Improved Lower Bounds for the Online Bin Packing Problem with Cardinality Constraints

Hiroshi Fujiwara   Koji Kobayashi

Abstract

The bin packing problem has been extensively studied and numerous variants have been considered. The *k-item bin packing* problem is one of the variants introduced by Krause et al. in Journal of the ACM 22(4). In addition to the formulation of the classical bin packing problem, this problem imposes a *cardinality constraint* that the number of items packed into each bin must be at most k. For the *online* setting of this problem, in which the items are given one by one, Babel et al. provided lower bounds $\sqrt{2} \approx 1.41421$ and 1.5 on the asymptotic competitive ratio for $k = 2$ and 3, respectively, in Discrete Applied Mathematics 143(1-3). For $k \geq 4$, some lower bounds (e.g., by van Vliet in Information Processing Letters 43(5)) for the online bin packing problem, i.e., a problem without cardinality constraints, can be applied to this problem.

In this paper we consider the online k-item bin packing problem. First, we improve the previous lower bound 1.41421 to 1.42764 for $k = 2$. Moreover, we propose a new method to derive lower bounds for general k and present improved bounds for various cases of $k \geq 4$. For example, we improve 1.33333 to 1.5 for $k = 4$, and 1.33333 to 1.47058 for $k = 5$.

1 Introduction

The bin packing problem is one of the most extensively studied problems in the field of computer science, which is defined as follows. We are given a sequence of *items*, each of *size* in the range $(0, 1]$, as an input, and an infinite number of *bins*. Each item has to be packed into one of the bins, and the sum of sizes of items packed into each bin has to be at most one. A bin that contains at least one item is said to be *non-empty*. The goal of this problem is to minimize the number of non-empty bins.

The bin packing problem has been studied also in the *online* setting: the items are given one by one, and each item has to be packed before the next one is given. This version is quite important in both theoretical and applied aspects, and much work has been done on it (e.g. [RBLL89, vV92, Sei02, BBG12]). Online algorithms are usually evaluated using competitive analysis [BE98, ST85]. For any sequence σ of items, and any algorithm ALG, let $C_{ALG}(\sigma)$ denote the number of ALG's non-empty bins for σ. Let OPT be an optimal offline algorithm. Then, for any online algorithm ON, define $R_{ON} = \limsup_{n\to\infty} \sup_{\sigma}\{C_{ON}(\sigma)/C_{OPT}(\sigma) \mid C_{OPT}(\sigma) = n\}$, which we call the *asymptotic competitive ratio* (also known as the *asymptotic performance ratio*) of ON.

A constraint that the number of items packed in a bin is somehow restricted seems quite realistic in application. For example, since there exists the minimum size of files used by a computer, the number of files stored on the computer is bounded. In light of this situation, Krause et al. [KSS75, KSS77] introduced the *k-item bin packing* problem, in which the *cardinality*

[0]This work was supported by KAKENHI (23700014 and 23500014). The final publication is available at `http://dx.doi.org/10.1007/s10878-013-9679-8`.

Table 1: Previous results and our results for the online k-item bin packing problem

k	lower bound	upper bound
2	$\sqrt{2}(\approx 1.41421)$ [BCKK04] $\rightarrow$ 1.42764 [this paper]	$1+\frac{\sqrt{5}}{5}(\approx 1.44721)$ [BCKK04]
3	1.5 [BCKK04]	1.75 [Eps06]
4	$\frac{4}{3}(\approx 1.33333)$ [vV92] $\rightarrow$ 1.5 [this paper]	$\frac{71}{38}(\approx 1.86843)$ [Eps06]
5	$\frac{4}{3}(\approx 1.33333)$ [vV92] $\rightarrow$ $\frac{25}{17}(\approx 1.47058)$ [this paper]	$\frac{771}{398}(\approx 1.93719)$ [Eps06]
6	1.5 [Yao80]	$\frac{287}{144}(\approx 1.99306)$ [Eps06]
7 to 9	1.5 [Yao80]	2 [BCKK04]
10 to 41	1.5 [Yao80] $\rightarrow$ (See Table 2 in Section 3.) [this paper]	
42 to 293	$\frac{217}{141}(\approx 1.53900)$ [vV92]	
294 to 2057	$\frac{10633}{6903}(\approx 1.54034)$ [BBG12]	
∞	$\frac{248}{161}(\approx 1.54037)$ [BBG12]	1.58889 [Sei02]

constraint that each bin can contain at most k items is imposed. (They defined this problem as a scheduling problem.) This problem has been well studied in both the offline and online settings.

Previous Results and Our Results. In the *online k-item bin packing* problem, in which items are given in an online manner and the number of items in a bin has to be at most k, Babel et al. [BCKK04] showed that for $k = 2$, the asymptotic competitive ratio of any online algorithm is at least $\sqrt{2} \approx 1.41421$. Also, they presented a lower bound of 1.5 when $k = 3$ using the method by Yao [Yao80]. Moreover, for larger k, lower bounds by van Vliet [vV92], Yao [Yao80], and Balogh et al. [BBG12] for the online bin packing problem, i.e., a problem without cardinality constraints, can be applied to the online k-item bin packing problem. We mention that the lower bounds for $k = 4$ and 5 are straightforwardly given by manipulating the method in [vV92]. (See Table 1.)

In this paper, we consider the online k-item bin packing problem. First, we show that the asymptotic competitive ratio of any algorithm is at least $\overline{r} \approx 1.42764$ for $k = 2$, where $\overline{r}$ is the root of the equation $2r^3 - 17r^2 + 30r - 14 = 0$ between $\frac{4}{3}$ and $\frac{3}{2}$, which improves the previous lower bound. The gap between the upper and lower bounds is reduced by about 40%. Second, we extend the method by van Vliet [vV92] and get improved lower bounds for various cases of $k \geq 4$. For example, we improve 1.33333 to 1.5 for $k = 4$, and 1.33333 to 1.47058 for $k = 5$. (See Table 1, and also Table 2 in Section 3.)

Related Results. In the online k-item bin packing problem, Krause et al. [KSS75, KSS77] showed that for any k, the asymptotic competitive ratio of the most basic algorithm FirstFit is at most $2.7 - 12/5k$. Babel et al. [BCKK04] established an algorithm whose asymptotic competitive ratio is at most 2 for any k. Moreover, Babel et al. [BCKK04] and Epstein [Eps06] designed algorithms for small k. These results are also presented in Table 1. In addition, Epstein [Eps06] established a bounded space algorithm. She showed that its asymptotic competitive ratio is at most 2.69104, and is asymptotically optimal. Note that while a bounded space algorithm always has only a constant number of bins available to accept items, the other results described above, including our new results, focus on unbounded space algorithms. There are some studies [KP99, CKP03, EL10] about approximation algorithms for the k-item bin packing problem. Needless to say, the online bin packing problem (without cardinality constraints) has been much studied. The best upper and lower bounds are 1.58889 by Seiden [Sei02] and $248/161 \approx 1.54037$ by Balogh et al. [BBG12], respectively.

2 A Lower Bound for $k = 2$

In this section we present a lower bound of 1.42764 for $k = 2$. We first define an adversary which determines the size of the next item adaptively according to the behavior of an online algorithm. The strategy of the adversary is chosen from the three strategies whose pseudocodes will be later given as ROUTINE1, 2, and 3, respectively.

We begin by describing three subroutines called by ROUTINE1, 2, and 3. See their pseudocodes SUBROUTINE1, 2, and 3 below. Roughly speaking, each subroutine gives a sequence of items while changing the size within a specified range. The only difference among the three subroutine is only the termination conditions. Let us see the details. Each subroutine is called with four arguments: an online algorithm ON and three values `Min`, `Max`, and `Length` with `Min` < `Max`. Each subroutine returns the current value of the internal variable `tmpMin`. The sizes of given items are chosen from the range (`Min`, `Max`). For ease of presentation, if an algorithm ALG is about to put an item into a bin that contains no item, we say that ALG *opens* the bin. The termination conditions of the three subroutines are as follows: SUBROUTINE1 finishes when it has given `Length` items to ON, SUBROUTINE2 finishes when ON has opened new `Length` bins, and SUBROUTINE3 finishes when ON has created `Length` bins with two items.

Before giving their pseudocodes, we define a function f used in these subroutines: for any $x, y \in (0, 1]$ with $x < y$, $f(x, y) = (x + y)/2$. (Indeed, f can be any function that maps x and y to a value between x and y.)

```
SUBROUTINE1($ON$, Min, Max, Length):
Step 1. $a_1 := f(\texttt{Min}, \texttt{Max})$, tmpMax := Max, tmpMin := Min, and $i := 1$.
Step 2. Give an item $b_i$ of size $a_i$, and do the following according to $ON$'s action.
  Case 2.1. If $ON$ opens a bin and puts $b_i$ into it,
    $a_{i+1} := f(\texttt{tmpMin}, a_i)$ and tmpMax := $a_i$.
  Case 2.2. Otherwise,
    $a_{i+1} := f(a_i, \texttt{tmpMax})$ and tmpMin := $a_i$.
Step 3. If Length = $i$, then return tmpMin. Otherwise, $i := i + 1$, and go to Step 2.
```

```
SUBROUTINE2($ON$, Min, Max, Length):
Step 1. $a_1 := f(\texttt{Min}, \texttt{Max})$, tmpMax := Max, tmpMin := Min, and $i := 1$.
Step 2. Give $ON$ an item $b_i$ of size $a_i$, and do the following according to $ON$'s action.
  Case 2.1. If $ON$ opens a bin and puts $b_i$ into it, then
    $a_{i+1} := f(\texttt{tmpMin}, a_i)$ and tmpMax := $a_i$.
  Case 2.2. Otherwise,
    $a_{i+1} := f(a_i, \texttt{tmpMax})$ and tmpMin := $a_i$.
Step 3. If the number of bins that were opened by $ON$ at Case 2.1 is Length, then return
  tmpMin. Otherwise, $i := i + 1$, and go to Step 2.
```

```
SUBROUTINE3($ON$, Min, Max, Length):
Step 1. $a_1 := f(\texttt{Min}, \texttt{Max})$, tmpMax := Max, tmpMin := Min, and $i := 1$.
Step 2. Give $ON$ an item $b_i$ of size $a_i$, and do the following according to $ON$'s action.
```

```
    Case 2.1. If $ON$ opens a bin and puts $b_i$ into it, then
        $a_{i+1} := f(\texttt{tmpMin}, a_i)$ and $\texttt{tmpMax} := a_i$.
    Case 2.2. Otherwise,
        $a_{i+1} := f(a_i, \texttt{tmpMax})$ and $\texttt{tmpMin} := a_i$.
Step 3. If the number of bins with two items both of which are given at Step 2 is Length, then
return tmpMin. Otherwise, $i := i + 1$, and go to Step 2.
```

The purpose of these subroutines is to construct a sequence that has the following property.

Lemma 1. *Suppose that* Subroutine1 *(*Subroutine2*,* Subroutine3*, respectively) is called with some ON,* `Max`*, and* `Min` *with* `Max` $>$ `Min`*. Let β_j be the size of the j-th item $(1 \le j \le n)$ that is put into a bin at Case 2.1, and let $\gamma_{j'}$ be the size of the j'-th item $(1 \le j' \le m)$ that is put into a bin at Case 2.2. Also, let $\beta_0 :=$* `Max` *and $\gamma_0 :=$* `Min`*. Denote by t_{min} (t_{max}) the value of* `tmpMin` *(*`tmpMax`*, respectively) at the moment when the subroutine returns. Then, $\beta_0 > \beta_1 > \cdots > \beta_n = t_{max} > t_{min} = \gamma_m > \cdots > \gamma_1 > \gamma_0$.*

Proof. Although the terminal conditions of Subroutine1, 2, and 3 are different from one another, their pseudocodes of Step2, where the sizes of items are determined, are the same. Thus, it is only necessary to consider Subroutine1. The proof is by induction on the number of items given at Step 2 in Subroutine1. Let $t_{min,j}(t_{max,j})$ denote the value of `tmpMin` (`tmpMax`) just before the j-th item is given. First, we consider just before the first item of size a_1 is given. By the assumption that `Max` $>$ `Min` and the definition of Step 1, `Max` $= \beta_0 = t_{max,1} > t_{min,1} = \gamma_0 =$ `Min`, which proves that the base case is true.

We suppose that the statement is true just before the $i(\geq 1)$-th item b_i whose size is a_i is given, and prove the statement holds just after b_i is given. Let j' (j'') denote the number of items which are packed at Case 2.1 (2.2) just before b_i is given. By the induction hypothesis, (a) $\beta_{j'} = t_{max,i} > t_{min,i} = \gamma_{j''}$. Also, when b_i is packed, (b-1) $\beta_{j'+1} = t_{max,i+1} = a_i$ and (b-2) $t_{min,i+1} = t_{min,i}$ hold if Case 2.1 applies. If Case 2.2 applies, then (b-3) $\gamma_{j''+1} = t_{min,i+1} = a_i$ and (b-4) $t_{max,i+1} = t_{max,i}$.

Next, we consider the size a_i of b_i. If $i = 1$, then (c) $\beta_0 > a_1 > \gamma_0$. In the case $i \geq 2$, a_i is determined depending on which of Case 2.1 or Case 2.2 applies to the $(i-1)$-th item b_{i-1} whose size is a_{i-1}. If Case 2.1 applies to b_{i-1}, then (d-1) $a_{i-1} = \beta_{j'} > a_i > t_{min,i-1} = t_{min,i}$. If Case 2.2 applies, (d-2) $t_{max,i-1} = t_{max,i} > a_i > \gamma_{j''} = a_{i-1}$.

Now we prove each case. First, we consider $i = 1$, that is, the moment just after the first item b_1 whose size is a_1 is given. Since no item has been given yet at this moment, $j' = j'' = 0$. If Case 2.1 applies to b_1, then $\beta_0 > a_1 = \beta_1 = t_{max,2}$ by (c) and (b-1). Furthermore, $a_1 > \gamma_0 = t_{min,1} = t_{min,2}$ by (c), (a), and (b-2). Hence, $\beta_0 > \beta_1 = t_{max,2} > t_{min,2} = \gamma_0$ by these inequalities, which means that the statement is true. In the same way, if Case 2.2 applies to b_1, then $\gamma_0 < \gamma_1 = t_{min,2} < t_{max,2} = \beta_0$ by (a), (c), (b-3) and (b-4).

Next, we discuss the case $i \geq 2$. Firstly, we discuss the case where Case 2.1 applies to b_{i-1} and b_i. By (a), (b-1), (b-2) and (d-1), $\beta_{j'} > \beta_{j'+1} = t_{max,i+1} = a_i > t_{min,i} = t_{min,i+1} = \gamma_{j''}$ holds. Thus, the statement holds. Second, if Cases 2.1 and 2.2 apply for b_{i-1} and b_i, respectively, then $\beta_{j'} = t_{max,i+1} = t_{max,i} > a_i = t_{min,i+1} = \gamma_{j''+1} > \gamma_{j''}$ by (a), (b-3), (b-4) and (d-1). We have shown that the statement is true when Case 2.1 applies to b_{i-1}.

The case where Case 2.2 applies to b_{i-1} can be shown in a similar way to the above argument. We have shown that the statement is true just after b_i is given, which completes the proof. □

Now we are ready to describe the main routines any of which the adversary chooses as its strategy. We remark here that Routine1 outputs an equivalent sequence to that used for getting a lower bound for $k = 2$ in [BCKK04]. In that analysis the competitiveness of an online algorithm depends on how it packs the items given in the counterpart of Step 1. Our analysis, in addition, examines how to deal with those given in Step 3 and Step 4 of ROUTINE2 and 3.

The variables t, b, s, x, y, u, z, w, and v appearing in the pseudocodes are used both for the execution of the routine and for the later analysis. The symbol "#" stands for "the number of".

```
ROUTINE1($ON$, Length):
Step 1. Call SUBROUTINE1($ON$, $\frac{1}{10}$, $\frac{1}{9}$, Length), and $t :=$ (the return value).
    Then, $\frac{x}{2} :=$ (# bins with two items), and $y :=$ (# bins with exactly one item).
Step 2. Give $ON$ $\frac{x}{2}$ items of size $1 - t$.
```

```
ROUTINE2($ON$, Length):
Step 1. Call SUBROUTINE1($ON$, $\frac{1}{10}$, $\frac{1}{9}$, Length), and $t :=$ (the return value).
    Then, $\frac{x}{2} :=$ (# bins with two items), and $y :=$ (# bins with exactly one item).
Step 2. Give $ON$ $\frac{x}{2}$ items of size $1 - t$.
Step 3. Call SUBROUTINE2($ON$, $\frac{4}{5}$, $\frac{7}{8}$, $y + \frac{x}{2}$), and $b :=$ (the return value).
    Then, $u :=$ (# bins with one item given in Step 1 and one given in Step 3).
Step 4. Call SUBROUTINE1($ON$, $\frac{1}{6}$, $1 - b$, $u$).
    Then, $z :=$ (# bins with one item given in Step 1 and one given in Step 4).
```

```
ROUTINE3($ON$, Length):
Step 1. Call SUBROUTINE1($ON$, $\frac{1}{10}$, $\frac{1}{9}$, Length), and $t :=$ (the return value).
    Then, $\frac{x}{2} :=$ (# bins with two items), and $y :=$ (# bins with exactly one item).
Step 2. Give $ON$ $\frac{x}{2}$ items of size $1 - t$.
Step 3. Call SUBROUTINE2($ON$, $\frac{4}{5}$, $\frac{7}{8}$, $y + \frac{x}{2}$), and $b :=$ (the return value).
    Then, $u :=$ (# bins with one item given in Step 1 and one given in Step 3).
Step 4. Call SUBROUTINE3($ON$, $\frac{1}{6}$, $1 - b$, $u$), and $s :=$ (the return value).
    Then, $z + w :=$ (# bins with one item given in Step 1 and one given in Step 4), and
$v :=$ (# bins with exactly one item given in Step 4).
Step 5. Give $ON$ $u + z + w$ items of size $1 - s$.
```

For an arbitrary online algorithm ALG and a positive integer `Length`, let ROUTINE1, 2, and 3 run and generate sequences of items σ_1, σ_2, and σ_3, respectively. What should be remarked upon here is that σ_1 is a prefix of σ_2 and σ_2 is a prefix of σ_3. Compare Step 4 in ROUTINE2 and 3. At this step, the routines call different subroutines but with the same arguments. By the terminal conditions, we know that SUBROUTINE3 gives no fewer items than SUBROUTINE1. This verifies the consistency of the setting of the variables $z(\geq 0)$ and $w(\geq 0)$.

Now we see what items are included in the longest sequence σ_3. According to the values t, b, and s determined through the execution of ROUTINE3, we classify all items into the following eight categories:

- $\mathtt{t}_-$-*items*, those which are of size in $(\frac{1}{10}, t]$ and given in Step 1,
- $\mathtt{t}_+$-*items*, those which are of size in $(t, \frac{1}{9})$ and given in Step 1,
- $(\mathtt{1} - \mathtt{t})$-*items*, those which are of size $(1 - t)$ and given in Step 2,
- $\mathtt{b}_-$-*items*, those which are of size in $(\frac{4}{5}, b]$ and given in Step 3,
- $\mathtt{b}_+$-*items*, those which are of size in $(b, \frac{7}{8})$ and given in Step 3,

- $\mathtt{s}_-$-*items*, those which are of size in $(\frac{1}{6}, s]$ and given in Step 4,
- $\mathtt{s}_+$-*items*, those which are of size in $(s, 1-b)$ and given in Step 4, and
- $(\mathtt{1}-\mathtt{s})$-*items*, those which are of size $(1-s)$ and given in Step 5.

Lemma 1 clarifies the magnitude relation among items given in each subroutine. Together with the categorization above, we have the next fact:

Fact. *In the execution of* Subroutine1 *(2, 3, respectively) called by* Routine3*, every item that is put into a bin at Case 2.1 is classified as a* $\mathtt{t}_+$*-item (*$\mathtt{b}_+$*-item,* $\mathtt{s}_+$*-item, respectively), while every item that is put into a bin at Case 2.2 is classified as a* $\mathtt{t}_-$*-item (*$\mathtt{b}_-$*-item,* $\mathtt{s}_-$*-item, respectively).*

The lemma below counts the numbers of non-empty bins of *ALG* and *OPT*. See Figure 1 for the packings.

Lemma 2. *For ALG, OPT, and* (x, y, u, z, w, v) *determined by each of* Routine1*, 2, and 3, it holds that:*

$$\begin{aligned}
\mathtt{Length} &= x + y,\\
C_{ALG}(\sigma_1) &= x + y,\\
C_{OPT}(\sigma_1) &= \frac{1}{2}x + \left\lceil \frac{1}{4}x + \frac{1}{2}y \right\rceil \le \frac{3}{4}x + \frac{1}{2}y + \frac{1}{2},\\
C_{ALG}(\sigma_2) &\ge \frac{3}{2}x + 2y + \frac{1}{2}(u - z),\\
C_{OPT}(\sigma_2) &= x + y + u,\\
C_{ALG}(\sigma_3) &\ge \frac{3}{2}x + y + 3u + v + 2z + 2w,\\
C_{OPT}(\sigma_3) &= x + y + 2u + z + w + \left\lceil \frac{1}{2}v \right\rceil \le x + y + 2u + z + w + \frac{1}{2}v + \frac{1}{2}.
\end{aligned}$$

Proof. We first evaluate $C_{ALG}(\sigma_3)$. As a result of Routine3, *ALG* has packed all items in σ_3 as below (see also Figure 1):

- $\frac{x}{2}$ bins with a $\mathtt{t}_+$-item and a $\mathtt{t}_-$-item,
- $\frac{x}{2}$ bins with only a $(\mathtt{1}-\mathtt{t})$-item,
- u bins with a $\mathtt{t}_+$-item and a $\mathtt{b}_-$-item,
- $z + w$ bins with a $\mathtt{t}_+$-item and a $\mathtt{s}_-$-item,
- some bins with a $\mathtt{t}_+$-item and a $(\mathtt{1}-\mathtt{s})$-item,
- some bins with only a $\mathtt{t}_+$-item,
- $y + \frac{x}{2}$ bins with only a $\mathtt{b}_+$-item,
- u bins with a $\mathtt{s}_+$-item and a $\mathtt{s}_-$-item,
- v bins with a $\mathtt{s}_+$-item, and
- some bins with only a $(\mathtt{1}-\mathtt{s})$-item.

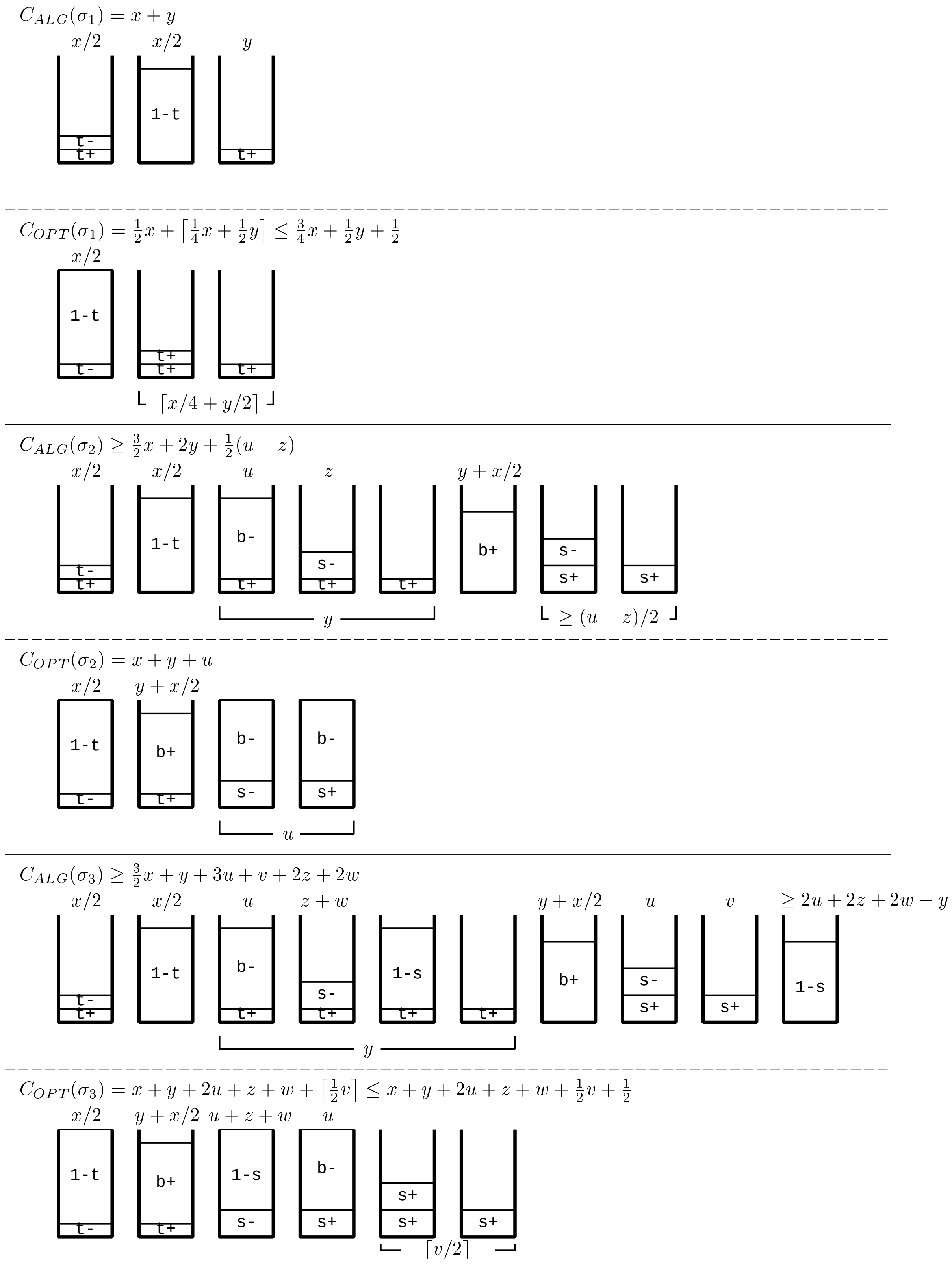
$C_{ALG}(\sigma_1) = x + y$
$x/2$ $x/2$ y
t- t+ 1-t t+
$C_{OPT}(\sigma_1) = \frac{1}{2}x + \lceil \frac{1}{4}x + \frac{1}{2}y \rceil \le \frac{3}{4}x + \frac{1}{2}y + \frac{1}{2}$
$x/2$
1-t t- t+ t+ t+
$\lceil x/4 + y/2 \rceil$
$C_{ALG}(\sigma_2) \ge \frac{3}{2}x + 2y + \frac{1}{2}(u - z)$
$x/2$ $x/2$ u z $y + x/2$
t- t+ 1-t b- t+ s- t+ t+ b+ s- s+ s+
y
$\ge (u - z)/2$
$C_{OPT}(\sigma_2) = x + y + u$
$x/2$ $y + x/2$
1-t t- b+ t+ b- s- b- s+
u
$C_{ALG}(\sigma_3) \ge \frac{3}{2}x + y + 3u + v + 2z + 2w$
$x/2$ $x/2$ u $z + w$ $y + x/2$ u v $\ge 2u + 2z + 2w - y$
t- t+ 1-t b- t+ s- t+ 1-s t+ t+ b+ s- s+ s+ 1-s
y
$C_{OPT}(\sigma_3) = x + y + 2u + z + w + \lceil \frac{1}{2}v \rceil \le x + y + 2u + z + w + \frac{1}{2}v + \frac{1}{2}$
$x/2$ $y + x/2$ $u + z + w$ u
1-t t- b+ t+ 1-s s- b- s+ s+ s+ s+
$\lceil v/2 \rceil$

Figure 1: Packings by an arbitrary online algorithm and an optimal offline algorithm for the input sequences generated by Routine1, 2, and 3. The expression above (or below) a bin indicates the number of such bins, that is, those which contain items as depicted.

Note that all possible packings are found in this list; by the preceding **Fact**, we know that *ALG* never combines, for example, a pair of a $\mathtt{t}_-$-item and a $(\mathtt{1}-\mathtt{t})$-item, or a pair of a $\mathtt{s}_-$-item and a $\mathtt{b}_+$-item.

By tracing the execution of ROUTINE3, we can easily confirm the number of bins belonging to each type as the above list (see also Figure 1). To evaluate $C_{ALG}(\sigma_3)$, it is sufficient to bound *the number of bins containing solely a* $(\mathtt{1}-\mathtt{s})$*-item.* That is to say, we need not count the bins with a $\mathtt{t}_+$-item and a $(\mathtt{1}-\mathtt{s})$-item, or those with only a $\mathtt{t}_+$-item; we know that there are y $\mathtt{t}_+$-items in total, as shown in Figure 1.

Consider the situation immediately before Step 5. Now there are $(y-(u+z+w))$ bins which contain only a $\mathtt{t}_+$-item. Each of $(u+z+w)$ $(\mathtt{1}-\mathtt{s})$-items, which are about to arrive, can be packed with one of these $\mathtt{t}_+$-items. *ALG* can pack all $(\mathtt{1}-\mathtt{s})$-items successfully if it opens at least $(u+z+w-(y-(u+z+w)))=(2u+2z+2w-y)$ new bins. Therefore, after Step 5, there must be at least $(2u+2z+2w-y)$ bins which contain only a $(\mathtt{1}-\mathtt{s})$-item. Note that this claim holds true even if $(2u+2z+2w-y)$ is negative; this case is consistent with the situation that every $(\mathtt{1}-\mathtt{s})$-item is packed with one of the $\mathtt{t}_+$-items and consequently there is no need to open a new bin. We thus evaluate $C_{ALG}(\sigma_3)$ by the above inequality.

On the other hand, *OPT* packs all items in σ_3 as below (see also Figure 1):

- $\frac{x}{2}$ bins with a $\mathtt{t}_-$-item and a $(\mathtt{1}-\mathtt{t})$-item,
- $y+\frac{x}{2}$ bins with a $\mathtt{t}_+$-item and a $\mathtt{b}_+$-item,
- $u+z+w$ bins with a $\mathtt{s}_-$-item and a $(\mathtt{1}-\mathtt{s})$-item,
- u bins with a $\mathtt{s}_+$-item and a $\mathtt{b}_-$-item, and
- $\left\lceil \frac{1}{2}v \right\rceil$ bins with one or two $\mathtt{s}_+$-items.

We give a supplementary explanation on how to pack $\mathtt{s}_+$-items. *OPT* first packs as many $\mathtt{s}_+$-items with $\mathtt{b}_-$-items as possible. It next tries to create couples of the remaining $\mathtt{s}_+$-items, that is to say, creates as many bins with two $\mathtt{s}_+$-items as possible and then a bin with just a $\mathtt{s}_+$-item if a single $\mathtt{s}_+$-item is left at the end. The ceiling function is employed to represent inclusively the number of bins containing only one or two $\mathtt{s}_+$-items without regard to the parity.

Since σ_1 and σ_2 are prefixes of σ_3, the evaluations with respect to $C_{ALG}(\sigma_1)$, $C_{OPT}(\sigma_1)$, $C_{ALG}(\sigma_2)$, and $C_{OPT}(\sigma_2)$ are done straightforwardly. Please see Figure 1 for the packings of σ_1 and σ_2. We here mention $C_{OPT}(\sigma_1)$ and $C_{ALG}(\sigma_2)$, which need some care.

$C_{OPT}(\sigma_1)$ is evaluated as follows: *OPT* lets $\frac{x}{2}$ bins contain a $\mathtt{t}_-$-item and a $(\mathtt{1}-\mathtt{t})$-item. It then tries to create couples of $\mathtt{t}_+$-items. Depending on the parity of the number, we have two different representations. Thus, we use the ceiling function for this.

$C_{ALG}(\sigma_2)$ is evaluated as follows: ROUTINE2 Step 4 issues u items in total. *ALG* puts z items out of them together with $\mathtt{t}_+$-items. To pack the remaining $(u-z)$ items requires at least $\frac{1}{2}(u-z)$ new bins.

SUBROUTINE1 called by Step 1 of ROUTINE1, 2, and 3 issues `Length` items. Then, *ALG* creates $\frac{x}{2}$ bins with a $\mathtt{t}_-$-item and a $\mathtt{t}_+$-item, and y bins with only a $\mathtt{t}_+$-item. Counting the total number of items, we easily have $\mathtt{Length}=x+y$. □

The next lemma is the heart of our analysis.

Lemma 3. *For an arbitrary online algorithm ALG and any $\varepsilon>0$, there exists a positive integer* `Length` *such that: Let* ROUTINE1*, 2, and* 3 *run with ALG and* `Length` *as arguments, and generate sequences of items σ_1, σ_2, and σ_3, respectively. Then, it follows that*

$$\max\left\{\frac{C_{ALG}(\sigma_1)}{C_{OPT}(\sigma_1)},\frac{C_{ALG}(\sigma_2)}{C_{OPT}(\sigma_2)},\frac{C_{ALG}(\sigma_3)}{C_{OPT}(\sigma_3)}\right\}\geq\overline{r}-\varepsilon, \tag{1}$$

where $\overline{r}(\approx 1.42764)$ *is the root of the cubic equation* $2r^3 - 17r^2 + 30r - 14 = 0$ *which lies between* $\frac{4}{3}$ *and* $\frac{3}{2}$.

Proof. Lemma 2 implies that it suffices to prove that: For any $\varepsilon > 0$, there exists a positive integer `Length` such that for any nonnegative integers x, y, u, z, w, and v with $x + y = \texttt{Length}$,

$$\max\left\{\frac{x+y}{\frac{3}{4}x+\frac{1}{2}y+\frac{1}{2}}, \frac{\frac{3}{2}x+2y+\frac{1}{2}(u-z)}{x+y+u}, \frac{\frac{3}{2}x+y+3u+v+2z+2w}{x+y+2u+z+w+\frac{1}{2}v+\frac{1}{2}}\right\} \geq \overline{r} - \varepsilon. \qquad (2)$$

This follows from the following Lemmas 4 and 5. By the setting of variables in ROUTINE1, 2, and 3, it implicitly holds that x is even and $u + z + w \leq y$. Note that although Lemmas 4 and 5 hold true even without these restrictions, this is of course sufficient. □

The following lemma claims that when sufficiently many items are given, the constant $\frac{1}{2}$ appearing in $C_{OPT}(\sigma_1)$ and $C_{OPT}(\sigma_3)$ is negligible.

Lemma 4. *For any* $\varepsilon > 0$, *there exists a positive integer* `Length` *such that for any nonnegative integers* x, y, u, z, w, *and* v *with* $x + y = \texttt{Length}$,

$$\frac{x+y}{\frac{3}{4}x+\frac{1}{2}y} - \frac{x+y}{\frac{3}{4}x+\frac{1}{2}y+\frac{1}{2}} \leq \varepsilon, \qquad (3)$$

$$\frac{\frac{3}{2}x+y+3u+v+2z+2w}{x+y+2u+z+w+\frac{1}{2}v} - \frac{\frac{3}{2}x+y+3u+v+2z+2w}{x+y+2u+z+w+\frac{1}{2}v+\frac{1}{2}} \leq \varepsilon. \qquad (4)$$

Proof. We first give a simple observation: Let A, B, c, and ε be positive numbers.

$$\frac{A}{B} - \frac{A}{B+c} \leq \varepsilon$$

if and only if

$$\frac{B}{A}\left(1 + \frac{B}{c}\right) \geq \frac{1}{\varepsilon}.$$

The proof is done by simply taking the inverse of both sides.

First, for any $\varepsilon > 0$ we show the inequality (4) by choosing `Length` properly. Let $A := \frac{3}{2}x + y + 3u + v + 2z + 2w$ and $B := x + y + 2u + z + w + \frac{1}{2}v$. Since x, y, u, z, w, and v are nonnegative, $A \leq 2x + 2y + 4u + 2z + 2w + v = 2B$ and $B \geq x + y = \texttt{Length}$ hold true. We then have $\frac{B}{A}(1 + \frac{B}{\frac{1}{2}}) \geq \frac{1}{2}(1 + \frac{\texttt{Length}}{\frac{1}{2}})$. The above observation with $c = \frac{1}{2}$ implies that (4) holds if we choose $\texttt{Length} \geq \frac{1}{\varepsilon} - \frac{1}{2}$.

Similarly we do with (3). Let $A := x + y$ and $B := \frac{3}{4}x + \frac{1}{2}y$. $A \leq \frac{3}{2}x + y = 2B$ and $B \geq \frac{1}{2}x + \frac{1}{2}y = \frac{1}{2}\texttt{Length}$ imply that $\frac{B}{A}\left(1 + \frac{B}{\frac{1}{2}}\right) \geq \frac{1}{2}\left(1 + \texttt{Length}\right)$. The above observation states that we should choose $\texttt{Length} \geq \frac{2}{\varepsilon} - 1$ for (3). □

Lemma 5. *For any nonnegative integers* x, y, u, z, w, *and* v *with* $x + y > 0$,

$$\max\left\{\frac{x+y}{\frac{3}{4}x+\frac{1}{2}y}, \frac{\frac{3}{2}x+2y+\frac{1}{2}(u-z)}{x+y+u}, \frac{\frac{3}{2}x+y+3u+v+2z+2w}{x+y+2u+z+w+\frac{1}{2}v}\right\} \geq \overline{r}, \qquad (5)$$

where $\overline{r}(\approx 1.42764)$ *is the root of the cubic equation* $2r^3 - 17r^2 + 30r - 14 = 0$ *which lies between* $\frac{4}{3}$ *and* $\frac{3}{2}$.

Proof. The proof is done by contradiction. Assume the lemma to be false. Then, all of the operands of the max operation in (5) can fall below $\overline{r}$ at the same time. That is to say, there exists a tuple of nonnegative integers (x, y, u, z, w, v) with $x + y > 0$ such that

$$f_1 := x + y - \overline{r}\left(\frac{3}{4}x + \frac{1}{2}y\right) < 0, \tag{6}$$

$$f_2 := \frac{3}{2}x + 2y + \frac{1}{2}(u - z) - \overline{r}\,(x + y + u) < 0, \tag{7}$$

$$f_3 := \frac{3}{2}x + y + 3u + v + 2z + 2w - \overline{r}\left(x + y + 2u + z + w + \frac{1}{2}v\right) < 0. \tag{8}$$

In what follows we show that there is no such (x, y, u, z, w, v). Specifically, we derive an inequality that does not contain either x, y, or u from the inequalities (6), (7), and (8). We then claim that there do not exist z, w, and v which satisfy the derived inequality.

Recall $\frac{4}{3} < \overline{r} < \frac{3}{2}$. Noting that $3 - 2\overline{r}$ and $2\overline{r} - 1$ are both positive, we have an inequality without u from (7) and (8).

$$\begin{aligned} f_4 :=& 4(3 - 2\overline{r})f_2 + 2(2\overline{r} - 1)f_3 \\ =& \left(-2\overline{r}^2 + 5\overline{r} - 2\right) v + \left(-4\overline{r}^2 + 10\overline{r} - 4\right) w + \left(4\overline{r}^2 - 16\overline{r} + 15\right) x \\ &+ \left(4\overline{r}^2 - 22\overline{r} + 22\right) y + \left(-4\overline{r}^2 + 14\overline{r} - 10\right) z \\ <& 0. \end{aligned}$$

(Please see that the elimination is done so that the resulting inequality sign makes sense.) Next, let us eliminate x. The coefficient of x in the above inequality $4\overline{r}^2 - 16\overline{r} + 15 = (5 - 2\overline{r})(3 - 2\overline{r})$ is confirmed to be positive. Together with positivity of $3\overline{r} - 4$, we eliminate x using (6).

$$\begin{aligned} f_5 :=& 4(4\overline{r}^2 - 16\overline{r} + 15)f_1 + (3\overline{r} - 4)\, f_4 \\ =& \left(-6\overline{r}^3 + 23\overline{r}^2 - 26\overline{r} + 8\right)(v + 2w) + \\ & 2\left(2\overline{r}^3 - 17\overline{r}^2 + 30\overline{r} - 14\right) y + 2\left(-6\overline{r}^3 + 29\overline{r}^2 - 43\overline{r} + 20\right) z \\ =& \left(-6\overline{r}^3 + 23\overline{r}^2 - 26\overline{r} + 8\right)(v + 2w) + 2\left(-6\overline{r}^3 + 29\overline{r}^2 - 43\overline{r} + 20\right) z \\ <& 0. \end{aligned}$$

The reason why y has gone is because $\overline{r}$ is a root of $2r^3 - 17r^2 + 30r - 14 = 0$. The inequality $\frac{4}{3} < \overline{r} < \frac{3}{2}$ leads to that both $(-6\overline{r}^3 + 23\overline{r}^2 - 26\overline{r} + 8) = (2 - \overline{r})(2\overline{r} - 1)(3\overline{r} - 4)$ and $(-6\overline{r}^3 + 29\overline{r}^2 - 43\overline{r} + 20) = (\overline{r} - 1)(5 - 2\overline{r})(3\overline{r} - 4)$ are positive. Therefore, for fulfilling $f_5 < 0$, either z, w, or v should be negative. This contradicts the assumption that z, w, and v are all nonnegative. □

Our new lower bound is obtained almost as a corollary from Lemma 3.

Theorem 1. *Any online algorithm for the online 2-item bin packing problem has an asymptotic competitive ratio of at least $\overline{r}$, where $\overline{r}(\approx 1.42764)$ is the root of the cubic equation $2r^3 - 17r^2 + 30r - 14 = 0$ which lies between $\frac{4}{3}$ and $\frac{3}{2}$.*

Proof. Let n_0 be an arbitrary positive integer and

$$T_n := \sup_{\sigma}\left\{\frac{C_{ALG}(\sigma)}{C_{OPT}(\sigma)} \;\middle|\; C_{OPT}(\sigma) \geq n\right\}.$$

Run Routine1, 2, and 3 with ALG and `Length` $:= 2n_0$ as arguments and obtain sequences of items σ_1, σ_2, and σ_3, respectively. Since

$$\begin{aligned} C_{OPT}(\sigma_3) \geq C_{OPT}(\sigma_2) \geq C_{OPT}(\sigma_1) = \frac{1}{2}x + \left\lceil \frac{1}{4}x + \frac{1}{2}y \right\rceil &\geq \frac{3}{4}x + \frac{1}{2}y \\ &\geq \frac{1}{2}(x + y) = \frac{1}{2}\texttt{Length} = n_0, \end{aligned}$$

we have

$$T_{n_0} \geq \max\Big\{\frac{C_{ALG}(\sigma_1)}{C_{OPT}(\sigma_1)}, \frac{C_{ALG}(\sigma_2)}{C_{OPT}(\sigma_2)}, \frac{C_{ALG}(\sigma_3)}{C_{OPT}(\sigma_3)}\Big\}.$$

This inequality and Lemma 3 imply that for any $\varepsilon > 0$, there exists n_0 such that $T_{n_0} \geq \overline{r} - \varepsilon$.

On the other hand, by definition, the sequence $\{T_n\}$ is non-increasing. Therefore,

$$R_{ALG} = \limsup_{n\to\infty} \sup_{\sigma} \Big\{\frac{C_{ALG}(\sigma)}{C_{OPT}(\sigma)} \;\Big|\; C_{OPT}(\sigma) = n\Big\} = \lim_{n\to\infty} T_n = \inf T_n \geq \overline{r}.$$

□

3 Lower Bounds for $k \geq 4$

We propose an approach for deriving a lower bound for the online k-item bin packing problem for each $k \geq 4$, expanding the method of van Vliet [vV92] for the problem without a cardinality constraint. His method was to solve a linear program in which variables represent the packings by an arbitrary online algorithm given some patterns of input sequences. We illustrate how to embed a cardinality constraint into the linear program.

Some existing lower bounds for the problem without a cardinality constraint, such as [Yao80, vV92, BBG12], can be interpreted as lower bounds for the online k-item bin packing for some ranges of k; if the possible item size is restricted to be at least s, then the problem can be seen as the online k-item bin packing for $k \geq \lfloor\frac{1}{s}\rfloor$, since there is no chance that more than $\lfloor\frac{1}{s}\rfloor$ items are packed together. Such results include: A lower bound of $\frac{4}{3}$ for $4 \leq k \leq 5$ [vV92], $\frac{3}{2}$ for $6 \leq k \leq 41$ [Yao80], $\frac{217}{141}$ for $42 \leq k \leq 293$ [vV92], and $\frac{10633}{6903}$ for $294 \leq k \leq 2057$ [BBG12]. See Table 1 in Section 1. Note that although the paper [vV92] does not provide the value of $\frac{4}{3}$ explicitly, it is given just by slightly changing the settings of his method. In the derivation of these results, the arbitrary online algorithm behaves without taking care of the cardinality constraint; due to the setting of item size, the resulting packing naturally satisfies the cardinality constraint. In this section, after the reformulation of a linear program, we set $k < \lfloor\frac{1}{s}\rfloor$ and try to obtain a better lower bound.

We first give our new formulation with the cardinality constraint. We are given a tuple of item sizes $(s_1, \dots, s_l)$ with $\sum_{i=1}^{l} s_i \leq 1$. Set N a positive integer divisible by k and $\lfloor\frac{1}{\sum_{h=i}^{l} s_h}\rfloor$ for all $1 \leq i \leq l$. Let L_i be a sequence of N items of size s_i for each $1 \leq i \leq l$. The adversary issues any of $L_l \cdots L_i$ $(1 \leq i \leq l)$.

We denote by a vector $(t_1, \dots, t_l)^T$ a packing of a bin that consists of t_i items of size s_i for $1 \leq i \leq l$. Any packing has to satisfy the following constraints: (i) the capacity constraint $\sum_{i=1}^{l} t_i s_i \leq 1$, (ii) the cardinality constraint $\sum_{i=1}^{l} t_i \leq k$, and (iii) the constraint that only non-empty bins are taken into account $\sum_{i=1}^{l} t_i > 0$.

Sort all feasible packings in a lexicographical order with a later entry having higher priority. For example, for $(s_1, s_2, s_3) = (\frac{1}{2}+\varepsilon, \frac{1}{3}+\varepsilon, \frac{1}{7}+\varepsilon)$ and $k = 4$, we have a list of feasible packings sorted as $(1,0,0)^T, (0,1,0)^T, (1,1,0)^T, \dots, (1,0,3)^T, (0,1,3)^T, (0,0,4)^T$. Denote by $t_{i,j}$ the i-th entry of the j-th packing in the sorted list. The set of $t_{i,j}$'s is regarded as a matrix. For the above example,

$$(t_{i,j}) = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 2 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 1 & 2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 4 \end{pmatrix}. \quad (9)$$

Let m be the number of feasible packings, which is 17 for this example.

Fix an online algorithm ALG arbitrarily. Suppose that given the input sequence $L_l \cdots L_1$, ALG creates n_j bins according to the j-th packing (i.e., $(t_{1,j}, \dots, t_{l,j})^T$). Define p_i as the index of the first column that has a non-zero entry in the i-th row of the matrix $(t_{i,j})$. Then it holds that for $i \geq 2$ a packing before the p_i-th is one that does not contain any items of size $s_l, \dots, s_i$ and begins packing items of size s_{i-1}, and that $p_1 = 1$. For the above example, $p_1 = 1$, $p_2 = 2$, and $p_3 = 5$. Thus, we can describe the total number of non-empty bins of ALG for $L_l \cdots L_i$ $(1 \leq i \leq l)$ as

$$C_{ALG}(L_l \cdots L_i) = \sum_{j=p_i}^{m} n_j. \tag{10}$$

The total number of non-empty bins of an optimal offline algorithm OPT is bounded by a simple but nontrivial lemma.

Lemma 6. *For $1 \leq i \leq l$, $C_{OPT}(L_l \cdots L_i) \leq \max\{\frac{N}{\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor}, \frac{N(l-i+1)}{k}\}$.*

Proof. Sort items in the input sequence $L_l \cdots L_i$ so that the sizes are ordered as

$$\overbrace{s_i, s_{i+1}, \dots, s_l}^{l-i+1}, \overbrace{s_i, s_{i+1}, \dots, s_l}^{l-i+1}, \cdots, \overbrace{s_i, s_{i+1}, \dots, s_l}^{l-i+1},$$

and denote the sorted sequence by σ. The proof is done by presenting an offline algorithm OFF that repeatedly takes a fixed number of items from the front of σ and puts them into a new bin.

We call any set of $(l - i + 1)$ items whose sizes are $s_i, \dots, s_l$, respectively, a *collection.* As a preparation we remark the following. Suppose that the cardinality constraint can be ignored and OFF puts as many disjoint collections into a bin as possible only subject to the capacity constraint. Then, $\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$ disjoint collections, which consist of $(l - i + 1)\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$ items in total, can be packed into a bin.

(i) Case $k \geq (l - i + 1)\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$. We let OFF repeatedly take every $(l - i + 1)\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$ items from the front of σ and put them into a new bin. Since the total number of items is $N(l - i + 1)$ and N is divisible by $\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$, we have

$$C_{OPT}(L_l \cdots L_i) \leq C_{OFF}(L_l \cdots L_i) = \frac{N(l - i + 1)}{(l - i + 1)\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor} = \frac{N}{\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor}.$$

(ii) Case $k < (l - i + 1)\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$. We let OFF repeatedly take every k items from the front of σ and put them into a new bin. In what follows, we show that in each bin, the total size of items does not exceed one.

It is observed that when σ is separated into subsequences with length k, each of the subsequences consists of $\lfloor \frac{k}{l-i+1} \rfloor$ disjoint collections and a subset of a collection that contains $(k - \lfloor \frac{k}{l-i+1} \rfloor)$ items. Therefore, the total size of items in each subsequence is no greater than the total size of items in $\lceil \frac{k}{l-i+1} \rceil$ disjoint collections, which is $\lceil \frac{k}{l-i+1} \rceil \sum_{h=i}^{l} s_h$. We next claim that $\lceil \frac{k}{l-i+1} \rceil \sum_{h=i}^{l} s_h \leq 1$.

By $k < (l-i+1)\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$ and integrality of $\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$, we have $\frac{k}{l-i+1} \leq \lceil \frac{k}{l-i+1} \rceil \leq \lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$. Since $\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor \sum_{h=i}^{l} s_h \leq 1$, we conclude $\lceil \frac{k}{l-i+1} \rceil \sum_{h=i}^{l} s_h \leq 1$.

Hence OFF puts exactly k items into each bin. Since N is divisible by k, we have

$$C_{OPT}(L_l \cdots L_i) \leq C_{OFF}(L_l \cdots L_i) = \frac{N(l - i + 1)}{k}.$$

□

As a matter of course, the whole set of bins created by ALG for $L_l \cdots L_1$ contains N items of size s_i for each $1 \leq i \leq l$. This fact is described as $\sum_{j=1}^{m} t_{i,j} n_j = N$ for all $1 \leq i \leq l$. Note that as long as this equation holds, the packings for $L_l \cdots L_i$ $(1 \leq i \leq l-1)$ are consistent as well. For later formulation, we rewrite this as

$$\sum_{j=1}^{m} \frac{t_{i,j} n_j}{N} - 1 = 0, \quad 1 \leq i \leq l. \tag{11}$$

The asymptotic competitive ratio R_{ALG} is asymptotically lower-bounded by R such that

$$\frac{C_{ALG}(L_l \cdots L_i)}{C_{OPT}(L_l \cdots L_i)} - R \leq 0, \quad 1 \leq i \leq l. \tag{12}$$

A sufficient condition for (12) with slack variables $(u_1, \ldots, u_l)$ is

$$\min\Big\{ \Big\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \Big\rfloor, \frac{k}{l-i+1} \Big\} \sum_{j=p_i}^{m} \frac{n_j}{N} + u_i - R = 0, \quad 1 \leq i \leq l \tag{13}$$

for some $u_i \geq 0$ $(1 \leq i \leq l)$. The derivation follows from (10) and $\frac{N}{C_{OPT}(L_l \cdots L_i)} \geq \min\{\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor, \frac{k}{l-i+1}\}$ obtained from Lemma 6.

The problem of finding the minimum R that satisfies (11) and (13) is formulated as a mathematical program $(\mathcal{P}_N)$ with a $2l \times (m+l+1)$-matrix $A = (a_{i,j})$ and vectors $\boldsymbol{x}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ as below.

$$a_{i,j} = \begin{cases} t_{i,j}, & 1 \leq i \leq l, 1 \leq j \leq m; \\ 0, & 1 \leq i \leq l, m+1 \leq j \leq m+l+1; \\ 0, & l+1 \leq i \leq 2l, 1 \leq j \leq p_{i-l} - 1; \\ \min\{\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor, \frac{k}{l-i+1}\}, & l+1 \leq i \leq 2l, p_{i-l} \leq j \leq m; \\ \delta_{i-l,j-m}, & l+1 \leq i \leq 2l, m+1 \leq j \leq m+l; \\ -1, & l+1 \leq i \leq 2l, j = m+l+1. \end{cases} \tag{14}$$

$$\boldsymbol{x} = \Big(\frac{n_1}{N}, \ldots, \frac{n_m}{N}, u_1, \ldots, u_l, R\Big)^T, \boldsymbol{b} = (\overbrace{1, \ldots, 1}^{l}, \overbrace{0, \ldots, 0}^{l})^T, \boldsymbol{c} = (\overbrace{0, \ldots, 0}^{m+l}, 1)^T. \tag{15}$$

$$\begin{aligned} (\mathcal{P}_N) \quad & \text{minimize } \boldsymbol{c}^T \boldsymbol{x} \\ & \text{subject to } A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{x} = \Big(\frac{n_1}{N}, \ldots, \frac{n_m}{N}, u_1, \ldots, u_l, R\Big) \\ & (n_1, \ldots, n_m) \in \mathbb{Z}^m, (u_1, \ldots, u_l, R) \in \mathbb{R}^{l+1} \end{aligned}$$

Here $\delta_{\cdot,\cdot}$ is Kronecker delta, that is, if $i = j$, $\delta_{i,j} = 1$; otherwise, $\delta_{i,j} = 0$. We will later give an example of A, $\boldsymbol{b}$, and $\boldsymbol{c}$ for $(s_1, s_2, s_3) = (\frac{1}{2} + \varepsilon, \frac{1}{3} + \varepsilon, \frac{1}{7} + \varepsilon)$ and $k = 4$.

In the constraint $A\boldsymbol{x} = \boldsymbol{b}$, the first l rows correspond to (11), while the $(l+1)$-th to $2l$-th rows correspond to (13). $\delta_{i-l,j-m}$ lets the slack variable u_i appear in the equation of the $(l+i)$-th row. The objective function is $\boldsymbol{c}^T \boldsymbol{x} = R$. Note that A, $\boldsymbol{b}$, and $\boldsymbol{c}$ are independent of the choice of N.

Apparently, the optimal value of the following linear program $(\mathcal{P})$ is a lower bound on the optimal value of $(\mathcal{P}_N)$.

$$\begin{aligned} (\mathcal{P}) \quad & \text{minimize } \boldsymbol{c}^T \boldsymbol{x} \\ & \text{subject to } A\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{x} \in \mathbb{R}^{m+l+1} \end{aligned}$$

Lemma 7. *For given k and a tuple of item sizes $(s_1, \dots, s_l)$ with $\sum_{i=1}^{l} s_i \le 1$, formulate $(\mathcal{P})$. Then, any online algorithm for the online k-item bin packing problem has an asymptotic competitive ratio of at least the optimal value of $(\mathcal{P})$.*

Proof. Let ALG be an arbitrary online algorithm and

$$T_n := \sup_{\sigma} \Big\{ \frac{C_{ALG}(\sigma)}{C_{OPT}(\sigma)} \;\Big|\; C_{OPT}(\sigma) \ge n \Big\}.$$

Choose N to be a positive integer divisible by k and $\lfloor \frac{1}{\sum_{h=i}^{l} s_h} \rfloor$ for all $1 \le i \le l$. From the cardinality constraint, it follows that for $1 \le i \le l$,

$$C_{OPT}(L_l \cdots L_i) \ge C_{OPT}(L_l) \ge \frac{N}{k}.$$

Hence, we have for $n_0 = \frac{N}{k}$,

$$\begin{aligned} T_{n_0} &\ge \max \Big\{ \frac{C_{ALG}(L_l \cdots L_i)}{C_{OPT}(L_l \cdots L_i)} \;\Big|\; 1 \le i \le l \Big\} \\ &= (\text{the optimal value of } (\mathcal{P}_N)) \ge (\text{the optimal value of } (\mathcal{P})). \end{aligned}$$

According to the choice of N, n_0 can take an arbitrary large value.

On the other hand, by definition, the sequence $\{T_n\}$ is non-increasing. Therefore,

$$R_{ALG} = \limsup_{n \to \infty} \sup_{\sigma} \Big\{ \frac{C_{ALG}(\sigma)}{C_{OPT}(\sigma)} \;\Big|\; C_{OPT}(\sigma) = n \Big\} = \lim_{n \to \infty} T_n = \inf T_n \ge (\text{the optimal value of } (\mathcal{P})).$$

□

The next theorem provides a lower bound for each $4 \le k \le 45$. The reason why we do not mention $k \ge 46$ is simply because of space limitation. Note that as long as the computer power is available, one can calculate a lower bound for arbitrary k using our method.

The tuples of item sizes we employ are those proposed by Balogh et al. [BBG12] for obtaining a lower bound for the problem without a cardinality constraint, which were derived by modifying those of van Vliet [vV92]. (Although Balogh et al. gave a different representation of formulation, the nature is the same.) As mentioned before, in our analysis k is set smaller than the inverse of the possible smallest item size.

Theorem 2. *For each $4 \le k \le 45$, any online algorithm for the online k-item bin packing problem has an asymptotic competitive ratio of at least the value of "lower bound" in Table 2.*

Proof. The proof is based on Lemma 7. Given k, we use a tuple of item sizes $(\frac{1}{2}+\varepsilon, \frac{1}{3}+\varepsilon, \frac{1}{7}+\varepsilon)$ for $k = 4$ and 5, $(\frac{1}{2}+\varepsilon, \frac{1}{3}+\varepsilon, \frac{1}{7}+\varepsilon, \frac{1}{43}+\varepsilon)$ for $6 \le k \le 41$, and $(\frac{1}{2}+\varepsilon, \frac{1}{3}+\varepsilon, \frac{1}{7}+\varepsilon, \frac{1}{49}+\varepsilon, \frac{1}{295}+\varepsilon)$ for $42 \le k \le 45$, where ε is set to be such a positive number that the sum of the sizes is no larger than one. Using this setting we consider the linear program $(\mathcal{P})$. Lemma 8 guarantees the optimal value of $(\mathcal{P})$ is at least the value of "lower bound" in Table 2. □

We look into the results before presenting Lemma 8. Table 2 says that for the cases $k = 4$, 5, and $10 \le k \le 41$, our results improve the known bounds. In contrast, one can see two types of anomaly. (A) For each of $k = 5$, $13 \le k \le 30$, and $42 \le k \le 45$, the obtained bound does not exceed that for some smaller k. (B) Any bound for $42 \le k \le 45$ is smaller than the known bound $\frac{217}{141}(\approx 1.53900)$ [vV92].

It is unlikely that the value of the matching upper and lower bound is not increasing with respect to k. The above anomalies suggest a limit of our method; for some values of k, our

Table 2: Our new lower bounds for each $4 \leq k \leq 45$. Bold font indicates improvement.

k	lower bound	k	lower bound	k	lower bound
4	$\boldsymbol{\frac{3}{2}(= 1.5)}$	18	$\boldsymbol{\frac{93}{61}(\approx 1.52459)}$	32	$\boldsymbol{\frac{496}{323}(\approx 1.53560)}$
5	$\boldsymbol{\frac{25}{17}(\approx 1.47058)}$	19	$\boldsymbol{\frac{171}{112}(\approx 1.52678)}$	33	$\boldsymbol{\frac{341}{222}(\approx 1.53603)}$
6	$\frac{3}{2}(= 1.5)$	20	$\boldsymbol{\frac{315}{206}(\approx 1.52912)}$	34	$\boldsymbol{\frac{527}{343}(\approx 1.53644)}$
7	$\frac{3}{2}(= 1.5)$	21	$\boldsymbol{\frac{26}{17}(\approx 1.52941)}$	35	$\boldsymbol{\frac{1085}{706}(\approx 1.53682)}$
8	$\frac{3}{2}(= 1.5)$	22	$\boldsymbol{\frac{341}{223}(\approx 1.52914)}$	36	$\boldsymbol{\frac{186}{121}(\approx 1.53719)}$
9	$\frac{3}{2}(= 1.5)$	23	$\boldsymbol{\frac{713}{466}(\approx 1.53004)}$	37	$\boldsymbol{\frac{1147}{746}(\approx 1.53753)}$
10	$\boldsymbol{\frac{80}{53}(\approx 1.50943)}$	24	$\boldsymbol{\frac{124}{81}(\approx 1.53086)}$	38	$\boldsymbol{\frac{589}{383}(\approx 1.53785)}$
11	$\boldsymbol{\frac{44}{29}(\approx 1.51724)}$	25	$\boldsymbol{\frac{775}{506}(\approx 1.53162)}$	39	$\boldsymbol{\frac{403}{262}(\approx 1.53816)}$
12	$\boldsymbol{\frac{66}{43}(\approx 1.53488)}$	26	$\boldsymbol{\frac{403}{263}(\approx 1.53231)}$	40	$\boldsymbol{\frac{20}{13}(\approx 1.53846)}$
13	$\boldsymbol{\frac{26}{17}(\approx 1.52941)}$	27	$\boldsymbol{\frac{279}{182}(\approx 1.53296)}$	41	$\boldsymbol{\frac{1271}{826}(\approx 1.53874)}$
14	$\boldsymbol{\frac{441}{289}(\approx 1.52595)}$	28	$\boldsymbol{\frac{434}{283}(\approx 1.53356)}$	42	$\frac{1519}{993}(\approx 1.52970)$
15	$\boldsymbol{\frac{315}{206}(\approx 1.52912)}$	29	$\boldsymbol{\frac{899}{586}(\approx 1.53412)}$	43	$\frac{9331}{6098}(\approx 1.53017)$
16	$\boldsymbol{\frac{624}{409}(\approx 1.52567)}$	30	$\boldsymbol{\frac{155}{101}(\approx 1.53465)}$	44	$\frac{4774}{3119}(\approx 1.53061)$
17	$\boldsymbol{\frac{527}{346}(\approx 1.52312)}$	31	$\boldsymbol{\frac{961}{626}(\approx 1.53514)}$	45	$\frac{3255}{2126}(\approx 1.53104)$

method fails to derive a good lower bound. It is an interesting open problem to construct a better scheme for whatever k.

We here add that for $k = 5$, we searched for a tuple of item sizes which is different from the tuple of Balogh et al., with the aim of obtaining an even larger bound. Nevertheless, as far as we tried, we could not find a better tuple. We also mention that although we do not prove it here, we found that for each $80 \leq k \leq 100$, a better bound than $\frac{217}{141}$ is obtained using the same tuple of item sizes as that for $42 \leq k \leq 45$.

Lemma 8. *For given k $(4 \leq k \leq 45)$, formulate $(\mathcal{P})$ using $(s_1, s_2, s_3) = (\frac{1}{2} + \varepsilon, \frac{1}{3} + \varepsilon, \frac{1}{7} + \varepsilon)$ if $k = 4$ or 5, $(s_1, s_2, s_3, s_4) = (\frac{1}{2} + \varepsilon, \frac{1}{3} + \varepsilon, \frac{1}{7} + \varepsilon, \frac{1}{43} + \varepsilon)$ if $6 \leq k \leq 41$, and $(s_1, s_2, s_3, s_4, s_5) = (\frac{1}{2} + \varepsilon, \frac{1}{3} + \varepsilon, \frac{1}{7} + \varepsilon, \frac{1}{49} + \varepsilon, \frac{1}{295} + \varepsilon)$ if $42 \leq k \leq 45$. Set ε to be a positive number such that $\sum s_i \leq 1$. Then, the optimal value of $(\mathcal{P})$ is at least the value of "lower bound" in Table 2.*

Proof. Consider the dual program $(\mathcal{D})$ of $(\mathcal{P})$:

$$\begin{aligned} (\mathcal{D}) \quad & \text{maximize } \boldsymbol{b}^T \boldsymbol{y} \\ & \text{subject to } A^T \boldsymbol{y} \leq \boldsymbol{c} \\ & \boldsymbol{y} \in \mathbb{R}^{2l} \end{aligned}$$

Pick up the "solution" corresponding to k from Table 3 and denote it by $\overline{\boldsymbol{y}}$. Since $A^T \overline{\boldsymbol{y}} \leq \boldsymbol{c}$, $\overline{\boldsymbol{y}}$ is feasible for $(\mathcal{D})$. The value of $\boldsymbol{b}^T \overline{\boldsymbol{y}}$ is confirmed to be equal to the value of "lower bound" for k in Table 2.

By the fact that for all $\boldsymbol{x} \geq \boldsymbol{0}$ with $A\boldsymbol{x} = \boldsymbol{b}$, $\boldsymbol{c}^T \boldsymbol{x} \geq (A^T \overline{\boldsymbol{y}})^T \boldsymbol{x} = \overline{\boldsymbol{y}}^T A \boldsymbol{x} = \overline{\boldsymbol{y}}^T \boldsymbol{b} = \boldsymbol{b}^T \overline{\boldsymbol{y}}$ holds true, which is known as the *weak duality theorem*, we have the optimal value of $(\mathcal{P})$ is at least the value of "lower bound" for k in Table 2. $\square$

Table 3: Solution to the dual program $(\mathcal{D})$ formulated for each $4 \le k \le 45$.

k	solution	k	solution
4	$(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{2}, -\tfrac{1}{2}, -\tfrac{1}{4}, -\tfrac{1}{4})^T$	25	$(\tfrac{150}{253}, \tfrac{150}{253}, \tfrac{75}{253}, \tfrac{25}{506}, -\tfrac{150}{253}, -\tfrac{75}{253}, -\tfrac{25}{253}, -\tfrac{3}{253})^T$
5	$(\tfrac{10}{17}, \tfrac{10}{17}, \tfrac{5}{17}, -\tfrac{10}{17}, -\tfrac{5}{17}, -\tfrac{2}{17})^T$	26	$(\tfrac{156}{263}, \tfrac{156}{263}, \tfrac{78}{263}, \tfrac{13}{263}, -\tfrac{156}{263}, -\tfrac{78}{263}, -\tfrac{26}{263}, -\tfrac{3}{263})^T$
6	$(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4}, -\tfrac{1}{2}, -\tfrac{1}{4}, -\tfrac{1}{6}, -\tfrac{1}{12})^T$	27	$(\tfrac{54}{91}, \tfrac{54}{91}, \tfrac{27}{91}, \tfrac{9}{182}, -\tfrac{54}{91}, -\tfrac{27}{91}, -\tfrac{9}{91}, -\tfrac{1}{91})^T$
7	$(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4}, -\tfrac{1}{2}, -\tfrac{1}{4}, -\tfrac{1}{7}, -\tfrac{3}{28})^T$	28	$(\tfrac{168}{283}, \tfrac{168}{283}, \tfrac{84}{283}, \tfrac{14}{283}, -\tfrac{168}{283}, -\tfrac{84}{283}, -\tfrac{28}{283}, -\tfrac{3}{283})^T$
8	$(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4}, -\tfrac{1}{2}, -\tfrac{1}{4}, -\tfrac{1}{8}, -\tfrac{1}{8})^T$	29	$(\tfrac{174}{293}, \tfrac{174}{293}, \tfrac{87}{293}, \tfrac{29}{586}, -\tfrac{174}{293}, -\tfrac{87}{293}, -\tfrac{29}{293}, -\tfrac{3}{293})^T$
9	$(\tfrac{1}{2}, \tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{4}, -\tfrac{1}{2}, -\tfrac{1}{4}, -\tfrac{1}{9}, -\tfrac{5}{36})^T$	30	$(\tfrac{60}{101}, \tfrac{60}{101}, \tfrac{30}{101}, \tfrac{5}{101}, -\tfrac{60}{101}, -\tfrac{30}{101}, -\tfrac{10}{101}, -\tfrac{1}{101})^T$
10	$(\tfrac{30}{53}, \tfrac{30}{53}, \tfrac{15}{53}, \tfrac{5}{53}, -\tfrac{30}{53}, -\tfrac{15}{53}, -\tfrac{6}{53}, -\tfrac{2}{53})^T$	31	$(\tfrac{186}{313}, \tfrac{186}{313}, \tfrac{93}{313}, \tfrac{31}{626}, -\tfrac{186}{313}, -\tfrac{93}{313}, -\tfrac{31}{313}, -\tfrac{3}{313})^T$
11	$(\tfrac{33}{58}, \tfrac{33}{58}, \tfrac{33}{116}, \tfrac{11}{116}, -\tfrac{33}{58}, -\tfrac{33}{116}, -\tfrac{3}{29}, -\tfrac{5}{116})^T$	32	$(\tfrac{192}{323}, \tfrac{192}{323}, \tfrac{96}{323}, \tfrac{16}{323}, -\tfrac{192}{323}, -\tfrac{96}{323}, -\tfrac{32}{323}, -\tfrac{3}{323})^T$
12	$(\tfrac{24}{43}, \tfrac{24}{43}, \tfrac{12}{43}, \tfrac{6}{43}, -\tfrac{24}{43}, -\tfrac{12}{43}, -\tfrac{4}{43}, -\tfrac{3}{43})^T$	33	$(\tfrac{22}{37}, \tfrac{22}{37}, \tfrac{11}{37}, \tfrac{11}{222}, -\tfrac{22}{37}, -\tfrac{11}{37}, -\tfrac{11}{111}, -\tfrac{1}{111})^T$
13	$(\tfrac{39}{68}, \tfrac{39}{68}, \tfrac{39}{136}, \tfrac{13}{136}, -\tfrac{39}{68}, -\tfrac{39}{136}, -\tfrac{13}{136}, -\tfrac{3}{68})^T$	34	$(\tfrac{204}{343}, \tfrac{204}{343}, \tfrac{102}{343}, \tfrac{17}{343}, -\tfrac{204}{343}, -\tfrac{102}{343}, -\tfrac{34}{343}, -\tfrac{3}{343})^T$
14	$(\tfrac{168}{289}, \tfrac{168}{289}, \tfrac{84}{289}, \tfrac{21}{289}, -\tfrac{168}{289}, -\tfrac{84}{289}, -\tfrac{28}{289}, -\tfrac{9}{289})^T$	35	$(\tfrac{210}{353}, \tfrac{210}{353}, \tfrac{105}{353}, \tfrac{35}{706}, -\tfrac{210}{353}, -\tfrac{105}{353}, -\tfrac{35}{353}, -\tfrac{3}{353})^T$
15	$(\tfrac{60}{103}, \tfrac{60}{103}, \tfrac{30}{103}, \tfrac{15}{206}, -\tfrac{60}{103}, -\tfrac{30}{103}, -\tfrac{10}{103}, -\tfrac{3}{103})^T$	36	$(\tfrac{72}{121}, \tfrac{72}{121}, \tfrac{36}{121}, \tfrac{6}{121}, -\tfrac{72}{121}, -\tfrac{36}{121}, -\tfrac{12}{121}, -\tfrac{1}{121})^T$
16	$(\tfrac{240}{409}, \tfrac{240}{409}, \tfrac{120}{409}, \tfrac{24}{409}, -\tfrac{240}{409}, -\tfrac{120}{409}, -\tfrac{40}{409}, -\tfrac{9}{409})^T$	37	$(\tfrac{222}{373}, \tfrac{222}{373}, \tfrac{111}{373}, \tfrac{37}{746}, -\tfrac{222}{373}, -\tfrac{111}{373}, -\tfrac{37}{373}, -\tfrac{3}{373})^T$
17	$(\tfrac{102}{173}, \tfrac{102}{173}, \tfrac{51}{173}, \tfrac{17}{346}, -\tfrac{102}{173}, -\tfrac{51}{173}, -\tfrac{17}{173}, -\tfrac{3}{173})^T$	38	$(\tfrac{228}{383}, \tfrac{228}{383}, \tfrac{114}{383}, \tfrac{19}{383}, -\tfrac{228}{383}, -\tfrac{114}{383}, -\tfrac{38}{383}, -\tfrac{3}{383})^T$
18	$(\tfrac{36}{61}, \tfrac{36}{61}, \tfrac{18}{61}, \tfrac{3}{61}, -\tfrac{36}{61}, -\tfrac{18}{61}, -\tfrac{6}{61}, -\tfrac{1}{61})^T$	39	$(\tfrac{78}{131}, \tfrac{78}{131}, \tfrac{39}{131}, \tfrac{13}{262}, -\tfrac{78}{131}, -\tfrac{39}{131}, -\tfrac{13}{131}, -\tfrac{1}{131})^T$
19	$(\tfrac{57}{98}, \tfrac{57}{98}, \tfrac{57}{196}, \tfrac{57}{784}, -\tfrac{57}{98}, -\tfrac{57}{196}, -\tfrac{19}{196}, -\tfrac{3}{98})^T$	40	$(\tfrac{240}{403}, \tfrac{240}{403}, \tfrac{120}{403}, \tfrac{20}{403}, -\tfrac{240}{403}, -\tfrac{120}{403}, -\tfrac{40}{403}, -\tfrac{3}{403})^T$
20	$(\tfrac{60}{103}, \tfrac{60}{103}, \tfrac{30}{103}, \tfrac{15}{206}, -\tfrac{60}{103}, -\tfrac{30}{103}, -\tfrac{10}{103}, -\tfrac{3}{103})^T$	41	$(\tfrac{246}{413}, \tfrac{246}{413}, \tfrac{123}{413}, \tfrac{41}{826}, -\tfrac{246}{413}, -\tfrac{123}{413}, -\tfrac{41}{413}, -\tfrac{3}{413})^T$
21	$(\tfrac{10}{17}, \tfrac{10}{17}, \tfrac{5}{17}, \tfrac{1}{17}, -\tfrac{10}{17}, -\tfrac{5}{17}, -\tfrac{5}{51}, -\tfrac{1}{51})^T$	42	$(\tfrac{196}{331}, \tfrac{196}{331}, \tfrac{98}{331}, \tfrac{14}{331}, \tfrac{7}{993}, -\tfrac{196}{331}, -\tfrac{98}{331}, -\tfrac{98}{993}, -\tfrac{4}{331}, -\tfrac{1}{993})^T$
22	$(\tfrac{132}{223}, \tfrac{132}{223}, \tfrac{66}{223}, \tfrac{11}{223}, -\tfrac{132}{223}, -\tfrac{66}{223}, -\tfrac{22}{223}, -\tfrac{3}{223})^T$	43	$(\tfrac{1806}{3049}, \tfrac{1806}{3049}, \tfrac{903}{3049}, \tfrac{129}{3049}, \tfrac{43}{6098}, -\tfrac{1806}{3049}, -\tfrac{903}{3049}, -\tfrac{301}{3049}, -\tfrac{36}{3049}, -\tfrac{3}{3049})^T$
23	$(\tfrac{138}{233}, \tfrac{138}{233}, \tfrac{69}{233}, \tfrac{23}{466}, -\tfrac{138}{233}, -\tfrac{69}{233}, -\tfrac{23}{233}, -\tfrac{3}{233})^T$	44	$(\tfrac{1848}{3119}, \tfrac{1848}{3119}, \tfrac{924}{3119}, \tfrac{132}{3119}, \tfrac{22}{3119}, -\tfrac{1848}{3119}, -\tfrac{924}{3119}, -\tfrac{308}{3119}, -\tfrac{36}{3119}, -\tfrac{3}{3119})^T$
24	$(\tfrac{16}{27}, \tfrac{16}{27}, \tfrac{8}{27}, \tfrac{4}{81}, -\tfrac{16}{27}, -\tfrac{8}{27}, -\tfrac{8}{81}, -\tfrac{1}{81})^T$	45	$(\tfrac{630}{1063}, \tfrac{630}{1063}, \tfrac{315}{1063}, \tfrac{45}{1063}, \tfrac{15}{2126}, -\tfrac{630}{1063}, -\tfrac{315}{1063}, -\tfrac{105}{1063}, -\tfrac{12}{1063}, -\tfrac{1}{1063})^T$

We below demonstrate the case $k = 4$. The tuple of $(s_1, s_2, s_3) = (\frac{1}{2} + \varepsilon, \frac{1}{3} + \varepsilon, \frac{1}{7} + \varepsilon)$ is used as item sizes. Applying $(t_{i,j})$ in (9) to (14), we have

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 4 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & -1 \\ 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 0 & 0 & 1 & -1 \end{pmatrix}.$$

From (15) we also have

$$\begin{aligned} \boldsymbol{b} &= (1, 1, 1, 0, 0, 0)^T \\ \boldsymbol{c} &= (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)^T. \end{aligned}$$

We find $\overline{\boldsymbol{y}} = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{4}, -\frac{1}{4})^T$ in Table 3. This is feasible for $(\mathcal{D})$, since

$$A^T\overline{\boldsymbol{y}} = (0, -\frac{1}{2}, 0, 0, -\frac{3}{2}, -1, -1, -\frac{1}{2}, -\frac{1}{2}, -1, -\frac{1}{2}, -\frac{1}{2}, 0, -\frac{1}{2}, 0, 0, 0, -\frac{1}{2}, -\frac{1}{4}, -\frac{1}{4}, 1)^T \le \boldsymbol{c}.$$

Also, $\boldsymbol{b}^T\overline{\boldsymbol{y}} = \frac{3}{2}$ is indeed equal to the value of "lower bound" for $k = 4$ in Table 2. The weak duality theorem states that the optimal value of $(\mathcal{P})$ is at least $\frac{3}{2}$.

Remark. For each k, the "solution" in Table 3 is in fact an optimal solution to $(\mathcal{D})$. To prove Lemma 8, it is sufficient just to present a feasible solution to $(\mathcal{D})$. We therefore omit the proof of optimality.

4 Concluding Remarks

While we have narrowed the gap between the upper and lower bounds for the case $k = 2$, the gaps for $k \geq 3$ still remain large. As we discussed in Section 3, there should be room for improvement in our method. It is interesting to combine the method with a trick of making slight differences in item size, such as subroutines in Section 2.

References

[BBG12] J. Balogh, J. Békési, and G. Galambos. New lower bounds for certain classes of bin packing algorithms. *Theor. Comput. Sci.*, 440-441:1–13, 2012.

[BCKK04] L. Babel, B. Chen, H. Kellerer, and V. Kotov. Algorithms for on-line bin-packing problems with cardinality constraints. *Discrete Applied Mathematics*, 143(1-3):238–251, 2004.

[BE98] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.

[CKP03] A. Caprara, H. Kellerer, and U. Pferschy. Approximation schemes for ordered vector packing problems. *Naval Research Logistics*, 50(1):58–69, 2003.

[EL10] L. Epstein and A. Levin. AFPTAS results for common variants of bin packing: A new method for handling the small items. *SIAM Journal on Optimization*, 20(6):3121–3145, 2010.

[Eps06] L. Epstein. Online bin packing with cardinality constraints. *SIAM J. Discrete Math.*, 20(4):1015–1030, 2006.

[KP99] H. Kellerer and U. Pferschy. Cardinality constrained bin-packing problems. *Annals of Operations Research*, 92(0):335–348, 1999.

[KSS75] K. L. Krause, V. Y. Shen, and H. D. Schwetman. Analysis of several task-scheduling algorithms for a model of multiprogramming computer systems. *J. ACM*, 22(4):522–550, 1975.

[KSS77] K. L. Krause, V. Y. Shen, and H. D. Schwetman. Errata: "Analysis of several task-scheduling algorithms for a model of multiprogramming computer systems". *J. ACM*, 24(3):527, 1977.

[RBLL89] P. V. Ramanan, D. J. Brown, C. C. Lee, and D. T. Lee. On-line bin packing in linear time. *J. Algorithms*, 10(3):305–326, 1989.

[Sei02] S. S. Seiden. On the online bin packing problem. *J. ACM*, 49(5):640–671, 2002.

[ST85] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985.

[vV92] A. van Vliet. An improved lower bound for on-line bin packing algorithms. *Inf. Process. Lett.*, 43(5):277–284, 1992.

[Yao80] A. C. Yao. New algorithms for bin packing. *J. ACM*, 27(2):207–227, 1980.