Noname manuscript No.
(will be inserted by the editor)

Online Results for Black and White Bin Packing

János Balogh · József Békési · György Dósa · Leah Epstein · Hans Kellerer · Zsolt Tuza

Received: date / Accepted: date

Abstract In online bin packing problems, items of sizes in $[0, 1]$ are to be partitioned into subsets of total size at most 1, called bins. We introduce a new variant where items are of two types, called black and white, and the item types must alternate in each bin, that is, two items of the same type cannot be assigned consecutively into a bin. We design an online algorithm with the absolute competitive ratio 3. We further show that a number of well-known algorithms cannot have a better performance, even in the asymptotic sense. Additionally, we prove a surprising general lower bound $1 + \frac{1}{2\ln 2} \approx 1.7213$ on the asymptotic competitive ratio of any deterministic or randomized online algorithm. This lower bound significantly exceeds the known *upper* bound 1.58889 for classic online bin packing.

A proceedings version where some of the results in this article were announced appeared as [1]. Research supported in part by the Stiftung Aktion Österreich-Ungarn, project No. 82öu9, by the project K-TET: 10-1-2011-0115, and by the Hungarian Scientific Research Fund, grant OTKA 81493.

János Balogh and József Békési
Department of Applied Informatics, Gyula Juhász Faculty of Education, University of Szeged, H-6701 Szeged, POB 396, Hungary. E-mail: {balogh,bekesi}@jgypk.u-szeged.hu

György Dósa
Department of Mathematics, University of Pannonia, H-8200 Veszprém, Egyetem u. 10, Hungary. E-mail: dosagy@almos.vein.hu

Leah Epstein
Department of Mathematics, University of Haifa, 31905 Haifa, Israel. E-mail: lea@math.haifa.ac.il

Hans Kellerer
Institut für Statistik und Operations Research, Universität Graz, Universitätsstraße 15, 8010 Graz, Austria. E-mail: hans.kellerer@uni-graz.at

Zsolt Tuza
Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences, H-1053 Budapest, Reáltanoda u. 13–15; and Department of Computer Science and Systems Technology, University of Pannonia, H-8200 Veszprém, Egyetem u. 10, Hungary. E-mail: tuza@dcs.vein.hu

Keywords Bin packing · Black and white bin packing · Online algorithms · Competitive analysis · Any Fit · Conflict graphs

1 Introduction

We deal with a new problem, that we call *Black and White Bin Packing*, abbreviated as *BWBP*. As in standard bin packing, items characterized by their sizes $p_1, p_2, \ldots$ are given, where $0 \leq p_i \leq 1$[1] and the goal is to pack them into the minimum number of unit capacity bins. This well-known problem is NP-hard (see [13]), and was widely studied.

In our problem the items are divided into two types or classes, i.e., every item is either black or white. In addition to the capacity constraint, we require that no two items of the same color are packed into a bin consecutively.

There are three variants of interest, as follows:

Unrestricted offline: In this setting the entire set of items is known in advance, and they are given as a set. Thus, it is possible to pack them in any order. In this case the items packed into a bin can be reordered, and the assumption that no two items of the same type are packed consecutively is equivalent to the assumption that given a bin A, if it contains $b(A)$ black items, and $w(A)$ white items, then $|w(A)-b(A)| \leq 1$. This variant was studied in the conference version of this submission [1].

Online: The items are presented one by one according to a list L, and no information on future items is given in advance; each item can be packed only into a bin where it fits both with respect to its size, and where the packing does not violate the condition on types, i.e., if it is packed into a non-empty bin, then it can be packed only if the last item packed into the bin is of a different type. Obviously, if a new item cannot be packed into a non-empty bin, then it must be packed into a new empty bin (but it is always allowed to pack a new item into a new bin).

Restricted offline: This variant belongs to a class of offline problems where the input is seen as a sequence that is known to the algorithm in advance (see the terminology of [3]). In this case the items still are given as a list L, and they have to be packed sequentially according to this list, but the algorithm has complete information, that is, the order of items and their properties (sizes and types) are known in advance.

Motivation. One application of black and white bin packing is optimization of the distribution of TV or radio programs and their commercial breaks, or music and another kind of program contents on a radio channel, mainly for online radio programs. The bins correspond to the blocks of programs (at many stations these are one-hour intervals) and the black and white items represent the two kinds of contents, item sizes meaning program duration. The model

[1] A more standard assumption is $0 < p_i \leq 1$, however in the problem studied here, zero sized items cannot be neglected, and thus we consider such items as well.

also makes it possible to optimize similar online contents (as an example, information and advertisement alternately, like on many video-content sharing portals, e.g. Youtube) onto modern mobile phone devices. On mobile phones the contents often are observed in a band-like arrangement because on the tiny display they fit under each other only. Here bin size means the maximum amount of information fitting on one screen.

In this work we study the online problem. We compare our algorithms to restricted optimal offline algorithms. This is motivated by the following example, showing the difference between the unrestricted offline and the restricted offline variants. Consider an input consisting of N white items, each of size zero, followed by N black items, each of size zero. A solution for the unrestricted offline problem can pack all the items into a single bin, but a solution to the restricted offline problem uses N bins, as the first N items must be packed into N different bins. An online algorithm must pack each item as it arrives and cannot reorder the items, thus, naturally, we compare it to restricted offline algorithms, that have the same restriction.

An interesting feature of the problem is that unlike many other packing problems, is it non-monotone in the following sense. Given a sequence, it is possible that as a result of deleting some items from the sequence, the cost of an optimal solution can increase. This is true for all the variants. Consider of $2N$ zero sized items as above (N items of each color), where the items of each color alternate. These items can be packed into one bin for all variants, but if all white items are removed from the input, then any feasible packing requires N bins.

For an algorithm ALG, let ALG denote both the algorithm and its cost (the number of bins). We also use $ALG(L)$ for the cost of ALG and an input list L. An optimal restricted offline solution is denoted simply as OPT (or $OPT(L)$ for an input list L). For any set S of items let $P(S)$ denote the total size of items in the set S.

During the packing procedure, at each time, we call a bin where the last packed item is black, a *black bin*, and analogously *white bin* if the last packed item is white. The *level* or *load* of a bin means the total size of items already packed into the bin.

To evaluate the efficiency of an *online* algorithm, *competitive ratio* is one of the standard measures. If an online algorithm always achieves a solution within a factor ρ of the cost of an optimal (restricted) offline algorithm, we say that the online algorithm is *ρ-competitive*. More accurately, the *absolute competitive ratio* of A is defined as

$$R_{A,abs} = \sup_{L} \left\{ A(L)/OPT(L) \right\},$$

while

$$R_{A,as} = \lim_{n\to\infty} \sup \sup_{L} \left\{ A(L)/OPT(L) \mid OPT(L) \geq n \right\},$$

is called the *asymptotic competitive ratio* of A, and when the competitive ratio of an algorithm does not exceed R', we say that it is R'-competitive. In the *offline* setting the analogous measures are called *approximation ratio*. For both settings together we simply use the term *performance guarantee*.

Classic algorithms and results for the standard bin packing problem. Bin packing (without item types) was introduced by Johnson [16] and by Ullman [25] who introduced a number of natural online algorithms (see also [18,17,7,6]). The algorithm Next Fit [17] keeps only one open bin at any time, and if the next item cannot be packed into the open bin, then the bin gets closed and this item is packed into a newly opened bin. With a somewhat different approach where opened bins do not get closed, in the case of the First Fit, Best Fit, or Worst Fit algorithms [17,25,18], the next item is always packed into the first bin where it fits, into a bin with highest level where it fits, or into a bin with lowest level where it fits, respectively; and if there is no such bin, the item is packed into a new bin. The generalization of the latter three algorithms is called Any Fit; these algorithms are allowed to pack the new item into any open bin where it fits. The item is packed into a new bin if and only if there is no such bin (where it can be packed). The algorithms are abbreviated as NF, FF, BF, WF, and AF, respectively. Clearly, FF, BF and WF are restricted versions of AF.

A totally different packing idea is used by the algorithm $Harmonic(K)$ [20]. The items here are classified according to their sizes, items from interval $(\frac{1}{i+1}, \frac{1}{i}]$ belong to *Class* i for $1 \leq i \leq K-1$, where $K \geq 2$ is a fixed integer. The smaller items, i.e., items with sizes at most $\frac{1}{K}$, belong to *Class* K. Through the packing process, any bin only contains items from one class. If an incoming item fits into a bin of the class of the item then the item is packed there, otherwise a new bin is opened for this class of items. Thus, in this algorithm NF is applied on each class independently. While for a class $i < K$ applying an AF algorithm would get the same result (as exactly i items can be packed into one bin), for class K it is possible that applying some AF algorithms we can get slightly different outputs. The asymptotic performance guarantee of $Harmonic(K)$ is approximately 1.69103 if K is chosen to be sufficiently large, and the performance guarantee improves by increasing K.

As for AF algorithms, FF has an asymptotic performance guarantee 1.7, and a parametric performance guarantee $1 + 1/d$ for any parameter value of $d \geq 2$ such that item sizes do not exceed $\frac{1}{d}$, see [18]. A sequence of papers studied the absolute performance guarantee and the additive constant C such that $FF(L) \leq \frac{17}{10} OPT(L) + C$, see e.g. [24,27,4,24]. Finally, Dósa and Sgall [8] answered the question, that as it was conjectured by many people $FF(L)/OPT(L) \leq 1.7$ holds for any input L. Thus, not only that the asymptotic ratio 1.7 is tight for FF, but this is also the absolute performance bound. The current champion algorithm among algorithms for standard online bin packing is Harmonic++ by Seiden [23], which has an asymptotic competitive ratio of 1.58889. The best lower bound on the asymptotic competitive ratio was 1.5401 [26] for many years, and recently Balogh et al. improved it to $\frac{248}{161} \approx 1.5403$ [2]. The offline version of standard bin packing admits an APTAS and an AFPTAS [12,19].

Other variants. Many variants of bin packing have been studied (see e.g. [6]), and we briefly mention one variant where the input must be treated as a sequence rather than a set. The *bin packing problem with LIB constraint* (Largest Item in Bottom) was introduced in [21,22]. In this model the items are given by a list, and must be packed in this order. The items in a bin must be packed sorted by non-increasing order. That is, an item can only be packed into a bin if the previous item packed there is no smaller (and if the load of the bin is sufficiently small). The best lower bound on the asymptotic competitive ratio known for this problem is 2 [10], while FF was studied in several papers [21,22,10], and finally it was shown in [9] that the absolute competitive ratio of FF is at most $\frac{13}{6}$ and at most $2 + 1/d(d + 2)$ for the parametric case, where no item has size above $\frac{1}{d}$, for a given integer d.
Lower bounds on the number of bins. It is obvious that the total size of the items is a lower bound on the cost of any solution, this lower bound is sometimes used in the analysis of bin packing problems. Let this lower bound be denoted as LB_1. Thus, for any input L, the value $LB_1(L) = P(L)$ is a valid lower bound. The examples above demonstrate that this lower bound can be much smaller than the cost of an optimal solution (even an optimal unrestricted offline solution), due to the constraint on types. While for standard bin packing an analysis using this lower bound only already shows that all AF algorithms and NF are 2-competitive, BWBP requires stronger lower bounds (for example, if all items are of one type, then the optimal number of bins is equal to the number of input items). In Section 4 we develop other lower bounds on the cost of a restricted offline solution that are based in particular on the types of items, and we use it to analyze a new online algorithm that we design.

Our results. First, we consider the classic algorithms. We show that the performance of NF and $Harmonic(K)$ is unbounded. This holds for any $K \geq 2$ if $Harmonic(K)$ applies NF on each class, and as K grows to infinity even if $Harmonic(K)$ applies AF on class K. We also show that FF, BF, and WF have asymptotic competitive ratios of at least 3, and even for the parametric case, the asymptotic competitive ratio exceeds 2. Nevertheless, we prove the AF algorithms have constant absolute competitive ratios of at most 5. We design a different online algorithm, called Pseudo, and show that its absolute competitive ratio is exactly 3. We also show a surprising lower bound of $1 + \frac{1}{2 \ln 2} \approx 1.7213$ on the asymptotic competitive ratio of any deterministic or randomized online algorithm.

2 Lower bounds for classic algorithms

There is a natural way to define the appropriate versions of NF and all the AF algorithms for BWBP. The algorithms are defined in the same way as in the pure online case, in the sense that if NF cannot pack an item into its open bin, no matter if this is due the the constraint on types, or due to load, it closes this bin and opens a new bin. Similarly, for AF algorithms, an item is packed into one of the bins where it can fit and the last packed item is of another type, and a new bin is opened if and only if such a suitable bin does not exist. We can define $Harmonic(K)$ similarly as earlier, however in this case an incoming item from class i ($i < K$) is packed into an existing bin of type i if there are less than i items in the bin, and the colors of the new item and the bin are different. Otherwise a new bin of this class i is opened and the item is packed there. The situation is similar in class K: a new bin is opened in this class and the incoming item is packed there, if the item does not fit into the last open bin, or in the case if the last bin and the incoming item have the same color.

On the negative side, we observe that neither of the algorithms mentioned above can work well. The situation is worst in case of NF.

Proposition 1 *NF and $Harmonic(K)$ do not have constant asymptotic competitive ratios.*

Proof Let us consider the next sequence. Let M be a large integer. The items are presented in M batches (the items in a batch are obviously presented one by one), any item has size 0, and there are $M + 1$ items in each batch, all having the same color. Batches of an odd index have black items, and other batches have white items. We find that NF can only combine the last item of a batch with the first item of the following batch. Therefore, NF uses $M + 1$ bins for the first batch, and M additional bins for any other batch, for a total of $M^2 + 1$ bins. A restricted offline algorithm can pack all items into $M + 1$ bins, where the bin of index i will have every item that has the index i in its batch.

If $Harmonic(K)$ applies NF to class $K \geq 2$, then this example is valid for it too. Otherwise, consider an input where there are two kinds of items. Black items, each of size $\frac{2K-1}{2K(K-1)}$ (we have $\frac{1}{K} < \frac{2K-1}{2K(K-1)} < \frac{1}{K-1}$), and white items, each of size $\frac{1}{2K(K-1)}$. There are $2M(K-1)$ items, and the sequences alternates between items of the two kinds, starting with a black item. As the white items belong to class K while black items belong to class $K-1$, the algorithm is forced to use a dedicated bin for each item. A packing created by running NF on the input has $2M$ bins. Thus, the asymptotic competitive ratio is at least $K - 1$, and it grows to infinity as K grows. □

Theorem 1 *The asymptotic competitive ratios of algorithms FF, BF, and WF are at least 3. Moreover, in the parametric case, if all item sizes are at most $1/d$, the parametric asymptotic competitive ratio of algorithms FF, BF and WF are at least 3, 3, and $1 + \frac{d}{d-1}$, respectively.*

Proof We start with a lower bound for FF. Let n be a fixed big integer. The lower bound construction of an input L' consists of three batches of items. The first batch contains $2n^2$ items, each of size $\frac{2}{4n+1}$, alternating between black and white (the first item is black). FF packs them into n white bins, each such bin contains n black and n white items, and their loads are $\frac{4n}{4n+1}$. Next, in the second batch there come n white items of size $\frac{1}{4n+1}$ each, and FF is forced to open a new bin for each of them. These bins are white as well, and have loads of $\frac{1}{4n+1}$. Finally, in the third batch there are $2n$ items, each of size $\frac{1}{4n+1}$, again alternating between white and black, starting with a black item. The black items can be packed into the first n bins. Packing each such item results in the load 1, so no such bin will contain any further items (as all the items of this input have positive sizes), and thus all bins that can still receive items are white at all times, so the white items of the last batch will be packed into dedicated bins. We get a total of $3n$ bins, i.e. $FF(L') = 3n$.

Now we show that the items in list L' can be packed into $n+1$ bins (i.e., the restricted optimal offline solution requires at most $n+1$ bins). The first item of the list (which is black) is packed into a a special bin. The same bin will contain the first item of the second batch, and all the items of the third batch. The first item packed into the special bin is black, the second one is white, and the remaining items alternate between the colors, starting with a black item. The total size of these items is $\frac{2}{4n+1} + \frac{1}{4n+1} + \frac{2n}{4n+1} = \frac{2n+3}{4n+1} \leq 1$ for any n.

The items that remain unpacked so far are those in the first batch (except for first item) and the items in the second batch, except for the first such item. The $2n^2 - 1$ items of the first batch (each of size $\frac{2}{4n+1}$ where the first item is white) are packed into n bins. The first $n-1$ bins among these n bins are black, and they have levels of $\frac{4n}{4n+1}$, and the level of the last such bin is $\frac{4n-2}{4n+1}$, and it is white. Thus the $n-1$ unpacked white items from the second batch can be packed into the first $n-1$ bins among these n bins, and each of them will be the last item of the bin. Thus $OPT(L') \leq n+1$.

It is simple to check that BF makes the same packing as FF. Thus we get that the asymptotic competitive ratios of algorithms FF and BF are at least 3. The value of n can be chosen to be arbitrarily large, and thus this lower bound holds even in the parametric case for any d.

Applying WF to the previous construction creates a different packing that uses $2n$ bins. We use a different construction to show that WF has an asymptotic competitive ratio of at least 3, and furthermore, in the parametric case the competitive ratio of WF is at least $1 + \frac{d}{d-1}$.

Let the list L consist of $dn+2$ batches, where $d \geq 2$ is some fixed integer, N is a sufficiently large integer, and $n = N(d-1) + d - 2$. We have that $nd + 1 = dN(d-1) + d(d-2) + 1 = (d-1)(Nd + d - 1)$ is divisible by $d-1$. Let δ be a small positive value, where $\delta \leq \frac{1}{6dn+3}$. In batch i, $(i = 1, \ldots, dn+1)$ the next four items come in the following order:

- A_i, a white item with size $1/d - 4\delta$,
- B_i, a black item with size 3δ,
- C_i, a white item with size 3δ,
- D_i, a black item with size 3δ.

In the last batch, i.e., the $(dn+2)$-th batch, n black items are presented, each one with size δ.

WF packs the items as follows. The items in the first $d-1$ batches are packed into the first WF bin, as the colors of these items are alternating, and they fit into one bin. The level at this point of the first WF bin is $(d-1)(1/d+5\delta) = 1 - \frac{1}{d} + 5(d-1)\delta \geq 1 - \frac{1}{d} + 5\delta$ since $d \geq 2$. Thus no additional A_i item will fit into this bin. Since the resulting bin is black, no further black items will be packed into the bin, and white items will always be packed into the current last bin, since its load will be smaller. Therefore, the items from the next $d-1$ batches are packed into the second WF bin, and the process continues in the same way. After packing the first $dn+1$ batches there are $\frac{dn+1}{d-1}$ black bins. After this moment WF creates n further bins with one black item of size δ packed into each such bin. Thus $WF(L) = \frac{dn+1}{d-1} + n$. (Note that FF and BF would make much better packing, as the small white and black items would be packed into such earlier bins what are not used later by WF.)

We define a feasible packing as follows. The items C_i and D_i for $1 \leq i \leq dn+1$ are packed into a single bin. This is possible since their colors alternate, and their total size is $3(dn+1)\delta \leq \frac{3dn+3}{6dn+3} < 1$. The remaining subsequence for the first $dn+1$ batches also contains items of alternating colors. The first item (A_1) is packed into a bin, and the remaining items are packed such that each bin contains $2d$ items, and the last item (B_{dn+1}) is combined into the bin of A_1. That is, for $i = 1, \ldots, n$, items $B_{(i-1)d+1}, A_{(i-1)d+2}, B_{(i-1)d+2}, \ldots, B_{id}, A_{id+1}$ are packed into a bin. The load of such a bin is $d(\frac{1}{d} - 4\delta + 3\delta) = 1 - d\delta$. These bins are white, and therefore each such bin can receive one black item of the last batch. We find that $OPT(L) \leq n+2$, and $WF(L)/OPT(L) \geq \frac{\frac{dn+1}{d-1}+n}{n+2}$ holds. This ratio can be arbitrarily close (from below) to $1 + \frac{d}{d-1}$ as n tends to infinity. This value is 3 if $d = 2$, implying the lower bound for $d = 1$ as well. □

3 A lower bound on the competitive ratio of any online algorithm

We prove a lower bound for arbitrary deterministic or randomized online algorithms. The proof is given for deterministic algorithms, while replacing the variables used in the proof by their expected values does not harm the correctness, and using Yao's method [28] (that states that it is possible to assume that the algorithm is deterministic while the input is randomized) the second part of the input is picked uniformly at random among all the options.

Theorem 2 *There is no online algorithm for BWBP with an asymptotic competitive ratio smaller than* $1 + \frac{1}{2\ln 2} \approx 1.7213$.

Proof The construction consists of a list of very small items, and one list concatenated with it, where the second list has larger items, all of the same type. The lists are defined as follows. Let $a = \frac{1}{4k}$, where k is large positive integer, and let $y_i = \frac{i}{2k}$ for $k \leq i < 2k$. We have $\frac{1}{2} \leq y_i < 1$. Items of size a are called small. Now let $n > k$ be an additional large positive integer. Define the first list, L_0, as a list of $4kn$ items of size a. The list L_0 contains a large number identical very small items of size a, where the cumulative size of the small items is n. The colors of the items are alternating, i.e., the items with odd indices are white, while the items with even indices are black (thus, the first small item is white).

Define L_{iw} as a list of $n_i = \frac{n}{y_i}$ white items, each of size $0 < 1 - y_i \leq 1/2$, and we have $n_i > n$. Similarly define L_{ib} to be a list of $n_i = \frac{n}{y_i}$ black items, each of size $1 - y_i$. The proof of the lower bound is based on the analysis of the behavior of an arbitrary online algorithm for the lists L_0L_{iw} and L_0L_{ib} and all possible values of i ($k \leq i < 2k$), that is, $2k$ inputs in total.

Lemma 1 *$OPT(L_0L_{iw}) = n_i$ and $n_i \leq OPT(L_0L_{ib}) \leq n_i + 1$.*

Proof Clearly, all the items of the lists L_{ib} and L_{iw} must be packed into different bins, and thus $OPT(L_0L_{iw}) \geq n_i$ and $OPT(L_0L_{ib}) \geq n_i$ hold.

We present an algorithm that packs the lists L_0L_{ib} and L_0L_{iw} using the stated number of bins. For L_0L_{iw}, we create n_i identical bins, such that each bin contains $2i$ items of alternating colors, starting with a white item. The resulting bins are black. The total number of packed items is $2i \cdot n_i = 2i \cdot n \cdot 2k/i = 4nk$. The loads of these bins are $\frac{2i}{4k} = \frac{i}{2k}$, thus each bin can receive a white item of size $1 - y_i = 1 - \frac{i}{2k}$.

The packing for L_0L_{ib} is similar, but the bins with $2i$ small items should be white, thus, the very first white item is packed into the first bin, then $n_i - 1$ white bins, each containing $2i$ items (the first of which is black), are created, and a white bin with $2i-2$ items is created as well (these are the bins of indices $2, \ldots, n_i + 1$). The last small black item is combined with the first small item into the same bin, while the black items of size $1 - y_i$ are combined into the other bins. □

We introduce the following notation. Consider the state of an online algorithm A just after all small items were packed. Let z_{kb} be the number of black bins that contain at most $2k$ small items. Let z_{lb} for $k+1 \leq l \leq 2k$ denote the number of black bins that contain $2l$ or $2l-1$ items. The variables z_{kw} and z_{lw} for $k+1 \leq l \leq 2k$ are defined analogously for white bins.

Lemma 2 *The next inequality must hold:*

$$4nk \leq \sum_{l=k}^{2k} 2l(z_{lw} + z_{lb})$$

Proof There are $z_{lw} + z_{lb}$ bins that contain at most l items, and the bins were partitioned such that every bin is counted in one variable. □

Next, we analyze the cost of the algorithm in each one of the cases.

Lemma 3 *The cost of the algorithm for the different inputs satisfies:*

$$A(L_0L_{iw}) \geq n_i + \sum_{l=k}^{2k} z_{lw} + \sum_{l=i+1}^{2k} z_{lb}$$

and

$$A(L_0L_{ib}) \geq n_i + \sum_{l=k}^{2k} z_{lb} + \sum_{l=i+1}^{2k} z_{lw} \ .$$

Proof Consider the list L_0L_{iw} for a given value of i (the proof for L_0L_{ib} is the same). The last n_i items are white, and each one of them must be packed into a separate bin. A white item of size $1-y_i$ can only combined into a black bin that contains at most $2i$ small items. □

Let $x_l = z_{lw} + z_{lb}$, and $A_i = A(L_0L_{iw}) + A(L_0L_{ib})$. We find that $A_i \geq 2n_i + \sum_{l=k}^{2k} x_l + \sum_{l=i+1}^{2k} x_l$, and $4nk \leq \sum_{l=k}^{2k} 2lx_l$.

Assume that R is the asymptotic competitive ratio of A. Since in all cases the optimal cost is at least n, there exists a value ε_n such that $A(L) \leq (R+\varepsilon_n)OPT(L)$ for all the lists considered here (and $\lim_{n\to\infty} \varepsilon_n = 0$). Thus, $A_i \leq 2(R+\varepsilon_n)(n_i+1)$. Using the lower bound on A_i we find $2n_i + \sum_{l=k}^{2k} x_l + \sum_{l=i+1}^{2k} x_l \leq A_i \leq 2(R+\varepsilon_n)(n_i+1)$, or alternatively, $\sum_{l=k}^{2k} x_l + \sum_{l=i+1}^{2k} x_l \leq (2R+2\varepsilon_n-2)(n_i+1)+2$. Taking the sum over all values of i ($k \leq i < 2k$), we find:

$$\sum_{i=k}^{2k-1}\left(\sum_{l=k}^{2k} x_l + \sum_{l=i+1}^{2k} x_l\right) \leq (2R+2\varepsilon_n-2)(\sum_{i=k}^{2k-1}(n_i+1)) + 2k.$$

Rearranging the left hand size, writing $n_i = \frac{n}{y_i} = \frac{2kn}{i}$, and using $2nk \leq \sum_{i=k}^{2k} ix_i$ we get:

$$2nk \leq \sum_{i=k}^{2k-1} ix_i \leq (2R+2\varepsilon_n-2)(2nk\sum_{i=k}^{2k-1}\frac{1}{i} + k) + 2k \ .$$

Now let $f(k) = \sum_{i=k}^{2k-1} \frac{1}{i}$. We get $2k(n-1) \leq 2(R+\varepsilon_n-1)(2nkf(k)+k)$, or alternatively, $R \geq \frac{n-1}{2nf(k)+1} - \varepsilon_n + 1 = \frac{1-1/n}{2f(k)+1/n} - \varepsilon_n + 1$. We have $\lim_{k\to\infty} f(k) - \ln 2 = 0$, thus, $f(k) = \ln 2 + \delta_k$, where δ_k tends to zero when k tends to infinity. Thus, $R \geq \frac{1-1/n}{2\ln 2 + 2\delta_k + 1/n} - \varepsilon_n + 1$. Letting k (and therefore also n) tend to infinity we find $R \geq \frac{1}{2\ln 2} + 1$. □

An important aspect of this lower bound is the comparison to the known 1.58889-competitive algorithm (that we mentioned earlier) for classic online bin packing. It follows that BWBP is conceptually harder.

4 Competitive algorithms

For the analysis of algorithms, we start with defining a new lower bound for optimal solutions.

4.1 A second lower bound for optima of restricted offline instances

The lower bound LB_1 mentioned above is computed based on item sizes only. Here, we present a further bound, which is determined by the color pattern of list L, that is, by the list of item types. These lower bounds on restricted offline algorithms will allow us to analyze online algorithms. At the end of the paper we define a third lower bound which one takes both the color pattern and the item sizes into account. In this section the number of items in an input is denoted by n, and the list of items is $L = p_1, p_2, \dots, p_n$.
The lower bound LB_2. We have seen in the introduction that by deleting some items, the optimal cost can possibly increase. This is not the case, however, if we delete items from the beginning of the sequence, or from the end of the sequence. More exactly, the next lemma holds.

Lemma 4 *Given an input L, partition it into $L = L_1L_2L_3$ (where L_1 and L_3 may be empty). We have $OPT(L_2) \leq OPT(L)$.*

Proof It is sufficient to show that given an input L, and L' that results from L by removing the first item and L" results from L by removing the last item, then $OPT(L') \leq OPT(L)$ and $OPT(L") \leq OPT(L)$ must hold.

Consider an optimal solution for L, and remove the last item. This item was packed last in a bin, and thus it can be removed without violating the condition on types, and we get a valid packing for L". This proves $OPT(L") \leq OPT(L)$. The proof for L' is similar, only in this case the first item is removed. The first item is packed first in some bin, and thus its removal does not violate the condition on the types, and we get a valid packing for L'. This proves $OPT(L') \leq OPT(L)$. □

Let $s_i = 1$ if the ith item is black and let $s_i = -1$ if it is white. Let $1 \leq i < j \leq n$ be two arbitrary indices, and let $LB_2(i, j) = \left|\sum_{k=i}^{j} s_k\right|$. We call 'segment' a set of subsequent elements of the list without any gaps. If S means that segment which starts with i and ends with j, then $LB_2(i, j)$ will be also denoted as $LB_2(S)$. Moreover the maximum value of $LB_2(i, j)$ will be denoted by LB_2. Then the next lemma holds.

Lemma 5 *LB_2 is a lower bound on the optimal cost, that is, $OPT(L) \geq LB_2$.*

Proof By Lemma 4, it suffices to consider a shortest subsequence (with the smallest value of $j-i$) attaining the maximum value of $LB_2(i, j)$, i.e. LB_2. We assume that the sum is positive, i.e., that the number of black items exceeds the number of white ones (the other case is symmetric), and without loss of generality we assume $i = 1$ and $j = n$.

Consider a specific algorithm that packed the n items into a certain number of bins. Let us introduce the following notation:

$a_k = \sum_{i=1}^{k} s_i$, and under the assumed packing procedure, let b_k denote the number of *white* items among the first k items that started a new bin, i.e., each of which has been packed as a first item into a bin. By the minimality of $j-i$, $a_k > 0$ holds for all $1 \le k \le n$. Additionally, by definition, $0 \le b_1 \le b_2 \le ... \le b_n$ must hold. We will prove the following claim by induction on k. The lemma will follow from it.

Claim After packing the first k items, the number of *black bins* is exactly $a_k + b_k$.

Proof of the claim The first item is black, hence we have $a_1 = 1$, $b_1 = 0$, and the number of black bins is $a_1 + b_1 = 1$. Suppose that the claim is valid after the packing of the first $k-1$ items, and consider the kth item. If it is black, then $a_k = a_{k-1} + 1$ and $b_k = b_{k-1}$, hence $a_k + b_k = a_{k-1} + b_{k-1} + 1$. We calculate the change in the number of black bins. The kth item cannot be packed into a black bin. If the kth item is packed into an empty bin, then this new bin is a new black bin. If the kth item it is packed into a white bin, then this bin becomes black, and the number of black bins also increases by 1 in this case as well, as claimed.

If the kth item is white, then $a_k = a_{k-1} - 1$. The kth item cannot be packed into a white bin. If it is packed into a new bin, then $b_k = b_{k-1} + 1$, and hence the sum $a_k + b_k = a_{k-1} + b_{k-1}$ remains unchanged, and so does the number of black bins. Otherwise, it is packed into a black bin. The number of black bins decreases by precisely 1. But then we have $b_k = b_{k-1}$, therefore $a_k + b_k = a_{k-1} + b_{k-1} - 1$ holds, hence the sum decreases by 1 as well. This proves the claim.

The lemma follows by letting $k = n$, and applying the claim on a restricted optimal algorithm; we have $OPT \ge a_n = LB_2$. □

Lemma 6 *Let L be an instance of BWBP for which AF creates s bins, all of them having level at most* $1/2$. *Then $LB_2 \ge s$.*

Proof Denote the s bins by $A_1, \ldots, A_s$. Let us introduce the following notation:

- $b(i)$: the bottom item in A_i, $1 \le i \le s$;
- $t_i(j)$: the top item in A_j at the moment when A_i is opened (i.e., when $b(i)$ arrives), $1 \le j \le i-1$. For formal reasons we also write $t_i(i) := b(i)$.

Note that all $t_i(j)$ have the same color as $b(i)$, for otherwise item $b(i)$ could be packed into A_j and hence AF would not be allowed to open the bin A_i.

The key point in the proof is the following observation, from which the lemma follows.

Claim There exists a segment S of L that ends with item $b(s)$ and satisfies $LB_2(S) \ge s$ and it is the color of $b(s)$ that occurs at least s more times in S than the other color.

Proof of the claim We prove the *Claim* by induction on s. It clearly holds for $s = 1$ because $b(1)$ is the very first item in L.

Let $s > 1$ and assume that the claim is valid for the smaller value $s' = s-1$. That is, there exists a segment S' of L ending with $b(s-1)$, in which the color of $b(s-1)$ occurs at least $s-1$ more times than the other color.

Case 1. Items $b(s-1)$ and $b(s)$ have the same color.

In this case we extend S' to the segment with the items of L arriving between $b(s-1)$ and $b(s)$, also including $b(s)$. In this larger segment S, inside each A_j $(1 \leq j \leq s-1)$ the number of black and white items arriving after $t_{s-1}(j)$ is the same because they alternate and all the four items $b(s-1), b(s), t_{s-1}(j), t_s(j)$ have the same color. Hence, the appearance of $b(s)$ in S increases the black-white difference by 1 and we obtain $LB_2(S) = LB_2(S') + 1 \geq s$.

Case 2. Item $b(s-1)$ is black and $b(s)$ is white (or vice versa).

Now we consider the segment S ending with $b(s)$, which consists of all the items arriving after $b(s-1)$. Since $t_{s-1}(j)$ is black and $t_s(j)$ is white for all $1 \leq j \leq s-1$, alternating colors imply that inside each A_j the number of white items packed above $t_{s-1}(j)$ is larger by 1 than the number of black items packed there. Counting also $b(s)$, which is the single item of S in A_s, this immediately implies the inequality $LB_2(S) \geq s$. □

Corollary 1 *If all items have zero sizes (or alternatively, the bins have infinite capacities), then LB_2 is equal to the restricted offline optimal value. Moreover, any AF algorithm creates a solution with exactly LB_2 bins.*

Proof Suppose AF creates s bins. Let us realize that all AF bins have zero level, thus the condition of Lemma 6 is satisfied, thus $s \leq LB_2$ follows. On the other hand, from Lemma 5 we know $LB_2 \leq OPT$. Thus $s \leq LB_2 \leq OPT$, and since s cannot be smaller than OPT as AF creates s bins, $AF = s = LB_2 = OPT$ follows. □

We need another lemma for the behavior of algorithm AF in the general case.

Lemma 7 *Let L be an instance of BWBP for which the algorithm AF creates $s+t$ bins, s of them having level at most $1/2$ and t of them with level exceeding $1/2$. Then $LB_2(L) \geq s-t$.*

Proof Denote the s bins of level at most $1/2$ by $A_1, \ldots, A_s$ and the t bins of level higher than $1/2$ by $B_1, \ldots, B_t$. First we restrict our attention to the shorter list L_A which consists of the items packed into $A_1, \ldots, A_s$. The order of items in L_A is supposed to be the same as they appear in L. It is clear that AF generates the solution $(A_1, \ldots, A_s)$ for the instance L_A.

We have seen in Lemma 6 that there exists a segment S of L_A which satisfies $LB_2(S) \geq s$. We now extend S with those items of $L \setminus L_A$ which arrive later than the first item of S and earlier than the last item of S. This yields a segment S^* of the original list L. The items packed from S^* into any B_i $(1 \leq i \leq t)$ respect their order in L, with no interruption by items from $L \setminus S^*$. Hence, their colors alternate in B_i, therefore the number of black items in $B_i \cap S^*$ is at most one larger and at most one smaller than the number of white items in $B_i \cap S^*$. Thus, $LB_2(S^*) \geq LB_2(S) - t \geq s - t$. This completes the proof of the lemma. □

4.2 An efficient online algorithm Pseudo

Before defining a new algorithm, we prove that the absolute approximation ratio of AF is at most 5 (for items of arbitrary sizes).

Theorem 3 *AF has an absolute approximation ratio of at most* 5.

Proof We use the notations of Lemma 7, hence let the number of bins created by AF be $s+t$, s of them having level at most $1/2$ and t of them with level exceeding $1/2$. Then $LB_2 \geq s-t$. We get

$$\frac{AF}{OPT} \leq \frac{s+t}{\max\{LB_1, LB_2\}} \leq \frac{s+t}{\max\{t/2, s-t\}} \leq \frac{s-t}{s-t} + \frac{2t}{t/2} = 5.$$

□

Next, we define a simple online algorithm with an absolute competitive ratio of 3, and a parametric absolute competitiveness of $1+\frac{d}{d-1}$ if all sizes are at most $1/d$. As $d \to \infty$, this competitiveness approaches 2 from above, while we have seen that FF never can be better than 3-competitive, even for very small items. The algorithm uses the concept of pseudo-bins. These are bins of unbounded capacity. These bins are split further into valid bins in an online fashion.

Algorithm Pseudo

1. Arriving items are packed into pseudo-bins of infinite size using FF.
2. The items of each pseudo-bin are packed into bins using NF (NF is applied obliviously of item types).

Note that the choice of NF for the packing of items of each pseudo-bin is not arbitrary. The subsequence of items of each pseudo-bin has items of alternating colors. As this subsequence is split into subsequences of consecutive items using NF, the resulting bins also have items of alternating colors, and thus the output packing is valid (even though NF is applied obviously of the types). The correctness follows from this observation. On the other hand, while the algorithm uses FF for packing the pseudo-bins, replacing FF by some AF algorithm does not harm the validity of the upper bounds proved in the next theorem.

Theorem 4 *Algorithm Pseudo has (an asymptotic and absolute) competitive ratio of* 3*, and it has (an asymptotic and absolute) competitive ratio of* $1+\frac{d}{d-1}$ *in the parametric case, where all items have sizes at most* $1/d$.

Proof We start with upper bounds on the absolute competitive ratio, and afterwards we prove lower bounds on the asymptotic competitive ratio.

Let L be an input. By Corollary 1, the number of created pseudo-bins is exactly $LB_2(L)$. Suppose that pseudo-bin i is split into exactly $m_i \geq 1$ valid bins. Then, the total number of bins packed by the algorithm is exactly $\sum_{i=1}^{LB_2(L)} m_i$.

As the items of each pseudo-bin are packed using NF, every two consecutive bins of the same pseudo-bin have items of a total size exceeding 1. Moreover, if all item sizes are at most $\frac{1}{d}$ and $m_i > 1$, then all packed bins but the last one (for a given pseudo-bin) are packed with a total size that exceeds $1 - \frac{1}{d}$. Partitioning the bins into consecutive pairs (and neglecting the last bin, if m_i is odd), in the absolute case ($d = 1$) we find

$$LB_1(L) \geq \sum_{i=1}^{LB_2(L)} \left\lfloor \frac{m_i}{2} \right\rfloor \geq \sum_{i=1}^{LB_2(L)} \frac{m_i - 1}{2} = \sum_{i=1}^{LB_2(L)} \frac{m_i}{2} - \frac{LB_2(L)}{2} .$$

Thus, we have $\sum_{i=1}^{LB_2(L)} m_i \leq 2LB_1(L) + LB_2(L) \leq 3OPT(L)$. In the parametric case we find

$$LB_1(L) \geq \sum_{i=1}^{LB_2(L)} (1 - \frac{1}{d})(m_i - 1) \geq (\frac{d-1}{d}) \sum_{i=1}^{LB_2(L)} (m_i - 1)$$

$$= (\frac{d-1}{d})(\sum_{i=1}^{LB_2(L)} m_i - LB_2(L)) .$$

Thus, we have $\sum_{i=1}^{LB_2(L)} m_i \leq \frac{d}{d-1} LB_1(L) + LB_2(L) \leq (1 + \frac{d}{d-1})OPT(L)$.

Now we present a simple example showing that the analysis of the algorithm is tight, and already the asymptotic competitive ratio of the algorithm is 3. The sequence consists of $3N$ items for a large integer N. For $i = 1, \ldots, N$, the item of index $3i-2$ is white and the items of indices $3i-1$ and $3i$ are black (i.e., the sequence of item colors is W,B,B,W,B,B,...). Thus Pseudo creates one bin with the items of indices $1, 2, 4, 5, 7, 8, \ldots$ (the items of indices $3i-2$ and $3i-1$ for all i are in this bin), and N bins with the items of indices $3, 6, \ldots$ (i.e., for all i, the item of index $3i$ is packed into a dedicated pseudo-bin). The sizes of the items of indices $3i-2$ and $3i$ (for $1 \leq i \leq N$) are $1/(2N)$ and the sizes of the items of indices $3i-1$ are 1 (i.e. the middle item in any triple is a large item and the two other items are small items). As a result, the first pseudo-bin of Pseudo is split into valid bins, the algorithm packs every item in a separate bin. So $Pseudo(L) = 3N$. A restricted optimal offline solution, however, packs all items of size $1/(2N)$ into one bin, and every larger item into a separate bin. Thus $OPT = N + 1$. Note that this example is not valid for AF algorithms, that obtain optimal solutions.

For the parametric case with $d \geq 2$, we show that the asymptotic competitive ratio is at least $1 + \frac{d}{d-1}$. The sequence consists of $(4d-1)(d-1)N$ items, partitioned into $N(d-1)$ batches. Each batch has $4d-1$ items. The first two items of each batch are white, and the remaining colors are alternating, starting and ending with black items. Let $\gamma = \frac{1}{4d^2(d-1)N}$. The first item of each batch, as well as the items of indices of the form $4\ell+2$ for $1 \le \ell \le d-1$, i.e., indices $6, 10, \ldots, 4d-2$ (inside the batch) have sizes of $\frac{1}{d} - (d-1)\gamma$ (according to the definition above, they are all white). All items of indices of the form $4\ell+1$ for $1 \le \ell \le d-1$, i.e., indices $5, 9, \ldots, 4d-3$ (inside the batch) have sizes of $d\gamma$ (according to the definition above, they are all black). The remaining items have sizes of $2d\gamma$. There are $2d$ such items, and their colors alternate, starting with a white item, and ending with a black item. An optimal solution packs all items of sizes $2d\gamma$ in one bin. There are $2d(d-1)N$ such items, and their total size is $2d(d-1)N \cdot 2d \cdot \frac{1}{4d^2(d-1)N} = 1$. Moreover, their colors are alternating, and thus this bin is valid. The remaining items of each batch alternate between white large items, having sizes of $\frac{1}{d} - (d-1)\gamma$ and black small items of sizes $d\gamma$. There are d large items and $d-1$ small items in each batch, giving a total of $N(d-1)$ bins. The total size of these items (for one batch) is $d(\frac{1}{d} - (d-1)\gamma) + (d-1)d\gamma = 1$. Pseudo packs all items into one pseudo-bin, except for the items that are second items in their batches, that are packed into dedicated bins. This creates $N(d-1)$ additional bins. We claim that the first pseudo-bin is split into at least Nd bins, and more specifically, that every bin receives at most $d-1$ items of size $\frac{1}{d} - (d-1)\gamma$. Assume by contradiction that some bin receives d such items. As the first pseudo-bin is split into subsequences of items that are packed into it consecutively, there is at least one item of size $d\gamma$ between every pair of items of size $\frac{1}{d} - (d-1)\gamma$. Thus, the total size of items in the bin is at least $d(\frac{1}{d} - (d-1)\gamma) + (d-1)d\gamma = 1 + d(d-1)\gamma > 1$, which is a contradiction. Thus, the algorithm creates at least $N(2d-1)$ bins, for a ratio of $\frac{2d-1}{d-1} = 1 + \frac{d}{d-1}$.

□

4.3 A third lower bound for optima of restricted offline instances

In this section we use the concept of conflict graphs. Such graphs are undirected graphs, where the set of vertices is the set of items, and an edge represents the constraint that the two items cannot be packed into the same bin. Unlike the problem of *bin packing with conflicts* [15, 14, 11], here the conflicts result from analyzing the input rather than from constraints given as an input.

For any list $L = \{p_1, \ldots, p_n\}$ of items, an instance of the problem, we denote by $L = L_B \cup L_W$ the partition into the sets of black and white items.

Definition 1 The conflict graph of black items in an instance of the online or restricted offline problem is an undirected graph, denoted by G_B. It has vertex set L_B. Two items $p_i, p_j \in L_B$ with $1 \le i < j \le n$ are joined by an edge if and only if, for every white item p_k in the range $i < k < j$ we have $p_i + p_j + p_k > 1$. The conflict graph of white items, denoted by G_W, is defined analogously on the vertex set L_W.

For any graph G, we use the standard notation $\omega(G)$ for the *clique number* (largest number of mutually adjacent vertices) and $\chi(G)$ for the *chromatic number* (smallest number of independent sets into which the vertex set can be partitioned). Since no two vertices adjacent by an edge of G_B or G_W can be packed into the same bin, it is immediate by definition that

$$opt(L) \geq \max\{\chi(G_B), \chi(G_W)\} \geq \max\{\omega(G_B), \omega(G_W)\} =: LB_3 \tag{1}$$

holds for all instances $L = L_B \cup L_W$.

Remark 1 Consider an input of $n = 4N$ items, where the items of odd indices are black and have sizes of $\frac{1}{2}$, and the items of even indices are white and have sizes of $\frac{1}{2N}$. For this list L, we have $LB_1(L) = N + 1$. Moreover, as the items have alternating colors, we find that $LB_2(L) = 1$. For this instance, the conflict graph of black items is complete, and thus $LB_3(L) = 2N$. This shows that LB_3 provides new information on optimal solutions.

Definition 2 A graph $G = (V, E)$ is said to be chordal if every induced cycle in G has length 3. In other words, each cycle longer than 3 has a chord. A vertex $v \in V$ is called a simplicial vertex if its neighbors are mutually adjacent. A simplicial order is a linear ordering of the vertex set, say $v_1, v_2, \ldots, v_{|V|}$, such that each v_i is a simplicial vertex in the subgraph induced by $\{v_i, v_{i+1}, \ldots, v_{|V|}\}$.

It is easy to see that every graph admitting a simplicial order is chordal, because no vertex of an induced cycle longer than 3 can occur as the first vertex in any simplicial order.[2] Let us mention some further well-known properties of graphs having a simplicial order. For references, see e.g. [5]. In a fixed ordering $v_1, \ldots, v_{|V|}$ of the vertex set we denote by d_i^+ the number of vertices v_j such that $j > i$ and v_iv_j is an edge.

1. The clique number is equal to $\max_{1\leq i\leq |V|} d_i^+ + 1$.
2. The chromatic number is equal to $\max_{1\leq i\leq |V|} d_i^+ + 1$.

To see property 1, observe that the d_i^+ neighbors following v_i together with v_i induce a complete subgraph, moreover each non-extendable complete subgraph is exactly the closed neighborhood of its vertex that appears first in the simplicial order. In this way a lower bound on the chromatic number is also obtained. Moreover, one can color the graph with that many colors, and hence prove property 2, by assigning the smallest available color to each vertex in *reverse* simplicial order because v_i has at most $\max d_i^+$ colored neighbors when the procedure reaches v_i, hence the '+1' term ensures that there is a free color available for v_i.

Lemma 8 *Suppose that $p_i \in L_B$ is a smallest black item, i.e., let $p_i = \min_{1\leq j\leq |L_B|} p_j$. Then vertex p_i is simplicial in the conflict graph G_B.*

[2] The existence of simplicial order is actually not only sufficient but also necessary for a graph to be chordal; but we shall not need this theorem in our discussion.

Proof Assume that p_i has two non-adjacent neighbors p_j and $p_{j'}$ in G_B. Let $j < j'$. Since $p_j p_{j'}$ is a non-edge, there exists some $p_k \in L_W$ such that $j < k < j'$ and $p_j + p_{j'} + p_k \leq 1$. If $i < k$, using $p_i \leq p_j$ we obtain

$$p_i + p_{j'} + p_k \leq p_j + p_{j'} + p_k \leq 1,$$

and since $i < k < j'$, the inequality above contradicts the assumption that p_i and $p_{j'}$ are adjacent. If $k < i$, then in a similar way we obtain the analogous contradiction that $p_i p_j$ should not be an edge of G_B. □

Lemma 9 *If $p_{i_1} \leq p_{i_2} \leq \cdots \leq p_{i_{|L_B|}}$ is a non-decreasing order of black items, then it is a simplicial order of G_B.*

Proof The assertion follows by the repeated application of the previous lemma. □

Certainly, the analogous properties can be proved in the same way for the conflict graph G_W of the white vertices, too. From this, we can derive the following result.

Theorem 5 *Both G_B and G_W are chordal, and we have $\chi(G_B) = \omega(G_B)$ and $\chi(G_W) = \omega(G_W)$. Moreover, the lower bound in* (1) *can be computed in $O(n^2)$ time.*

Proof The first part of the theorem follows by the properties of the simplicial order as described above. The crucial part to prove in the assertion is that an algorithm with guaranteed quadratic running time can be designed. We apply Algorithm 1. We describe it for G_B; the procedure for G_W is completely analogous. The following variables are introduced:

LB — the currently best lower bound on $\omega(G_B)$, initialized to 1 and updated each time when the treatment of a vertex of G_B is completed.

B — the set of black vertices not treated completely yet.

q_W — currently smallest size of the white items which separate the black vertex under treatment from the members of B that have not yet been considered in connection with the treated one; re-initialized[3] to 2 for each black vertex at the beginning of its treatment.

n_B — number of neighbors of the black item under treatment, in the subgraph induced by the set B in G_B; initialized to 0 for each black vertex at the beginning of its treatment.

Sorting of L_B requires just $O(n \log n)$ comparisons, and it determines a simplicial order on G_B. Then, each round of the main **for** loop determines the number of neighbors of the jth vertex of G_B in the jth induced subgraph (after the deletion of its first $j-1$ vertices). Steps 6–11 proceed forward (by increasing index) in the original order of items in the input, while steps 13–18

[3] The value 2 is chosen to make sure that two black items of size 0 cannot be packed into the same bin if they are not separated by a white item. If all black items have positive size, then also $q_W = 1$ is a suitable initialization.

Algorithm 1 LOWER BOUND $\chi(G_B) = \omega(G_B)$

Require: Black-and-white instance $L = L_B \cup L_W = \{p_1, p_2, \ldots, p_n\}$
Ensure: Value of lower bound $\chi(G_B) = \omega(G_B)$ for $OPT(L)$
1: Sort the black items to ensure $p_{i_1} \le p_{i_2} \le \cdots \le p_{i_{|L_B|}}$
2: $LB := 1$, $B := \{i_1, i_2, \ldots, i_{|L_B|}\}$
3: **for** $j = 1$ to $|L_B|$ **do**
4: $\quad n_B := 0$
5: $\quad q_W := 2$
6: $\quad$ **for** $k = i_j + 1$ to n **do**
7: $\quad\quad$ **if** $p_k \in L_W$ **then**
8: $\quad\quad\quad q_W := \min\{q_W, p_k\}$
9: $\quad\quad$ **else**
10: $\quad\quad\quad$ **if** $p_{i_j} + p_k + q_W > 1$ **then**
11: $\quad\quad\quad\quad n_B := n_B + 1$
12: $\quad q_W := 2$
13: $\quad$ **for** $k = i_j - 1$ downto 1 **do**
14: $\quad\quad$ **if** $p_k \in L_W$ **then**
15: $\quad\quad\quad q_W := \min\{q_W, p_k\}$
16: $\quad\quad$ **else**
17: $\quad\quad\quad$ **if** $p_{i_j} + p_k + q_W > 1$ **then**
18: $\quad\quad\quad\quad n_B := n_B + 1$
19: $\quad LB := \max\{LB, n_B + 1\}$
20: $\quad B := B \setminus \{p_{i_j}\}$
21: **print** LB and **STOP**

proceed backward. The set of neighbors is the union of those in both directions, this is the reason why n_B has to be initialized only once (Step 4). On the other hand, the separating white items that ensure the packability of two black items into the same bin, need to occur on the proper side of the jth vertex of G_B, therefore their size has to be re-initialized for both halves of the main **for** loop (Steps 5 and 12).

Soundness of the algorithm follows from the properties of chordal graphs as quoted above. Each execution of the main **for** loop takes $O(n)$ time because each individual step requires constant time only. Hence, the overall running time is quadratic in the number of items. □

Remark 2 It is worth noting that one may need as much as $\Theta(n^2)$ *steps to check whether a given single vertex of a graph is simplicial. Therefore, it is a substantial improvement in the running time that we find a simplicial order in* $O(n \log n)$ steps. This efficiency is obtained by using the special structure of graphs obtained from instances of Black and White Bin Packing.

An interesting aspect of the time bound $O(n^2)$ is that already checking the adjacency of a single vertex pair in the conflict graph G_B or G_W may take $\Omega(n)$ time.

The following problem may be of interest on its own right: *Characterize the structure of graphs that can occur as conflict graphs of black (or white) items.*

5 Conclusion

We studied a new variant of online bin packing. We showed that it admits constant competitive algorithms. In particular, we have defined a new online algorithm Pseudo, and showed that its (absolute and asymptotic) competitive ratio is 3 and $1 + \frac{d}{d-1}$ in the parametric case, while for FF, BF, and WF, the asymptotic competitive ratios are at least 3, 3, and $1+\frac{d}{d-1}$, respectively, in the parametric case. We have proved, however, that AF algorithms are constant competitive, and their absolute competitive ratios are at most 5. A similar analysis can be applied in the parametric case, yielding an upper bound of $\frac{3d-1}{d-1}$ on the competitive ratio (in this case, in the proofs, the threshold level of bins should be $\frac{d-1}{d}$ instead of $\frac{1}{2}$). This shows that the analysis of FF and BF for very small items is almost tight. Additionally, the worst-case performance of Pseudo is at least as good as the performance of FF, BF, and WF. In the parametric cases with $d \geq 3$ Pseudo performs strictly better than FF and BF, and at least as well as WF. There is still a gap between the bounds on the best competitive ratio for BWBP, and it is interesting to find whether the true bound is below 2, above 2, or possibly simply 2. Furthermore, providing a tight analysis of the asymptotic and absolute competitive ratios of AF algorithms remains an open problem.

References

1. Balogh, J., Békési, J., Dósa, Gy., Kellerer H., Tuza, Zs.: Black and white bin packing. In: Proc. of WAOA'12, Lecture Notes in Computer Science, to appear.
2. Balogh, J., Békési, J., Galambos G.: New lower bounds for certain classes of bin packing algorithms. Theoretical Computer Science 440—441, 1–13 (2012)
3. Benkő, A. Dósa, Gy., Tuza, Zs.: Bin covering with a general profit function: approximability results. Central European Journal of Operations Research, online published (2012). doi:: 10.1007/s10100-012-0269-0
4. Boyar, J., Dósa, Gy., Epstein, L.: On the absolute approximation ratio for First Fit and related results. Discrete Applied Mathematics 160(13–14), 1914–1923 (2012)
5. Brandstädt, A., Le, V.B., Spinrad, J.P., Graph Classes – A Survey. SIAM Monographs on Discrete Mathematics and Applications, SIAM Press, Philadelphia, PA (1999)
6. Coffman, E.G., Jr, Garey, M., Johnson, D.: Approximation algorithms for bin packing: A survey. In: Hochbaum, D. (ed.) Approximation algorithms for NP-hard problems, pp. 46–93. PWS Publishing Co., Boston (1996)
7. Csirik J., Woeginger G.J.: On-line packing and covering problems. In: A. Fiat and G. J. Woeginger, editors, Online Algorithms: The State of the Art, pp. 147–177 (1998)
8. Dósa, Gy., Sgall, J.: First Fit bin packing: A tight analysis. STACS 2013, to appear.
9. Dósa, Gy., Tuza, Zs., Ye, D.: Bin packing with "Largest In Bottom" constraint: tighter bounds and generalizations. Journal of Combinatorial Optimization, online published (2012). doi: 10.1007/s10878-011-9408-0
10. Epstein, L.: On online bin packing with LIB constraints. Naval Research Logistics 56(8), 780–786 (2009)
11. Epstein, L., Levin, A.: On bin packing with conflicts. SIAM Journal on Optimization 19(3), 1270–1298 (2008)
12. Fernandez de la Vega, W., Lueker, G.S.: Bin packing can be solved within $1+\varepsilon$ in linear time. Combinatorica 1, 349–355 (1981)
13. Garey, M.R., Johnson, D.S.: Computer and Intractability: A Guide to the theory of NP-Completeness. Freeman, New York (1979)

14. Jansen, K.: An approximation scheme for bin packing with conflicts. Journal of Combinatorial Optimization 3(4), 363–377 (1999)
15. Jansen, K., Öhring, S.: Approximation algorithms for time constrained scheduling. Information and Computation 132(2), 85–108 (1997)
16. Johnson, D.S.: Near-optimal bin-packing algorithms. Doctoral Thesis, MIT, Cambridge (1973)
17. Johnson, D.S.: Fast algorithms for bin packing. Journal of Computer and System Sciences 8(3), 272–314 (1974)
18. Johnson, D., Demers, A., Ullman, J., Garey, M., Graham, R.: Worst-case performance bounds for simple one-dimensional packing algorithms. SIAM Journal on Computing 3, 25–278 (1974)
19. Karmarkar, N., Karp, R.: An efficient approximation scheme for the one-dimensional bin packing problem. In: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS'82), pp. 312–320 (1982)
20. Lee, C.C., Lee, D.T., A simple on-line packing algorithm. J. ACM 32, 562–572 (1985)
21. Manyem, P.: Uniform sized bin packing and covering: Online version. In: J.C. Misra (ed.) Topics in Industrial Mathematics, pp. 447–485. Narosa Publishing House, New Delhi (2003)
22. Manyem, P., Salt, R., Visser, M.: Approximation lower bounds in online LIB bin packing and covering. Journal of Automata, Languages and Combinatorics 8(4), 663–674 (2003)
23. Seiden, S.: On the online bin packing problem. Journal of the ACM 49(5), 640–671 (2002)
24. Simchi-Levi, D.: New worst-case results for the bin-packing problem. Naval Research Logistics 41(4), 579–585 (1994)
25. Ullman J.D.: The performance of a memory allocation algorithm. Technical Report 100, Princeton University, Princeton, NJ, 1971.
26. van Vliet, A.: An improved lower bound for on-line bin packing algorithms. Information Processing Letters 43(5), 277–284 (1992)
27. Xia, B.Z., Tan, Z.Y.: Tighter bounds of the First Fit algorithm for the bin-packing problem. Discrete Applied Mathematics 158(15), 1668–1675 (2010)
28. Yao A.C.C.: Probabilistic computations: Toward a unified measure of complexity (extended abstract). In: Proc. of the 18th Annual Symposium on Foundations of Computer Science (FOCS'77), pp. 222–227 (1977)