Online Bin Packing with Cardinality Constraints *

Leah Epstein†

Abstract

We consider a one dimensional storage system where each container can store a bounded amount of capacity as well as a bounded number of items $k \geq 2$. This defines the (standard) bin packing problem with cardinality constraints which is an important version of bin packing, introduced by Krause, Shen and Schwetman already in 1975. Following previous work on the unbounded space online problem, we establish the *exact* best competitive ratio for bounded space online algorithms for every value of k. This competitive ratio is a strictly increasing function of k which tends to $\Pi_\infty + 1 \approx 2.69103$ for large k. Lee and Lee showed in 1985 that the best possible competitive ratio for online bounded space algorithms for the classical bin packing problem is the sum of a series, and tends to Π_∞ as the allowed space (number of open bins) tends to infinity. We further design optimal online bounded space algorithms for *variable sized bin packing*, where each allowed bin size may have a distinct cardinality constraint, and for the *resource augmentation* model. All algorithms achieve the *exact* best possible competitive ratio possible for the given problem, and use constant numbers of open bins. Finally, we introduce unbounded space online algorithms with smaller competitive ratios than the previously known best algorithms for small values of k, for the standard cardinality constrained problem. These are the first algorithms with competitive ratio below 2 for $k = 4, 5, 6$.

1 Introduction

The classical bin packing problem [19, 5, 3] assumes no limit on the *number* of items which may be packed in a single bin. In practice, many applications require such a bound either due to overheads or additional constrains that are not modeled. For example, a disk cannot keep more than a certain number of files, even if these files are indeed very small. A processor cannot run more than a given number of tasks during a given time, even if all tasks are very short. The problem where there is a given bound $k > 1$ on the number of items which can co-exist in one bin, is called "Bin Packing with Cardinality Constraints" [11, 1]. We consider several versions of this problem.

We first define the classic online bin packing problem. In this problem, we receive a sequence σ of *items* $p_1, p_2 \ldots p_n$, arriving one by one. The values p_i are the *sizes* of the items. We have an infinite supply of *bins*, each of which is of unit size. An item must be assigned to a bin upon arrival, so that the sum of items in no bin exceeds 1. A bin is *empty* if no item is assigned to it, otherwise it is *used*. The goal is to minimize the number of bins used. In the **cardinality constrained bin packing**

*A preliminary version of this paper appears in Proceedings of the 13th European Symposium on Algorithms, 2005.

†Department of Mathematics, University of Haifa, 31905 Haifa, Israel. lea@math.haifa.ac.il. Research supported by Israel Science Foundation (grant no. 250/01).

problem, an additional constraint is introduced. A parameter k bounds the number of items that can be assigned to a single bin.

The standard measure of algorithm quality for online bin packing is the *asymptotic competitive ratio*, which we now define. For a given input sequence σ, let $\mathcal{A}(\sigma)$ (or $\mathcal{A}$) be the number of bins used by algorithm $\mathcal{A}$ on σ. Let $OPT(\sigma)$ (or OPT) be the cost of an optimal offline algorithm which knows the complete sequence of items in advance, i.e., the minimum possible number of bins used to pack items in σ. The *asymptotic performance ratio* for an algorithm $\mathcal{A}$ is defined to be

$$\mathcal{R}(\mathcal{A}) = \limsup_{n\to\infty} \sup_{\sigma} \left\{ \frac{\mathcal{A}(\sigma)}{OPT(\sigma)} | OPT(\sigma) = n \right\} .$$

In the *resource augmented* bin packing problem, the online algorithm is supplied with larger bins at its disposal than those of the offline algorithm that it is compared to. The competitive ratio then becomes a function of the bin size. All online bins are of the same size, and all the offline bins are of the same size, but these two sizes are not necessarily the same.

In the *variable-sized* bin packing problem, there is a supply of several bin sizes that can be used to pack the items. The cost of an algorithm is the sum of sizes of used bins. In this problem, the generalization into cardinality constrained packing assumes that each bin size $s_i \leq 1$ is associated a parameter k_i which bounds the number of items that can be packed into such a bin.

We stress the fact that items arrive *online*, this means that each item must be assigned in turn, without knowledge of the next items. We consider *bounded space* algorithms, which have the property that they only have a constant number of bins available to accept items at any point during processing, these bins are also called "open bins". The bounded space assumption is a quite natural one. Essentially the bounded space restriction guarantees that output of packed bins is steady, and that the packer does not accumulate an enormous backlog of bins which are only output at the end of processing.

Previous results. Cardinality constrained bin packing was studied in the offline environment already in 1975 by Krause, Shen and Schwetman [12, 13]. They showed that the performance guarantee of the well known First Fit algorithm is at most $2.7 - \frac{12}{5k}$. Additional results were offline approximation algorithms of performance guarantee 2. These results were later improved in two ways. Kellerer and Pferschy [11] designed an improved offline approximation algorithm with performance guarantee 1.5 and finally a PTAS was designed in [2] (for a more general problem). On the other hand, Babel et al. [1], designed a simple *online* algorithm with competitive ratio 2 for any value of k. They also designed improved algorithms for $k = 2, 3$ of competitive ratios $1 + \frac{\sqrt{5}}{5} \approx 1.44721$ and 1.8 respectively. The same paper [1] also proved an almost matching lower bound of $\sqrt{2} \approx 1.41421$ for $k = 2$ and mentioned that the lower bounds of [22, 20] for the classic problem hold for cardinality constrained bin packing as well. The lower bound of 1.5 given by Yao [22] holds for small values of $k > 2$ and the lower bound of 1.5401 given by Van Vliet [20] holds for sufficiently large k. No other lower bounds are known.

For the classic bin packing problem, Lee and Lee [14] presented an algorithm called HARMONIC, which partitions items into $m > 1$ classes and uses bounded space of at most $m-1$ open bins. For any $\varepsilon > 0$, there is a number m such that the HARMONIC algorithm that uses m classes has a performance ratio of at most $(1+\varepsilon)\Pi_\infty$ [14], where $\Pi_\infty \approx 1.69103$ is the sum of series (see Section 2). They also showed there is no bounded space algorithm with a performance ratio below Π_∞. Currently the best known unbounded space upper bound is 1.58889 due to Seiden [17].

The first to investigate the variable sized bin packing problem were Friesen and Langston [10]. Csirik [4] proposed the VARIABLE HARMONIC algorithm and showed that it has performance ratio at most Π_∞. Seiden [16] showed that this algorithm is optimal among bounded space algorithms. Unbounded space variable sized bin packing was studied also in [18].

The resource augmented bin packing problem was studied by Csirik and Woeginger [6]. They showed that the optimal bounded space asymptotic performance ratio is a function $\rho(b)$ of the online bin size b. Unbounded space resource augmented bin packing was studied also in [8].

Our results. We consider bounded space algorithms. For every value of k, we find the best competitive ratio of any online bounded space algorithm. The competitive ratio is a strictly increasing function of k and for large enough k it approaches $1 + \Pi_\infty \approx 2.69103$ where Π_∞ is the best competitive ratio shown by [14] for the classic bounded space problem. This is a surprising feature of the problem, since one would expect this value to simply tend to Π_∞ as k grows.

We further consider the resource augmented problem where the online algorithm may use larger bins compared to the optimal offline algorithm. We design optimal online algorithms for this problem as well. For large enough values of k, the competitive ratios again approach values which differ by 1 from the best competitive ratios for the classic resource augmented problem [6]. We show that the competitive ratios for our problem never drops below 1 (unlike the case studied in [6]) and identify the cases where the competitive ratio is exactly 1.

For the variable sized bin packing problem, we design algorithms of the exact optimal competitive ratios (among bounded space algorithms) for any set of bins and cardinality constraints. An interesting feature is that we prove the algorithms have optimal competitive ratios, even though we do not know what these ratios are.

A main difference between our results for bounded space algorithms and the results of [14, 6, 16] is that our algorithms have exactly the best possible competitive ratio achievable by bounded space online algorithms. The algorithms for variants of the classical problem have competitive ratios which tend to the best competitive ratio as the number of open bins grows without bound. Our algorithms just need a constant number of open bins to achieve the best competitive ratios. Therefore we need to be very careful in the analysis since unlike the classic problem, we may not lose any small constants, which depend on the number of open bins, in the analysis.

For small values of k we design several new unbounded space algorithms, based on combination of large and small items together in bins (see [14, 15, 17]), according to sizes of small items. We prove the competitive ratios of our algorithms for $k = 3, 4, 5, 6$ are $\frac{7}{4} = 1.75, \frac{71}{38} \approx 1.86842, \frac{771}{398} \approx 1.93719, \frac{287}{144} \approx 1.99306$ (respectively). This improves on the bounds $\frac{9}{5} = 1.8$ ($k = 3$) and 2 ($k = 4, 5, 6$) of [1].

2 Optimal Algorithms for Bounded Space Packing

In this section we define bounded space algorithms of optimal competitive ratio for each value of $k > 1$. For every $k > 1$, we define an online bounded space algorithm which packs at most k items in each bin and uses at most $k - 1$ open bins. We show that this algorithm is the best possible among bounded space algorithms. We use the well known sequence π_i, $i \geq 1$ which is often used for bin

packing, let $\pi_1 = 2, \pi_{i+1} = \pi_i(\pi_i - 1) + 1$ and let $\Pi_\infty = \sum_{i=1}^{\infty} \frac{1}{\pi_i - 1} \approx 1.69103$.

This sequence was used by Lee and Lee in [14] and by Van Vliet [20]. Adaptations of this sequence were later used in several papers including [6, 18]. The sequence is constructed in a way that $1 - \sum_{i=1}^{j} \frac{1}{\pi_i} = \frac{1}{\pi_{j+1} - 1}$ (which can be easily shown by induction using the sequence definition). This means that each time the next value π_i is picked to be an integer, such that all items $\frac{1}{\pi_j}$ for $j \leq i$ can fit together in a bin leaving some empty space. Note that Π_∞ is a lower bound on the best competitive ratio for classical bounded space bin packing, and there exists a sequence of bounded space algorithms, with an increasing sequence of open bins whose competitive ratios tend to this value [14, 21]. The algorithms in this section are based on the algorithms in [14] with some differences in the construction and proof due to the cardinality constraint (which also increases the competitive ratio by 1 for large values of k). We also would like to achieve the best possible bound for every value of k separately, and not only in the limit.

Let $\mathcal{R}_k = \sum_{i=1}^{k} \max\left\{\frac{1}{\pi_i - 1}, \frac{1}{k}\right\}$. We show that for every value of k, the best competitive ratio is *exactly* $\mathcal{R}_k$. We start with some properties of $\mathcal{R}_k$ as a function of k.

Theorem 1 *The value of $\mathcal{R}_k$ is a strictly increasing function of k, such that $\frac{3}{2} \leq \mathcal{R}_k < \Pi_\infty + 1$, and $\lim_{k\to\infty} \mathcal{R}_k = \Pi_\infty + 1 \approx 2.69103$.*

Proof. We first find the value of $\mathcal{R}_2$. Since $\pi_1 = 2$ and $\pi_2 = 3$, we have $\mathcal{R}_2 = \frac{3}{2} = 1.5$. Note also that $\mathcal{R}_3 = \frac{11}{6} \approx 1.83333, \mathcal{R}_4 = 2, \mathcal{R}_5 = 2.1$ and $\mathcal{R}_6 = \frac{13}{6} \approx 2.16666$. Next we show the monotonicity of $\mathcal{R}_k$. For a given k, let $j_k = \min_{1\leq j\leq k}\{j | \frac{1}{k} \geq \frac{1}{\pi_j - 1}\}$. The value j_k exists for all k since $\pi_k - 1 \geq k$ for all k. Then we have $\mathcal{R}_k = \sum_{i=1}^{j_k - 1} \frac{1}{\pi_i - 1} + \sum_{i=j_k}^{k} \frac{1}{k} = \sum_{i=1}^{j_k - 1} \frac{1}{\pi_i - 1} + \frac{k - j_k + 1}{k}$. By definition of the values j_i, clearly $j_k \leq j_{k+1}$. Therefore $\mathcal{R}_{k+1} - \mathcal{R}_k = \sum_{j_k}^{j_{k+1} - 1} \frac{1}{\pi_i - 1} + \frac{k - j_{k+1} + 2}{k+1} - \frac{k - j_k + 1}{k} > \frac{j_{k+1} - j_k}{k+1} + \frac{1 - j_{k+1}}{k+1} - \frac{1 - j_k}{k} = \frac{j_k - 1}{k} - \frac{j_k - 1}{k+1} \geq 0$. We deduce the strict inequality above by $\pi_i - 1 < k + 1$ which holds for $i < j_{k+1}$.

An upper bound on $\mathcal{R}_k$ follows from $\mathcal{R}_k = \sum_{i=1}^{k} \max\left\{\frac{1}{\pi_i - 1}, \frac{1}{k}\right\} < \sum_{i=1}^{k} \frac{1}{\pi_i - 1} + \sum_{i=1}^{k} \frac{1}{k} < \Pi_\infty + 1$. We next show that $\mathcal{R}_k$ tends to this value. For a given $\varepsilon > 0$, let ℓ be a value such that $\sum_{i=1}^{\ell} \frac{1}{\pi_i - 1} \geq \Pi_\infty - \frac{\varepsilon}{2}$, and $\ell \geq \frac{2}{\varepsilon}$. Let $k = \ell^2$, then $\mathcal{R}_k \geq \sum_{i=1}^{\ell} \frac{1}{\pi_i - 1} + \sum_{i=\ell+1}^{k} \frac{1}{\ell^2} \geq \Pi_\infty - \frac{\varepsilon}{2} + \frac{\ell^2 - \ell}{\ell^2} \geq \Pi_\infty - \frac{\varepsilon}{2} + 1 - \frac{\varepsilon}{2} = \Pi_\infty + 1 - \varepsilon$ ■

Next we define the algorithm CARDINALITY CONSTRAINED HARMONIC$_k$ (CCH$_k$) which is an adaptation of the algorithm HARMONIC$_k$ defined originally by Lee and Lee [14]. The fundamental idea of "harmonic-based" algorithms is to first classify items by size, and then pack an item according to its class (as opposed to letting the exact size influence packing decisions).

For the classification of items, we partition the interval $(0, 1]$ into sub-intervals. We use $k - 1$ sub-intervals of the form $(\frac{1}{i+1}, \frac{1}{i}]$ for $i = 1, \ldots, k - 1$ and one final sub-interval $(0, \frac{1}{k}]$. Each bin will contain only items from one sub-interval (type). Items in sub-interval i are packed i to a bin for

$i = 1, \ldots, k-1$, thus keeping the cardinality constraint. The items in interval k are packed k to a bin. A bin which received the full amount of items (according to its type) is closed, therefore at most $k-1$ bins are open simultaneously (one per interval, except for $(\frac{1}{2}, 1]$).

To prove the upper bound on the competitive ratio, we use a simplified version of a theorem 9 stated in section 5. We use the technique of weighting functions. This technique was originally introduced for one-dimensional bin packing algorithms [19]. The version we use is as follows.

Theorem 2 *Consider a bin packing algorithm. Let w be a weight measure. Assume that for every output of the algorithm, the number of bins used by an algorithm ALG is bounded by $X(\sigma) + c$ for some constant c, where $X(\sigma)$ is the sum of weights of all items in the sequence according to weight measure w. Denote by W the maximum amount of weight that can be packed into a single bin of an offline algorithm according to measure w. Then the competitive ratio of the algorithm is upper bounded by W.*

We define weights as follows. The weight of item x is denoted $w(x)$. The weight of an item in interval $(\frac{1}{i+1}, \frac{1}{i}]$, for $i = 1, \ldots, k-1$, is $\frac{1}{i}$. The weight of an item in interval $(0, \frac{1}{k}]$ is $\frac{1}{k}$. Recall that except for $k-1$ open bins that may not receive the full amount of items, each output bin receives a total weight of 1. A closed bin for items in interval $(\frac{1}{i+1}, \frac{1}{i}]$ receives i items, of weight $\frac{1}{i}$ each. A closed bin for items in interval $(0, \frac{1}{k}]$ receives k items, of weight $\frac{1}{k}$ each. Therefore we get $CCH_k(\sigma) \leq X(\sigma) + k - 1$.

Theorem 3 *For every k, the competitive ratio of CCH_k is $\mathcal{R}_k$, and no online algorithm which uses bounded space can have a better competitive ratio.*

Proof. We prove the upper bound first. Let $\varepsilon > 0$ a very small constant, such that $\varepsilon << \frac{1}{k\pi_{k+1}}$ We claim that the maximum weight of a single bin is achieved for the following set of items. $f_1 \geq \ldots \geq f_k$, so that $f_i = \frac{1}{\pi_i} + \varepsilon$. This set of items fits in a single bin according to the definition of the sequence π_j. Their sum of weights is exactly $\mathcal{R}_k = \sum_{i=1}^{k} \max\left\{\frac{1}{\pi_i - 1}, \frac{1}{k}\right\}$.

To show that the maximum weight of any bin is indeed $\mathcal{R}_k$, consider an arbitrary set S of $\ell \leq k$ items which fits into one bin. If $\ell \neq k$, we add $k - \ell$ items of size zero and give them weight $\frac{1}{k}$. This may only increase the sum of weights. Let $g_1 \geq \ldots \geq g_k$ be the sorted list of items. If $g_i \in (\frac{1}{\pi_i}, \frac{1}{\pi_i - 1}]$ holds for all i such that $\pi_i \leq k$ and $g_i \in [0, \frac{1}{k}]$ for all i such that $\pi_i > k$, then the weight of the items of S is exactly $\mathcal{R}_k$. Otherwise let i be the first index of item that does not satisfy the above. If $w(g_i) = \frac{1}{k}$ we get that $\sum_{j=1}^{k} w(g_j) = \sum_{j=1}^{i-1} w(g_j) + \sum_{j=i}^{k} w(g_j) = \sum_{j=1}^{i-1} w(f_j) + \sum_{j=i}^{k} \frac{1}{k} \leq \sum_{j=1}^{k} w(f_j) = \mathcal{R}_k$.

Otherwise, assume $w(g_i) > \frac{1}{k}$. Due to the greedy construction of the sequence π_j, and since $g_i \notin (\frac{1}{\pi_i}, \frac{1}{\pi_i - 1}]$, we get that $g_i \leq \frac{1}{\pi_i}$ and therefore $w(g_i) < \frac{1}{\pi_i - 1} = w(f_i)$. Let i' be the smallest index such that $w(g_{i'}) = \frac{1}{k}$ (this value exists as mentioned above since $k \leq \pi_k - 1$). If $i' = i + 1$ we get that $w(g_i) < w(f_i)$, and for $j \geq i'$, $\frac{1}{k} = w(g_j) \leq w(f_j)$. In this case we have $\sum_{j=1}^{k} w(g_i) < \sum_{j=1}^{k} w(f_i) = \mathcal{R}_k$. Otherwise consider the values of j such that $i \leq j \leq i' - 1$. We have $g_j \leq \frac{1}{\pi_i}$, and therefore according to the weight definition for $x > \frac{1}{k}$, $\frac{w(g_j)}{g_j} \leq \frac{\pi_i + 1}{\pi_i}$ for $i \leq j \leq i' - 1$. Given

that for $j < i$, $g_j \in (\frac{1}{\pi_j}, \frac{1}{\pi_j - 1}]$, we have $\sum_{j=i}^{k} g_j \leq \frac{1}{\pi_i - 1}$ and therefore $\sum_{j=i}^{i'-1} w(g_j) \leq \frac{\pi_i + 1}{\pi_i^2 - \pi_i}$. However $w(f_i) + w(f_{i+1}) = \frac{1}{\pi_i - 1} + \frac{1}{\pi_{i+1} - 1} = \frac{1}{\pi_i - 1} + \frac{1}{\pi_i^2 - \pi_i} = \frac{\pi_i + 1}{\pi_i^2 - \pi_i}$. Summarizing, we get $\sum_{j=1}^{k} w(g_j) = \sum_{j=1}^{i-1} w(g_j) + \sum_{j=i}^{i'-1} w(g_j) + \sum_{j=i'}^{k} w(g_j) \leq \sum_{j=1}^{i-1} w(f_j) + w(f_i) + w(f_{i+1}) + \sum_{j=i'}^{k} \frac{1}{k} \leq \sum_{j=1}^{k} w(f_i) = \mathcal{R}_k$.

The proof of the lower bound is similar to previously known lower bound proofs for bounded space algorithms, see [14, 6]. To prove the lower bound, let N be a large constant, and $\delta > 0$ a very small constant, such that $\delta << \frac{1}{k\pi_{k+1}}$. We construct the following sequence. The sequence has k phases. Phase i contains N items of size $\frac{1}{\pi_i} + \delta$. Let K be the number of bins that may be open simultaneously. Except for at most K bins, all bins of each phase are closed after the phase. Such bins can be filled by a maximum amount of $\min\{\pi_i - 1, k\}$ items. Therefore phase i contributes at least $\frac{N}{\min\{\pi_i - 1, k\}} - K = N \max\{\frac{1}{\pi_i - 1}, \frac{1}{k}\} - K$ closed bins to the output. The optimal packing of the sequence contains N identically packed bins with one item of each phase per bin. We get that the competitive ratio is at least $\mathcal{R}_k - \frac{kK}{N}$. This approaches $\mathcal{R}_k$ for large enough N. ■

3 Extension to Resource Augmentation

Following the work of [6] which studied resource augmentation for the classic bin packing problem, we show that the algorithms defined in the previous section are optimal in a resource augmented environment as well.

We compare an online algorithm which uses bins of size 1 to an optimal offline algorithm whose bins are of size $\frac{1}{b}$. We assume that all item sizes are bounded by $\frac{1}{b}$. This problem definition is equivalent to the alternative definition where items have sizes in $(0, 1]$, the online algorithm uses bins of size b and the offline algorithm uses bins of size 1. The competitive ratio for bounded cardinality k is measured as a function of $b > 1$. The best competitive ratio for bounded space algorithms and unrestricted online algorithms are denoted $\mathcal{R}_k(b)$ and $r_k(b)$ (respectively). We note a fundamental difference between the resource augmented problem associated with the classic bin packing problem and the problem studied in this paper. As we show later in this section, the competitive ratio is never below 1 for our problem, whereas the classic problem has a competitive ratio below 1 for $b \geq 2$ [6, 8].

We show that the competitive ratio (even for unbounded space algorithms) cannot actually reach 1 if $b < k$ and is exactly 1 for $b = k$.

Theorem 4 *For all values of b, k such that $b < k$ we have $\mathcal{R}_k(b) \geq r_k(b) > 1$. For all values of b, k such that $b \geq k$, we have $r_k(b) = \mathcal{R}_k(b) = 1$.*

Proof. It is easy to see that the functions $r_k(b)$ and $\mathcal{R}_k(b)$ are monotonically decreasing in b, and $\mathcal{R}_k(b) \geq r_k(b)$. Therefore given $b < k$, we can prove the first part for $b' = \max\{b, k - \frac{1}{2}\} \geq b$. I.e., we prove that $r_k(b') > 1$ and therefore $r_k(b) > 1$. Let $x = \frac{b' + k}{2kb'} < \frac{1}{b'}$ and let $\varepsilon = \frac{1 - xb'}{b'(k-1)} > 0$. Let N be a large enough integer. The input sequence consists of a first phase with $Nk(k-1)$ items of size ε possibly followed by a second phase with Nk items of size x. Denote the optimal offline cost after the first phase by OPT_1 and after the second phase by OPT_2. We get that $OPT_1 = N(k-1)$, since $k\varepsilon < \frac{1}{b'}$ and $OPT_2 = Nk$, since $x + (k-1)\varepsilon = \frac{1}{b'}$. Let R be the competitive ratio of an online

algorithm $\mathcal{A}$. Let Y_i ($1 \leq i \leq k$) be the number of bins $\mathcal{A}$ has with exactly i items after the first phase. Note that $\sum_{i=1}^{k} iY_i = Nk(k-1)$. If the sequence stops here, we have $\sum_{i=1}^{k} Y_i \leq R \cdot OPT_1 = RN(k-1)$. If $\sum_{i=1}^{k} Y_i(k-i) > Nk$, we get $\sum_{i=1}^{k} kY_i > Nk(k-1) + Nk = Nk^2$, which gives $R > \frac{k}{k-1}$. Otherwise if the sequence continues, $\sum_{i=1}^{k} Y_i(k-i)$ is exactly the number of larger items can join the bins of the online algorithm. Since this number is at most Nk, the other items need to be packed into new bins. Note that $kx = \frac{b'+k}{2b'} > 1$. Therefore the best packing can be with $k-1$ items per bin. This results in a packing of size $\sum_{i=1}^{k} Y_i + \frac{Nk - \sum_{i=1}^{k} Y_i(k-i)}{k-1} \leq R \cdot OPT_2 = RNk$. We get $\sum_{i=1}^{k}(k-1)Y_i + Nk - \sum_{i=1}^{k} Y_i(k-i) = Nk + \sum_{i=1}^{k}(i-1)Y_i \leq RNk(k-1)$. Combining with $\sum_{i=1}^{k} Y_i \leq RN(k-1)$ we have $Nk^2 = Nk + Nk(k-1) = Nk + \sum_{i=1}^{k} iY_i \leq RN(k^2-1)$ or $R \geq \frac{k^2}{k^2-1} > 1$.

For the second part we simply use the algorithm Next-Fit. Since $\frac{1}{b} \leq \frac{1}{k}$, and all item sizes are at most $\frac{1}{b}$, each bin receives exactly k items. Given a sequence of f items, we get $\left\lceil \frac{f}{k} \right\rceil$ packed bins. However due to the cardinality constraint $OPT \geq \left\lceil \frac{f}{k} \right\rceil$ and therefore the competitive ratio is at most 1. Consider now a sequence of Nk items of size $\frac{1}{b}$. No algorithm can pack them into less than N bins (no matter how large b is). Since $OPT = N$ as well we get that $r_k(b) = \mathcal{R}_k(b) = 1$ for the case $b \geq k$. ■

The algorithms are defined exactly as in the previous section. However this means that some of the defined classes do not exist if b is large enough. Note that the algorithm for the case $b \geq k$ becomes exactly Next Fit as described in Theorem 4.

To define the competitive ratio, we first define sequences $\pi_i(b)$ and $\Pi_i(b)$, originally defined by [6] as follows. $\Pi_0(b) = 0$, $\pi_1(b) = \lfloor b \rfloor + 1$, $\Pi_1(b) = \frac{1}{\pi_1(b)}$, $\pi_i(b) = \left\lfloor \frac{1}{\frac{1}{b} - \Pi_{i-1}(b)} \right\rfloor + 1$ and $\Pi_i(b) = \Pi_{i-1}(b) + \frac{1}{\pi_i(b)}$. The intuition behind this function is to find a sequence of integers, such that the next integer at each point is picked greedily to be minimal, and the sum of their reciprocals is less than $\frac{1}{b}$. The values of $\pi_i(b)$ satisfy $\pi_i(b) > b$. We can show that the values are strictly increasing as a function of b. Clearly the values are non-decreasing. If two values are the same we let $\pi_i(b) = \pi_{i+1}(b) = f$ be these identical values. Then we argue that $\pi_i(b)$ should have been chosen to be at most $f-1$. To see that note that $\frac{1}{b} - \Pi_{i-1}(b) > \frac{2}{f} \geq \frac{1}{f-1}$. This holds for all $f \geq 2$.

Csirik and Woeginger [6] introduced the function $\rho(b) = \sum_{i=1}^{\infty} \frac{1}{\pi_i(b)-1}$ and showed that this is the best possible competitive ratio with resource augmentation b for the classic bin packing problem. Note that $\rho(1) = \Pi_\infty \approx 1.69103$. We can prove the following theorems.

Theorem 5 *For every k, the competitive ratio of CCH_k (defined in the previous section) is $\mathcal{R}_k(b) = \sum_{i=1}^{k} \max\left\{\frac{1}{\pi_i(b)-1}, \frac{1}{k}\right\}$, and no online algorithm which uses bounded space can have a better competitive ratio.*

Theorem 6 *The value of $\mathcal{R}_k(b)$ for a fixed value of b is an increasing function of k, such that $1 \leq \mathcal{R}_k(b) < \rho(b)+1$, and $\lim_{k\to\infty} \mathcal{R}_k = \rho(b)+1$.*

Proof. (of Theorem 5). We again use Theorem 2. The weights are defined as in the previous section. We prove the upper bound first. Let $\varepsilon > 0$ a very small constant, such that $\varepsilon << \frac{1}{k\pi_{k+1}(b)}$ We claim that the maximum weight of a single bin is achieved for the following set of items. $f_1 \geq \ldots \geq f_k$, so that $f_i = \frac{1}{\pi_i(b)} + \varepsilon$. This set of items fits in a single bin of size $\frac{1}{b}$ according to the definition of the sequence $\pi_j(b)$. Their sum of weights is exactly $\mathcal{R}_k(b) = \sum_{i=1}^{k} \max\left\{\frac{1}{\pi_i(b)-1}, \frac{1}{k}\right\}$.

To show the maximum weight of any bin is indeed $\mathcal{R}_k(b)$ consider an arbitrary set S of $\ell \leq k$ items which fits into one bin of size $\frac{1}{b}$. If $\ell \neq k$, we add $k-\ell$ items of size zero and give them weight $\frac{1}{k}$. This may only increase the sum of weights. Let $g_1 \geq \ldots \geq g_k$ be the sorted list of items. If $g_i \in (\frac{1}{\pi_i(b)}, \frac{1}{\pi_i(b)-1}]$ holds for all i such that $\pi_i(b) \leq k$ and $g_i \in [0, \frac{1}{k}]$ for all i such that $\pi_i(b) > k$, then the weight of the items of S is exactly $\mathcal{R}_k(b)$.

Otherwise let i be the first index of item that does not satisfy the above. If $w(g_i) = \frac{1}{k}$ we get that $\sum_{j=1}^{k} w(g_j) = \sum_{j=1}^{i-1} w(g_j) + \sum_{j=i}^{k} w(g_j) = \sum_{j=1}^{i-1} w(f_j) + \sum_{j=i}^{k} \frac{1}{k} \leq \sum_{j=1}^{k} w(f_j) = \mathcal{R}_k(b)$. Otherwise, assume $w(g_i) > \frac{1}{k}$. Due to the greedy construction of the sequence π_j, and since $g_i \notin (\frac{1}{\pi_i(b)}, \frac{1}{\pi_i(b)-1}]$ we get that $g_i \leq \frac{1}{\pi_i(b)}$ and therefore $w(g_i) < \frac{1}{\pi_i(b)-1} = w(f_i)$. Let i' be the smallest index such that $w(g'_i) = \frac{1}{k}$. If such an index does not exist we let $i' = k+1$. If $i' = i+1$ we get that $w(g_i) < w(f_i)$, and for $i' \leq j \leq k$, $\frac{1}{k} = w(g_j) \leq w(f_j)$. In this case we have $\sum_{j=1}^{k} w(g_i) < \sum_{j=1}^{k} w(f_i) = \mathcal{R}_k(b)$. Otherwise consider the values of j such that $i \leq j \leq i'-1$. We have $g_j \leq \frac{1}{\pi_i(b)}$, and therefore according to the weight definition for $x > \frac{1}{k}$, $\frac{w(g_j)}{g_j} \leq \frac{\pi_i(b)+1}{\pi_i(b)}$. Given that for $j < i$, $g_j \in (\frac{1}{\pi_j(b)}, \frac{1}{\pi_j(b)-1}]$, we have $\sum_{j=i}^{k} g_j \leq \frac{1}{b} - \Pi_{i-1}(b) \leq \frac{1}{\pi_i(b)-1}$ and $\sum_{j=i+1}^{k} g_j \leq \frac{1}{b} - \Pi_i(b) = \frac{1}{b} - \Pi_{i-1}(b) - \frac{1}{\pi_i(b)} \leq \frac{1}{\pi_{i+1}(b)-1}$. Using $g_j \leq \frac{1}{\pi_i(b)}$ again we get $\sum_{j=i}^{k} g_j \leq \frac{1}{\pi_{i+1}(b)-1} + \frac{1}{\pi_i(b)}$. This gives $\sum_{j=i}^{i'-1} w(g_j) \leq \left(\frac{\pi_i(b)+1}{\pi_i(b)}\right)\left(\sum_{j=i+1}^{k} g_j\right) \leq \frac{1}{\pi_i(b)(\pi_i(b)-1)} + \frac{1}{\pi_i(b)} + \frac{1}{\pi_{i+1}(b)-1} = \frac{1}{\pi_i(b)-1} + \frac{1}{\pi_{i+1}(b)-1}$. However we have $w(f_i) + w(f_{i+1}) = \frac{1}{\pi_i(b)-1} + \frac{1}{\pi_{i+1}(b)-1}$. Summarizing, we get $\sum_{j=1}^{k} w(g_j) = \sum_{j=1}^{i-1} w(g_j) + \sum_{j=i}^{i'-1} w(g_j) + \sum_{j=i'}^{k} w(g_j) \leq \sum_{j=1}^{i-1} w(f_j) + w(f_i) + w(f_{i+1}) + \sum_{j=i'}^{k} \frac{1}{k} \leq \sum_{j=1}^{k} w(f_i) = \mathcal{R}_k(b)$.

To prove the lower bound, let N be a large constant, and $\delta > 0$ a very small constant, such that $\delta << \frac{1}{k\pi_{k+1}(b)}$. We construct the following sequence. The sequence has k phases. Phase i contains N items of size $\frac{1}{\pi_i(b)} + \delta$. Let K be the number of bins that may be open simultaneously. Except for at most K bins, all bins of each phase are closed after the phase. Such bins can be filled by a maximum amount of $\min\{\pi_i(b)-1, k\}$ items. Therefore phase i contributes at least $\frac{N}{\min\{\pi_i(b)-1,k\}} - K = N\max\{\frac{1}{\pi_i-1}, \frac{1}{k}\} - K$ closed bins to the output. The optimal packing of the sequence contains N identically packed bins with one items of each phase per bin. We get that the competitive ratio is at least $\mathcal{R}_k(b) - \frac{kK}{N}$. This approaches $\mathcal{R}_k(b)$ for large enough N. ■

Proof. (of Theorem 6). As shown above the value of $\mathcal{R}_k(b)$ is at least 1. Next we show the monotonicity of $\mathcal{R}_k(b)$ for a fixed value of b as a function of k. If $k \leq b$ the value of the function is 1, therefore we need to prove monotonicity for $k > b$. Note that for every k and b, $\pi_k(b) > k$. This holds since if $\pi_k \leq k$ we get that $\sum_{t=1}^{k} \frac{1}{\pi_k(b)} \geq 1 > b$. We therefore need to consider the case $\pi_k(b) \geq k+1$, $\pi_{k+1}(b) \geq k+2$. For $k' = k, k+1$, let $j'_k = \min_{1 \leq j \leq k'} \{j | \frac{1}{k'} \geq \frac{1}{\pi_j(b)-1}\}$. We have $\mathcal{R}_k = \sum_{i=1}^{j_k-1} \frac{1}{\pi_i(b)-1} + \sum_{i=j_k}^{k} \frac{1}{k} = \sum_{i=1}^{j_k-1} \frac{1}{\pi_i(b)-1} + \frac{k-j_k+1}{k}$. By definition of the values j_i, clearly $j_k \leq j_{k+1}$. Therefore $\mathcal{R}_{k+1} - \mathcal{R}_k = \sum_{j_k}^{j_{k+1}-1} \frac{1}{\pi_i(b)-1} + \frac{k-j_{k+1}+2}{k+1} - \frac{k-j_k+1}{k} > \frac{j_{k+1}-j_k}{k+1} + \frac{1-j_{k+1}}{k+1} - \frac{1-j_k}{k} = \frac{j_k-1}{k} - \frac{j_k-1}{k+1} \geq 0$. The strict inequality above follows from $\pi_i(b) - 1 < k+1$ for $i < j_{k+1}$.

An upper bound on $\mathcal{R}_k$ follows from $\mathcal{R}_k = \sum_{i=1}^{k} \max\left\{\frac{1}{\pi_i(b)-1}, \frac{1}{k}\right\} \leq \sum_{i=1}^{k} \frac{1}{\pi_i(b)-1} + \sum_{i=1}^{k} \frac{1}{k} < \rho(b) + 1$. We next show that $\mathcal{R}_k$ tends to this value. For a given $\varepsilon > 0$, let ℓ be a value such that $\sum_{i=1}^{\ell} \frac{1}{\pi_i(b)-1} \geq \rho(b) - \frac{\varepsilon}{2}$, and $\ell \geq \frac{2}{\varepsilon}$. Let $k = \ell^2$, then $\mathcal{R}_k \geq \sum_{i=1}^{\ell} \frac{1}{\pi_i(b)-1} + \sum_{i=\ell+1}^{k} \frac{1}{\ell^2} \geq \rho(b) - \frac{\varepsilon}{2} + \frac{\ell^2-\ell}{\ell^2} \geq \rho(b) - \frac{\varepsilon}{2} + 1 - \frac{\varepsilon}{2} = \rho(b) + 1 - \varepsilon$ ■

4 Extension to Variable Sized Bins

Following the work of Seiden [16] we design optimal online bounded space algorithm for the case of variable sized bins. Similarly to that case and other work on variable sized bins [7], we design algorithms for any set of bin sizes, we prove their optimality, however we do not know their competitive ratios. Our algorithms are based on the VARIABLE HARMONIC algorithms of Csirik [4]. The optimality of these algorithms among the class of bounded space algorithms was proved in [16]. As in previous sections, the main difference between these algorithms and our algorithms is in the way that small items are packed. As in previous sections, our algorithms have the exact best possible competitive ratio for a given set of bins and cardinality constraints, this with a constant number of open bins that can be easily computed (as a function of the bins sizes and constraints). The algorithms for the classical problem get close to the best possible competitive ratio as the number of open bins grows without bound.

In order to define our general algorithm CARDINALITY CONSTRAINED VARIABLE HARMONIC (CCVH) we use some definitions. Let the bins sizes be $s_1 < \ldots s_m = 1$. Let their cardinality constraints be $k_1, \ldots, k_m$ (respectively). We define a set of critical sizes for each bin in the following way. Let $T_i = \{\frac{s_i}{j} | 1 \leq j \leq k_i\}$ and $T = \bigcup_{1 \leq i \leq m} T_i$. Let $|T| = M$ and the members of T be $1 = t_1 > t_2 > \ldots > t_M$. The type of a size t_r is defined to a value $i(r)$ such that $t_r \in T_{i(r)}$ (ties are broken arbitrarily). In this case the order of t_r is $\ell(r) \leq k_i$ such that $t_r = \frac{s_{i(r)}}{\ell(r)}$.

We again classify items into intervals whose right endpoint is a critical size. This associates an item with an type and order. Afterwards we pack an item according to its type and order (here as well as in the previous sections, the exact size does not influence packing decisions). Each bin will contain items of a single interval.

Since $M = |T| \leq \sum_{i=1}^{m} k_i$, there is a bounded number of pairs of type and order. For the classifi-

cation of items, we partition the interval $(0, 1]$ into sub-intervals. The "small" interval is $(0, t_M]$. The other intervals are $(t_{j+1}, t_j]$ for $j = 1, \ldots, M-1$. Each bin will contain only items from one pair of type and order. Items in the sub-interval whose right endpoint is t_r are packed into bins of size $s_{i(r)}$. The items in this interval are packed $\ell(r)$ to a bin, thus keeping the cardinality constraints. Note that at most $M - m$ bins are open simultaneously, since a bin which received the full amount of items (according to its type) is closed.

The differences with algorithms for the classic variable sized bin packing problem are as follows. The condition for an item to be "small" (i.e. in the "small" interval) is determined by the cardinality constraints. Items cannot be packed using Next Fit due to these constraints. Moreover, in [16] the smallest items are packed into bins of size 1. In that case it is actually possible to pack the small items into any type of bin. Here the type of bin for the small items must be $s_{i(M)}$ (if there exists another size i' such that $t_M \in T'_i$, that size can be used for the small items as well).

The following theorem is used in [16] to prove upper bounds on the competitive ratio of algorithms for variable sized bins.

Theorem 7 *Consider a bin packing algorithm. Let w be a weight measure. Assume that for every output of the algorithm, the cost of all the bins used by the algorithm ALG is bounded by $X(\sigma) + c$ for some constant c, where $X(\sigma)$ is the sum of weights of all items in the sequence according to weight measure w. Denote by W_i the maximum amount of weight that can be packed into a single bin of size s_i of an offline algorithm according to measure w. Then the competitive ratio of the algorithm is upper bounded by $\max_{1 \le i \le m} \left\{ \frac{W_i}{s_i} \right\}$.*

We assign weights to items in the following way. A weight of an item x is again denoted by $w(x)$. An item of interval $(0, t_M]$ receives weight $\frac{s_{i(M)}}{\ell(M)}$ (note that $\ell(M) = k_{i(M)}$). An item of interval $(t_{j+1}, t_j]$ receives weight $\frac{s_{i(j)}}{\ell(j)}$. Each closed bin of interval $(0, t_M]$ is of size $s_{i(M)}$, it receives $\ell(M)$ items and thus the weight of items packed in it is equal to its size. Each closed bin of interval $(t_{j+1}, t_j]$ is of size $s_{i(j)}$. It receives $\ell(j)$ items and thus the weight of items packed in it is equal to its size. Therefore the cost of the algorithm differs from the total weight of all items by the cost of all open bins, which is clearly bounded by $M - m$.

We can now use Theorem 7 to prove the following theorem.

Theorem 8 *For a given set of bins sizes and cardinality constraints, the algorithm CVH is an optimal online algorithm.*

Proof. Let $s = s_i$ be the bin size which maximizes the expression $\max_{1 \le i \le m} \left\{ \frac{W_i}{s_i} \right\}$. Let $k = k_i$ be the cardinality constraint of this bin size. We allow the bin to contain items of size 0 and we give them the weight $\frac{s_{i(M)}}{\ell(M)}$ as the weight of other very small items. Assume therefore that a bin which contains a maximum amount of weight has exactly k items. Let $b_1, \ldots, b_k$ be their sizes. Let N be a large enough integer. Consider an offline packing with N bins of size s identically packed with items $b_1, \ldots, b_k$. The cost of this algorithm is Ns.

We show that any bounded space online algorithm is forced to have competitive ratio of at least $\frac{\sum_{y=1}^{k} w(b_y)}{s}$. The input sequence is sorted so that it consists of k phases. Phase y has N identical items

of size b_y. Let K be the number of bins that can be open simultaneously. For each bin size s_a, we compute the maximum number of items of size b_y that can be packed in a closed bin of size s_a. This number is $Q(y,a) = \min\{k_a, \lfloor \frac{s_a}{b_y} \rfloor\}$. Let $t_{j(y)}$ be the upper bound of the interval for b_y. According to the above weight definitions, $w(b_y) = t_{j(y)}$. For $1 \leq a \leq m$, such that $s_a \geq b_y$, let $x(y,a)$ be the smallest integer such that $b_y \leq t_{x(y,a)}$, and $i(x(y,a)) = a$.

We charge an item of size b_y, which the online algorithm packs it in a bin of size s_a, with $\frac{s_a}{Q(y,a)}$. In this way the cost for all items packed in closed bins is exactly the cost of the online algorithm for the closed bins. We claim that for pairs y, a for which $x(y,a)$ is defined, $t_{x(y,a)} = \frac{s_a}{Q(y,a)}$ and $x(y,a) \leq j(y)$ hold. If $Q(y,a) = k_a$ then $\frac{s_a}{b_y} \geq k_a$. Therefore $\frac{s_a}{k_a} \geq b_y$ and $t_{x(y,a)} = \frac{s_a}{k_a}$. Otherwise $\frac{s_a}{b_y} - 1 < Q(y,a) \leq \frac{s_a}{b_y}$. Therefore $\frac{s_a}{Q(y,a)+1} < b_y$ and $\frac{s_a}{Q(y,a)} \geq b_y$. This is exactly the definition of $t_{x(y,a)}$. Since $j(y)$ is the largest index that satisfies $t_j(y) \geq b_y$, we get that $x(y,a) \leq j(y)$. We got that an item of size b_y in a bin of size b_a is charged with $\frac{s_a}{Q(y,a)} = t_{x(y,a)} \geq t_j(y)$. Let $\kappa = \max_{1 \leq i \leq m} \{k_i\}$. At most $K\kappa$ items are in open bins after phase y, therefore the cost for this phase is at least $Nt_j(y) - K\kappa = Nw(b_y) - K\kappa$. Summing over all phases we get the cost $\sum_{y=1}^{k} (Nw(b_y) - K\kappa) = N \sum_{y=1}^{k} w(b_y) - Kk\kappa$. The competitive ratio is therefore at least $\frac{\sum_{y=1}^{k} w(b_y)}{s} - \frac{Kk\kappa}{sN}$. This value approaches $\frac{\sum_{y=1}^{k} w(b_y)}{s}$ for large enough N. ■

5 Improved Unbounded Space Algorithms for Small Values of k

5.1 $k = 3$

In this section we design an algorithm for $k = 3$. Already the algorithm of [1] has a competitive ratio lower than the best bounded space algorithm ($\frac{9}{5} = 1.8$ which is smaller than $\frac{11}{6}$). We design an algorithm which uses a more careful partition into classes and has competitive ratio $\frac{7}{4} = 1.75$. The algorithm is based on the idea of the HARMONIC algorithm, and its generalizations (see [14, 15, 17, 9]). In these generalizations, items of two intervals are combined together in the same bins. We would like to use a similar approach, however the boundaries of intervals are chosen with accord to cardinality constraints.

We use the following five intervals. $A = (\frac{2}{3}, 1]$, $B = (\frac{1}{2}, \frac{2}{3}]$, $C = (\frac{1}{3}, \frac{1}{2}]$, $D = (\frac{1}{6}, \frac{1}{3}]$, $E = (0, \frac{1}{6}]$. Items which belong to an interval I are called items of type I, type I items, or simply I items. Items of types A, C and D are packed independently of any other items, one, two and three items per bin, respectively. Note that it is always possible to combine one item of type B with two items of type E. Therefore, each item of type E receives a color upon arrival, white or red. White items are packed in separate bins (three per bin) whereas red items are packed two per bin, and combined with one type B item. If there exists such an open bin, the red type E items are added there. Otherwise once a type B item arrives later, it is added to a bin with two type E items. The colors are assigned so that an α fraction of the type E items are red. We use $\alpha = \frac{1}{4}$. Therefore every fourth type E item is red, and all others are white.

We define a bin as incomplete in the four following packings.

- A bin with a single C item.
- A bin with only one or two D items.
- A bin with one or two white E items.
- A bin with a single red E item (and possibly a B item as well).

At every time, the algorithm can have at most four incomplete bins, one for each combination. Therefore upon termination, except for at most four incomplete bins, all bins can be packed as follows.

- A single A item.
- Two C items.
- Three D items.
- One B item.
- Two red E items
- Three white E items
- One B item and two red E items.

According to the definition of the algorithm, we never have a situation where one bin has only a B item, and another bin has two red E items. This is true since a new bin is opened for such items only if they cannot join a previously opened bin.

The algorithm is therefore at one of the following two situations. 1. There are no bins with two red E items with no B item. 2. There are no bins with one B item and no E items.

We assign two weights to each item, according to the two scenarios. The weights are assigned according to types of items. We use $w_1(I)$ and $w_2(I)$ to denote the weights of type I items according to the two weight functions. Let

$$w_1(A) = w_2(A) = 1,$$

$$w_1(B) = 1,\ w_2(B) = 0,$$

$$w_1(C) = w_2(C) = \frac{1}{2},$$

$$w_1(D) = w_2(D) = \frac{1}{3},$$

$$w_1(E) = \frac{1-\alpha}{3} = \frac{1}{4},\ w_2(E) = \frac{1-\alpha}{3} + \frac{\alpha}{2} = \frac{\alpha+2}{6} = \frac{3}{8}.$$

The weights are defined so that in the first scenario, on average all bins (but at most four) have a total amount of weight of at least 1 packed into them according to the first weight measure, and otherwise the same property holds according to the second weight measure.

We use the following theorem, see Seiden [17].

Theorem 9 *Consider a bin packing algorithm. Let w_1, w_2 be two weight measures. Assume that for every output of the algorithm, there exists i ($i = 1$ or $i = 2$) such that the number of bins used by the algorithm ALG is bounded by $X_i(\sigma) + c$ for some constant c, where $X_i(\sigma)$ is the sum of weights of all items in the sequence according to weight measure w_i. Denote by W_i the maximum amount of weight that can be packed into a single bin according to measure w_i ($i = 1, 2$). Then the competitive ratio of the algorithm is upper bounded by* $\max(W_1, W_2)$.

Proof. Given an input, let i be the value that satisfies the theorem for this input. Clearly $OPT(\sigma) \geq \frac{X_i(\sigma)}{W_i}$. We get $ALG \leq X_i(\sigma) + c \leq W_i OPT + c$. ■

To use the theorem, we need to prove that for every input $ALG \leq X_i(\sigma) + c$ for some i. We ignore the (at most four) incomplete bins, which adds at most 5 to the constant c. The weight of a bin is the sum of weights of items assigned to it. In both scenarios, bins with one A item have weight 1, bins with two C items have weight 1, and so do bins with three D items.

We remove from the sequence items of incomplete bins. Denote the amounts of B items by $n(B)$, and of E items by $n(E)$. The number of red E items is denoted $n(ER)$, and the number of white E items $n(EW)$, (i.e., $n(E) = n(EW) + n(ER)$). According to the color assignments, and since at most two white items and one red item were removed, $3n(ER) \leq n(EW) \leq 3n(ER)+6$. In the first scenario, no bins contain red E items only. The total weight of B and E items is $n(B) + \frac{n(E)}{4}$. The number of bins used for these types is $n(B)+\frac{n(EW)}{3} \leq n(B)+\frac{n(E)+2}{4}$ (using $n(EW) \leq 3n(ER)+6$ which gives $4n(EW) \leq 3n(E) + 6$). In this case we get $ALG < X_1 + 5$. In the second scenario, no bins contain a B item only. The total weight of B and E items is $\frac{3n(E)}{8}$. The number of bins used for these types is $\frac{n(ER)}{2} + \frac{n(EW)}{3} = \frac{n(E)}{3} + \frac{n(ER)}{6} \leq n(E)(\frac{1}{3} + \frac{1}{24}) = \frac{3n(E)}{8}$ (using $3n(ER) \leq n(EW)$ which gives $4n(ER) \leq n(E)$). In this case we get $ALG < X_2 + 4$.

Next we analyze the maximum amount of weight that a bin can contain according to the two weight measures. In both weight measures, if no item has weight 1, the total weight of three items does not exceed $\frac{3}{2}$. Using w_1, the smallest item of weight 1 is slightly larger than $\frac{1}{2}$. If there is a C item, then there can be no D item but only a E item. We get therefore $1 + \frac{1}{2} + \frac{1}{4}$. If there is no C item, the worst case is two extra D items. This gives $1 + \frac{2}{3}$. We get therefore $W_1 = \frac{7}{4} = 1.75$. Using w_2, the smallest item of weight 1 is slightly larger than $\frac{2}{3}$. There can be no B or C items. The worst case is two extra E items, and we get $W_2 = 1 + 2 \cdot \frac{3}{8} = 1.75$.

We proved the following theorem.

Theorem 10 *The competitive ratio of the above algorithm for $k = 3$ is at most* 1.75.

5.2 $k = 4, 5, 6$

In this section we introduce a general algorithm and analyze it for three values of k. The algorithm is a generalization of the algorithm for $k = 3$ with additional options. The intervals (also called classes) are defined as follows. The interval of largest items is $A = (1 - \frac{1}{k}, 1]$. The next interval, of smaller large items is $B = (\frac{1}{2}, 1 - \frac{1}{k}]$. Intervals $C_2, \ldots, C_{k-1}$ are $C_i = (\frac{1}{i+1}, \frac{1}{i}]$. Intervals $E_1, \ldots, E_{k-1}$ are defined to be $E_i = (\frac{1}{k(i+1)}, \frac{1}{ki}]$ for $i < k - 1$ and $E_{k-1} = (0, \frac{1}{k(k-1)}]$.

We use parameters α_i for intervals E_i. An α_i fraction of the items of interval E_i are colored red and all others are colored white. All these values are rational, so if $\alpha_i = \frac{p_i}{q_i}$ is a minimal rational

representation of α_i, then the input items of this intervals are partitioned into sets of q_i items, out of which, the first $q_i - p_i$ are colored white, and the next p_i are colored red.

The packing is done as follows. Items of class C_i are packed i per bin. White items of classes E_i are packed k per bin. Red items of class E_i are packed i per bin. This means that a bin never contains more than $k-1$ red items, and they occupy a space of at most $\frac{1}{k}$. These items can always be combined with type B items. Basically, items of class B are packed one per bin, but when possible, they are combined with one of the types E_i. When we need to open a bin for red E_i items for some i, we first check whether there exists a bin with only a class B item, and if so the red items are added to that bin. Otherwise a new bin is opened for them. When an item of class B arrives, we try to add it into a bin of red items that still did not receive a B item, and open a new bin if it does not exist.

A bin is complete if it received its full amount of items, or if it contains a B item or if it contains the full amount of red items (possibly without a B item). We can neglect bins that are not complete, since their amount is at most $3k-4$. This amount is caused by at most $k-1$ bins for intervals C_i for $1 \leq i \leq k-1$, $k-1$ bins for white items of $k-1$ types, and $k-2$ bins for red items of $k-2$ types (a bin with a red E_1 item cannot be incomplete). As in the algorithm for $k=3$, only one of the two situations can occur. Either there are no complete bins with red items without a class B item, or there are no bins with a class B item and no red items.

We define weights as follows. Assign two weights to each item, according to the two scenarios. The weights are assigned according to types of items. We again use $w_1(I)$ and $w_2(I)$ to denote the weights of type I items according to the two weight functions. Let $w_1(A) = w_2(A) = 1,\ w_1(B) = 1,\ w_2(B) = 0,\ w_1(C_i) = w_2(C_i) = \frac{1}{i},\ w_1(E_i) = \frac{1-\alpha_i}{k},\ w_2(E_i) = \frac{1-\alpha_i}{k} + \frac{\alpha_i}{i} = \frac{i+(k-i)\alpha_i}{ik}$.

The weights are defined so that in the first scenario, on average all bins (neglecting the bins which are not complete) have a total amount of weight of at least 1 packed in them according to the first weight measure, and otherwise the same property holds according to the second weight measure.

To use Theorem 9, we need to prove the conditions of the theorem hold.

Lemma 11 *For every input σ, $ALG(\sigma) \leq X_i(\sigma) + c$ holds for some i.*

Neglecting the incomplete bins (which affect only the constant c), we would like to show that $ALG \leq X_i + c$. For both weight measures cases, bins with one A item have weight 1, and bins with i class C items have weight 1. Denote the numbers of B items by $n(B)$, and of E_i items by $n(E_i)$. The number of red E_i items is denoted $n(ER_i)$, and the number of white E_i items $n(EW_i)$, (i.e., $(n(E_i) = n(EW_i) + n(ER_i))$.

According to the color assignments, let $\alpha_i = \frac{p_i}{q_i}$ (a minimal rational representation of α_i). Then $\alpha_i(n(E_i) - (q_i - p_i)) \leq n(ER_i) \leq \alpha_i n(E_i)$, and $(1-\alpha_i)n(E_i) \leq n(EW_i) \leq (1-\alpha_i)n(E_i) + q_i - p_i$. In the first scenario, no complete bins contain red E_i items only. The total weight of B and E_i items for all i is $n(B) + \sum_{i=1}^{k-1} \frac{1-\alpha_i}{k} \cdot n(E_i)$. The number of bins used for these types is $n(B) + \sum_{i=1}^{k-1} \frac{n(EW_i)}{k} \leq n(B) + \cdot \sum_{i=1}^{k-1} (\frac{1-\alpha_i}{k} n(E_i) + \frac{q_i - p_i}{k})$. In this case we get $ALG < X_1 + c_1$, where c_1 depends on the number of neglected incomplete bins which is constant (for a given choice of the p_i, q_i values). In the second scenario, no bins contain a B item only. The total weight of B and E items is $\sum_{i=1}^{k-1} \frac{i+(k-i)\alpha_i}{ik} \cdot n(E_i)$. The number of bins used for these types is $\sum_{i=1}^{k-1} \left(\frac{n(EW_i)}{k} + \frac{n(ER_i)}{i} \right) \leq$

$\sum_{i=1}^{k-1} \frac{(1-\alpha_i)n(E_i)+q_i-p_i}{k} + \frac{\alpha_i n(E_i)}{i} = \sum_{i=1}^{k-1} n(E_i)\frac{i+(k-i)\alpha_i}{ki} + \frac{q_i-p_i}{k}$. In this case we get $ALG < X_2 + c_2$, where c_2 is a constant which depends on the number of incomplete bins, and on the values chosen for q_i, p_i, $1 \leq i \leq k-1$.

Next we would like to analyze the maximum amount of weight that a bin can contain according to the two weight measures. We do that separately for $k = 4, 5, 6$. We always assume that there are exactly k items in each bin. This is done by allowing items of size 0 that belong to the class E_{k-1}. Note also that we will have ranges of sizes where weights are fixed to be monotonically non-decreasing functions of size, therefore in these cases, we do not need to consider options where a single item can be replaced by a smaller one.

The case $k = 4$. We are aiming at the competitive ratio $\mathcal{R}(4) = \frac{71}{38} \approx 1.86842$. Define the following values. $\alpha_1 = \frac{1}{19}$, $\alpha_2 = \frac{3}{19}$, $\alpha_3 = \frac{9}{19}$. This implies the weights, $w_1(E_1) = \frac{9}{38}$, $w_1(E_2) = \frac{8}{38}$, $w_1(E_3) = \frac{5}{38}$, $w_2(E_i) = \frac{11}{38}$ for $i = 1, 2, 3$.

We compute the maximum amount of weight in a single bin with respect to w_2 first. If no item in the bin is of class A, then the largest weight of any item can be $\frac{1}{2}$. However, a bin can contain at most two such items. All other items have weights of at most $\frac{1}{3}$. This gives a total of at most $\frac{5}{3} < \mathcal{R}(4)$. Next, if a class A item is present, all other others are of classes E_1, E_2, E_3. They all have identical weight. At most three more items can exist, thus we get the total weight $1 + 3 \cdot \frac{11}{38} = \mathcal{R}(4)$.

Next, we compute the maximum weight with respect to w_1. If no item of weight 1 is present, then all weights are upper bounded by the weights of the same items with respect to w_2 and therefore this case is covered by the calculation done for w_2. Otherwise, an item of weight 1 occupies a space of more than $\frac{1}{2}$. If an item of class C_2 exists, it occupies a space of more than $\frac{1}{3}$, and the two other items are of types E_1, E_2, E_3. Moreover, there is room for only one item of either class E_1 or E_2 (these items are larger than $\frac{1}{12}$). Since weights are monotone for all sizes, the worst case is one item of each class B, C_2, E_1, E_3 whose sum of weights is $1 + \frac{1}{2} + \frac{9}{38} + \frac{5}{38} = \mathcal{R}(4)$. If there is no item of C_2, there are three other items, only one of them can be a C_3 item, and which gives the worst case $1 + \frac{1}{3} + 2 \cdot \frac{9}{38} < \frac{69}{38} < \mathcal{R}(4)$.

The case $k = 5$. We are aiming at the competitive ratio $\mathcal{R}(5) = \frac{771}{398} \approx 1.93719$. Define the following values. $\alpha_1 = \frac{9}{199}$, $\alpha_2 = \frac{24}{199}$, $\alpha_3 = \frac{54}{199}$, $\alpha_4 = \frac{114}{199}$. This implies the following weights. $w_1(E_1) = \frac{76}{398}$, $w_1(E_2) = \frac{70}{398}$, $w_1(E_3) = \frac{58}{398}$, $w_1(E_4) = \frac{34}{398}$, $w_2(E_i) = \frac{94}{398}$ for $i = 1, 2, 3$ and $w_2(E_4) = \frac{91}{398}$.

We compute the maximum amount of weight in a single bin with respect to w_2 first. If no item in the bin is of class A, then the largest weight of any item can be $\frac{1}{2}$. However, a bin can contain at most two such items, and at most three items larger than $\frac{1}{4}$ (two of which may be of size larger than $\frac{1}{3}$). The weight of three items larger then $\frac{1}{4}$ is therefore at most $2 \cdot \frac{1}{2} + \frac{1}{3}$. All other items have weight of at most $\frac{1}{4}$, which gives a total of at most $\frac{4}{3} + 2 \cdot \frac{1}{4} = \frac{11}{6} < \mathcal{R}(5)$. Next, if a class A item is present, all other others are of classes E_1, E_2, E_3, E_4. Four more items are present, but at least one of them must be in class E_4. Items in E_1, E_2, E_3 all have weight $\frac{47}{199}$, thus we get the total weight of at most $1 + 3 \cdot \frac{47}{199} + \frac{91}{398} = \mathcal{R}(5)$.

Next, we compute the maximum weight with respect to w_1. If no item of weight 1 is present, then again all weights are bounded from above by the weights of the same items with respect to w_2 and therefore this case is covered by the calculation done for w_2. Otherwise, an item of weight 1 occupies

a space of more than $\frac{1}{2}$.

If an item of class C_2 exists, it occupies a space of more than $\frac{1}{3}$, and the three other items are of types E_1, E_2, E_3, E_4. Moreover, if there is a class E_1 item, then there is no class E_2 item and at most one class E_3 item. This gives a total weight of $1 + \frac{1}{2} + \frac{38}{199} + \frac{29}{199} + \frac{17}{199} = \frac{765}{398} < \mathcal{R}(5)$. If there is no class E_1 item, then if we have a class E_2 item, we can have another item of either class E_2 or E_3, and a class E_4 item, which gives the weight of at most $1 + \frac{1}{2} + 2 \cdot \frac{35}{199} + \frac{17}{199} = \mathcal{R}(5)$. Finally if there is no E_1 and E_2 items, then the weight is at most $1 + \frac{1}{2} + 3 \cdot \frac{29}{199} = \mathcal{R}(5)$.

If no item of class C_2 exists, but there is a class C_3 item, we have the following options. If there is a C_4 item as well, then the occupied area is already more than 0.95 so the other two items are of classes E_4 and this gives weight of at most $1 + \frac{1}{3} + \frac{1}{4} + 2 \cdot \frac{17}{199} < \frac{699}{398} < \mathcal{R}(5)$. If there is no C_4 item, then the largest weight of the additional three items can be $\frac{38}{199}$ each, which bounds the weight by $1 + \frac{1}{3} + 3 \cdot \frac{38}{199} < \frac{759}{398} < \mathcal{R}(5)$.

If no items of classes C_2, C_3 exist, there can be at most two C_4 items and other items have weight at most $\frac{38}{199}$, which together bounds the weight by $1 + 2 \cdot \frac{1}{4} + 2 \cdot \frac{38}{199} < \frac{749}{398} < \mathcal{R}(5)$.

The case $k = 6$. We are aiming at the competitive ratio $\mathcal{R} = \frac{287}{144} \approx 1.99306$. Define the following values. $\alpha_1 = \frac{2}{48} = \frac{1}{24}$, $\alpha_2 = \frac{5}{48}$, $\alpha_3 = \frac{10}{48} = \frac{5}{24}$, $\alpha_4 = \frac{20}{48} = \frac{5}{12}$, $\alpha_5 = \frac{30}{48} = \frac{5}{8}$. This implies the following weights. $w_1(E_1) = \frac{46}{288}$, $w_1(E_2) = \frac{43}{288}$, $w_1(E_3) = \frac{38}{288}$, $w_1(E_4) = \frac{28}{288}$, $w_1(E_5) = \frac{18}{288}$, $w_2(E_i) = \frac{58}{288}$ for $i = 1, 2, 3, 4$ and $w_2(E_5) = \frac{54}{288} = \frac{3}{16}$.

We compute the maximum amount of weight in a single bin with respect to w_2 first. If no item in the bin is of class A, then a bin can contain at most two such items larger than $\frac{1}{3}$, or at most three items larger than $\frac{1}{4}$ or at most four items larger than $\frac{1}{5}$. The worst case gives two items of class C_2, one of C_3 and one of C_4. The two other items have weight of at most $\frac{29}{144}$ (since $\frac{1}{5} < \frac{29}{144}$), which gives a total of at most $2 \cdot \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + 2 \cdot \frac{29}{144} = \frac{286}{144} < \mathcal{R}(6)$. Next, if a class A item is present, all other others are of classes E_i, $1 \leq i \leq 5$. Five more items are present, but at least one of them must be in class E_5. Items in E_1, E_2, E_3, E_4 all have weight $\frac{29}{144}$, thus we get the total weight of at most $1 + 4 \cdot \frac{29}{144} + \frac{27}{144} = \mathcal{R}(6)$.

Next, we compute the maximum weight with respect to w_1. If no item of weight 1 is present, then again all weights are upper bounded by the weights of the same items with respect to w_2 and therefore this case is covered by the calculation done for w_2. Otherwise, an item of weight 1 occupies a space of more than $\frac{1}{2}$. Consider the other contents of the bin. We replace an item of class C_i with an item of size $\frac{1}{i+1}$ (without changing its weight). Similarly we replace an item of class E_i with an item of size $\frac{1}{6(i+1)}$ for $i < k - 1$ and with an item of size 0 if $i = 5$. We only decreased sizes of items therefore they all fit into the bin. We define the expansion of an item of size x of weight w to be $r(x, w) = \frac{w - \frac{1}{16}}{x}$, and for $x = 0$, the expansion is 0. Note that the weight of a set of i items, of total size S and of maximum expansion s is at most $Ss + \frac{i}{16}$.

The expansions for classes $C_2, \ldots, C_5$ are $\frac{189}{144} = 1.3125, \frac{156}{144} \approx 1.08333, \frac{135}{144} = 0.9375, \frac{33}{40} = 0.825$ (respectively). The expansions for classes $E_1, \ldots, E_5$ are $\frac{7}{6} \approx 1.166667, \frac{25}{16} = 1.5625, \frac{5}{3} \approx 1.66667, \frac{150}{144} \approx 1.041667, 0$ (respectively).

Let e_2 and e_3 be the amounts of items of classes E_2 and E_3. If there is no class C_2 item, we can bound the weight as follows. There are $i - e_2 - e_3$ other items, therefore the weight is bounded by $1 + e_2 \frac{43}{288} + e_3 \frac{19}{144} + \frac{5 - e_1 - e_2}{16} + (\frac{1}{2} - \frac{e_2}{18} - \frac{e_3}{24}) \cdot \frac{7}{6} = \frac{91}{48} + \frac{19}{864} e_2 + \frac{18}{864} e_3$. If $e_2 + e_3 \leq 4$ we get at most $\frac{857}{432} < \mathcal{R}(6)$. Otherwise, if $e_1 + e_2 = 5$ we do not have any other items except for an item of

weight 1 and five items of weight $\frac{43}{288}$ or $\frac{38}{288}$, which gives a total of at most $\frac{503}{288} < \mathcal{R}(6)$.

If there is an item of class C_2 the empty space left is less than $\frac{1}{6}$. This means that $i = e_2 + e_3 \leq 3$ and $e_2 \leq 2$. We get a total weight of at most $1.5 + e_2\frac{43}{288} + e_3\frac{19}{144} + \frac{4-e_2-e_3}{16} + (\frac{1}{6} - \frac{e_2}{18} + \frac{e_3}{24}) \cdot \frac{7}{6} = \frac{35}{18} + \frac{19}{864}e_2 + \frac{18}{864}e_3$. If $e_2 + e_3 \leq 2$ we can bound the weight by $\frac{859}{432} < \mathcal{R}(6)$. We are left with the cases $e_2 = 2, e_3 = 1$, $e_2 = 1, e_3 = 2$, $e_2 = 0, e_3 = 3$. In the first two cases, only an item of class E_5 can be added to the bin. In the last case, an item of class E_4 or E_5 can be added. Therefore we need to consider two cases, where the four small items are of classes E_2, E_2, E_3, E_5 and E_3, E_3, E_3, E_4. We get total weights $1.5 + 2 \cdot \frac{43}{288} + \frac{19}{144} + \frac{1}{16} = \mathcal{R}(6)$, and $1.5 + 3 \cdot \frac{19}{144} + \frac{14}{144} = \mathcal{R}(6)$.

We summarize with the following theorem.

Theorem 12 *The competitive ratios of the above algorithm are at most* $\frac{71}{38} \approx 1.86842$ *for* $k = 4$, $\frac{771}{398} \approx 1.93719$ *for* $k = 5$ *and* $\frac{287}{144} \approx 1.99306$ *for* $k = 6$.

6 Conclusion

The main open question is whether an algorithm with competitive ratio strictly better than 2 can be designed for all values of k. In this paper we showed that such an algorithm cannot be bounded space (unless $k \leq 3$). We note that the methods used in this paper for small values of k cannot be extended for larger k.

References

[1] 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.

[2] A. Caprara, H. Kellerer, and U. Pferschy. Approximation schemes for ordered vector packing problems. *Naval Research Logistics*, 92:58–69, 2003.

[3] E. G. Coffman, M. R. Garey, and D. S. Johnson. Approximation algorithms for bin packing: A survey. In D. Hochbaum, editor, *Approximation algorithms*. PWS Publishing Company, 1997.

[4] J. Csirik. An online algorithm for variable-sized bin packing. *Acta Informatica*, 26:697–709, 1989.

[5] J. Csirik and G. J. Woeginger. On-line packing and covering problems. In *A. Fiat and G. J. Woeginger, editors*, Online Algorithms: The State of the Art, pages 147–177, 1998.

[6] J. Csirik and G. J. Woeginger. Resource augmentation for online bounded space bin packing. *Journal of Algorithms*, 44(2):308–320, 2002.

[7] L. Epstein and R. van Stee. On variable-sized multidimensional packing. In *Proc. of the 12th Annual European Symposium on Algorithms (ESA2004)*, pages 287–298, 2004.

[8] L. Epstein and R. van Stee. Online bin packing with resource augmentation. In *Proceedings of the 2nd Workshop on Approximation and Online Algorithms (WAOA 2004)*, pages 48–60, 2004.

[9] L. Epstein and R. van Stee. Optimal online bounded space multidimensional packing. In *Proc. of 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'04)*, pages 207–216, 2004.

[10] D. K. Friesen and M. A. Langston. Variable sized bin packing. *SIAM Journal on Computing*, 15:222–230, 1986.

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

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

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

[14] C. C. Lee and D. T. Lee. A simple online bin packing algorithm. *Journal of the ACM*, 32(3):562–572, 1985.

[15] P. Ramanan, D. J. Brown, C. C. Lee, and D. T. Lee. Online bin packing in linear time. *Journal of Algorithms*, 10:305–326, 1989.

[16] S. S. Seiden. An optimal online algorithm for bounded space variable-sized bin packing. *SIAM Journal on Discrete Mathematics*, 14(4):458–470, 2001.

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

[18] S. S. Seiden, R. van Stee, and L. Epstein. New bounds for variable-sized online bin packing. *SIAM Journal on Computing*, 32(2):455–469, 2003.

[19] J. D. Ullman. The performance of a memory allocation algorithm. Technical Report 100, Princeton University, Princeton, NJ, 1971.

[20] A. van Vliet. An improved lower bound for online bin packing algorithms. *Information Processing Letters*, 43(5):277–284, 1992.

[21] G. J. Woeginger. Improved space for bounded-space online bin packing. *SIAM Journal on Discrete Mathematics*, 6:575–581, 1993.

[22] A. C. C. Yao. New algorithms for bin packing. *Journal of the ACM*, 27:207–227, 1980.