Theoretical Computer Science 556 (2014) 71–84

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Online bin covering: Expectations vs. guarantees ☆

Marie G. Christ, Lene M. Favrholdt, Kim S. Larsen *

Department of Mathematics and Computer Science, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark

ARTICLE INFO

Article history:
Received 28 February 2014
Received in revised form 12 May 2014
Accepted 18 June 2014
Available online 25 June 2014

Keywords:
Online algorithms
Bin covering
Performance measures
Competitive analysis

ABSTRACT

Bin covering is a dual version of classic bin packing. Thus, the goal is to cover as many bins as possible, where covering a bin means packing items of total size at least one in the bin. For online bin covering, competitive analysis fails to distinguish between most algorithms of interest; all "reasonable" algorithms have a competitive ratio of $\frac{1}{2}$. Thus, in order to get a better understanding of the combinatorial difficulties in solving this problem, we turn to other performance measures, namely relative worst order, random order, and max/max analysis, as well as analyzing input with restricted or uniformly distributed item sizes. In this way, our study also supplements the ongoing systematic studies of the relative strengths of various performance measures.

Two classic algorithms for online bin packing that have natural dual versions are Harmonic$_k$ and Next-Fit. Even though the algorithms are quite different in nature, the dual versions are not separated by competitive analysis. We make the case that when guarantees are needed, even under restricted input sequences, dual Harmonic$_k$ is preferable. In addition, we establish quite robust theoretical results showing that if items come from a uniform distribution or even if just the ordering of items is uniformly random, then dual Next-Fit is the right choice.

1. Introduction

Bin covering [2] is a dual version of classic bin packing. As usual, bins have size one and items with sizes between zero and one must be packed. However, in bin covering, the objective is to cover as many bins as possible, where a bin is covered if the sizes of items placed in the bin sum up to at least one. We are considering the online version of bin covering. A problem is online if the input sequence is presented to the algorithm one item at a time, and the algorithm must make an irrevocable decision regarding the current item without knowledge of future items.

Bin covering algorithms have numerous important applications. For instance, when packing or canning food items guaranteeing a minimum weight or volume, reductions in the overpacking of even a few percent may have a large economic impact. If items arrive on a conveyor belt, for instance, the problem becomes online.

Classic algorithms for online bin packing are Next-Fit and the parameterized family Harmonic$_k$ [21]. Next-Fit is a very simple and natural algorithm, and Harmonic$_k$ was designed to obtain a competitive ratio [19,24] better than any Any-Fit algorithm (First-Fit and Best-Fit are examples of Any-Fit algorithms for bin packing, and the competitive ratio of Next-Fit is worse than both these algorithms). Harmonic$_k$ and variations of it have been analyzed extensively [22,23,25].

☆ A preliminary version of this paper appeared in the proceedings of the Seventh Annual International Conference on Combinatorial Optimization and Applications, 2013. Supported in part by the Danish Council for Independent Research and the Villum Foundation.

* Corresponding author.
E-mail addresses: christm@imada.sdu.dk (M.G. Christ), lenem@imada.sdu.dk (L.M. Favrholdt), kslarsen@imada.sdu.dk (K.S. Larsen).

http://dx.doi.org/10.1016/j.tcs.2014.06.029
0304-3975/

We consider the obvious dual version of these, DNF [2] and DH_k [12]. These algorithms are quite different in nature and the bin packing versions are clearly separated, having competitive ratios of 2 and approximately 1.691, respectively. However, for bin covering, competitive analysis does not distinguish between them! In fact, for bin covering, competitive analysis categorizes both algorithms as being optimal among deterministic algorithms, but also worst possible among "reasonable" algorithms for the problem. This is unlike the situation in bin packing, and in general, results from bin packing do not transfer directly to bin covering.

To understand the algorithmic differences better, it is therefore necessary to employ different techniques, and we turn to other generally applicable performance measures, namely relative worst order analysis, random order analysis, and max/max analysis. As for almost all performance measures, the idea is to abstract away some details of the problem to enable comparisons. Without some abstraction, it is hard to ever, analytically, claim that one algorithm is better than another, since almost any algorithm performs better than any other algorithm on at least one input sequence. For all the measures considered here, the abstraction can be viewed as being defined via first a partitioning of the set of input sequences of a given length and then an aggregation of the results from each partition. For each sequence length, competitive analysis, for instance, considers all the ratios of the online performance to the optimal offline performance obtained for each sequence of that length, and then takes the worst ratio of all of these. The measures above employ a less fine-grained partitioning of the input space. Worst order and random order analysis group permutations of the same sequence together instead of considering each sequence separately, deriving worst-case or average-case performance, respectively, within each partition. With max/max analysis the partitioning of the input space is even coarser: for each sequence length n, the online worst-case behavior over all sequences of length n is compared to the worst-case optimal offline behavior over all sequences of length n. There is no one correct way to compare algorithms, but since these measures focus on different aspects of algorithmic behavior, considering all of the ones above lead to a very broad analysis of the problem. Extensive motivational sections can be found in the papers introducing these measures and in the survey [13]. As a further supplement, we analyze restricted input sequences, where items have similar size, which is likely to happen in practice if one is packing products with an origin in nature, for instance. Finally, we consider input sequences containing items having uniformly distributed sizes.

Relative worst order analysis [4,5] has been applied to many problems; a recent list can be found in [15]. In [16], bin covering was analyzed, but using a version of the problem allowing items of size 1. We analyze the more commonly studied version for bin covering, where all items are strictly smaller than 1. Since worst-case sequences from [16] contain items of size 1, this leads to slightly different results. For completeness, we include these results. Random order analysis [20] was introduced for classic bin packing, but has also been used for other problems; a server problem, for instance [8]. Max/max analysis [3] was introduced as an early step towards refining the results from competitive analysis for paging and a server problem.

Relative worst order analysis emphasizes the fact that there exist multisets of input items where DNF can perform $\frac{3}{2}$ times as poorly as DH_k. On the other hand, DH_k's method of limiting the worst-case also means that it has less of an opportunity to reach the best case, as opposed to DNF. This is reflected in the random order analysis, where DNF comes out at least as well as DH_k. Another way of approaching randomness is to analyze a uniform distribution. We establish new results on DH_k showing that its performance here is slightly worse than that of DNF, in line with the random order results. With the max/max analysis, a distinction between the two algorithms can only be achieved, when the item sizes are limited, and DH_k is the algorithm selected as best by this measure. With respect to competitive analysis, we also consider restricted input in the sense that item sizes may only vary across one or two consecutive DH_k partitioning points. This is a formal way of treating the case where items are of similar size, while allowing greater variation when this size is large. We show that with this restricted form of input, considering the worst-case measure of competitive analysis, DH_k is deemed better than DNF, as DNF is more vulnerable to worst-case sequences.

This study also contributes to the ongoing systematic studies of the relative strengths of various performance measures, initiated in [8]. Up until that paper, most performance measures were introduced for a specific problem to overcome the limitations of competitive analysis. In [8], comparisons of performance measures different from competitive analysis were initiated, and this line of work has been continued in [6,7,9], among others. Our results supplement results in [11], showing that no deterministic algorithm for the bin covering problem can be better than $\frac{1}{2}$-competitive and giving an asymptotically optimal algorithm for the case of items being uniformly distributed on $(0, 1)$. For DNF, [10] established an expected competitive ratio of $\frac{2}{e}$ under the same conditions.

In the following, we formally define the bin covering problem and the algorithms DNF and DH_k, the performance of which we compare under different performance measures. The performance measures themselves are defined in each their section. We conclude on our findings in the final section.

1.1. Bin covering

In the one dimensional bin covering problem, the algorithm gets an input sequence $I = \langle i_1, i_2, \ldots \rangle$ of item sizes, where for all j, $0 < i_j < 1$. The items are to be packed in bins of size 1. A bin is *covered*, if items of total size at least 1 have been packed in it, and the goal is to cover as many bins as possible.

Requiring items to be strictly smaller than 1 corresponds to assuming that items of size 1 are treated separately. This makes sense, since there is no advantage in combining an item of size 1 with any other items in a bin. In other words, any algorithm not giving special treatment to items of size 1 could trivially be improved by doing so.

Worst-case	Competitive analysis	$DH_k = DH_2 = DNF$
	for some (a, b) restrictions	$DH_k > DH_2 > DNF$
	Max/max analysis	$DH_k = DH_2 = DNF$
	for some (a, b) restrictions	$DH_k = DH_2 > DNF$
	Relative worst order analysis	$DH_k > DH_2 > DNF$
Expected	Uniform input distribution	$DH_k < DH_2 < DNF$
	Random order analysis	$DH_k = DH_2 \leq DNF$

Fig. 1. This is an illustration of the results obtained. The definition of the different measures can be found in the individual sections. The relation ">" should be read as "better than". In DH_k, k should be read as some sufficiently large $k > 2$. Finally, (a, b) indicates a restriction on the item sizes to an interval smaller than $(0, 1)$.

For a bin covering algorithm A, we let $A(I)$ denote the number of covered bins when given the sequence I of items. We let Opt denote an optimal offline algorithm. Thus, $Opt(I)$ is the largest number of bins that can be covered by any algorithm processing I.

In algorithms for bin packing and covering, it is standard to use the following terminology. A bin that has received at least one item is *open* if it may receive more items, and *closed* if the algorithm will not consider that bin again for future items.

1.1.1. The Dual Next-Fit algorithm

Assmann, Johnson, Kleitman, and Leung [2] introduced the Dual Next-Fit algorithm (DNF), an adaptation of the Next-Fit algorithm for bin packing. DNF always keeps at most one open bin. When a new item arrives, it is packed in the currently open bin, if any. Otherwise, a new bin is opened. A bin is closed when it has received items of total size at least one.

1.1.2. The Dual Harmonic algorithm

The algorithm Harmonic$_k$ was introduced for bin packing by Lee and Lee [21]. This algorithm partitions the interval $(0, 1)$ into k subintervals, with the partitioning points at $\frac{1}{2}, \frac{1}{3}, \ldots, \frac{1}{k}$, resulting in the different sized intervals $(0, \frac{1}{k}], (\frac{1}{k}, \frac{1}{k-1}], \ldots, (\frac{1}{2}, 1)$. Harmonic$_k$ packs items from each of these k subintervals in separate bins. This means that each closed bin for the interval $(\frac{1}{j}, \frac{1}{j-1}]$ contains exactly $j - 1$ items. The natural adaptation to the bin covering problem is to use the intervals

$$\left(0, \frac{1}{k}\right), \left[\frac{1}{k}, \frac{1}{k-1}\right), \ldots, \left[\frac{1}{2}, 1\right).$$

The resulting algorithm, DHarmonic$_k$ (DH_k) [12], uses exactly j items from the interval $[\frac{1}{j}, \frac{1}{j-1})$ to cover a bin. Note that if the input consists only of items in the interval $(0, \frac{1}{k})$, then DH_k and DNF behave identically. This also means that we assume that $k \geq 2$ throughout the paper, since for $k = 1$, DH_k becomes DNF.

Fig. 1 gives an illustration of our results. We refer to the individual sections for definitions of the different measures.

2. Competitive analysis

In competitive analysis [19,24], the performance of an online algorithm is compared to that of an optimal offline algorithm Opt. An algorithm A for a maximization problem is called *c-competitive* if there exists a fixed constant b such that for any input sequence I, it holds that $A(I) \geq c\,Opt(I) + b$. The supremum over all such c is the *competitive ratio* CR(A) of A. Note that some authors reverse the order of the algorithm and Opt to get ratios larger than one.

For bin covering, Csirik and Totik [11] showed that no deterministic online algorithm can be better than $\frac{1}{2}$-competitive. DNF was shown to be $\frac{1}{2}$-competitive in [2], and the same result for DH_k was noted in [16]. For completeness, to show that this result is tight for a large class of algorithms, we define a *reasonable* algorithm to be one that closes bins as soon as they are covered, does not close bins before they are covered, and does not have more than a constant number of open bins at any point.

Theorem 1. *Any deterministic reasonable algorithm has a competitive ratio of* $\frac{1}{2}$.

Proof. The upper bound follows from [11]. For the lower bound, note that the only item that can overfill a bin is the last item to go into that bin, by the definition of a reasonable algorithm. Since that item has size less than one, all bins will contain items of total size less than two. Thus, Opt could not cover more than twice as many bins, using items from the closed bins. Being reasonable also means that there are only a constant number of open bins, so the items in there can only enable Opt to cover an additive constant of further bins. Thus, no reasonable algorithm can be worse than $\frac{1}{2}$-competitive. □

2.1. Limiting the item sizes

In some applications of the bin covering problem it is likely that the sizes of the items contained in an input sequence differ only slightly, e.g., packing similar food items into a container, guaranteeing the consumer a minimum weight. As an

example, consider selling boxes of 1 kg of chicken thighs at a fixed price. The consumer must be guaranteed that there is at least 1 kg and the producer wants to produce (cover) as many boxes as possibly, so this is a bin covering problem. To derive a guarantee using competitive analysis, we get useless results from pathological sequences containing items with chicken thighs of weight ε kg or $1-\varepsilon$ kg. Since some algorithms are better for some types of sequences than others, in order to get useful results, we should restrict item sizes to some reasonable interval such as 80–120 g, depending on the type of chicken.

In the following, we investigate the performance of DNF and DH_k on sequences with similar-sized items. Since it seems reasonable to allow larger variance in size when the considered sizes are large, we consider sequences containing item sizes from two or three consecutive DH_k intervals.

We first consider intervals $(a,b) \subseteq (0,1)$ that contain exactly one DH_k partitioning point. Afterwards, we consider sequences with exactly two DH_k partitioning points. We emphasize that there are no restrictions on the endpoints a and b, which can be any real numbers, as long as the interval between them contains exactly one or two DH_k partitioning points. In both cases, DH_k turns out to have the better ratio.

For any $(a,b) \subseteq (0,1)$, we let $\mathrm{CR}_{a,b}$ denote the competitive ratio on sequences where all item sizes are in (a,b).

If (a,b) does not contain at least one of the interval borders used by DH_k, then DH_k packs exactly like DNF. If (a,b) contains a DH_k border, then we define

$$\frac{1}{p} = \max\left\{\frac{1}{l} \;\middle|\; l \in \mathbb{N}, \frac{1}{l} < b\right\},$$

and refer to $\frac{1}{p}$ as the *maximal border in* (a,b).

Note that if (a,b) contains exactly one of the interval borders used by DH_k, then $\frac{1}{p+1} \le a < \frac{1}{p}$. The next two theorems and the corollary deal with this case.

Proposition 1. *For any integer $x \ge 2$ such that $\frac{1}{x} \in (a,b)$, we have that $\mathrm{CR}_{a,b}(\mathrm{DNF}) \le \frac{x}{x+1}$.*

Proof. Consider the sequence $\langle\langle \frac{1}{x}\rangle^{x-1}, \frac{1}{x}-\varepsilon, \frac{1}{x}+\varepsilon\rangle^{xn}$. For this sequence, DNF covers only xn bins, whereas Opt can place exactly one small and one large item in each bin, filling up with items of size $\frac{1}{x}$, to cover $(x+1)n$ bins. □

Theorem 2. *If $\frac{1}{p+1} \le a < \frac{1}{p}$, then*

$$\mathrm{CR}_{a,b}(\mathrm{DNF}) = \frac{p}{p+1}$$

Proof. The lower bound follows directly from the fact that it takes at least p and at most $p+1$ items to cover a bin, and the upper bound follows from Proposition 1. □

Theorem 3. *If $\frac{1}{p+1} \le a < \frac{1}{p}$, then*

$$\mathrm{CR}_{a,b}(\mathrm{DH}_k) = \begin{cases} \frac{p^2+1}{p(p+1)}, & \text{if } k \ge p \\ \frac{p}{p+1}, & \text{otherwise.} \end{cases}$$

Proof. For $k < p$, (a,b) is contained in $(0, \frac{1}{k})$, and the result follows from Theorem 2.

For $k \ge p$, we consider the lower bound first. Since $k \ge p$, DH_k packs the items of size larger than or equal to $\frac{1}{p}$ in separate bins. For any sequence I, let t denote the total number of items in I and let ℓ denote the number of items of size larger than or equal to $\frac{1}{p}$. Then, DH_k covers at least $\lfloor \frac{\ell}{p} \rfloor + \lfloor \frac{t-\ell}{p+1} \rfloor > \frac{\ell+tp}{p(p+1)} - 2$ bins. Thus, letting $n = \mathrm{Opt}(I)$, we obtain

$$\mathrm{CR}_{a,b}(\mathrm{DH}_k) \ge \frac{\ell+tp}{np(p+1)}.$$

We treat this in two cases:

Case $\ell < n$: At least $n-\ell$ bins covered by Opt contain more than p items. Hence, $t \ge np+n-\ell$, and

$$\begin{aligned} \mathrm{CR}_{a,b}(\mathrm{DH}_k) &\ge \frac{\ell+tp}{np(p+1)} \ge \frac{\ell+np^2+np-\ell p}{np(p+1)} = \frac{np^2+n+n(p-1)-\ell(p-1)}{np(p+1)} \\ &> \frac{np^2+n}{np(p+1)}, \quad \text{since } \ell < n \\ &= \frac{p^2+1}{p(p+1)}. \end{aligned}$$

Case $\ell \geq n$**:** Here we can only use $t \geq np$, obtaining

$$\mathrm{CR}_{a,b}(\mathrm{DH}_k) \geq \frac{\ell + tp}{np(p+1)} \geq \frac{n + tp}{np(p+1)} \geq \frac{n(1+p^2)}{np(p+1)} = \frac{p^2+1}{p(p+1)}.$$

For the upper bound, we consider the sequence $\langle\langle \frac{1}{p} - \frac{\varepsilon}{p-1}\rangle^{p-1}, \frac{1}{p} + \varepsilon\rangle^n$, where $0 < \varepsilon < \min\{(p-1)(\frac{1}{p} - a), b - \frac{1}{p}\}$, ensuring that both item sizes belong to (a, b). Opt covers n bins, whereas DH_k packs the different sized items in separate bins, and covers $\frac{n(p-1)}{p+1} + \frac{n}{p} = \frac{p^2+1}{p(p+1)}n$ bins, up to an additive constant independent of n which is due to rounding. □

It follows that if (a, b) contains exactly one DH_k partitioning point, $\frac{1}{p}$, and $k \geq p$, then DH_k has a better competitive ratio than DNF:

Corollary 1. *If* $\frac{1}{p+1} \leq a < \frac{1}{p}$ *and* $k \geq p$*, then*

$$\mathrm{CR}_{a,b}(\mathrm{DH}_k) > \mathrm{CR}_{a,b}(\mathrm{DNF}).$$

Proof. The result follows from Theorems 2 and 3, since $\mathrm{CR}_{a,b}(\mathrm{DH}_k) = \frac{p^2+1}{p(p+1)} = \frac{p}{p+1} + \frac{1}{p(p+1)} = \mathrm{CR}_{a,b}(\mathrm{DNF}) + \frac{1}{p(p+1)}$. □

We now consider intervals $(a, b) \subseteq (0, 1)$ that contain exactly two DH_k partitioning points, $\frac{1}{p}$ and $\frac{1}{p+1}$. Including the extra partitioning point, $\frac{1}{p+1}$, results in a lower competitive ratio for DNF, with an upper bound depending on whether b is smaller or larger than $\frac{p+2}{p(p+1)}$. The competitive ratio of DH_k becomes lower than with just one partitioning point, only if $b > \frac{p+2}{p(p+1)}$.

Theorem 4. *If* $a < \frac{1}{p+1}$*, then*

$$\mathrm{CR}_{a,b}(\mathrm{DNF}) \leq \begin{cases} \frac{p+1}{p+2}, & \text{if } b \leq \frac{p+2}{p(p+1)} \\ \frac{p^2+p}{p^2+2p+2}, & \text{otherwise} \end{cases}$$

Proof. Applying Proposition 1, using $x = p + 1$, we get an upper bound of $\frac{p+1}{p+2}$.

If $b > \frac{p+2}{p(p+1)}$, we can strengthen the upper bound further. Since

$$a < \frac{1}{p+1} < \frac{1}{p} < \frac{p+2}{p(p+1)} < b < \frac{1}{p-1},$$

we can choose an $\varepsilon > 0$ small enough so that for any fraction r in this sequence of inequalities, and any constant, c, used below, $r - c\varepsilon$ and $r + c\varepsilon$ respect the same inequalities as r. Now, we consider a sequence I consisting of the following subsequences, for some integer n:

- $\langle\langle \frac{1}{p}\rangle^{p-1}, \frac{1}{p} - 2\varepsilon, \frac{p+2}{p(p+1)} + \varepsilon\rangle^{(p+1)(p-2)n}$
- $\langle\langle \frac{1}{p+1}\rangle^{p}, \frac{1}{p+1} - \varepsilon, \frac{p+2}{p(p+1)} + \varepsilon\rangle^{(p+1)n}$
- $\langle \frac{1}{p+1} + i(p-2)\varepsilon, \frac{1}{p+1} - (i+1)(p-2)\varepsilon, \langle \frac{1}{p+1} + \varepsilon\rangle^{p-2}, \frac{1}{p+1} - \varepsilon, \frac{p+2}{p(p+1)} + \varepsilon\rangle$ for $i = 1, 2, \ldots, (p+1)n - 1$
- $\langle \frac{1}{p+1} - (p-2)\varepsilon, \langle \frac{1}{p+1} + \varepsilon\rangle^{p-2}, \frac{1}{p+1} - \varepsilon, \frac{p+2}{p(p+1)} + \varepsilon\rangle$
- $\langle \frac{1}{p+1} + (p+1)n(p-2)\varepsilon\rangle$

Giving the items in this order, the number of bins covered by DNF is

$$\mathrm{DNF}(I) = (p+1)(p-2)n + (p+1)n + \big((p+1)n - 1\big) + 1 = p(p+1)n.$$

Opt just puts the items into the correct bins as they arrive, but for verification purposes, we list an order of the items which is optimal for DNF, but emphasize that this is Opt obtaining this result. Below, we use that $\frac{1}{p+1} + \frac{p+2}{p(p+1)} = \frac{2}{p}$. The following order illustrates the optimal packing:

- $\langle \frac{p+2}{p(p+1)} + \varepsilon, \frac{1}{p} - 2\varepsilon, \langle \frac{1}{p}\rangle^{p-3}, \frac{1}{p+1} + \varepsilon\rangle^{(p+1)(p-2)n}$
- $\langle \frac{p+2}{p(p+1)} + \varepsilon, \langle \frac{1}{p}\rangle^{p-2}, \frac{1}{p+1} - \varepsilon\rangle^{2(p+1)n}$

- $\langle \frac{1}{p+1} + i(p-2)\varepsilon, \frac{1}{p+1} - i(p-2)\varepsilon, \langle \frac{1}{p+1} \rangle^{p-1} \rangle$ for $i = 1, 2, \ldots, (p+1)n$
- $\langle \langle \frac{1}{p+1} \rangle^{p+1} \rangle^n$

The number of bins covered by Opt is

$$\textsc{Opt}(I) = (p+1)(p-2)n + 2(p+1)n + (p+1)n + n = \left(p^2 + 2p + 2\right)n. \qquad \square$$

Theorem 5. *If $\frac{1}{p+2} \le a < \frac{1}{p+1}$ and $k \ge p+1$, then*

$$\mathrm{CR}_{a,b}(\mathrm{DH}_k) = \begin{cases} \frac{p^3+2p^2+p+2}{p(p+1)(p+2)} = \frac{p^2+1}{p(p+1)}, & \text{if } b \le \frac{p+2}{p(p+1)} \\ \frac{p^3+2p^2+2}{p(p+1)(p+2)}, & \text{otherwise} \end{cases}$$

Proof. We prove the lower bound first.

Items of size less than $\frac{1}{p+1}$ are called *small*, items of size at least $\frac{1}{p}$ are called *large*, and the remaining items are called *medium*. Let s, m, and ℓ denote the number of small, medium, and large items, respectively.

Consider an optimal packing. For $i \in \{1, 2, 3\}$, let n_i denote the number of bins with exactly $p + i - 1$ items. Then, $n = n_1 + n_2 + n_3$ is the number of bins covered by Opt. Since DH_k covers exactly $\lfloor \frac{s}{p+2} \rfloor + \lfloor \frac{m}{p+1} \rfloor + \lfloor \frac{\ell}{p} \rfloor$ bins, independent of the order of the items, we can consider items from the three types of bins separately.

Bins with p items: Let s_1, m_1, and ℓ_1 denote the number of small, medium, and large items, respectively, packed in these n_1 bins by Opt. Further, let $t_1 = s_1 + m_1 + \ell_1 = pn_1$ denote the total number of items packed here.

In each of these bins, the items have an average size of at least $\frac{1}{p}$. This means that $\ell_1 \ge n_1$, since each bin has to contain at least one item of size at least $\frac{1}{p}$.

We now prove the inequality $s_1 \le \ell_1$. We do this by proving the stronger result that for each bin β with exactly p items, $s(\beta) \le \ell(\beta)$, where $s(\beta)$ and $\ell(\beta)$ denote the number of small and large items in β, respectively.

Small items deviate from the average size with strictly more than

$$\varepsilon_s = \frac{1}{p} - \frac{1}{p+1} = \frac{1}{p(p+1)} = \frac{p-1}{(p-1)p(p+1)},$$

and large items deviate with at most

$$\varepsilon_\ell = \frac{1}{p-1} - \frac{1}{p} = \frac{1}{(p-1)p} = \frac{p+1}{(p-1)p(p+1)}.$$

Thus, having an average item size of at least $\frac{1}{p}$ within a bin β requires $\ell(\beta)\varepsilon_\ell > s(\beta)\varepsilon_s$. Assume that β contains more small items than large items, i.e., $s(\beta) \ge \ell(\beta) + 1$. Then, $\ell(\beta)\varepsilon_\ell > (\ell(\beta)+1)\varepsilon_s$, which is equivalent to $\ell(\beta) > \frac{\varepsilon_s}{\varepsilon_\ell - \varepsilon_s}$, implying that $\ell(\beta) > \frac{p-1}{2}$, using the equation above. Since $\ell(\beta)$ is an integer, this means that $\ell(\beta) \ge \frac{p}{2}$, and since β contains exactly p items, this proves that $\ell(\beta) \ge s(\beta)$.

The contribution to the number of bins covered by DH_k from the t_1 items considered here is more than $d_1 - 3$, where the -3 comes from a possible fractional part in the three addends below.

$$\begin{aligned}
d_1 &= \frac{s_1}{p+2} + \frac{m_1}{p+1} + \frac{\ell_1}{p} \\
&= \frac{s_1}{p+2} + \frac{pn_1 - s_1 - \ell_1}{p+1} + \frac{\ell_1}{p} \\
&= \frac{(p+1)s_1}{(p+1)(p+2)} + \frac{pn_1}{p+1} - \frac{(p+2)s_1}{(p+1)(p+2)} - \frac{p\ell_1}{p(p+1)} + \frac{(p+1)\ell_1}{p(p+1)} \\
&= \frac{pn_1}{p+1} - \frac{s_1}{(p+1)(p+2)} + \frac{\ell_1}{p(p+1)} \\
&\ge \frac{pn_1}{p+1} - \frac{p\ell_1}{p(p+1)(p+2)} + \frac{(p+2)\ell_1}{p(p+1)(p+2)}, \quad \text{since } s_1 \le \ell_1 \\
&= \frac{p^2(p+2)n_1}{p(p+1)(p+2)} + \frac{2\ell_1}{p(p+1)(p+2)} \\
&\ge \frac{(p^3+2p^2)n_1}{p(p+1)(p+2)} + \frac{2n_1}{p(p+1)(p+2)}, \quad \text{since } \ell_1 \ge n_1 \\
&= \frac{p^3+2p^2+2}{p(p+1)(p+2)} n_1
\end{aligned}$$

If $b \le \frac{p+2}{p(p+1)} = \frac{1}{p} + \frac{1}{p(p+1)}$, one large item is not large enough to compensate for the loss of contribution to the average that a small item generates (recall that this loss is strictly larger than $\varepsilon_s = \frac{1}{p(p+1)}$). Therefore, additional to the n_1 large items, there has to be at least one more large item for each small item, i.e., $\ell_1 \ge s_1 + n_1$. In this case, we can strengthen the calculations above from a certain point:

$$\begin{aligned}
d_1 &= \frac{pn_1}{p+1} - \frac{s_1}{(p+1)(p+2)} + \frac{\ell_1}{p(p+1)} \\
&\ge \frac{pn_1}{p+1} - \frac{s_1}{(p+1)(p+2)} + \frac{s_1+n_1}{p(p+1)}, \quad \text{since } \ell_1 \ge s_1 + n_1 \\
&= \frac{(p^2+1)n_1}{p(p+1)} + \frac{2s_1}{p(p+1)(p+2)} \\
&\ge \frac{p^2+1}{p(p+1)} n_1, \quad \text{since } s_1 \ge 0 \\
&= \frac{p^3+2p^2+p+2}{p(p+1)(p+2)} n_1
\end{aligned}$$

Bins with $p+1$ items: Let s_2, m_2, and ℓ_2 denote the number of small, medium, and large items, respectively, packed in these n_2 bins. Further, let $t_2 = s_2 + m_2 + \ell_2 = (p+1)n_2$ denote the total number of items packed here.

In each of these bins, the items have an average size of at least $\frac{1}{p+1}$. This means that $s_2 \le pn_2$, as each bin has to contain at least one item of size at least $\frac{1}{p+1}$.

The contribution to the number of bins covered by DH_k from the t_2 items considered here is more than $d_2 - 3$, where

$$\begin{aligned}
d_2 &= \frac{s_2}{p+2} + \frac{m_2}{p+1} + \frac{\ell_2}{p} \\
&= \frac{s_2}{p+2} + \frac{(p+1)n_2 - s_2 - \ell_2}{p+1} + \frac{\ell_2}{p} \\
&= \frac{(p+1)s_2}{(p+1)(p+2)} + n_2 - \frac{(p+2)s_2}{(p+1)(p+2)} - \frac{p\ell_2}{p(p+1)} + \frac{(p+1)\ell_2}{p(p+1)} \\
&= n_2 - \frac{s_2}{(p+1)(p+2)} + \frac{\ell_2}{p(p+1)} \\
&\ge n_2 - \frac{pn_2}{(p+1)(p+2)} + \frac{\ell_2}{p(p+1)}, \quad \text{since } s_2 \le pn_2 \\
&\ge \frac{(p^2+2p+2)n_2}{(p+1)(p+2)}, \quad \text{since } \ell_2 \ge 0 \\
&\ge \frac{p^3+2p^2+2p}{p(p+1)(p+2)} n_2 \\
&\ge \frac{p^3+2p^2+p+2}{p(p+1)(p+2)} n_2, \quad \text{since } p \ge 2
\end{aligned}$$

Bins with $p+2$ items: Since DH_k cannot be forced to pack more than $p+2$ items in each bin, the contribution to the number of bins covered by DH_k from the items considered here is exactly n_3.

Now, we turn to the upper bound. Assume first that $b > \frac{p+2}{p(p+1)}$. Consider the sequence

$$\left\langle \left\langle \frac{1}{p+1} - \varepsilon \right\rangle^n, \left\langle \frac{p+2}{p(p+1)} + (p-1)\varepsilon \right\rangle^n, \left\langle \frac{1}{p} - \varepsilon \right\rangle^{n(p-2)} \right\rangle$$

for some $\varepsilon > 0$, sufficiently small such that all the items in the sequence are in the range (a, b). Since $\frac{1}{p+1} + \frac{p+2}{p(p+1)} = \frac{2}{p}$, Opt can cover n bins by combining one item of size $\frac{1}{p+1} - \varepsilon$, one item of size $\frac{p+2}{p(p+1)} + (p-1)\varepsilon$, and $(p-2)$ items of size $\frac{1}{p} - \varepsilon$. DH_k packs each kind of item separately, covering $\frac{n}{p+2} + \frac{n}{p} + \frac{n(p-2)}{p+1} = \frac{p^3+2p^2+2}{p(p+1)(p+2)} n$ bins.

If $b \le \frac{p+2}{p(p+1)}$, we do not need small items to get this weaker upper bound. It is sufficient to consider the two larger intervals and use Theorem 3, since $\frac{p^2+1}{p(p+1)} = \frac{p^3+2p^2+p+2}{p(p+1)(p+2)}$. □

It follows that if (a, b) contains exactly two DH_k partitioning points, then DH_k has a better competitive ratio than DNF:

Corollary 2. *If* $\frac{1}{p+2} \le a < \frac{1}{p+1}$, *then* $\mathrm{CR}_{a,b}(\mathrm{DH}_k) > \mathrm{CR}_{a,b}(\mathrm{DNF})$.

Proof. The result follows from Theorems 4 and 5, since if $b \le \frac{p+2}{p(p+1)}$, then

$$\begin{aligned}\mathrm{CR}_{a,b}(\mathrm{DH}_k) &= \frac{p^3+2p^2+p+2}{p(p+1)(p+2)} = \frac{p+1}{p+2} + \frac{2}{p(p+1)(p+2)} \\ &> \frac{p+1}{p+2} \ge \mathrm{CR}_{a,b}(\mathrm{DNF})\end{aligned}$$

and otherwise,

$$\begin{aligned}\mathrm{CR}_{a,b}(\mathrm{DH}_k) &= \frac{p^3+2p^2+2}{p(p+1)(p+2)} \\ &= \frac{p(p+1)}{p^2+2p+2} + \frac{2p^3+4p^2+4p+4}{(p^2+2p+2)p(p+1)(p+2)} \\ &> \frac{p(p+1)}{p^2+2p+2} \ge \mathrm{CR}_{a,b}(\mathrm{DNF}) \qquad \square\end{aligned}$$

3. Relative worst order analysis

Relative worst order analysis was introduced by Boyar and Favrholdt [4] and it compares the performance of two algorithms A and B directly instead of via the comparison to Opt. Algorithms are compared on the same input sequence I, but on the worst possible permutation of I for each algorithm.

Formally, if n is the length of I, and σ is a permutation on n elements, then $\sigma(I)$ denotes I permuted by σ, and we define $\mathrm{A}_W(I) = \min_\sigma \mathrm{A}(\sigma(I))$. If there exists a fixed constant b such that, for any input sequence I, $\mathrm{A}_W(I) \ge \mathrm{B}_W(I) - b$, then A and B are *comparable* and the *relative worst order ratio* of A to B is defined as follows:

$$\mathrm{WR}(\mathrm{A}, \mathrm{B}) = \sup\left\{c \;\middle|\; \exists b\ \forall I\colon \mathrm{A}_W(I) \ge c\,\mathrm{B}_W(I) - b\right\}$$

Note that since the performance of DH_k does not depend on the order in which the items are given, relative worst order analysis of DNF versus DH_k gives the same result as simply comparing the two algorithms on each sequence separately, just as competitive analysis with Opt replaced by DH_k.

In [16], a relative worst order analysis of DH_k and DNF is given for the model that allows items of size 1, showing that for $i < j$, $\mathrm{WR}(\mathrm{DH}_j, \mathrm{DH}_i) = \frac{i+1}{i}$. Hence, in this model, $\mathrm{WR}(\mathrm{DH}_k, \mathrm{DNF}) = 2$, for $k \ge 2$, since DNF and DH_1 are equivalent. Note that, for $i \ge 2$, the result from [16] holds for our model too, since the lower bound sequences for these cases do not contain items of size 1.

We first show that DH_k and DNF are comparable. This is a special case of the corresponding result in [16].

Lemma 1. *For any* $k \ge 1$ *and any input sequence* I,

$$\mathrm{DH}_{kW}(I) \ge \mathrm{DNF}_W(I) - (k-1)$$

Proof. For any sequence I, we can construct an input sequence for DNF by giving the items in the order they are packed in the bins by DH_k; first the covered bins and afterwards the items within the uncovered bins. For the closed bins, DNF then does the same as DH_k. DNF can cover at most $k-1$ additional bins, because DH_k has at most k open bins at the end. Thus, for any I, if $\sigma_{\mathrm{DH}_k}(I)$ and $\sigma_{\mathrm{DNF}}(I)$ denote the worst permutations of I with respect to the two algorithms, then $\mathrm{DH}_{kW}(I) = \mathrm{DH}_k(\sigma_{\mathrm{DH}_k}(I)) \ge \mathrm{DNF}(\sigma_{\mathrm{DH}_k}(I)) - (k-1) \ge \mathrm{DNF}(\sigma_{\mathrm{DNF}}(I)) - (k-1) = \mathrm{DNF}_W(I) - (k-1)$. $\square$

Thus, according to relative worst order analysis, DH_k is at least as good as DNF. The next lemma establishes a separation between the two algorithms in our model.

Lemma 2. *For any* $k \ge 2$, $\mathrm{WR}(\mathrm{DH}_k, \mathrm{DNF}) \ge \frac{3}{2}$.

Proof. It follows from Lemma 1 that the algorithms are comparable.

We prove that the ratio cannot be smaller than $\frac{3}{2}$ by exhibiting a family of sequences $\{I_n\}$ such that the following two conditions hold:

- $\lim_{n\to\infty} \mathrm{DH}_k(I_n) = \infty$.
- For all I_n, $\mathrm{DH}_{kW}(I_n) \ge \frac{3}{2} \cdot \mathrm{DNF}_W(I_n) - 1$.

For each $n \geq 1$, we define $I_n = \langle \frac{1}{2}, \langle \frac{1}{2n} \rangle^{n-1}, \frac{1}{2} \rangle^{2n}$. DH_k covers $2n + (n-1) = 3n-1$ bins, whereas DNF covers only $2n$ bins. Thus, for all I_n,

$$\mathrm{DH}_{kW}(I_n) \geq \frac{3}{2} \cdot \mathrm{DNF}_W(I_n) - 1. \qquad \square$$

By providing a matching upper bound, we determine the exact relative worst order ratio of the two algorithms.

Theorem 6. $\mathrm{WR}(\mathrm{DH}_k, \mathrm{DNF}) = \frac{3}{2}$.

Proof. Lemma 2 shows that $\mathrm{WR}(\mathrm{DH}_k, \mathrm{DNF}) \geq \frac{3}{2}$. Thus, it remains to be established that $\mathrm{WR}(\mathrm{DH}_k, \mathrm{DNF}) \leq \frac{3}{2}$.

Assume that an input sequence I has a total volume of n, and assume that DNF covers xn bins.

Case $x < \frac{1}{2}$**:** To cover fewer than $\frac{n}{2}$ bins, a volume of more than $\frac{n}{2}$ has to be wasted by overpacking fewer than $\frac{n}{2}$ bins. Thus, some item of size larger than one must exist, which is a contradiction.

Case $\frac{1}{2} \leq x < \frac{2}{3}$**:** If DNF covers only xn bins, it wastes a volume of $(1-x)n$ by overpacking at most xn bins. Therefore, the average size of an item that is packed as the last item in a bin by DNF is at least $\frac{(1-x)n}{xn} > \frac{1}{2}$. Since items larger than $\frac{1}{2}$ are packed with another item of size at least $\frac{1}{2}$ by DH_k, the volume above $\frac{1}{2}$ is also wasted for DH_k. Thus, DH_k wastes at least a volume of $(\frac{(1-x)n}{xn} - \frac{1}{2})xn = n - \frac{3}{2}xn$. So, $\mathrm{DH}_k(I) \leq n - (n - \frac{3}{2}xn) = \frac{3}{2}xn = \frac{3}{2}\mathrm{DNF}(I)$.

Case $\frac{2}{3} \leq x \leq 1$**:** The performance of DH_k is bounded by the volume n of the sequence I, so $\mathrm{DH}_k(I) \leq n$. Thus, $\mathrm{DH}_k(I) \leq n = \frac{3}{2} \cdot \frac{2}{3}n \leq \frac{3}{2}xn = \frac{3}{2}\mathrm{DNF}(I)$. $\square$

We conclude that according to relative worst order analysis, DH_k is a better algorithm than DNF.

4. Random order analysis

The random order ratio was introduced by Kenyon [20] as the worst ratio obtained over all sequences I, comparing the expected value of an algorithm A, with respect to a uniform distribution of all permutations, σ, of I, to the value of Opt on I:

$$\mathrm{RR(A)} = \liminf_{\mathrm{Opt}(I) \to \infty} \frac{E_\sigma[\mathrm{A}(\sigma(I))]}{\mathrm{Opt}(I)}$$

Note that Opt is still assumed to know the entire sequence in advance, so there is no expectation involved in computing $\mathrm{Opt}(I)$.

The following theorem gives a bound on how well DNF can perform with respect to the random order ratio.

Theorem 7. *The random order ratio of* DNF *is at most* $\frac{4}{5}$.

Proof. Let S^n denote all sequences of length n with item sizes from $\mathcal{I}$, where $\mathcal{I} = \{\varepsilon, 1-\varepsilon\}$ for an ε such that $0 < \varepsilon < \frac{1}{n}$. Define

$$S_i^n = \left\{ I \in S^n \;\middle|\; I \text{ contains } i \text{ items of size } \varepsilon \text{ and } n-i \text{ items of size } 1-\varepsilon \right\}.$$

Then we can consider the following disjoint partitioning $S^n = \bigcup_{0 \leq i \leq n} S_i^n$. We let R^n denote the set of all sequences of length n.

The first inequality below follows from two facts:

- For any pair of sequences, $I, I' \in S_i^n$, $\mathrm{Opt}(I) = \mathrm{Opt}(I')$.
- For two sums $A = \sum_{i=1}^n a_i$ and $B = \sum_{i=1}^n b_i$, $\frac{A}{B} \geq \min_{1 \leq i \leq n} \frac{a_i}{b_i}$.

$$\begin{aligned} \frac{\mathrm{E}_{I \in S^n}[\mathrm{DNF}(I)]}{\mathrm{E}_{I \in S^n}[\mathrm{Opt}(I)]} &\geq \min_{0 \leq i \leq n} \frac{\mathrm{E}_{I \in S_i^n}[\mathrm{DNF}(I)]}{\mathrm{Opt}(I_i^n)}, \quad \text{where } I_i^n \in S_i^n \\ &= \min_{I \in S^n} \frac{\mathrm{E}_\sigma[\mathrm{DNF}(\sigma(I))]}{\mathrm{Opt}(I)} \geq \min_{I \in R^n} \frac{\mathrm{E}_\sigma[\mathrm{DNF}(\sigma(I))]}{\mathrm{Opt}(I)} \end{aligned}$$

Hence,

$$\lim_{n \to \infty} \frac{\mathrm{E}_{I \in S^n}[\mathrm{DNF}(I)]}{\mathrm{E}_{I \in S^n}[\mathrm{Opt}(I)]} \geq \liminf_{\mathrm{Opt}(I) \to \infty} \frac{\mathrm{E}_\sigma[\mathrm{DNF}(\sigma(I))]}{\mathrm{Opt}(I)} = \mathrm{RR(DNF)}.$$

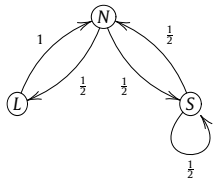

Fig. 2. A Markov chain describing DNF's behavior on the considered sequences.

In the rest of the proof, we bound the leftmost expression from the above, which then gives us an upper bound on the random order ratio of DNF.

There is no difference between choosing some element from S^n uniformly at random and generating a length n sequence iteratively by choosing the next item from $\mathcal{I}$ with equal probability. Thus, we can analyze the behavior of DNF by considering a Markov chain, where the state of the system after i items have been processed is determined by the state of the open bin. The Markov chain is finite and has just three states: either there is no open bin (N – for "No"), one open bin containing one large item of size $1-\varepsilon$ (L – for "Large"), or one bin with a number of small items, each of size ε (S – for "Small") (Fig. 2). Note that since $\varepsilon < \frac{1}{n}$, there is room for all the small items in one bin, if necessary.

This is an irreducible chain, where all states are positive recurrent, which implies that it has a stationary (equilibrium) distribution, and the probability of ending up in each of the states converges independently of the starting state [14]. The probability of being in one of the states N, L, or S can be calculated from the following equations:

$$\begin{aligned} 1 &= \text{Prob}[N] + \text{Prob}[L] + \text{Prob}[S] \\ \text{Prob}[N] &= \text{Prob}[L] + \frac{\text{Prob}[S]}{2} \\ \text{Prob}[L] &= \frac{\text{Prob}[N]}{2} \\ \text{Prob}[S] &= \frac{\text{Prob}[N]}{2} + \frac{\text{Prob}[S]}{2} \end{aligned}$$

This system has the solution $\text{Prob}[N] = \text{Prob}[S] = \frac{2}{5}$ and $\text{Prob}[L] = \frac{1}{5}$. From this it follows that $\text{E}_{I\in S^n}[\text{DNF}(I)]$ tends to $\text{Prob}[N]n = \frac{2}{5}n$.

For the optimal algorithm, note that its result only depends on the number of items of each size. In particular, after n items, it can cover $\lfloor \frac{n}{2} \rfloor$ bins, unless there are more small than large items. All the extra small items would be wasted.

Using random walks, it is easy to see that the expected difference between the number of large and small items is a low order term compared with n, and therefore does not affect the limit. A sequence of independent stochastic variables $\{X_i\}_{i\geq 1}$, where $\text{Prob}[X_i = 1] = \text{Prob}[X_i = -1] = \frac{1}{2}$, is called a *simple random walk* [14]. It is well known that if we define $T_n = \sum_{i=1}^{n} X_i$, then $\lim_{n\to\infty} \frac{\text{E}[|T_n|]}{\sqrt{n}} = \sqrt{\frac{2}{\pi}}$ [17]. Hence, $\text{E}[|T_n|] \in O(\sqrt{n})$, and then $\text{E}_{I\in S^n}[\text{OPT}(I)] = \frac{n}{2} - O(\sqrt{n})$.

In conclusion, we get

$$\lim_{n\to\infty} \frac{\text{E}_{I\in S^n}[\text{DNF}(I)]}{\text{E}_{I\in S^n}[\text{OPT}(I)]} = \lim_{n\to\infty} \frac{\frac{2}{5}n}{\frac{n}{2} - O(\sqrt{n})} = \frac{4}{5}. \qquad \square$$

Theorem 8. *The random order ratio of* DH_k *is* $\frac{1}{2}$.

Proof. The performance of DH_k does not depend on the order of the items in the sequence. Given a sequence containing n items of size $1-\varepsilon$ and n items of size ε, where $\varepsilon < \frac{1}{n}$, DH_k will always cover $\frac{n}{2}$ bins, while OPT will cover n bins. The lower bound is given by Theorem 1, since the random order ratio of a bin covering algorithm is never worse than its competitive ratio. $\square$

Thus, according to random order analysis, DNF is at least as good as DH_k. Though it seems hard to raise the lower bound on the random order ratio for DNF above $\frac{1}{2}$, and thereby separate the two algorithms, we conjecture that DNF is in fact strictly better than DH_k with respect to this measure. We discuss this further in the conclusion.

5. The max/max ratio

The max/max ratio was introduced by Ben-David and Borodin [3] and compares an algorithm's worst-case behavior on any sequence of length n with OPT's worst-case behavior on any sequence of length n.

The max/max ratio was introduced for the minimization problems paging and K-server. Since bin covering is a maximization problem, we actually need a min/min ratio. Additionally, since the input items can be arbitrarily small, letting the

sequence length approach infinity does not give interesting results. Thus, we modify the measure to consider the volume, $vol(I)$, of a sequence I, where $vol(I)$ is the sum of the sizes of all the items in I:

$$\mathrm{MR}_{vol}(\mathrm{A}) = \frac{\liminf_{v\to\infty} \min_{vol(I)=v} \mathrm{A}(I)/v}{\liminf_{v\to\infty} \min_{vol(I)=v} \textsc{Opt}(I)/v}$$

It turns out that this measure cannot distinguish between DNF and DH_k in the general case:

Theorem 9. *Both* DNF *and* DH_k *have a min/min ratio of* 1.

Proof. For any $\varepsilon > 0$, a sequence consisting only of items of size $1-\varepsilon$ will force any algorithm, including Opt, to put at least two items in each bin. As ε tends to 0, this gives an upper bound on the number of covered bins tending to $vol(I)/2$. Since both DNF and DH_k always cover at least $\lfloor vol(I)/2 \rfloor$ bins, this shows that their min/min ratios are 1. □

If the item sizes are restricted to an interval $(a, b) \subseteq (0, 1)$ containing at least one DH_k interval border, the min/min ratio can distinguish between DNF and DH_k. If (a, b) does not contain at least one of the interval borders used by DH_k, then DH_k packs exactly like DNF.

If (a, b) contains a DH_k border, then we define, as in Section 2, $\frac{1}{p}$ as the *maximal border in* (a, b). Throughout the paper, we assume that the constants k, a, b, and p have the meaning defined above.

Theorem 10. *With item sizes in* $(a, b) \subseteq (0, 1)$*, where* $a < \frac{1}{p}$, DH_k *has a min/min ratio of* 1.

Proof. The worst-case sequences for DH_k consist of items only of size either $b-\varepsilon$ or $\frac{1}{p}-\varepsilon$, for any small ε, and, since there are no choices in packing sequences with just one item size, Opt cannot pack them better than DH_k. □

Theorem 11. *With item sizes in* $(a, b) \subseteq (0, 1)$*, where* $\frac{1}{p} \in (a, b)$, DNF *has a min/min ratio of* $\max\{\frac{1+\frac{1}{p}}{1+b}, \frac{pb}{1+b}\}$.

Proof. To maximize the overpacking by DNF, the last item of each bin should have size close to b and be packed in a nearly full bin. Thus, we arrange that each bin gets p items of size $\frac{1}{p}-\varepsilon$ for some $0 < \varepsilon < \frac{1}{p} - a$, and then an item of size $b-\varepsilon$. Each bin receives a volume of $1 - p\varepsilon + b - \varepsilon$, so to use volume n, we repeat this $n/(1-(p+1)\varepsilon+b)$ times to get a sequence I_n. We may assume this is integral, since any rounding disappears in the limit,

$$\liminf_{n\to\infty} \min_{vol(I)=n} \frac{\mathrm{DNF}(I_n)}{n} = \frac{1}{1-(p+1)\varepsilon+b},$$

and, since we can use any ε, $0 < \varepsilon < \frac{1}{p} - a$, we arrive at $\frac{1}{1+b}$.

The worst-case for Opt follows by using one of the two types of sequences from the proof of Theorem 10, i.e., for each bin, $p+1$ items of size $\frac{1}{p}-\varepsilon$ or p items of size $b-\varepsilon$, for some ε. Similar to the calculations above, and letting ε approach zero, the limit for Opt becomes

$$\liminf_{n\to\infty} \min_{vol(I)=n} \frac{\textsc{Opt}(I_n)}{n} = \min\left\{\frac{1}{(p+1)\frac{1}{p}}, \frac{1}{pb}\right\} = \min\left\{\frac{1}{1+\frac{1}{p}}, \frac{1}{pb}\right\}.$$

Dividing the result for DNF with the result for Opt, we get the stated ratio. □

Note that $\frac{1+\frac{1}{p}}{1+b} < 1$ is equivalent to $\frac{1}{p} < b$, which follows from the definition and maximality of $\frac{1}{p}$. Furthermore, $\frac{pb}{1+b} < 1$ is equivalent to $b < \frac{1}{p-1}$, which is satisfied as long as b is not equal to $\frac{1}{p-1}$. Thus, according to min/min analysis, DH_k is better than DNF when item sizes are restricted to an interval $(a, b) \in (0, 1)$ containing at least one DH_k border, and $b \neq \frac{1}{p-1}$ where $\frac{1}{p}$ is the maximal border.

6. Uniform distribution

In this section, we study the expected performance ratio of DNF and DH_k on sequences containing items drawn uniformly at random from the interval $(0, 1)$. Let $\mathrm{E}_{I\in U_n(x,y)}[\mathrm{A}(I)]$ denote the expected value of some algorithm A on sequences of length n, where the items in the sequences are drawn independently and uniformly at random from the interval (x, y). Note that this definition will be used later for varying sequence lengths and we also use it on semi-open intervals $[x, y)$.

Now, the expected performance ratio $\mathrm{ER}_{\mathrm{U}}(\mathrm{A})$ is the limit for n approaching infinity of the ratio between the expected performance of the algorithms A and Opt on sequences of length n, containing items drawn uniformly at random from the interval $(0, 1)$.

Formally,

$$\mathrm{ER_U(A)} = \lim_{n\to\infty} \frac{\mathrm{E}_{I\in U_n(0,1)}[\mathrm{A}(I)]}{\mathrm{E}_{I\in U_n(0,1)}[\textsc{Opt}(I)]}.$$

Theorem 12. *On a sequence containing items drawn uniformly at random from the interval* $(0,1)$*,*

$$\mathrm{ER_U(DH_2)} = \frac{1}{2} + \frac{1}{e^2-e} \approx 0.7141 \quad \textit{and}$$
$$\lim_{k\to\infty} \mathrm{ER_U(\,DH}_k) = \frac{12-\pi^2}{3} \approx 0.7101.$$

Proof. For sequences of length n drawn uniformly at random from the interval $(0,1)$,

$$\begin{aligned}\mathrm{ER_U(DH}_k) &= \lim_{n\to\infty} \frac{\mathrm{E}_{I\in U_n(0,1)}[\mathrm{DH}_k(I)]}{\mathrm{E}_{I\in U_n(0,1)}[\textsc{Opt}(I)]}\\ &= \lim_{n\to\infty} \frac{\mathrm{E}_{I\in U_{\frac{(k-1)n}{k}}[\frac{1}{k},1)}[\mathrm{DH}_k(I)] + \mathrm{E}_{I\in U_{\frac{n}{k}}(0,\frac{1}{k})}[\mathrm{DH}_k(I)]}{\mathrm{E}_{I\in U_n(0,1)}[\textsc{Opt}(I)]}\\ &= R_{[\frac{1}{k},1)} + R_{(0,\frac{1}{k})},\end{aligned}$$

where the second equality follows from the fact that DH_k processes items smaller than $\frac{1}{k}$ separately from items of size at least $\frac{1}{k}$. Thus, these items can be treated separately. Since item sizes are chosen uniformly at random and the result of DH_k depends linearly on the number of items in each interval, this corresponds to scaling n using $\frac{k-1}{k}$ and $\frac{1}{k}$, respectively.

The final equality just defines the following two expressions as

$$R_{[\frac{1}{k},1)} = \lim_{n\to\infty} \frac{\mathrm{E}_{I\in U_{\frac{(k-1)n}{k}}[\frac{1}{k},1)}[\mathrm{DH}_k(I)]}{\mathrm{E}_{I\in U_n(0,1)}[\textsc{Opt}(I)]}$$

and

$$R_{(0,\frac{1}{k})} = \lim_{n\to\infty} \frac{\mathrm{E}_{I\in U_{\frac{n}{k}}(0,\frac{1}{k})}[\mathrm{DH}_k(I)]}{\mathrm{E}_{I\in U_n(0,1)}[\textsc{Opt}(I)]}.$$

Using a pairing heuristic, [10] shows that $\mathrm{E}_{I\in U_n[0,1)}[\textsc{Opt}(I)] = \frac{n}{2}$.

For a sequence with items drawn uniformly at random from $(0,1)$, the expected number of items with sizes in the interval $[\frac{1}{i}, \frac{1}{i-1})$ is $(\frac{1}{i-1} - \frac{1}{i})n = \frac{n}{i(i-1)}$. For $2 \le i \le k$, DH_k packs each of these items (except for at most $i-1$ items) in bins with exactly i items each. Hence, the expected number of bins that DH_k covers with such items is more than $\frac{n}{i^2(i-1)} - 1$, $2 \le i \le k$.

Hence,

$$R_{[\frac{1}{k},1)} = \lim_{n\to\infty} \frac{\sum_{i=2}^{k}(\frac{n}{i^2(i-1)} - 1)}{n/2} = 2\sum_{i=2}^{k} \frac{1}{i^2(i-1)}.$$

Using partial fraction decomposition, we get

$$\begin{aligned}R_{[\frac{1}{k},1)} &= 2\sum_{i=2}^{k}\left(\frac{1}{i-1} - \frac{1}{i} - \frac{1}{i^2}\right)\\ &= 2\left(\sum_{i=1}^{k-1}\frac{1}{i} - \sum_{i=2}^{k}\frac{1}{i} - \sum_{i=1}^{k}\frac{1}{i^2} + 1\right)\\ &= 2\left(1 - \frac{1}{k} - \sum_{i=0}^{k-1}\frac{1}{(i+1)^2} + 1\right)\\ &= 2\left(2 - \frac{1}{k} - \sum_{i=0}^{\infty}\frac{1}{(i+1)^2} + \sum_{i=0}^{\infty}\frac{1}{(i+1+k)^2}\right)\\ &= 2\left(2 - \frac{1}{k} - \psi_1(1) + \psi_1(k+1)\right),\end{aligned}$$

where ψ_1 is the trigamma function [1]. Some properties of ψ_1 are that $\psi_1(1) = \frac{\pi^2}{6}$, $\psi_1(k+1) = \psi_1(k) - \frac{1}{k^2}$, and $\psi_1(k) \to 0$ as $k \to \infty$. Now,

$$\begin{aligned} R_{[\frac{1}{k},1)} &= 2\left(2 - \frac{1}{k} - \frac{\pi^2}{6} + \psi_1(k) - \frac{1}{k^2}\right) \\ &= 2\left(\frac{12-\pi^2}{6} - \frac{1+k}{k^2} + \psi_1(k)\right). \end{aligned}$$

Since $R_{(0,\frac{1}{k})} \to 0$ as $k \to \infty$, it follows that

$$\lim_{k\to\infty} \mathrm{ER_U}(\mathrm{DH}_k) = \frac{12-\pi^2}{3} \approx 0.7101.$$

Since DH_k packs the items of sizes in $(0, \frac{1}{k})$ the same way DNF would, we can use a result from [10], stating that

$$\lim_{n\to\infty} \frac{\mathrm{E}_{I\in U_n[0,\frac{1}{k})}[\mathrm{DNF}(I)]}{n} = \frac{1}{\mu(k)}$$

where

$$\mu(k) = \lim_{\varepsilon\to 0} \sum_{l=0}^{k} (-1)^l \frac{1}{l!} \left(\frac{k}{1-\varepsilon} - l\right)^l e^{\frac{k}{1-\varepsilon}-l} = \sum_{l=1}^{k} \frac{e^l(-l)^{k-l}}{(k-l)!},$$

and the limit for $\varepsilon \to 0$ is due to our working with an open interval, where the interval in [10, Eq. (28)] is closed. Now,

$$R_{(0,\frac{1}{k})} = \frac{1}{k} \lim_{n\to\infty} \frac{\mathrm{E}_{I\in U_n[0,\frac{1}{k})}[\mathrm{DNF}(I)]}{n/2} = \frac{2}{\mu(k)k}.$$

Note that $\mu(2) = e^2 - e$ and $\psi_1(2) = \psi_1(1) - \frac{1}{1^2} = \frac{\pi^2}{6} - 1$.
Hence,

$$\begin{aligned} \mathrm{ER_U}(\mathrm{DH}_2) &= R_{[\frac{1}{2},1)} + R_{(0,\frac{1}{2})} \\ &= 2\left(\frac{12-\pi^2}{6} - \frac{1+2}{2^2} + \psi_1(2)\right) + \frac{2}{2\mu(2)} \\ &= \frac{12-\pi^2}{3} - \frac{3}{2} + \frac{\pi^2}{3} - 2 + \frac{1}{e^2-e} \\ &= \frac{1}{2} + \frac{1}{e^2-e} \approx 0.7141. \qquad \square \end{aligned}$$

This should be compared with a result from [10], showing that on a uniform distribution, DNF has an expected performance ratio of $\frac{2}{e} \approx 0.7358$. Thus, under this assumption, DNF is a little better than DH_k.

7. Concluding remarks

The starting point for this paper was the fact that bin covering algorithms as different as DNF and DH_k are not separated using competitive analysis. We are interested in the question of which algorithm to use in different scenarios. DH_k was designed to guard against worst-case sequences, and since these are often made up using pathological input, such as mixing very large and very small items, we have carried out analyses using the worst-case performance, but on restricted input of items of similar size. The comparison is still in DH_k's favor, albeit less so. Max/max analysis (under similar conditions) and relative worst order analysis also point to DH_k.

In contrast, if input is not organized into worst-case sequences by an adversary, we can show, by carrying out an analysis of the expected results under a uniform distribution that DNF performs a little better than DH_k. This seems to be very robust, since adding a small element of worst-case requirements in the form of random order analysis also points to DNF not being worse than DH_k. Thus, even if an adversary gets to choose the worst sequence for the algorithm, just the fact that the items are received in a random order removes DH_k's advantage over DNF.

The conclusion is that unless guarantees are desired or it is known that items do not arrive in a random order, it is worth considering DNF as the algorithm of choice.

DH_k has a random order ratio of $\frac{1}{2}$, which is worst possible, whereas the upper bound we have on DNF is $\frac{4}{5}$. We conjecture that these two algorithms can be separated, proving DNF to be best. Though it is not essential to the conclusion above, we leave this as an interesting open problem we would like to see solved, and use the rest of this section to discuss

some relevant issues regarding this. It seems intuitively almost obvious that DNF would always get a ratio larger than $\frac{1}{2}$. The difficulty in establishing this formally stems from problems of handling the size aspects using probability theory. In the hardest case, there are a linear number of very large items such that if they end up on top of each other pairwise, we get the ratio of $\frac{1}{2}$. Thus, we need to prove that some fraction of these large items do not end up pairwise on top of each other. The small items that would be packed with the large items in an optimal packing can be cut into very small pieces so there are orders of magnitude more small items than large items – but still of possibly dramatically varying size, relatively. Whereas we have strong theoretical tools for bounding the deviation from the expected number of items in certain locations in the form of Chebyshev's inequality, for instance, it is much harder to reason regarding deviations from the expected size, and it is exactly the sum of sizes of small items surrounding a large item that decides whether or not two large items end up on top of each other.

Results on the random order ratio are often difficult to establish. This is reflected in the rather small number of obtained results and also in published results being far from tight. In the paper [20] introducing the random order ratio, for example, the random order ratio of the bin packing algorithm Best-Fit is shown to lie between 1.08 and 1.5. An exceptionally tight result appears in [18], where it is shown that the random order ratio of Next-Fit for bin packing is exactly 2. Note, however, that this result does not give indication that the random order ratio of DNF for bin covering should be $\frac{1}{2}$. The sequence establishing the lower bound of 2 consists of n items of size $\frac{1}{2}$ and kn items of size $\epsilon < \frac{1}{kn}$, for some large k. For a random ordering of these items, each item of size $\frac{1}{2}$ has a high probability of being combined with at least one of the small items, leaving too little space in the bin for another large item. For bin covering, the problem is reversed; to prove an upper bound of $\frac{1}{2}$, we must prove that each large item has a significant probability of being surrounded by a sufficiently small volume of small items so that it will go into the same bin as a neighboring large item.

References

[1] Milton Abramowitz, Irene A. Stegun (Eds.), Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, Dover, 1964.
[2] S.F. Assmann, David S. Johnson, Daniel J. Kleitman, Joseph Y.-T. Leung, On a dual version of the one-dimensional bin packing problem, J. Algorithms 5 (4) (1984) 502–525.
[3] Shai Ben-David, Allan Borodin, A new measure for the study of on-line algorithms, Algorithmica 11 (1) (1994) 73–91.
[4] Joan Boyar, Lene M. Favrholdt, The relative worst order ratio for on-line algorithms, ACM Trans. Algorithms 3 (2) (2007), article 22.
[5] Joan Boyar, Lene M. Favrholdt, Kim S. Larsen, The relative worst order ratio applied to paging, J. Comput. System Sci. 73 (5) (2007) 818–843.
[6] Joan Boyar, Sushmita Gupta, Kim S. Larsen, Access graphs results for LRU versus FIFO under relative worst order analysis, in: Scandinavian Symposium and Workshops on Algorithm Theory, in: Lecture Notes in Computer Science, vol. 7357, Springer, 2012, pp. 328–339.
[7] Joan Boyar, Sushmita Gupta, Kim S. Larsen, Relative interval analysis of paging algorithms on access graphs, in: Workshop on Algorithms and Data Structures, in: Lecture Notes in Computer Science, vol. 8037, Springer, 2013, pp. 195–206.
[8] Joan Boyar, Sandy Irani, Kim S. Larsen, A comparison of performance measures for online algorithms, in: Workshop on Algorithms and Data Structures, in: Lecture Notes in Computer Science, vol. 5664, Springer, 2009, pp. 119–130.
[9] Joan Boyar, Kim S. Larsen, Abyayananda Maiti, A comparison of performance measures via online search, Theor. Comput. Sci. 532 (2014) 2–13.
[10] János Csirik, J.B.G. Frenk, Gábor Galambos, A.H.G. Rinnooy Kan, Probabilistic analysis of algorithms for dual bin packing problems, J. Algorithms 12 (2) (1991) 189–203.
[11] János Csirik, V. Totik, Online algorithms for a dual version of bin packing, Discrete Appl. Math. 21 (2) (1988) 163–167.
[12] János Csirik, Gerhard J. Woeginger, On-line packing and covering problems, in: Amos Fiat, Gerhard J. Woeginger (Eds.), Online Algorithms, in: Lecture Notes in Computer Science, vol. 1442, Springer, 1998, pp. 147–177.
[13] Reza Dorrigiv, Alejandro López-Ortiz, A survey of performance measures for on-line algorithms, SIGACT News 36 (3) (2005) 67–81.
[14] Rick Durrett, Probability: Theory and Examples, Dixbury Press, 1991.
[15] Martin R. Ehmsen, Jens S. Kohrt, Kim S. Larsen, List factoring and relative worst order analysis, Algorithmica 66 (2) (2013) 287–309.
[16] Leah Epstein, Lene M. Favrholdt, Jens S. Kohrt, Comparing online algorithms for bin packing problems, J. Sched. 15 (1) (2012) 13–21.
[17] Jørgen Hoffmann-Jørgensen, Probability with a View Towards Statistics, vol. I, Chapman & Hall, 1994.
[18] Edward G. Coffman Jr., János Csirik, Lajos Rónyai, Ambrus Zsbán, Random-order bin packing, Discrete Appl. Math. 156 (2008) 2810–2816.
[19] Anna R. Karlin, Mark S. Manasse, Larry Rudolph, Daniel D. Sleator, Competitive snoopy caching, Algorithmica 3 (1988) 79–119.
[20] Claire Kenyon, Best-fit bin-packing with random order, in: ACM-SIAM Symposium on Discrete Algorithms, 1996, pp. 359–364.
[21] C.C. Lee, D.T. Lee, A simple on-line bin-packing algorithm, J. ACM 32 (3) (1985) 562–572.
[22] Prakash V. Ramanan, Donna J. Brown, C.C. Lee, D.T. Lee, On-line bin packing in linear time, J. Algorithms 10 (3) (1989) 305–326.
[23] Steven S. Seiden, On the online bin packing problem, J. ACM 49 (5) (2002) 640–671.
[24] Daniel D. Sleator, Robert E. Tarjan, Amortized efficiency of list update and paging rules, Commun. ACM 28 (2) (1985) 202–208.
[25] Gerhard J. Woeginger, Improved space for bounded space, on-line bin-packing, SIAM J. Discrete Math. 6 (4) (1993) 575–581.