# Two-bounded-space bin packing revisited

Marek Chrobak * Jiří Sgall † Gerhard J. Woeginger ‡

**Abstract**

We analyze approximation algorithms for bounded-space bin packing by comparing them against the optimal bounded-space packing (instead of comparing them against the globally optimal packing that does not necessarily satisfy the bounded-space constraint). For 2-bounded-space bin packing we construct a polynomial time offline approximation algorithm with asymptotic worst case ratio 3/2, and we show a lower bound of 5/4 for this scenario. We show that no 2-bounded-space online algorithm can have an asymptotic worst case ratio better than 4/3.

## 1 Introduction

An instance of the classical bin packing problem consists of an ordered list $a_1, a_2, \ldots, a_n$ of items with rational sizes $0 \leq s(a_i) \leq 1$, and the goal is to pack these items into the smallest possible number of bins of unit size. Bin packing is a fundamental problem in combinatorial optimization, and it has been studied extensively since the early 1970s. Since bin packing is NP-hard, one particularly active branch of research has concentrated on approximation algorithms that find near-optimal packings; see for instance [1].

A bin packing algorithm works *offline* if it packs the items with full knowledge of the entire item list, and it works *online* if it packs every item $a_i$ solely on the basis of the preceding items $a_j$ with $1 \leq j \leq i$, and without any information on subsequent items in the list. A bin packing algorithm uses *$k$-bounded space*, if for each item $a_i$ the choice of bins in which it can be packed is restricted to a set of $k$ or fewer so-called *active* bins. Once an active bin is closed, it can never become active again. Bounded-space packings arise in many applications, as for instance in packing trucks at a loading dock or in communicating via channels with bounded buffer sizes.

Customarily, the performance of an approximation algorithm $A$ for bin packing is measured by comparing it against the optimal offline packing. Let $\textsc{Opt}(I)$ denote the number of bins used in an optimal (unconstrained) packing for instance $I$, and let $A(I)$ denote the number of bins used by algorithm $A$. Then the *absolute worst case ratio* of $A$ is defined as

$$R(A) := \sup_I A(I)/\textsc{Opt}(I),$$

and its *asymptotic worst case ratio* is defined as

$$R^\infty(A) := \lim_{Opt(I)\to\infty} \sup_I A(I)/\textsc{Opt}(I).$$

*Department of Computer Science, University of California, Riverside, USA
†Department of Applied Mathematics, Charles University, Prague, Czech Republic
‡Department of Mathematics and Computer Science, TU Eindhoven, Eindhoven, The Netherlands

The bin packing literature contains many articles that investigate and evaluate the (absolute and asymptotic) worst case ratios of bounded-space bin packing algorithms; see for instance [2, 3, 6, 7, 8]. Csirik & Johnson [3] show that the 2-space-bounded Best Fit algorithm $\mathrm{BBF}_2$ has the asymptotic worst case ratio $R^\infty(\mathrm{BBF}_2) = 1.7$. Among all 2-space-bounded algorithms (online and offline) in the bin packing literature, this online algorithm is the champion with respect to worst case ratios. A central result of Lee & Lee [6] designs $k$-bounded-space online bin packing algorithms whose asymptotic ratios come arbitrarily close to the magic harmonic number $h_\infty \approx 1.69103$, as the space bound $k$ tends to infinity. They also show that every bounded-space online bin packing algorithm $A$ satisfies $R^\infty(A) \geq h_\infty$.

**The trouble with the old worst case ratio.** The lower bound constructions in [6] use item lists on which *all* bounded-space algorithms must fail, no matter whether they are online or offline and no matter whether they run in polynomial time or in exponential time: In these constructions decent packings can only be reached by using unbounded space, and every bounded-space algorithm has asymptotic worst case ratio at least $h_\infty$.

A cleaner — and more realistic — way of measuring the performance of bounded-space bin packing algorithms is to compare them against optimal offline solutions *that also are subject to the bounded-space constraint.* Hence we let $\mathrm{OPT}_k(I)$ denote the smallest possible number of bins used in a packing produced by a $k$-bounded-space offline bin packing algorithm for instance $I$, and define the asymptotic *$k$-bounded-space ratio* of algorithm $A$ as

$$R_k^\infty(A) := \lim_{Opt_k(I)\to\infty} \sup_I A(I)/\mathrm{OPT}_k(I). \tag{1}$$

Note that the optimal $k$-bounded-space offline algorithm has a $k$-bounded-space ratio of 1. Furthermore $\mathrm{OPT}_k(I) \geq \mathrm{OPT}(I)$ implies $R_k^\infty(A) \leq R^\infty(A)$ for any algorithm.

**Our contributions.** Throughout the paper, we will mainly concentrate on 2-bounded-space scenarios. Our main goal is to beat the 2-bounded-space champion with respect to the classical asymptotic worst case ratio, the Best Fit algorithm $\mathrm{BBF}_2$ of [3] with $R^\infty(\mathrm{BBF}_2) = 1.7$. Indeed in Section 3 we construct an offline approximation algorithm $A$ for 2-bounded-space bin packing with $R_2^\infty(A) \leq 3/2 + \varepsilon$. The analysis goes deeply into the structure of 2-bounded-space packings. Some of the techniques involve exhaustive search of instances of size $1/\varepsilon$, thus the running time exponential in $\varepsilon$, but it is linear in $n$.

In Section 4 we show that resource augmentation makes $k$-bounded-space bin packing easy. If an approximation algorithm for $k$-bounded-space bin packing is allowed to use bins of size $1 + \varepsilon$, then in polynomial time it can pack an entire list $I$ into $\mathrm{OPT}_k(I)$ many bins. And if it is allowed to use $(k+1)$-bounded space, then in polynomial time it can pack an entire list $I$ into $(1+\varepsilon)\,\mathrm{OPT}_k(I)$ many bins.

Section 5 demonstrates that the $k$-bounded-space ratio asymptotic worst case ratio in (1) does not allow polynomial time approximation schemes; in fact every polynomial time algorithm $A$ must satisfy $R_2^\infty(A) \geq 5/4$.

In Section 6 we discuss 2-bounded-space bin packing for small instances $I$ with $\mathrm{OPT}_2(I) \leq 4$, and we provide a complete analysis for the absolute performance of polynomial time approximation algorithms for these cases. The problem of finding good packings is closely connected to our offline algorithm.

Finally Section 7 shows that every 2-bounded-space online algorithm must satisfy $R_2^\infty(A) \geq 4/3$. Subsequent to our work, Epstein and Levin improved this lower bound to $3/2$. The 2-bounded-space

ratio of 2-space-bounded Best Fit is known to be in $[1.5, 1.7]$. Improving its analysis or designing a new online algorithm matching the lower bound of 1.5 remains an interesting open problem.

## 2 Technical preliminaries

Throughout the paper, bins usually have size $\beta = 1$, and items usually have rational sizes in $[0, 1]$. In our lower bound constructions (in Sections 5, 6, and 7) we will work with bins of integer size $\beta$ and item sizes in $[0, \beta]$. Since bin and item sizes can be scaled by $\beta$, both models are computationally equivalent. For a set $B$ of of items (or for the set of items packed into some bin $B$), we denote by $s(B)$ the sum of the sizes of all the items in $B$.

For an instance $I$ of $k$-bounded-space bin packing, we define a *well-ordered* packing $\mathcal{B}$ to be a partition of the items into bins $B_{ij}$ (for $i = 1, \ldots, k$ and $j = 1, \ldots, m_i$), such that for any $i$ and for any $j < j'$, every item in bin $B_{ij}$ precedes every item in bin $B_{ij'}$. We denote by $|\mathcal{B}| = m_1 + \cdots + m_k$ the number of bins in a packing $\mathcal{B}$. A well-ordered packing $\mathcal{B}$ forms a feasible $k$-bounded-space packing, if every bin $B_{ij}$ satisfies $s(B_{ij}) \leq 1$.

For a given well-ordered packing $\mathcal{B}$, the sequence of bins $B_{ij}$ with $j = 1, \ldots, m_i$ is called *track* $i$. We will sometimes define a packing $\mathcal{B}$ by specifying an assignment of items to tracks $1, ..., k$ only, rather than to specific bins. The actual bin assignment can be uniquely determined by greedily assigning items to bins, for each track separately.

For the purpose of the design of the offline algorithm, it is convenient to specify when exactly the bins are closed, so that at each time exactly $k$ bins are open. We say that a bin $B_{ij}$ is open from (and including) the moment when the first item in $B_{ij}$ arrives until (not including) the moment when the first item in $B_{i,j+1}$ arrives or until the instance ends. As an exception, the first bin $B_{i1}$ in the track is considered to be open from the beginning of the instance.

## 3 An offline approximation algorithm

In this section we construct for every $\varepsilon > 0$ an offline approximation algorithm $A$ for 2-bounded-space bin packing whose asymptotic ratio is at most $3/2 + \varepsilon$. The crucial part, which is to convert certain approximate packings in the first 2 or 3 bins into exact packings with one additional bin, is postponed to Section 3.3. This is closely connected to Section 6 where we study packing of instances with small optimum. To achieve a better approximation using our approach, we would need to handle longer initial segments and thus to extend the positive results from Section 6.

The high-level strategy of our algorithm is as follows. We fix a suitable integer $V \geq 3/\varepsilon$ (a constant independent of the input). We cut the given instance $I$ greedily into $m$ sub-lists $I_1, I_2, \ldots, I_m$ with $s(I_j) \leq 2V$ for all $j$ and $V \leq s(I_j)$ for $j < m$. Thus $\textsc{Opt}_2(I_j) \leq 4V$ and $m \leq 1 + \textsc{Opt}_2(I)/V$. Since any 2-bounded-space packing of $I$ can be split into 2-bounded-space packings of the sublists $I_j$ $(1 \leq j \leq m)$ with a loss of at most two bins between adjacent sublists, we have

$$\sum_{j=1}^{m} \textsc{Opt}_2(I_j) \;\leq\; \textsc{Opt}_2(I) + 2(m-1) \;\leq\; \left(1 + \frac{2}{V}\right) \textsc{Opt}_2(I).$$

Hence, for getting a good approximation for $I$ it is enough to get a good approximation for every sublist $I_j$ separately.

In Sections 3.1, 3.2 and 3.3, we will show that in polynomial time we can find 2-bounded-space packings of each $I_j$ with at most $\frac{3}{2}\textsc{Opt}_2(I_j)+1$ bins. By concatenating these packings, we get the desired approximate 2-bounded-space packing for $I$ with at most

$$\sum_{j=1}^{m}\left(\frac{3}{2}\textsc{Opt}_2(I_j)+1\right) \;\le\; \frac{3}{2}\left(1+\frac{2}{V}\right)\textsc{Opt}_2(I)+m \;\le\; \left(\frac{3}{2}+\varepsilon\right)\textsc{Opt}_2(I)+1.$$

This yields the following theorem (and the main result of the paper).

**Theorem 3.1** *For every $\varepsilon > 0$ there exists a polynomial time approximationalgorithm $A$ for 2-bounded-space bin packing with asymptotic ratio $R_2^\infty(A) \le 3/2+\varepsilon$.*

### 3.1 Relaxed packings

We are left with the following problem: given an instance $I$ with $\textsc{Opt}_2(I) \le 4V$, find an approximate 2-bounded-space packing into at most $\frac{3}{2}\textsc{Opt}_2(I)+1$ bins. This problem is solved in two steps: first we find a *relaxed* packing of $I$ that may slightly overpack some bins (Lemma 3.3), and then we transform this relaxed packing into an *exact* packing that obeys the bin sizes (Lemma 3.4).

**Definition 3.2** *A $\delta$-relaxed (2-bounded-space) packing of an instance $I$ is a well-ordered partition such that for each bin $B_{ij}$ (with $1 \le i \le 2$) either*

- *$s(B_{ij}) \le 1$, in which case we set $D_{ij} = \emptyset$ and $\bar{B}_{ij} = B_{ij}$, or*
- *there exists $d \in B_{ij}$ such that $d \le \delta$ and $s(B_{ij} \setminus \{d\}) \le 1$, in which case we fix one such $d$, call it the* special item*, and set $D_{ij} = \{d\}$ and $\bar{B}_{ij} = B_{ij} \setminus D_{ij}$.*

In other words, each bin $B_{ij}$ is split into a set $\bar{B}_{ij}$ of size at most 1 and the set $D_{ij}$ with at most 1 item of size at most $\delta$. For the rest of Section 3 we set $\delta = 1/20 = 0.05$.

**Lemma 3.3** *For any instance $I$ with $\textsc{Opt}_2(I) \le 4V$, we can find in polynomial time a $\delta$-relaxed packing $\mathcal{B}$ into at most $\textsc{Opt}_2(I)$ bins.*

*Proof.* We search exhaustively the value $\textsc{Opt}_2(I)$ and the assignment of all items that are larger than $\delta$. Since the optimal number of bins is at most $4V$ and the number of large items is at most $4V/\delta$, i.e., they are both bounded by a constant, there is a constant number of assignments to try. For each assignment, we pack the items smaller than $\delta$ by adding them greedily from the beginning of the instance: Assign each item into the open bin that closes first, until and including the first item that overfills the bin; if the bin is already full then into the other bin. It can be seen that for the optimal assignment of the large items, the number of bins used is at most $\textsc{Opt}(I_j)$: Inductively we can verify that if a bin is full at the time of our assignment of an item, then the total size of this and all the previous bins is larger than or equal to the size of these bins in the optimum, thus the other bin cannot be full and we can assign the item there. □

**Lemma 3.4** *(Transformation lemma) Given a $\delta$-relaxed packing $\mathcal{B}$ of instance $I$, we can construct in polynomial time an exact packing of $I$ with at most $\frac{3}{2}|\mathcal{B}|+1$ bins.*

To prove the transformation lemma, we split the instance again. In each round, we process the $r$ initial bins of $\mathcal{B}$ for a suitable $r$ and find an exact packing of them into at most $\frac{3}{2}r$ bins. Then we glue this packing together with a packing of a slightly modified remaining instance. Some care needs to be taken to ensure that the overlapping parts of the packings use only one track each. The exact formulation of the needed building block follows.

**Lemma 3.5** *(Initial segment lemma) Given a $\delta$-relaxed packing $\mathcal{B}$ of instance $I$ with at least 2 bins, we can find in polynomial time $r$, an instance $I'$ which is a subsequence of $I$ and an exact packing of $\mathcal{B}'$ of $I'$ such that*

(i) *$I'$ contains all the items from the first $r$ closed bins of $\mathcal{B}$ and possibly some additional items that arrive before the $r$th bin of $\mathcal{B}$ is closed;*

(ii) *$\mathcal{B}'$ uses at most $\frac{3}{2}r$ bins;*

(iii) *in $\mathcal{B}'$, all the items that arrive after some item in $I \setminus I'$ are packed into bins in the same track and each of these bins has size at most $1 - \delta$.*

### 3.2 Proof of the transformation lemmafrom the initial segment lemma

We proceed by induction on the number of bins $m$ in the relaxed packing $\mathcal{B}$. If $I$ is empty, the statement is trivial. If $\mathcal{B}$ uses a single bin $B_{11}$, then there is an exact packing with bins $\bar{B}_{11}$ and $D_{11}$.

If $\mathcal{B}$ uses $m \geq 2$ bins, then use Lemma 3.5 to find $r$, $I'$, and $\mathcal{B}'$.

Let $C$ be the set of all items in $I \setminus I'$ that arrive before the last item of $I'$. Lemma 3.5 (i) implies that all these items belong to a single bin of $\mathcal{B}$, namely to the bin that remains open when the $r$th bin is closed. If $C$ contains a special item $d$ of that bin, we let $C' = C \setminus \{d\}$; otherwise we let $C' = C$.

Now create the instance $I''$ as follows. It starts by an item $a$ of size $s(C')$ followed by the items in $(I \setminus I') \setminus C$ in the same order as in $I$. We claim that we can construct a $\delta$-relaxed packing of $I''$ with $m - r$ bins: Take $\mathcal{B}$, remove the first $r$ bins, remove the items $C$ and put the item $a$ in their bin. If $C = C'$ the size of the bin does not change. If $C = C' \cup \{d\}$, the size of $d$ is not included in $a$, thus the size of the bin drops below 1. In each of these cases, this bin and whole packing satisfy the condition of $\delta$-relaxed packing.

By the induction assumption there exists an exact packing $\mathcal{B}''$ for $I''$ with $\frac{3}{2}(m - r) + 1$ bins.

Now we construct the exact packing $\bar{\mathcal{B}}$ for $I$. The packing starts by the bins of $\mathcal{B}'$ put in the same tracks as in $\mathcal{B}'$; Lemma 3.5 (i) implies that there are at most $\frac{3}{2}r$ bins. If the special item $d$ exists and is in $C$, Lemma 3.5 (ii) guarantees that the bin in $\mathcal{B}'$ that is open when $d$ arrives has size at most $1 - \delta$; we add $d$ in this bin without violating the size constraint. Next we consider the bin $B''_{i1}$ from $\mathcal{B}''$ that contains $a$, modified so that we replace $a$ by items $C'$, preserving its size. Lemma 3.5 (ii) guarantees that all the bins in $\mathcal{B}'$ that are open when the items of $C'$ arrive are in a single track, thus we can use the other track for $B''_{i1}$ in $\bar{\mathcal{B}}$. Finally, $\bar{\mathcal{B}}$ contains all the remaining bins from $\mathcal{B}''$. By the definition of $C$, all the items in these bins arrive after the last item from $I'$. Thus the only constraint for placing them in tracks is the placement of $B''_{i1}$: If $B''_{i1}$ is in track $i$ in $\bar{\mathcal{B}}$, all the remaining bins from $\mathcal{B}''$ are in $\bar{\mathcal{B}}$ in the same track as in $\mathcal{B}''$, otherwise they are in the opposite track. Thus $\bar{\mathcal{B}}$ is indeed 2-bounded packing for $I$. The total number of bins in $\bar{\mathcal{B}}$ is $\frac{3}{2}r + \frac{3}{2}(m - r) + 1 = \frac{3}{2}m + 1$. This completes the proof of the Transformation Lemma 3.4.

### 3.3 Proof of the initial segment lemma

Without loss of generality, assume that $B_{11}$ is closed no later than $B_{21}$. Let $B_{1t}$ be the last bin closed before or at the same time when $B_{21}$ is closed.

We proceed in several cases. The overall idea for finding the packing is always the same. We identify up to two large items and pack them in bins by themselves. These bins are placed in track 1 together with bins $\bar{B}_{1j}$ that are not in an ordering conflict with the large items. The set $G$ of

the remaining items is packed greedily in track 2. Since there are no large items among them, this packing is efficient, which is quantified in Claim 3.6. To guarantee Lemma 3.5 (iii), we make sure that the bins containing $G$ have each size at most $1-\delta$ and that $G$ contains all the items that arrive after the first item in $I \setminus I'$.

In most of the cases the instance $I'$ consists exactly of all items in the first $r$ closed bins, in which case Lemma 3.5 (i) follows immediately. Only in Case 3.1 we use a more complicated $I'$.

The value of various constants in the proof is somewhat arbitrary. For understanding the idea of the proof, one can assume that $\delta$ is set arbitrarily small. Also, the hard case is when $t = 1, 2$ as we can use only one additional bin. With increasing $t$, there is increasing slack in the construction.

**Claim 3.6** *Suppose that $b \leq 1-\delta$ is the largest item in an instance $G$ and $s(G) > 1$. Then there exists a 1-bounded packing of $G$ with at most $\lceil (s(G)-1-\delta)/(1-\delta-b) \rceil + 1$ bins, each of size at most $1-\delta$. In particular for $r \geq 2$, $\delta = 0.05$, and $b \leq 0.75$:*

(i) *If $s(G) \leq (1.75-b) + r\delta$ then $\lfloor 1 + r/2 \rfloor$ bins of size $1-\delta$ suffice for $G$.*

(ii) *If $s(G) \leq 0.8 + r\delta$ then $\lfloor r/2 \rfloor$ bins of size $1-\delta$ are sufficient for $G$.*

*Proof.* Pack the items greedily from the beginning of the sequence. When it is necessary to close a bin, its size is more than $1-\delta-b$. Iterating this, and observing that once the volume is below $1-\delta$, a single bin is sufficient, the general bound follows. To prove that the special cases (i) and (ii) follow, note that they are tight for $r = 3$ and check that this is the tightest case, as for $r = 2$ there is additional slack and whenever $r$ increases by 2, we have 1 additional bin but the additional size is only $2\delta < 1-\delta-b$. □

**Case 1:** All items in $B_{21}$ have size at most 0.75. We set $I' = B_{21} \cup B_{11} \cup \cdots \cup B_{1t}$, thus $r = t+1$ and Lemma 3.5 (i) holds. The packing $\mathcal{B}'$ uses track 1 for $t$ bins $\bar{B}_{1i}$, $i = 1, \ldots, t$. The remaining items $G = B_{21} \cup D_{11} \cup \cdots \cup D_{1t}$ are packed greedily using track 2. No item in $G$ is larger than 0.75 and $s(G) \leq 1 + r\delta$. Claim 3.6 (i) implies that $1 + r/2$ bins are sufficient for $G$. All items from the last open bin, i.e. $B_{21}$, are included in $G$, thus Lemma 3.5 (iii) is satisfied. Overall, we have at most $\frac{3}{2}r$ bins as required in Lemma 3.5 (ii).

**Case 2:** The largest item $a$ in $B_{21}$ has size at least 0.75 and arrives while the bin $B_{1j}$ is open for some $j \leq t$. We set $I' = B_{21} \cup B_{11} \cup \cdots \cup B_{1t}$, thus $r = t+1$ and Lemma 3.5 (i) holds. The packing $\mathcal{B}'$ uses track 1 for one bin containing just $a$ and $t-1$ bins $\bar{B}_{1i}$, $i = 1, \ldots, t$, $i \neq j$, in the appropriate order; it also uses at most $1 + r/2$ additional bins in both tracks depending on the two subcases. Altogether we use at most $\frac{3}{2}r$ bins as required.

To satisfy Lemma 3.5 (iii), we will make sure that all items from $B_{21}$ are packed greedily in track 2, with the exception of $a$. However, $a$ arrives while $B_{1j}$ is open, i.e., before any item in $I \setminus I'$ arrives, thus it does not violate the condition.

**Subcase 2.1:** The largest item $\bar{a}$ in $B_{1j}$ has size at least 0.5. The packing $\mathcal{B}'$ uses one additional bin containing just $\bar{a}$ in track 1. The remaining items $G = (B_{21} \cup B_{1j} \setminus \{a, \bar{a}\}) \cup D_{11} \cup \cdots \cup D_{1t}$ are packed greedily using track 2. No item in $G$ has size more than 0.5 and $s(G) \leq 0.75 + r\delta$. Claim 3.6 (ii) implies that $r/2$ bins are sufficient for $G$.

**Subcase 2.2:** No item in $B_{1j}$ has size 0.5 or more. All the remaining items $G = (B_{21} \setminus \{a\}) \cup B_{1j} \cup D_{11} \cup \cdots \cup D_{1t}$ are packed greedily using track 2. No item in $G$ has size more than 0.5 and $s(G) \leq 1.25 + r\delta$. Claim 3.6 (i) implies that $1 + r/2$ bins are sufficient for $G$.

**Case 3:** The largest item $a$ in $B_{21}$ has size at least 0.75 and arrives while the bin $B_{1,t+1}$ is open. Let $C$ be the set of items in $B_{1,t+1}$ that arrive before the large item $a$ arrives into $B_{21}$. Since at the time of arrival of $a$ we need to use both tracks (one for $a$ and one for the greedy assignment),

we have to include in $I'$ also items in $C$ to guarantee Lemma 3.5 (iii). We distinguish subcases according to $s(C)$.

**Subcase 3.1:** $s(C) \le 0.5$. We set $I' = B_{21} \cup B_{11} \cup \cdots \cup B_{1t} \cup C$, thus $r = t+1$ and Lemma 3.5 (i) holds because all items in $C$ arrive while $B_{21}$ is open. The packing $\mathcal{B}'$ uses track 1 for $t$ bins $\bar{B}_{1i}$, $i = 1, \ldots, t$ and one bin containing just $a$. The remaining items $G = (B_{21} \setminus \{a\}) \cup D_{11} \cup \cdots \cup D_{1t} \cup C$ are packed greedily using track 2. No item in $G$ has size more than 0.5 (the largest item may be in $C$) and $s(G) \le 0.75 + r\delta$. Claim 3.6 (ii) implies that we use at most $r/2$ bins for $G$, a total of $\frac{3}{2}r$ bins.

**Subcase 3.2:** $s(C) \ge 0.5$. Here we want to include in $I'$ the whole bin $B_{1,t+1}$, so that an additional bin is available to be used for $C$. This brings more problems, as many bins may close in track 2 before $B_{1,t+1}$ closes, and we have to include them as well to satisfy Lemma 3.5 (i).

Let $t'$ be such that $B_{2t'}$ is the last bin closed while $B_{1,t+1}$ is open. We set $I' = B_{21} \cup \cdots \cup B_{2t'} \cup B_{11} \cup \cdots \cup B_{1t} \cup B_{1,t+1}$, $r = t + t' + 1$, and Lemma 3.5 (i) is satisfied now. The packing $\mathcal{B}'$ uses track 1 for $t$ bins $\bar{B}_{1i}$, $i = 1, \ldots, t$, one bin containing $C \setminus D_{1,t+1}$, one bin containing just $a$, and $t'-1$ bins $\bar{B}_{2i}$, $i = 2, \ldots, t'$, in this order; this is $t + 2 + (t'-1) = r$ bins total. The remaining items $G = D_{11} \cup \cdots \cup D_{1t} \cup (B_{1,t+1} \setminus C) \cup (B_{21} \setminus \{a\}) \cup D_{22} \cup \cdots \cup D_{2t'}$ are packed greedily using track 2. No item in $G$ has size more than $0.5 + \delta$ (the largest item may be in $B_{1,t+1} \setminus C$) and $s(G) \le 0.8 + r\delta$. Claim 3.6 (ii) implies that we use at most $r/2$ bins for $G$, altogether at most $\frac{3}{2}r$ bins.

# 4 Complexity and resource augmentation

The complexity of $k$-bounded-space bin packing is as follows. If $k$ is part of the input, then finding an optimal packing is strongly $\mathbb{NP}$-hard (since 3-PARTITION [5] is a special case). For $k = 1$, the optimal 1-bounded-space packing can be found in polynomial time (by using the NEXT-FIT algorithm). For fixed $k \ge 2$, the problem is $\mathbb{NP}$-hard in the ordinary sense (since PARTITION is a special case) and solvable in pseudo-polynomial time:

**Theorem 4.1** *For every fixed integer $k$, the optimal $k$-bounded-space packing of an item list can be computed in pseudo-polynomial time.*

*Proof.* We scale the instance such that the bin size and all item sizes become integers, and then apply a straightforward dynamic program that works through the scaled list item by item. For every new item, the dynamic program first decides whether the fullest active bin should be closed, and then decides into which of the active bins the item should go. A state in the dynamic program stores the number of bins closed so far, and the filling level of all currently active bins. □

Csirik & Woeginger [4] discuss bounded-space bin packing (with the classical worst case ratio) in the resource augmentation model of competitive analysis, where the approximation algorithm can use bigger bins than the offline algorithm. By rounding the state space of this dynamic program and by applying some routine methods (as for instance discussed in [9]), we can turn the dynamic program from Theorem 4.1 into an FPTAS for this type of resource augmentation:

**Corollary 4.2** *For every fixed integer $k$, there exists an algorithm that computes an $k$-bounded-space packing of any instance $I$ into at most $\textsc{Opt}_k(I)$ bins of size $1 + \varepsilon$. The time complexity of this algorithm is polynomially bounded in the size of $I$ and $1/\varepsilon$.* □

Another natural type of resource augmentation allows the approximation algorithm to use space $k$ and compares it against the optimal offline packing with space $k - 1$.

**Theorem 4.3** *For every $\varepsilon > 0$, there exists a polynomial time algorithm that computes an $k$-bounded-space packing of an instance $I$ into at most $(1+\varepsilon)\textsc{Opt}_{k-1}(I)$ bins of unit size.* □

*Proof.* We split instance $I$ into sub-lists that can be packed into a constant number of, say, $1/\varepsilon$ bins, similarly as we did in Section 3. Each sub-list contains a constant number of big items of size $\varepsilon$ and larger, which are assigned by exhaustive search. The small items of size below $\varepsilon$ are then added greedily as long as they fit into the bins. The leftover small items are collected into one extra bin for the whole sub-list – this uses the extra space, and also adds one bin per sub-list. □

# 5 An asymptotic lower bound for offline algorithms

We now prove a lower bound of 1.25 on the asymptotic worst case ratio of any 2-space-bounded bin packing algorithm. We reduce from the $\mathbb{NP}$-hard PARTITION problem; see [5]. We consider an instance of PARTITION which is a set $P = \{q_1, \ldots, q_n\}$ of $n \geq 2$ positive integers that add up to $2Q$. The question is whether $P$ has an *equitable partition*, namely a partition into two sets each adding up to $Q$. Without loss of generality, we assume that each $q_i$ is a multiple of 3 and is at most $Q$, and that $Q$ is large enough, say $Q \geq 100$. Then $P$ has an equitable partition if and only if it has a partition into two sets each adding up to between $Q-2$ and $Q+2$.

We construct the following bin packing instance. The bins have size $\beta = 2Q+3$. The instance $I$ consists of $m$ copies of the following sub-list $J$ with $n+6$ items:

$$q_1, \ldots, q_n, Q+3, Q+3, Q+2, Q+2, Q+1, Q+1.$$

If $P$ has an equitable partition, we claim that there exists a 2-bounded-space packing of $I$ into $4m$ bins. It is sufficient to pack $J$ into 4 bins. Partition $q_1, \ldots, q_n$ into two subsets, each adding up to $Q$, and assign them to different tracks. In addition, each track has one copy of items $Q+3$, $Q+2$, $Q+1$ and needs only two bins.

In the rest of Section 5 we show that if $P$ does not have an equitable partition, then any 2-bounded-space packing of $I$ uses at least $5m$ bins of size $\beta = 2Q+3$.

We introduce first some terminology. For a given list $L$, we say that a packing $\mathcal{B}$ *dominates* a packing $\mathcal{B}'$ on $L$ if, for any list $L'$ with all items of size at least 3, $\mathcal{B}$ can be extended to a packing of $LL'$ that uses no more bins that any extension of $\mathcal{B}'$ to $LL'$. A set of packings is called *dominant* on $L$ if any other packing is dominated by some packing in this set.

By a *state* we mean a pair of integers $\{u, v\}$ representing the content of the two open bins. We consider 2-bounded-space packings with arbitrary initial states, not necessarily $\{0, 0\}$. The concept of dominance extends naturally to such packings. Note that to determine whether $\mathcal{B}$ dominates $\mathcal{B}'$ we only need to know for each of $\mathcal{B}$ and $\mathcal{B}'$ the final state and the number of bins closed by it. In particular, the following easy observation will be useful in showing that one packing dominates another.

**Claim 5.1** *Consider two packings $\mathcal{B}$, $\mathcal{B}'$, where $\mathcal{B}$ closes $b$ bins and ends in state $\{u, v\}$, and $\mathcal{B}'$ closes $b'$ bins and ends in state $\{u', v'\}$, for $u \leq v$ and $u' \leq v'$. Suppose that either* (i) *$b \leq b' - 2$, or* (ii) *$b = b' - 1$ and $u \leq v'$, or* (iii) *$b = b'$ and $u \leq u'$ and $v \leq v'$. Then $\mathcal{B}$ dominates $\mathcal{B}'$.*

We split $J$ into three parts:

$$J_1 \;=\; q_1, \ldots, q_n, Q+3, Q+3, \qquad J_2 \;=\; Q+2, Q+2, \qquad J_3 \;=\; Q+1, Q+1.$$

Figure 1 shows a state diagram, where transitions correspond to packings of sub-lists $J_1$, $J_2$, $J_3$ from certain states. A transition from $\{u, u'\}$ to $\{v, v'\}$ labeled by $J_i \,/\, b$ represents a packing that, starting from state $\{u, u'\}$, packs $J_i$ closing $b$ bins and ending in state $\{v, v'\}$.

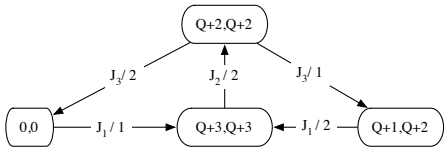


Figure 1: The state diagram representing dominant packings.

We claim that the packings represented in Figure 1 are dominant for the given initial states and lists $J_i$. Note that for the proof it is not necessary that these packings actually exist, since we only use them to lower-bound the actual packings. Another way to think about these transitions is as "sub-packings", where each item may take less space in the bin than its actual size.

Before proving this claim, we show that it implies the lemma. Indeed, since the packings in Figure 1 are dominant, an optimal packing of $I = (J_1 J_2 J_3)^m$ cannot do better than follow some path in that diagram, closing the numbers of bins indicated on the transitions, and ending either at state $\{0, 0\}$ or $\{Q + 1, Q + 2\}$. By a straightforward inspection of the cycles in the diagram, if this path ends at $\{0, 0\}$, the optimal packing will close $5m$ bins. If it ends at $\{Q + 1, Q + 2\}$, it will close $5m - 1$ bins, with two open non-empty bins. Thus the total number of used bins is at least $5m$, and the lemma follows.

To prove our claim, we consider transitions from each state. Since $q_i \geq 3$ for all $i$, we can assume that all bins with load at least $2Q + 1$ are immediately closed. We also assume that the bins open after packing each $J_i$ that have loads at most $2Q$ remain open until the next item arrives.

Consider a packing of $J_1$ starting from $\{0, 0\}$. If we put $Q+3$, $Q+3$ in different tracks, then we must close a bin that contains only items from $P$ (because $P$ does not have an equitable partition), so we can as well put them all in one bin, and we reach state $\{Q + 3, Q + 3\}$. If $Q + 3$, $Q + 3$ are in the same track, then we need to close one bin with the first item $Q + 3$, and the other track will contain items from $P$ that do not fit into that bin, so it will have load $x \geq Q + 3$, by our assumption. Thus the final state is $\{Q + 3, x\}$. In both cases, the packings are dominated by the packing represented by the one in Figure 1.

Consider a packing of $J_2$ from $\{Q + 3, Q + 3\}$. The items $Q + 2$, $Q + 2$ cannot share a bin and do not fit in the open bins, so any packing must close two bins. The dominant packing will end up in state with minimum load bins, that is $\{Q + 2, Q + 2\}$, same as the one in Figure 1.

Next, consider a packing from $\{Q + 2, Q + 2\}$ on $J_3$. If $Q + 1, Q + 1$ are assigned to different tracks then we close two bins and go to $\{0, 0\}$, otherwise we close one bin and go to $\{Q + 1, Q + 2\}$. These two packings are exactly those from Figure 1.

Finally, consider a packing of $J_1$ from $\{Q + 1, Q + 2\}$. Items $Q + 3$, $Q + 3$ cannot share a bin and cannot be assigned to already open bins, so we need to close two bins. If $Q + 3$, $Q + 3$ are assigned to different tracks, the two open bins will close and we will reach a state $\{Q + x, Q + x'\}$ for some $x, x' \geq 3$. If $Q + 3$, $Q + 3$ are on one track, the other track needs to be assigned some items from $P$, because $P$ does not have an equitable partition. Therefore the new state will be $\{Q + 3, Q + x\}$, for $x \geq 3$. This packing is dominated by the one in Figure 1.

Our reduction yields the main result of this section:

**Theorem 5.2** *Any polynomial time algorithm $A$ for 2-bounded-space bin packing has asymptotic ratio $R_2^\infty(A) \geq 1.25$ (unless $\mathbb{P}$=$\mathbb{NP}$).* □

## 6 Absolute bounds for offline algorithms

In this section we discuss 2-bounded-space bin packing for instances that can be packed into a small number $b$ of bins. For $b \geq 1$ we define $\mathcal{I}_b$ as set of all instances $I$ with $\textsc{Opt}_2(I) = b$. Furthermore, we define $f_2(b)$ as the smallest integer such that for every instance $I \in \mathcal{I}_b$ we can find in polynomial time a 2-bounded-space packing into $f_2(b)$ bins; note that in this definition we assume $\mathbb{P} \neq \mathbb{NP}$. Theorem 5.2 implies $f_2(b) \geq 5b/4$ for large values of $b$.

It is not hard to see that $f_2(1) = 1$, $f_2(2) = 3$, and $f_2(3) = 4$. As deciding containment in $\mathcal{I}_b$ with $b \geq 2$ is $\mathbb{NP}$-hard, we get $f_2(b) \geq b + 1$ for $b \geq 2$. The upper bounds $f_2(2) \leq 3$ and $f_2(3) \leq 4$ follow by ad hoc arguments that are similar to the arguments in the proof of Theorem 6.2 below. Theorems 6.2 and 6.3 imply $f_2(4) = 6$.

**Lemma 6.1** *For every fixed real $\varepsilon > 0$, there is a polynomial time algorithm that finds a 2-bounded-space packing*

(i) *with two bins for any instance of $\mathcal{I}_2$ with total item size at most $2 - \varepsilon$,*

(ii) *with three bins for any instance of $\mathcal{I}_3$ with total item size at most $3 - \varepsilon$.*

*Proof.* For statement (i), we apply any of the standard polynomial time approximation schemes for the SUBSET SUM problem. For statement (ii), we try all $O(n)$ possibilities for the opening point $p$ of the third bin. By using a PTAS for SUBSET SUM, we pack as much volume as possible from items $a_1, \ldots, a_{p-1}$ into the first bin and as much volume as possible from items $a_p, \ldots, a_n$ into the third bin, and put the rest into the second bin. □

**Theorem 6.2** *There is a polynomial time algorithm that finds a 2-bounded-space packing with six bins for any given instance of $\mathcal{I}_4$.*

*Proof.* For an instance $I \in \mathcal{I}_4$, the optimal 2-bounded-space packing is of one of the following two types: (i) one track consists of three bins and the other track of a single bin; (ii) the tracks consist of two bins each. For either type we can guess (that is, exhaustively try all possibilities for) the intervals during which the four bins are active.

Packings of type (i) are handled as follows. In the track with three bins, we use a PTAS for SUBSET SUM to pack as much volume as possible into each of the bins (picking $\varepsilon = 1/3$ will do). This leaves a volume of at most $1 + 3\varepsilon$ for the track with the single bin, which is easily packed into three bins.

Packings of type (ii) are handled as follows. We try all $O(n^2)$ possibilities for the opening points $p$ and $q$ (with $p \leq q$) of the third and fourth bin. Fix $\varepsilon = 1/5$. First assume that the total size of items $a_1, \ldots, a_{p-1}$ is at most $2 - \varepsilon$. Then we pack these items into two bins with Lemma 6.1.(i). The items $a_p, \ldots, a_n$ form an instance of $\mathcal{I}_3$, and hence can be packed into four bins as $f_2(3) = 4$. The case where the total size of items $a_q, \ldots, a_n$ is at most $2-\varepsilon$ is settled by a symmetric argument. In the remaining cases, the total size of $a_1, \ldots, a_{p-1}$ and $a_q, \ldots, a_n$ is at most 2, respectively, and the total size of $a_p, \ldots, a_{q-1}$ is at most $2\varepsilon$. Then $a_1, \ldots, a_{q-1}$ lies in $\mathcal{I}_3$ with total size at most $2 + \varepsilon$, and hence can be packed into three bins with Lemma 6.1.(ii). Furthermore $a_q, \ldots, a_n$ form an instance of $\mathcal{I}_2$, and hence can be packed into three bins as $f_2(2) = 3$. □

**Theorem 6.3** *It is $\mathbb{NP}$-hard to distinguish instances in $\mathcal{I}_4$ from those in $\mathcal{I}_6$.*

*Proof.* We consider an instance $P$ of the $\mathbb{NP}$-hard PARTITION problem (see Section 5) that consists of $n \geq 2$ positive integers $q_1, \ldots, q_n$ that add up to $2Q$. We construct a bin packing instance $I$ with bin size $\beta = 2Q + 1$ and the $2n + 4$ items with the following sizes:

$$q_1, \ldots, q_n, \quad Q+1, Q+1, Q+1, Q+1, \quad q_1, \ldots, q_n.$$

We claim that (i) if $P$ has an equitable partition then $\textsc{Opt}_2(I) = 4$, and (ii) if $I$ has a 2-bounded-space packing of $I$ into five bins then $P$ has an equitable partition.

Indeed, claim (i) is straightforward to verify. For claim (ii), consider some 2-bounded-space packing of $I$ into five bins. Clearly the four big items of size $Q+1$ must go into four separate bins. Below we refer to the remaining bin as the *fifth bin*. We distinguish two cases.

**Case 1:** The fifth bin contains items from both copies of $P$. Then two big items must be alone in their bins (as the packing uses 2-bounded space), and the remaining bins with a big item must each take only elements from one copy of $P$. Thus each copy of $P$ is spread over two bins: one bin with a big item, and the fifth bin. Since the bin size is $\beta = 2Q+1$, one of the copies of $P$ has items of total size at most $Q$ in the fifth bin and items of total size at most $Q$ in the other bin with the big item, producing an equitable partition of $P$.

**Case 2:** The fifth bin contains items from only one copy of $P$. Then the other copy of $P$ is spread over two bins with big items (as the packing uses 2-bounded space), producing an equitable partition of $P$.

Hence any optimal packing of the constructed instances uses either four or at least six bins. □

# 7 An asymptotic lower bound for online algorithms

Throughout this section we will consider an arbitrary online bin packing algorithm ON that uses 2-bounded space. Our goal is to establish a lower bound of 4/3 on the performance ratio $R_2^\infty(\text{ON})$. We will make excessive use of the following two straightforward observations. First: if a new item does not fit into any active bin, then it is never wrong to close the fullest active bin. Second: if an active bin cannot accommodate any of the future items, then it can be closed right away.

The proof is done in terms of a standard adversary argument. The adversary works in $p$ phases, where $p$ is a huge integer that can be chosen arbitrarily large. All item sizes in the $k$th phase ($1 \leq k \leq p$) depend on a small rational number $\varepsilon_k = 2^{k-p}/100$. Note that these values $\varepsilon_k$ form a strictly increasing sequence satisfying $\varepsilon_\ell \geq 2\varepsilon_k$ for all $\ell > k$. Note furthermore that the $\varepsilon_k$ are bounded from above by 1/100. During the $k$th phase, the adversary may issue items of the following three types: so-called 1-items of size $1 + \varepsilon_k$; so-called 2-items of size $2 - 2\varepsilon_k$; and so-called 3-items of size $3 - \varepsilon_k$.

The bins have size $\beta = 4$. When referring to the load or contents of a bin, we will usually ignore the dependence on the epsilons. Hence the active bins will be bins of load 1 (which contain a single 1-item), bins of load 2 (which contain a single 2-item, or two 1-items), or bins of load 3 (which contain a single 3-item, or three 1-items, or a 1-item together with a 2-item). Bins of load 4 can always be closed right away.

**Proposition 7.1** *Consider a partially packed bin that only contains items issued during the first $k$ phases.*
(i) *If the bin has load 1, then any 3-item from phase $k$ or later will fit into it.*
(ii) *If the bin has load 2, then any 2-item from phase $k$ or later will fit into it.*
(iii) *If the bin has load 3, then no item from phase $k+1$ or later will fit into it.*

*Proof.* Statements (i) and (ii) are straightforward. For statement (iii) we distinguish three cases: if the bin contains a single 3-item, the free space is at most $1 + \varepsilon_k$. If the bin contains three 1-items, the free space is at most $1 - 3\varepsilon_k$. If the bin contains a 1-item and a 2-item, the free space is at most $1 + 2\varepsilon_k - \varepsilon_k$. All items arriving in phase $k+1$ or later are of size at least $1 + 2\varepsilon_k$, and hence will not fit into the free space. □

Here is a central ingredient to the adversary's strategy: Whenever ON produces an active bin of load 3, the adversary starts a new phase and increases the value of $\varepsilon$. Then by Proposition 7.1.(iii) ON will never be able to use this bin again, and we may assume without loss of generality that whenever ON produces an active bin of load 3, this bin is closed right away.

The adversary will only close bins of load 4, either by adding a 2-item to an active bin of load 2, or by adding a 3-item to an active bin of load 1. By Proposition 7.1 these items will always fit into their respective bins.

The adversary will always issue one or two elements and make sure it ends in one of the following three states:

ADV$[0, 0]$: Both active bins are empty.

ADV$[0, 1]$: One empty bin; the other bin contains a single 1-item.

ADV$\{1, 1\}$: Two 1-items in the active bins. The adversary may decide later whether these items are in the same bin or in different bins.

Before a new item arrives, the active bins of ON will always have loads $X, Y \in \{0, 1, 2\}$, we may assume $X \leq Y$; these six states are denoted by ON$[X, Y]$.

The adversary's strategy is designed to guarantee that the combination of states ADV$\{1, 1\}$ and ON$[X, Y]$ never appears; all other combinations are possible. The following paragraphs provide a full description of the adversary's behavior, in dependence on its current state and the current state of ON.

ADV$[0, 0]$ and any ON$[X, Y]$: The adversary issues a 1-item and moves to ADV$[0, 1]$. Depending on its state, ON perhaps closes a bin of load 3. Since all ON states are legal with ADV$[0, 1]$, we end up in a legal combination.

ADV$[0, 1]$ or ADV$\{1, 1\}$ and ON$[X, Y]$ with $X, Y \in \{0, 2\}$: The adversary issues a 3-item, closes a bin of load 4, and moves to ADV$[0, 0]$ or ADV$[0, 1]$, while ON either closes a bin of load 3, or it closes a bin of load 2 and a bin of load 3. Again, all final state combinations are legal.

ADV$\{1, 1\}$ and ON$[0, 1]$ or ON$[1, 1]$: The adversary issues a 2-item, closes a bin of load 4 containing both 1-items, and moves to ADV$[0, 0]$, while ON cannot reach load 4. Perhaps it closes a bin of load 3. All ON states are a legal combination with ADV$[0, 0]$.

ADV$[0, 1]$ and ON$[0, 1]$ or ON$[1, 2]$: The adversary issues a 1-item and moves to ADV$\{1, 1\}$, while ON moves to one of the states ON$[1, 1]$, ON$[0, 2]$, or ON$[2, 2]$, or else it closes a bin of load 3 and moves to ON$[0, 1]$. In any case, ON final state is not ON$[1, 2]$, so the combination is legal.

ADV$[0, 1]$ and ON$[1, 1]$: The adversary issues two 2-items, closes a bin with them and remains in ADV$[0, 1]$, while ON closes one or two bins of load 3 and moves to some state. Again, any ON state is a legal combination with ADV$[0, 1]$.

This completes the description of the adversary's strategy, as all the cases are covered. To summarize: the adversary only closes bins of load 4, whereas ON always closes bins of load at most 3. This yields the following theorem.

**Theorem 7.2** *Any 2-bounded-space online bin packing algorithm* ON *has performance ratio* $R_2^\infty(\text{ON}) \geq 4/3$. □

**Acknowledgments.** This research has been partially supported by NSF Grant CCF-0729071; by Inst. for Theor. Comp. Sci., Prague (project 1M0545 of MŠMT ČR) and grant IAA100190902 of GA AV ČR, by the Netherlands Organization for Scientific Research (NWO), grant 639.033.403; by BSIK grant 03018 (BRICKS: Basic Research in Informatics for Creating the Knowledge Society).

## References

[1] E.G. Coffman, Jr., J. Csirik, G.J. Woeginger (2002). Approximate solutions to bin packing problems. In *Handbook of Applied Optimization*, P.M. Pardalos and M.G.C. Resende (eds.), Oxford University Press, New York, 607–615.

[2] J. Csirik and B. Imreh (1989). On the worst-case performance of the NkF bin packing heuristic. *Acta Cybernetica 9*, 89–105.

[3] J. Csirik and D.S. Johnson (2001). Bounded space on-line bin packing: Best is better than first. *Algorithmica 31*, 115–138.

[4] J. Csirik and G.J. Woeginger (2002). Resource augmentation for on-line bounded space bin packing. *Journal of Algorithms 44*, 308–320.

[5] M.R. Garey and D.S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco.

[6] C.C. Lee and D.T. Lee (1985). A simple on-line bin-packing algorithm. *Journal of the ACM 32*, 562–572.

[7] W. Mao (1993). Tight worst-case performance bounds for Next-$k$-Fit bin packing. *SIAM Journal on Computing 22*, 46–56.

[8] G.J. Woeginger (1993). Improved space for bounded-space, on-line bin-packing. *SIAM Journal on Discrete Mathematics 6*, 575–581.

[9] G.J. Woeginger (2000). When does a dynamic programming formulation guarantee the existence of a fully polynomial time approximation scheme (FPTAS)? *INFORMS Journal on Computing 12*, 57–75.