, and the bins are simply filled one by one. For $j < i$, H_i uses at most $j+1$ items of class j to cover a bin of that class, and applies NC to class i. Naturally, for any $i \geq 1$, H_i has a competitive ratio of 2 as well. This ratio follows from the fact that covered bins may have items of a total size of almost 2. However, it seems that even though such cases can occur for H_i, still it only happens in bins containing relatively large items. Below we find, unsurprisingly, that H_i is increasingly better for increasing i.

Lemma 2. *For any input sequence σ and $i < j$, $\mathrm{H}_{i\mathrm{W}}(\sigma) \leq \mathrm{H}_{j\mathrm{W}}(\sigma) + j - i + 1$, and thus H_i is comparable to H_j.*

Proof. Consider any input sequence σ and the packing of σ done by H_j. At most $j-i+1$ bins containing items of classes i to j are not closed. Move these bins to the end, and give all items to H_i in the order they appear in the bins. With this permutation of σ, H_i packs all the items in the same way as H_j except possibly for $j-i+1$ bins. □

Theorem 6. *For any $i < j$, $\mathrm{WR}_{\mathrm{H}_i,\mathrm{H}_j} = \frac{i+1}{i}$.*

Proof. The upper bound is similar to the one for standard bin packing. The number of bins covered using items of classes $1, 2, \ldots, i-1$ is equal for both algorithms on any permutation of the input. Thus we can assume that all items have a size of at most $\frac{1}{i}$. No bin receives a total size of more than $1 + \frac{1}{i}$, and the upper bound follows.
For the lower bound, consider the input sequence $\sigma = (\langle\frac{1}{n}\rangle^{n-1}\langle\frac{1}{i}\rangle)^{in}$ for some large integer n. For H_i, the items of sizes $\frac{1}{n}$ and $\frac{1}{i}$ belong to the same class, and using the above ordering H_i fills each bin to $\frac{n-1}{n} + \frac{1}{i}$ and covers in bins in total. For H_j, the items of the two sizes belong to two different classes, hence, no matter how σ is reordered, H_j only just fills all of its bins completely, yielding $n + i(n-1)$ covered bins. As n tends to infinity, this gives a ratio of $\frac{i+1}{i}$. □

4.3 Open-end bin packing

In the open-end bin packing problem, first described by Leung and Yang [24], the total size of items in a bin may exceed 1, as in bin covering. However, unlike bin covering, it is a minimization problem: The number of bins used should be minimized under the restriction that an item can be packed in a bin, only if the total size of the items already there is strictly less than 1.
For this problem, the HARMONIC algorithms partition the item sizes slightly differently: For H_i and $0 < j < i$, class j contains the items of sizes in $[\frac{1}{j+1}, \frac{1}{j})$. Class i contains items of sizes in $(0, \frac{1}{i})$, and class 0 contains items of size 1. The algorithm packs the items of class 0 into separate bins. Thus, a closed bin of class j, $0 < j < i$, contains $j+1$ items.
All the algorithms H_i have a competitive ratio of 2. However, using the relative worst-order ratio we show that the algorithms with larger indices are better.
A greedy algorithm NF can be defined again as an algorithm which uses a single active bin, and packs items into this bin as long as it is possible.
Note that in this case, NF is not exactly the same algorithm as H_1. However, we can prove that they are equivalent in terms of the relative worst-order ratio. Also, note that an item of size one is always the last item in its bin in any packing.

Lemma 3. *For any input sequence σ, $\mathrm{H}_{1\mathrm{W}}(\sigma) = \mathrm{NF}_{\mathrm{W}}(\sigma)$.*

Proof. Consider any input sequence σ. We claim that for NF there exists a worst permutation of σ in which all items of size one appear in the beginning. Among the worst permutations of σ for NF, consider a permutation σ' with a maximum number of items of size one in the beginning. Assume by contradiction that there is an item of size one which does not appear in the beginning. Move this item to the beginning and the items packed together with it (if they exist) to the end of the sequence. As a result, the item of size one is packed in a bin alone so the number of bins did not decrease, which is a contradiction.
The same is clearly true for H_1 (that the items of size one appear in the beginning of some worst permutation), since the exact location of these items is not important.
For a permutation with the items of size one appearing in the beginning, the packing of NF and H_1 are identical. This proves the lemma. □

Lemma 4. *For any input sequence σ and $i < j$, $\mathrm{H}_{i\mathrm{W}}(\sigma) \geq \mathrm{H}_{j\mathrm{W}}(\sigma) - j + i - 1$, and thus H_i is comparable to H_j.*

Proof. Consider any input sequence σ and the packing of σ done by H_j. At most $j - i + 1$ bins containing items of classes i to j are not closed.
Now, move these bins to the end, and give the items to H_i in the order they appear in the bins. With this permutation of σ, H_i packs all the items in the same way as H_j except possibly for $j - i + 1$ bins. □

Theorem 7. *For any $i < j$, $\mathrm{WR}_{\mathrm{H}_i,\mathrm{H}_j} = \frac{i+1}{i}$.*

Proof. The upper bound is similar to the one for standard bin packing. The number of bins packed using items of classes $1, 2, \ldots, i-1$ is equal for both algorithms and for any permutation on the input. Thus, we can assume that all items have a size of less than $\frac{1}{i}$. Consequently, a bin can only receive a total size of less than $1 + \frac{1}{i}$, and the upper bound follows.
For the lower bound, consider the input sequence $\sigma = \left(\langle\frac{1}{n}\rangle^i\langle\frac{1}{i} - \frac{1}{n}\rangle^i\right)^{(i+1)n}$ for some large integer n. For H_i, all items belong to the same class, and using the above ordering H_i packs the items of each subsequence into one bin using $(i+1)n$ bins in total. For H_j, the items of the two sizes belong to two different classes, hence, no matter how σ is reordered, H_j packs $i+1$ larger items in each bin of that class. The number of bins is therefore at most $in + i(i+1)$. As n tends to infinity, this gives a ratio of $\frac{i+1}{i}$. □

5 Bin coloring

In the bin coloring problem first defined by Krumke et al [20], we have q bins each capable of holding B items. The input sequence consists of items with only one property: their color. The items must be packed in bins. Each time a bin has been filled a new empty bin takes its place. The object is to minimize the maximum number of colors in any bin. The problem is motivated by a real life vehicle packing problem.
The paper [20] defines two algorithms for the problem: One-Bin (OB) simply fills the bins one by one, whereas Greedy-Fit (GF) places each item in a bin already containing an item of the same color, if possible, and otherwise in any bin currently containing the least number of colors. Intuitively, GF should be the better algorithm, but according to the (strict) competitive ratio, OB is strictly better: OB has a strict competitive ratio of at most $(2q-1)$ and GF has a strict competitive ratio of at least $2q$ [20]. We show that, according to the strict relative worst-order ratio, GF is better than OB. Interestingly, a related result was given by Hiller and Vredeveld [15] who showed that GF is better than OB, using probabilistic analysis.

Definition 7. *For any input sequence σ, define $C(\sigma)$ to be the number of different colors occurring in σ.*

Lemma 5. *For any algorithm* Alg *and for any input sequence σ,*

$$\min\left\{B, \left\lceil\frac{C(\sigma)}{q}\right\rceil\right\} \leq \mathrm{Alg}_{\mathrm{W}}(\sigma) \leq \min\{B, C(\sigma)\}.$$

Proof. The upper bound of $\min\{B, C(\sigma)\}$ follows immediately by the definition of the cost function of the problem.
For the lower bound, let Alg be any algorithm and σ be any input sequence for the problem. Reorder σ such that the first $C(\sigma)$ items all have different colors. Consider the packing of these first $C(\sigma)$ items. If the algorithm never closes a bin while processing these items, that is, they are packed into at most q bins, then by the pigeon hole principle, at least $\lceil C(\sigma)/q\rceil$ items are packed into a common bin. Otherwise, at least one bin is closed, and this bin contains B items of distinct colors.

Theorem 8. *For any input sequence σ,* $\mathrm{OB}_{\mathrm{W}}(\sigma) = \min\{B, C(\sigma)\}$.

Proof. Let σ be any input sequence. Reorder σ such that the first $C(\sigma)$ items all have different colors. This immediately yields the result. □

Theorem 9. *For any input sequence σ,* $\mathrm{GF}_{\mathrm{W}}(\sigma) = \min\left\{B, \lceil\frac{C(\sigma)}{q}\rceil\right\}$.

Proof. Let σ be any input sequence. By the definition of GF, items are always placed in a bin with same-colored items or in a bin with the least number of current colors.
No bin can contain items of more than B colors, therefore, if $B \leq \lceil C(\sigma)/q\rceil$, then we are done. Otherwise, by the pigeon hole principle, if GF needs to pack an item with a color not currently used in any of its open bins, it always has at least one bin with at most $\lfloor(C(\sigma)-1)/q\rfloor < \lceil C(\sigma)/q\rceil$ different colors, which can accommodate the item. □

By the above theorems, OB and GF are worst, respectively best, possible algorithms for the Bin Coloring Problem.

Theorem 10. $\mathrm{SWR}_{\mathrm{OB,GF}} = \min\{q, B\}$.

Proof. By Lemma 5 and Theorem 8, GF is always at least as good as OB on any input sequence. Hence, we only need to find a familiy of input sequences that maximizes the ratio between the cost of OB and GF.
Let σ_i be any input sequence with i different colors. By Lemma 8 and Theorem 9, we have $\mathrm{OB_W}(\sigma_i) = \min\{B, i\}$, and $\mathrm{GF_W}(\sigma_i) = \min\{B, \lceil i/q \rceil\}$. We need to find the worst possible i,

$$\max_{i\in\mathbb{N}} \left(\frac{\mathrm{OB_W}(\sigma_i)}{\mathrm{GF_W}(\sigma_i)} \right) = \max_{i\in\mathbb{N}} \left(\frac{\min\{B, i\}}{\min\{B, \lceil i/q \rceil\}} \right)$$

We can choose the value $i = q$ to obtain the lower bound $\min\{q, B\}$. Clearly, the worst possible values are obtained for i being multiples of q. Let $i = pq$ for some positive integer p. Then if $B \leq p$ we get the ratio 1, if $p < B \leq pq$, we get the ratio $\frac{B}{p}$ (and in this case $\frac{B}{p} \leq q$, $\frac{B}{p} \leq B$), and otherwise, we get the ratio q (and in this case $q < \frac{B}{p} \leq B$). □

6 Class-constrained packing

In the online class-constrained packing problem first described by Shachnai and Tamir [22,23], we again have to pack items in bins. Items have unit size and are each equipped with a color. Bins can contain up to at most v items (capacity constraint) and up to at most c different colors (class constraint). The object is to pack the items using as few bins as possible. Without loss of generality, in the following we only consider the case when $1 < c < v$, otherwise the problem is trivial.
Two 2-competitive algorithms were presented in [22]:

First-Fit (FF): An item is packed into the first bin that has less than v items, where packing it would not violate the class constraints.

ColorSets (CS): Denote the colors by $0, 1, 2, \ldots, K$ in the order of their first appearance in the input sequence.
Partition the colors into sets containing at most c colors such that the tth set, for $t \geq 0$, contains the colors $\{tc, tc+1, \ldots, tc+c-1\}$. CS packs each color set in separate bins.

Even though the two algorithms have the same competitive ratio, it seems obvious that they work well in different situations. This is reflected in the relative worst-order analysis: they are not comparable. More precisely, there are situations where one algorithm is essentially twice as good as the other and situations where it is the other way around.

Lemma 6. *For any $\varepsilon > 0$, there exist values of v and c, and an arbitrarily long input sequence σ, such that* $\mathrm{CS_W}(\sigma) \geq (2-\varepsilon)\,\mathrm{FF_W}(\sigma)$.

Proof. In the following we only consider the case when $v = ck - 1$, for an integer $k > 1$. For some large integer n, consider the input sequence σ consisting of cn colors each occurring k times, that is, the sequence contains kcn items in total.
No matter in which order the items are given, CS will produce n sets of colors, where the number of items in each set of colors is $ck = v + 1$, so two bins are used for each set, and $2n$ bins in total.
For FF, we need to consider all packing options:
Every bin which is full must contain v items. Since there exist only k items of each color, a bin which contains items of at most $c-1$ colors contains less than v items. Thus, a full bin contains items of exactly c colors, and therefore it must have exactly $c-1$ colors with k items and one color with $k-1$ items.
Every bin which is not full (except for possibly the last one) also contains items of c colors, since otherwise there is no need of opening an additional bin. Note that no color has items in more than two bins. If the first bin containing items of this color is full, then at most one item is left

for future packing. Otherwise, if already the first bin is not full, there is only one bin used for this color.
We assign costs to colors as follows. A color with k items in a full bin gets a cost of $\frac{c+1}{c^2}$. A color with $k-1$ items in a full bin gets a cost of $\frac{1}{c^2}$. A color with at least one item in a non-full bin (which is not the last bin) gets a cost of $\frac{1}{c}$. Note that this exactly covers the cost of all bins but the last, since $\frac{c+1}{c^2}(c-1)+\frac{1}{c^2}=1$, and $\frac{1}{c}\,c=1$.
Note also that a color may be assigned more than one cost. Specifically, a color with all items in one bin gets a cost of at most $\frac{c+1}{c^2}$, and a color with two bins gets a cost of $\frac{1}{c^2}+\frac{1}{c}=\frac{c+1}{c^2}$. Thus the total number of bins used by FF is at most the sum of costs of colors plus 1, which is at most $cn\frac{c+1}{c^2}+1=\frac{n(c+1)}{c}+1$.
For n approaching infinity, the ratio approaches $\frac{2c}{c+1}$ from below. This is close to 2 for large c. □

Lemma 7. *For any $\varepsilon>0$, there exist values of v and c, and an arbitrarily long input sequence σ, such that* $\mathrm{FF_W}(\sigma)\geq(2-\varepsilon)\,\mathrm{CS_W}(\sigma)$.

Proof. We consider values of c which divide v, and let $k=\frac{v}{c}$. For some large integer n, consider an input sequence consisting of items using $(c-1)cn+1$ different colors: Color 0 has $(k+c-1)cn$ items, and the remaining $(c-1)cn$ colors each has k items.
A possible bad ordering for FF is the following, where items are given in $cn+1$ phases: In Phase i, $0\leq i\leq cn-1$, $k+c-1$ items of color 0 are given together with $k-1$ items of each of the colors $i+1,i+2,\ldots,i+c-1$. In the last phase, one item of each color $1,2,\ldots,(c-1)cn$ is given.
FF uses one bin for each of the first cn phases. For the last phase, it uses $(c-1)n$ bins. This gives a total of $2cn-n$ bins.
For CS, independently of the ordering, almost all bins are completely filled. The color set containing color 0 may have a partially empty bin, and the last color set (which contains only $c-1$ colors) may have a partially empty bin. In total, the input sequence contains $c(c-1)n+cvn$ items, and CS uses at most $c(c-1)n/v+cn+2$ bins.
For any n, the ratio of the two is

$$\frac{2c-1}{\frac{c}{v}(c-1)+c+\frac{2}{n}}\,.$$

As $\frac{v}{c}$ and n tend to infinity, this tends to $2-\frac{1}{c}$. □

The previous two lemmas immediately imply the following theorem:

Theorem 11. FF *and* CS *are not comparable using the relative worst-order ratio.*

References

1. S. F. Assman, D. S. Johnson, D. J. Kleitman, and J. Y.-T. Leung. On a dual version of the one-dimensional bin packing problem. *Journal of Algorithms*, 5(4):502–525, 1984.
2. S. Ben-David and A. Borodin. A new measure for the study of on-line algorithms. *Algorithmica*, 11(1):73–91, 1994.
3. J. Boyar, M. R. Ehmsen, and K. S. Larsen. Theoretical evidence for the superiority of LRU-2 over LRU for the paging problem. In *Approximation and Online Algorithms*, pages 95–107, 2006.
4. J. Boyar and L. M. Favrholdt. The relative worst order ratio for online algorithms. *ACM Transactions on Algorithms*, 3(2):22, 2007.
5. J. Boyar, L. M. Favrholdt, and K. S. Larsen. The relative worst order ratio applied to paging. *Journal of Computer and System Sciences*, 73(5):818–843, 2007.
6. J. Boyar and P. Medvedev. The relative worst order ratio applied to seat reservation. *ACM Transactions on Algorithms*, 4(4):article 48, 22 pages, 2008.
7. E. G. Coffman Jr. and J. Csirik. Performance guarantees for one-dimensional bin packing. In T. F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*, chapter 32. Chapman & Hall/Crc, 2007. 18 pages.

8. J. Csirik, J. B. G. Frenk, M. Labbé, and S. Zhang. Two simple algorithms for bincovering. *Acta Cybernetica*, 14(1):13–25, 1999.
9. J. Csirik and J. Y.-T. Leung. Variable-sized bin packing and bin covering. In T. F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*, chapter 34. Chapman & Hall/Crc, 2007. 11 pages.
10. J. Csirik and J. Y.-T. Leung. Variants of classical one-dimensional bin packing. In T. F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*, chapter 33. Chapman & Hall/Crc, 2007. 13 pages.
11. J. Csirik and V. Totik. On-line algorithms for a dual version of bin packing. *Discrete Applied Mathematics*, 21:163–167, 1988.
12. M. R. Ehmsen, L. M. Favrholdt, J. S. Kohrt, and R. Mihai. Comparing First-Fit and Next-Fit for online edge coloring. In *19th International Symposium on Algorithms and Computation*, pages 89–99, 2008.
13. L. Epstein, L. M. Favrholdt, and J. S. Kohrt. Separating online scheduling algorithms with the relative worst order ratio. *Journal of Combinatorial Optimization*, 12(4):362–385, 2006.
14. R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell Systems Technical Journal*, 45:1563–1581, 1966.
15. B. Hiller and T. Vredeveld. Probabilistic analysis of online bin coloring algorithms via stochastic comparison. In *Proceedings of the 16th Annual European Symposium*, pages 528–539, 2008.
16. D. S. Johnson. Fast algorithms for bin packing. *Journal of Computer and System Sciences*, 8(3):272–314, 1974.
17. 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 Journal on Computing*, 3(4):299–325, 1974.
18. A. R. Karlin, M. S. Manasse, L. Rudolph, and D. D. Sleator. Competitive snoopy caching. *Algorithmica*, 3(1):79–119, 1988.
19. C. Kenyon. Best-fit bin-packing with random order. In *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 359–364, 1996.
20. S. O. Krumke, W. E. de Paepe, J. Rambau, and L. Stougie. Bincoloring. *Theoretical Computer Science*, 407(1-3):231–241, 2008.
21. C. C. Lee and D. T. Lee. A simple online bin packing algorithm. *Journal of the ACM*, 32(3):562–572, 1985.
22. H. Shachnai and T. Tamir. Tight bounds for online class-constrained packing. *Theoretical Comptuer Science*, 321:103–123, 2004.
23. E. C. Xavier and F. K. Miyazawa. The class constrained bin packing problem with applications to video-on-demand. *Theoretical Computer Science*, 393(1-3):240–259, 2008.
24. J. Yang and J. Y.-T. Leung. The ordered open-end bin packing problem. *Operations Research*, 51(5):759–770, 2003.