Discrete Mathematics and Theoretical Computer Science DMTCS vol. **11:1**, 2009, 141–152

On-line extensible bin packing with unequal bin sizes[†]

Deshi Ye[‡] and Guochuan Zhang[§]

College of Computer Science, Zhejiang University, Hangzhou 310027, China.

received , revised 13th February 2009, accepted .

In the extensible bin packing problem we are asked to pack a set of items into a given number of bins, each with an original size. However, the original bin sizes can be extended if necessary. The goal is to minimize the total size of the bins. We consider the problem with unequal (original) bin sizes and give the complete analysis on a list scheduling algorithm (LS). Namely we present tight bounds of LS for every collection of original bin sizes and every number of bins. We further show better on-line algorithms for the two-bin case and the three-bin case. Interestingly, it is proved that the on-line algorithms have better competitive ratios for unequal bins than for equal bins. Some variants of the problem are also discussed.

Keywords: On-line algorithms, extensible bin packing, competitive ratio

1 Introduction

We consider the following on-line extensible bin packing problem: there are m bins $B_1, B_2, \ldots, B_m$ with original bin sizes $b_1, b_2, \ldots, b_m$. The original bin sizes can be extended if needed. The (final) size of a bin is defined to be the maximum of the original bin size and the total size of the items assigned to it. Items $\{a_1, a_2, \ldots, a_n\}$ arrive over list, which are not known in advance. When an item a_i arrives it must immediately and irrevocably be assigned to one of the bins and the next item a_{i+1} becomes known only after a_i has been assigned. The size of item a_i is p_i. The goal is to assign all items to the bins such that the total size of the bins is minimized.

This problem arises in a wide variety of contexts. It has many applications in bin packing, storage allocation and scheduling problems. Consider, for instance, a set of workers is given with regular working times. In case of needed, the worker can do some overtime works. The problem is to assign duties to the workers in such a way that the total work (regular working times plus overtime works) is minimized.

[†]A preliminary version of this paper appeared in Proceedings of the First Workshop on Approximation and Online Algorithms (WAOA'03 235-247).

[‡]Research was supported by NSFC(10601048) and National 973 Fundamental Research Project of China(No. 2007CB310900). E-mail: *yedeshi@zju.edu.cn*.

[§]Research was supported by NSFC (60573020). E-mail: *zgc@zju.edu.cn*.

Competitive ratios. An instance of the problem consists of a list of items and m bins with original sizes $b_1, b_2, \ldots, b_m$. Let $\mathcal{B}$ be the collection of bin sizes, i.e., $\mathcal{B} = \{b_1, b_2, \cdots, b_m\}$. For any instance L, let $A_L(m, \mathcal{B})$ and $OPT_L(m, \mathcal{B})$ be the total size of the bins used by an on-line algorithm A and the total size of the bins used by an optimal off-line algorithm, respectively. Then the competitive ratio of algorithm A is

$$R_A(m, \mathcal{B}) = \sup_L A_L(m, \mathcal{B})/OPT_L(m, \mathcal{B}).$$

If the bin sizes are identical, i.e., $b_1 = b_2 = \cdots = b_m$, the competitive ratio is denoted as $R_A(m)$. Further we define the overall competitive ratios as follows.

$$R_A(m, \cdot) = \sup_{\mathcal{B}} R_A(m, \mathcal{B}), \quad R_A = \sup_{m, \mathcal{B}} R_A(m, \mathcal{B}).$$

Known results. The extensible bin packing problem with unequal bin sizes was first studied by Dell'Olmo and Speranza [5]. They considered both the off-line case and the on-line case under the assumption that the maximum item size is not larger than the minimum bin size. They showed an upper bound of $4 - 2\sqrt{2}$ for an LPT (Longest Processing Time) algorithm in the off-line case and an upper bound of $5/4$ for a list scheduling algorithm LS in the on-line case. The bound $5/4$ is tight in terms of the overall competitive ratio, i.e., $R_{LS} = 5/4$.

The special case that all bin sizes are equal to one has been extensively studied. It was proved that the off-line version of the problem is $\mathcal{NP}$-hard in the strong sense [4]. Actually, it can be easily reduced from the 3-partition problem [7]. Dell'Olmo et al. [4] proved that the worst-case ratio of the LPT algorithm is 13/12. If the number of the bins is fixed, it follows from the results of [11] that there exists a fully polynomial time approximation scheme. If m is not fixed, a polynomial time approximation scheme was provided in [1]. Recently, Coffman and Lueker [3] presented a fully polynomial time asymptotic approximation scheme. They also proved that the problem does not admit a fully polynomial time approximation scheme unless $\mathcal{P} = \mathcal{NP}$, when m is not fixed. The on-line version was studied by Speranza and Tuza [10]. They showed that the overall competitive ratio of a list scheduling heuristic is $5/4$. Then a class of heuristics H_x were presented which depend on a parameter x, $0 < x < 1$, and tend to load partially loaded bins until a limit total load of $1 + x$ is reached on the bin. For any number of bins, the algorithm has an overall competitive ratio bounded above by 1.228. For the lower bound of the on-line problem, they showed that no heuristic can have a competitive ratio smaller than $7/6$ by presenting an instance for the case that $m = 2$. Ye and Zhang [12] considered on-line scheduling on a small number m of bins. They proved lower bounds for $m = 3, 4$ and gave the competitive ratios $R_{H_x}(m)$ for $m = 2, 3$ and 4. For $m = 2$ the algorithm is best possible. An improved algorithm for $m = 3$ was also presented.

A similar problem, called *on-line bin stretching*, was introduced by Azar and Regev [2]. A set of bins of fixed sizes is given. Items arrive on-line while it is known that there exists a valid packing of all items into the given bins. One is asked to pack all items into the bins, which can be overloaded. The goal function is the stretch factor, which is the maximum ratio for any bin between the size to which any bin was stretched (the sum of all items assigned to it) and its original size. Azar and Regev [2] studied the problem of identical bins, i.e. all original bins are of size 1. Epstein [6] considered the two-bin case with unequal bin sizes.

Our results. In this paper, firstly we assume, as in [5], that the largest item size is at most the smallest bin size, i.e.,

$$\max_{i=1}^{n} p_i \le \min_{j=1}^{m} b_j. \tag{1}$$

We analyze the LS algorithm more carefully and prove the competitive ratios for each m and each collection $\mathcal{B}$ of bin sizes. The ratios differ for an even number of bins and an odd number of bins. It also shows that the competitive ratio of LS for the equal bins is exactly $5/4$ when m is even and $5/4 - 1/(4m^2)$ when m is odd. We then present an improved on-line algorithm for $m = 2$ and $m = 3$. Finally we discuss the problem without the assumption (1). A lower bound $6/5$ for the overall competitive ratio is presented. We prove the competitive ratio of LS for the two-bin case and present an improved on-line algorithm.

Notations and organization of the paper. Consider any algorithm A. During the execution of algorithm A, if a bin B_j has a load (the total size of items assigned to it) at least its original size b_j, B_j is called *heavy*; otherwise, it is called *light*. The difference between the load and the size b_j is *free space* if B_j is light or *excess* if B_j is heavy.

Consider any instance L where $\sum_{i=1}^{n} p_i < \sum_{j=1}^{m} b_j$. In an optimal packing there must be some bins with free space. Construct a new instance L' by adding new items such that no bins have free space in the optimal packing without changing the optimal value. Note that $A_{L'}(m, \mathcal{B}) \ge A_L(m, \mathcal{B})$ and $OPT_{L'}(m, \mathcal{B}) = OPT_L(m, \mathcal{B})$. Therefore,

$$A_L(m, \mathcal{B})/OPT_L(m, \mathcal{B}) \le A_{L'}(m, \mathcal{B})/OPT_{L'}(m, \mathcal{B}).$$

In instance L', the total size of items is at least the total original size of bins. It implies that we only need to consider the instances in which the total size of items is at least the total original size of bins to prove an upper bound. Throughout the paper in proving the competitive ratios we always assume

$$\sum_{i=1}^{n} p_i \ge \sum_{j=1}^{m} b_j. \tag{2}$$

The remainder of the paper is organized as follows. Section 2 gives the tight bound for a list scheduling algorithm. In Section 3, we present an on-line algorithm for $m = 2$ and $m = 3$. We discuss the problem without the item size assumption (1) in Section 4.

2 Tight bound for a list scheduling algorithm

List Scheduling was introduced by Graham [8] for parallel machine scheduling. The algorithm always schedules the current job to the machine with minimum load at this time. Applying list scheduling to extensible bin packing with unequal bin sizes may result in different versions. We first consider a list scheduling algorithm, denoted by LS, which always assigns the incoming item to the bin of largest free space. Dell'Olmo and Speranza [5] proved that $R_{LS} = 5/4$. In this section we figure out $R_{LS}(m, \mathcal{B})$. Then we will show that a different version of list scheduling, which always assigns the incoming item to the bin with the least load (the total size of items in a bin), is not as good as LS in terms of the overall competitive ratio. The two versions of list scheduling are indeed equivalent if all the bin sizes are the same.

Let b_{min} be the smallest bin sizes in the collection $\mathcal{B}$ and let p_{max} be the largest item size. Under the assumption (1), we have $p_{max} \leq b_{min}$. Consider the packing given by algorithm LS. For each bin B_j, let M_j be the total size of the items it accommodates after the schedule is over. If B_j is light, let s_j be its free space, i.e., $s_j = b_j - M_j$. If B_j is heavy, let r_j be its free space immediately before it becomes heavy, and e_j be its excess, i.e., $e_j = M_j - b_j$. We re-index the bins so that the first k bins are heavy, where k denotes the number of heavy bins.

Lemma 1 *The competitive ratio of the list scheduling algorithm*

$$R_{LS}(m, \mathcal{B}) \geq \begin{cases} 1 + \frac{mb_{min}}{4\sum_{j=1}^{m} b_j}, & \text{if } m \text{ is even,} \\ 1 + \frac{(m^2-1)b_{min}}{4m\sum_{j=1}^{m} b_j}, & \text{if } m \text{ is odd.} \end{cases}$$

Proof: Consider the following instances. If m is even, a large number of small enough items are coming first. The total size of these items is $\sum_{j=1}^{m} b_j - mb_{min}/2$ such that after assigning these items by LS, each bin has free space exactly $b_{min}/2$. Then $m/2$ items with size of b_{min} are coming. Clearly, the optimal value is $\sum_{j=1}^{m} b_j$, while LS generates a total size $\sum_{j=1}^{m} b_j + mb_{min}/4$. Thus $R_{LS}(m, \mathcal{B}) \geq 1 + \frac{mb_{min}}{4\sum_{j=1}^{m} b_j}$.

If m is odd, the instance is slightly different. A large number of small enough items are coming first. The total size of them is $\sum_{j=1}^{m} b_j - (m-1)b_{min}/2$, so that after assigning these items each bin has free space exactly $\frac{m-1}{2m} b_{min}$. Then $(m-1)/2$ items with size b_{min} are coming. Clearly, the optimal total bin size is $\sum_{j=1}^{m} b_j$, while LS generates a total bin size of $\sum_{j=1}^{m} b_j + (m^2-1)b_{min}/(4m)$. Thus $R_{LS}(m, \mathcal{B}) \geq 1 + \frac{(m^2-1)b_{min}}{4m\sum_{j=1}^{m} b_j}$. The lemma is proved. □

For an on-line algorithm in the resulting packing, if all bins are heavy or all bins are light, the packing is optimal. We only need to consider the packing with both heavy bin and light bin. It is crucial to estimate the total excess or the total free space of all bins. In proving an upper bound for algorithm LS we can further assume, without loss of generality, that a bin accepts no more items once it becomes heavy. Otherwise, all bins are heavy and the packing is optimal.

Theorem 2 *The competitive ratio of the list scheduling algorithm*

$$R_{LS}(m, \mathcal{B}) = \begin{cases} 1 + \frac{mb_{min}}{4\sum_{j=1}^{m} b_j}, & \text{if } m \text{ is even,} \\ 1 + \frac{(m^2-1)b_{min}}{4m\sum_{j=1}^{m} b_j}, & \text{if } m \text{ is odd.} \end{cases}$$

Proof: We only need to prove that

$$R_{LS}(m, \mathcal{B}) \leq \begin{cases} 1 + \frac{mb_{min}}{4\sum_{j=1}^{m} b_j}, & \text{if } m \text{ is even,} \\ 1 + \frac{(m^2-1)b_{min}}{4m\sum_{j=1}^{m} b_j}, & \text{if } m \text{ is odd.} \end{cases}$$

By the algorithm LS, we know that

$$\frac{\sum_{j=1}^{k} r_j}{k} \geq \min_{j=1,\cdots,k} r_j \geq \max_{j=k+1,\cdots,m} s_j \geq \frac{\sum_{j=k+1}^{m} s_j}{m-k}.$$

Then

$$(m-k)\sum_{j=1}^{k} r_j \geq k \sum_{j=k+1}^{m} s_j \tag{3}$$

From (2), we have

$$\sum_{i=j}^{k} e_j \geq \sum_{j=k+1}^{m} s_j. \tag{4}$$

We add $(m-k)\sum_{j=1}^{k} e_j$ to both sides of the inequality (3) and from (4), we get

$$\begin{aligned}(m-k)\sum_{j=1}^{k}(r_j+e_j) &\geq k\sum_{j=k+1}^{m} s_j + (m-k)\sum_{j=1}^{k} e_j \\ &\geq k\sum_{j=k+1}^{m} s_j + (m-k)\sum_{j=k+1}^{m} s_j \\ &= m\sum_{j=k+1}^{m} s_j.\end{aligned}$$

It means that $\sum_{j=k+1}^{m} s_j \leq \frac{(m-k)}{m}\sum_{j=1}^{k}(r_j+e_j)$. Note that the competitive ratio

$$R_{LS}(m,\mathcal{B}) \leq (\sum_{i=1}^{n} p_i + \sum_{j=k+1}^{m} s_j)/\sum_{i=1}^{n} p_i.$$

Then

$$R_{LS}(m,\mathcal{B}) \leq 1 + \frac{(m-k)\sum_{j=1}^{k}(r_j+e_j)}{m\sum_{i=1}^{n} p_i} \leq 1 + \frac{(m-k)\sum_{j=1}^{k}(r_j+e_j)}{m\sum_{j=1}^{m} b_j}.$$

Since $r_j + e_j \leq p_{\max} \leq b_{min}$, we obtain

$$R_{LS}(m,\mathcal{B}) \leq 1 + \frac{(m-k)kb_{min}}{m\sum_{j=1}^{m} b_j}.$$

If m is even, then $(m-k)k/m^2 \leq 1/4$. It follows

$$R_{LS}(m,\mathcal{B}) \leq 1 + \frac{mb_{min}}{4\sum_{j=1}^{m} b_j}.$$

If m is odd, then $(m-k)k \leq \frac{1}{4}(m^2-1)$. We have

$$R_{LS}(m,\mathcal{B}) \leq 1 + \frac{(m^2-1)b_{min}}{4m\sum_{j=1}^{m} b_j}.$$

$\square$

Corollary 3

$$R_{LS}(m,\mathcal{B}) \le R_{LS}(m) = \begin{cases} \frac{5}{4}, & \text{if } m \text{ is even,} \\ \frac{5}{4} - \frac{1}{4m^2}, & \text{if } m \text{ is odd.} \end{cases}$$

Moreover, $R_{LS} = 5/4$. □

Note that $R_{LS}(m,\mathcal{B}) < R_{LS}(m)$ if $m \ge 2$ and $b_i \ne b_j$ for some i, j. It means that the competitive ratio of LS becomes smaller if the bin sizes are unequal.

Before closing this section, we consider a different version of list scheduling: assign items to the bin of least load. Denote the algorithm as LS'. Consider the following simple instance L for two bins. Let $b_1 = 2b_2$ and let $\varepsilon > 0$ be a sufficiently small number. The first two items have size of $b_2 - \varepsilon$, which are followed by an item with size of $\varepsilon/2$. By algorithm LS', after assigning the first two items, each bin has a load of $b_2 - \varepsilon$. If the 3rd item goes to the first bin B_1, an item (the last one) with size of b_2 comes, which is assigned to B_2. If the 3rd item goes to the second bin B_2, then the 4th item has a size of ε and the 5th item (the last one) has size of b_2. The 4th is assigned to B_1 while the 5th is assigned to B_2. In both cases, $LS'_L(2,\mathcal{B}) \ge b_1 + b_2 + b_2 - \varepsilon$. In an optimal packing, we can assign the last item to B_2 and the others to B_1. Then $OPT_L(2,\mathcal{B}) = b_1 + b_2$, which shows that $R_{LS'}(2,\mathcal{B}) \to 4/3$ as ε tends to zero. It implies that $R_{LS'} \ge 4/3$.

3 The two- and three-bin cases

Assume that $b_1 \ge b_2 \ge \cdots \ge b_m \ge p_{max}$. In this section we consider the cases that $m = 2$ and $m = 3$.

Algorithm $A_m(\alpha)$: If all bins are heavy, then assign the incoming item a_i arbitrarily. If there is a light bin whose excess will be at most α in case of accepting, assign the item a_i to such a bin with lowest index; if such a bin does not exist, assign a_i to the bin with largest free space.

In this algorithm α is a user-specified parameter. Choosing different parameters results in different algorithms.

Lemma 4 *For any parameter $\alpha > 0$, the competitive ratio of algorithm $A_2(\alpha)$ is at least* $1 + b_2/(3(b_1 + b_2))$.

Proof: Let N be a sufficiently large integer. If $\alpha < b_2/3$, consider an instance L as follows. The first $\lfloor (b_1 - 2b_2/3)N \rfloor$ items are small items, each with size $1/N$, which are followed by items a_1, a_2, a_3 with size of $b_2/3, 2b_2/3$ and $2b_2/3$, respectively. Clearly, $A_2(\alpha)$ assigns all the small items together with a_1 to bin B_1, and a_2 to bin B_2. Thus no matter where a_3 is assigned, $A_2(\alpha)_L(2,\mathcal{B}) \ge b_1 + b_2 + b_2/3 - 1/N$. For an optimal solution, we can assign a_1 and a_2 to B_2, and the remaining items to B_1. Thus $OPT_L(2,\mathcal{B}) = b_1 + b_2$, which follows that $A_2(\alpha)_L(2,\mathcal{B})/OPT_L(2,\mathcal{B}) \to 1 + b_2/(3(b_1 + b_2))$ as N goes to infinity.

If $\alpha \ge b_2/3$, consider the following instance L: the first $\lfloor (b_1 - 2b_2/3)N \rfloor$ items with size $1/N$, are followed by item a_1 with size b_2. So $A_2(\alpha)_L(2,\mathcal{B}) \ge b_1 + b_2/3 + b_2 - 1/N$. In an optimal packing, we assign a_1 to B_2, and the remaining items to B_1. $OPT_L(2,\mathcal{B}) = b_1 + b_2$, which follows $A_2(\alpha)_L(2,\mathcal{B})/OPT_L(2,\mathcal{B}) \to 1 + b_2/(3(b_1 + b_2))$ as N tends to infinity. □

By setting $\alpha = b_2/3$, we prove that the competitive ratio of $A_2(\alpha)$ is $1 + b_2/(3(b_1 + b_2))$.

Theorem 5 *The competitive ratio of* $A_2(\alpha)$ *is* $1 + b_2/(3(b_1 + b_2))$ *if* $\alpha = b_2/3$. *Moreover the overall competitive ratio is* $7/6$.

Proof: If both bins are light or both are heavy, the packing is optimal. We only need to consider the case that exactly one bin is heavy. From the assumption (2), we have $M_1 + M_2 \geq b_1 + b_2$, where M_i is the load of bin B_i, $i = 1, 2$.

Let $\alpha = b_2/3$. Let T be the excess of the heavy bin and let X be the free space of the light bin. If $T \leq \alpha$, then $R_{A_2(\alpha)}(2, \mathcal{B}) \leq (T + b_1 + b_2)/(b_1 + b_2) \leq 1 + b_2/(3(b_1 + b_2))$. Now assume that $T > \alpha$. We want to prove $X \leq \alpha$.

Case 1. B_2 is light. $X = b_2 - M_2$. Let a_n be the last item assigned to B_1, which makes the excess over α. B_2 is not empty and before a_n is assigned B_1 is light and has an free space at least X. Otherwise, a_n should have been assigned to B_2. It implies that the size of the items in B_2 is larger than $X + \alpha$, otherwise all the items in B_2 can be assigned to B_1. Thus $X = b_2 - M_2 < b_2 - (X + \alpha)$. It follows that $X < \alpha$.

Case 2. B_1 is light. $X = b_1 - M_1$. Before the last item is assigned to B_2, B_2 is light and has an free space at least X. The total size of items in B_2 before the last item is assigned is at most $b_2 - X$. According to the algorithm, $b_2 - X + M_1 > b_1 + \alpha$. Then $b_2 - X > b_1 - M_1 + \alpha$ which implies that $X \leq \alpha$.

In both cases we have $R_{A_2(\alpha)}(2, \mathcal{B}) \leq (X + M_1 + M_2)/(M_1 + M_2) \leq 1 + b_2/(3(b_1 + b_2))$. It is easy to verify that $R_{A_2(\alpha)}(2, \cdot) = 7/6$. □

Since no on-line algorithm can have an overall competitive ratio less than 7/6 for two bins [5], $A_2(\alpha)$ is an optimal on-line algorithm for $m = 2$ in terms of the overall competitive ratio, setting $\alpha = b_2/3$.

We will show a lower bound for two bins under the assumption (1). Let $b_1 = kb_2/3 + x$, where $0 \leq x < b_2/3$ and $k \geq 3$ (k is an integer).

Lemma 6 *No on-line algorithm can achieve a competitive ratio smaller than*

$$R = \begin{cases} 1 + (b_2/3 - x)/(b_1 + b_2), & \text{if } 0 \leq x \leq b_2/6; \\ 1 + x/(b_1 + b_2), & \text{if } b_2/6 < x < b_2/3. \end{cases}$$

Proof: Let A be any on-line algorithm. Consider the following instance L. The items with size of $b_2/3$ come one by one until $n_2 = 2$ or $n_1 = k - 1$, where n_i is the number of items with size of $b_2/3$ placed into bin B_i by algorithm A for $i = 1, 2$.

Case 1. $n_2 = 0, n_1 = k - 1$. If $0 \leq x \leq b_2/6$, the next two items have size of $2b_2/3$. Thus $A_L(2, \mathcal{B}) \geq b_1 + b_2 + b_2/3 - x$.

If $b_2/6 < x < b_2/3$, the next item has size of x. If x goes to B_1, two items with size of $2b_2/3$ come. It follows that $A_L(2, \mathcal{B}) \geq b_1 + b_2 + b_2/3$. If x goes to B_2, an item with size of b_2 comes. Then $A_L(2, \mathcal{B}) \geq b_1 + b_2 + x$.

Case 2. $n_2 = 1, n_1 = k - 1$. The next item has size of b_2 and then $A_L(2, \mathcal{B}) \geq b_1 + b_2 + b_2/3$.

Case 3. $n_2 = 2, n_1 = p$, where $0 \leq p < k - 1$.

Subcase 3a) For the sake of proof, we also consider the case $p = k - 1$. In this case the next item has size of $2b_2/3 + x$, and thus $A_L(2, \mathcal{B}) \geq b_1 + b_2 + b_2/3$.

Subcase 3b) If $p = k-2$, we consider two subcases. If $0 \le x \le b_2/6$, the next two items have sizes of x and b_2 follows. If the first item is assigned to B_1, then $A_L(2, \mathcal{B}) \ge b_1 + b_2 + b_2/3$. If the first item is assigned to B_2, then we have $A_L(2, \mathcal{B}) \ge b_1 + b_2 + b_2/3 - x$. If $b_2/6 < x < b_2/3$, the next two items have size of $b_2/3 + x$ and $2b_2/3$. So, $A_L(2, \mathcal{B}) \ge b_1 + b_2 + x$.

Subcase 3c) If $p = k-3$, the next two items have size of $2b_2/3 + x/2$. $A_L(2, \mathcal{B}) \ge b_1 + b_2 + b_2/3$.

Subcase 3d) If $p \le k-4$, let $s = \lceil (k-p)/3 \rceil$-1. The next s items have size of b_2. If one of such items is packed into B_2, then $A_L(2, \mathcal{B}) \ge b_1 + b_2 + 2b_2/3$. Otherwise, all the s items go to B_1. The free space of B_1 becomes $(k-p)b_2/3 + x - (\lceil (k-p)/3 \rceil - 1)b_2$, which is either $b_2/3 + x$ or $2b_2/3 + x$ or $b_2 + x$. Then it is reduced to the above three subcases 3a) -3c).

Note that in any of the above cases, $OPT_L(2, \mathcal{B}) = b_1 + b_2$. Thus, the lemma is proved. □

Lemma 6 shows that $R_A(2, \mathcal{B}) \ge 1 + \frac{b_2}{6(b_1+b_2)}$ for any on-line algorithm A. It also shows that algorithm $A_2(\alpha)$ is optimal for the case that $x = 0$ (or $b_1 = kb_2/3$ for some integer $k \ge 4$), by setting $\alpha = b_2/3$. Now we consider the three-bin case.

Lemma 7 *For any parameter $\alpha > 0$, the competitive ratio of algorithm $A_3(\alpha)$ is at least* $1 + b_3/(2(b_1 + b_2 + b_3))$.

Proof: Let N be a sufficiently large integer. If $\alpha \ge b_3/2$, consider the following instance L. The first $\lfloor b_1 N - 1 \rfloor$ items with size of $1/N$ are followed by an item of size $b_3/2 + 1/N$. Clearly, $A_3(\alpha)_L(3, \mathcal{B}) \ge b_1 + b_2 + 3b_3/2 - 1/N$ and $OPT_L(3, \mathcal{B}) \le b_1 + b_2 + b_3 + 1/N$. It follows that $A_3(\alpha)_L(3, \mathcal{B})/OPT_L(3, \mathcal{B}) \to 1 + b_3/(2(b_1 + b_2 + b_3))$ as N tends to infinity.

If $\alpha < b_3/2$, consider the following instance L. $\lfloor b_1 N - 1 \rfloor$ items with size of $1/N$ are followed by item a_1 with size of $\alpha + 1/N$. Then $\lfloor (b_2 - b_3/2)N \rfloor + 1$ items with size of $1/N$ are coming, which are followed by the last two items a_2, a_3 with size of $b_3/2$, $b_3 - \alpha - 1/N$, respectively. Algorithm $A_3(\alpha)$ assigns the first $\lfloor b_1 N - 1 \rfloor$ items together with a_1 to B_1, $\lfloor (b_2 - b_3/2)N + 1 \rfloor$ items with size $1/N$ to B_2, and a_2 and a_3 to B_3. It shows $A_3(\alpha)_L(3, \mathcal{B}) \ge b_1 + b_2 + 3b_3/2 - 2/N$. In an optimal packing, we can assign a_1 and a_3 to B_3, the first $\lfloor b_1 N - 1 \rfloor$ items with size $1/N$ to B_1 and the remaining items to B_2. It follows that $OPT_L(3, \mathcal{B}) \le b_1 + b_2 + b_3$. Then $A_3(\alpha)_L(3, \mathcal{B})/OPT_L(3, \mathcal{B}) \to 1 + b_3/(2(b_1 + b_2 + b_3))$, as N tends to infinity. □

In the following, we prove that the competitive ratio of algorithm $A_3(\alpha)$ can reach the lower bound $1 + b_3/(2(b_1 + b_2 + b_3))$ by setting $\alpha = b_3/2$.

Theorem 8 *The competitive ratio of the algorithm $A_3(\alpha)$ is $1 + b_3/(2(b_1 + b_2 + b_3))$ if $\alpha = b_3/2$.*

Proof: Let $\alpha = b_3/2$. In the packing given by $A_3(\alpha)$, let M_i be the total size of the items assigned to bin B_i. If all the bins are heavy or all the bins are light, the algorithm gives an optimal packing. Hence, we only need to consider the following cases.

Case 1. Only one bin is heavy. If the excess of the heavy bin is at most α, we can get the competitive ratio $1 + b_3/(2(b_1 + b_2 + b_3))$ immediately. Before the last item of the heavy bin is assigned, all the three bins are light and the heavy bin has the largest free space at the moment.

Subcase 1a) B_1 is heavy. Then $M_1 > b_1 + \alpha$. Note that $M_2 + M_3 > b_2 + \alpha$. We have

$$R_{A_3(\alpha)}(3, \mathcal{B}) \le (M_1 + b_2 + b_3)/(M_1 + M_2 + M_3)$$

$$
\begin{aligned}
&= 1 + (b_2 + b_3 - M_2 - M_3)/(M_1 + M_2 + M_3) \\
&\leq 1 + (b_2 + b_3 - b_2 - \alpha)/(b_1 + b_2 + b_3) \\
&\leq 1 + b_3/(2(b_1 + b_2 + b_3))
\end{aligned}
$$

Subcase 1b) B_2 is heavy. $M_2 > b_2 + b_3/2$ and $M_1 + M_3 > b_1 + b_3/2$.

$$
\begin{aligned}
R_{A_3(\alpha)}(3, \mathcal{B}) &\leq (M_2 + b_1 + b_3)/(M_1 + M_2 + M_3) \\
&= 1 + (b_1 + b_3 - M_1 - M_3)/(M_1 + M_2 + M_3) \\
&\leq 1 + b_3/(2(M_1 + M_2 + M_3)) \\
&\leq 1 + b_3/(2(b_1 + b_2 + b_3))
\end{aligned}
$$

Subcase 1c) B_3 is heavy. $M_3 > b_3 + b_3/2$. Let X be the total free space of M_1 and M_2. Then $X = b_1 + b_2 - (M_1 + M_2)$. Let Y_3 be the load in B_3 before the last item is assigned. Thus

$$b_3 - Y_3 \geq b_2 - M_2,\ b_3 - Y_3 \geq b_1 - M_1.$$

We have $X = b_1 + b_2 - (M_1 + M_2) \leq 2(b_3 - Y_3)$. On the other hand,

$$Y_3 + M_1 \geq b_1 + b_3/2,\ Y_3 + M_2 \geq b_2 + b_3/2.$$

We have $X = b_1 + b_2 - (M_1 + M_2) \leq 2Y_3 - b_3$. So, $X \leq \min\{2(b_3 - Y_3), 2Y_3 - b_3\} \leq b_3/2$. Therefore,

$$
\begin{aligned}
R_{A_3(\alpha)}(3, \mathcal{B}) &\leq (X + M_1 + M_2 + M_3)/(M_1 + M_2 + M_3) \\
&\leq 1 + b_3/(2(b_1 + b_2 + b_3)).
\end{aligned}
$$

Case 2. Only one bin is light. If the free space X of the light bin is at most α or the total excess of the two heavy bins is at most α, we have the competitive ratio immediately. We only need to consider the case that $X > \alpha$ and the total excess exceeds α. However, we will show that it is impossible. Before considering the following two cases, we assume that $M_i > 0$ for $i = 1, 2, 3$. Otherwise, the total excess is at most α.

Subcase 2a) B_3 is light. $M_3 = b_3 - X < \alpha = b_3/2$. Let Y_i be the load of B_i immediately before it becomes heavy, for $i = 1, 2$. Then $b_i - Y_i \geq X$; otherwise the last item, which makes the total excess of B_1 and B_2 over α, would have been assigned to bin B_3, then any item less than b_3 can be assigned to B_1 or B_2. Thus all the items in B_3 should have been assigned to B_i by the algorithm. It is a contradiction.

Subcase 2b) B_3 is heavy. Recall that any item size is at most b_3. There are at least two items in B_3. Assume that a_k is the last item assigned to B_3. Those items assigned to B_3 have size at least $X > b_3/2$. Thus the idle space of B_3 before a_k is assigned is less than X, which means that a_k would have not been assigned to B_3. It is a contradiction. □

From Theorems 5 and 8 we realize that the competitive ratio of algorithm $A_m(\alpha)$ for unequal bins is better than that for equal bins.

4 Without the assumption on item sizes

In this section, we will discuss the problem without the assumption (1) for the two-bin case. However, it is reasonable to assume that all items with size at most b_1.

Lemma 9 *Without the assumption (1), no deterministic on-line algorithms can achieve an overall competitive ratio lower than* $6/5$ *for two bins.*

Proof: Let $b_1 = 3/2$ and $b_2 = 1$. The first item a_1 has size of 1/2. If a_1 is assigned to bin B_2, two items a_2, a_3 with size of 1 are coming. If a_1 is assigned to bin B_1, item a_2 with size of 3/2 is coming. For both cases, the overall competitive ratio is at least 6/5. □

Lemma 10 *For any on-line algorithm A the competitive ratio*

$$R_A(2, \mathcal{B}) \geq \begin{cases} 1 + (4b_2/3 - b_1)/(b_1 + b_2), & \text{if } b_1 \leq 7b_2/6, \\ 1 + (b_1 - b_2)/(b_1 + b_2), & \text{if } 7b_2/6 < b_1 < 4b_2/3, \\ 1 + b_2/(3(b_1 + b_2)), & \text{if } b_1 \geq 4b_2/3. \end{cases}$$

Proof: Let A be any on-line algorithm. Consider first the case that $b_1 \geq 4b_2/3$ with the following instance L. The first two items have size of $b_2/3$. If both items go to B_1, the next item has size of b_1; if both items go to B_2, two items with size of $2b_2/3$ and $b_1 - b_2/3$, respectively, are coming; if the first two items go to different bins, the next item has size of b_1. For any of the above cases, $A_L(2, \mathcal{B}) \geq b_1 + 4b_2/3$ and the optimal value $OPT_L(2, \mathcal{B}) = b_1 + b_2$. It implies that the lemma is true.

Now turn to the case that $b_1 < 4b_2/3$. Let $b_1 = b_2 + x$, where $0 \leq x < b_2/3$. The lemma follows from Lemma 6 by setting $k = 3$. □

Now we turn to upper bounds. Assume that all items have size of at most b_1 ($b_1 \geq b_2$).

Theorem 11 *The competitive ratio of the list scheduling algorithm LS for two bins is* $1+\min\{b_2, b_1/2\}/(b_1+b_2)$.

Proof: We first show the upper bound. Without loss of generality, we consider the case only one bin is heavy.

Case 1. $b_1/2 \geq b_2$. The LS algorithm assigns items to the bin with largest free space. Recall that the total size of items is not less than the total size of bins, we get that the free space is at most b_2. Then, $LS_L(2, \mathcal{B}) \leq \sum p_i + b_2$, $OPT_L(2, \mathcal{B}) \geq \sum p_i \geq b_1 + b_2$, which implies $R_{LS}(2, \mathcal{B}) \leq 1 + b_2/(b_1 + b_2)$.

Case 2. $b_1/2 < b_2$. Let X be the free space and T be the excess. We only consider the case that both X and T are greater than $b_1/2$. Otherwise, it follows that $R_{LS}(2, \mathcal{B}) \leq 1 + b_1/(2(b_1 + b_2))$. Let p_n be the size of the last item assigned to the bin which has an excess over $b_1/2$. Recall that all items have size of at most b_1. Let Y_i be the load of bin B_i before the last item is assigned. Then $b_i - Y_i \geq X > b_1/2$. Since $p_n \leq b_1$, if $X > b_1/2$, then T is less than $b_1/2$, which implies that either X or T is at most $b_1/2$.

The following simple instance shows that the bound is tight. Consider three items with sizes of $\min\{b_2, b_1/2\}$, $\max\{0, b_2 - b_1/2\}$, and b_1, respectively. The optimal value is $b_1 + b_2$ while LS costs $b_1 + b_2 + \min\{b_2, b_1/2\}$. □

In the following, we present an on-line algorithm to improve the upper bound in some cases.

Algorithm $A2$**:**

- If $b_1 \leq 4b_2/3$, apply algorithm $A_2(\alpha)$ by setting $\alpha = b_2/3$;
- if $4b_2/3 < b_1 \leq 2b_2$, apply $A_2(\alpha)$ by setting $\alpha = b_1 - b_2$;

- if $b_1 > 2b_2$, apply algorithm LS.

Theorem 12 *The competitive ratio of algorithm $A2$ is*

$$R_{A2}(2, \mathcal{B}) \le \begin{cases} 1 + b_2/(3(b_1+b_2)), & \text{if } b_1 \le 4b_2/3, \\ 1 + (b_1 - b_2)/(b_1 + b_2), & \text{if } 4b_2/3 < b_1 \le 2b_2, \\ 1 + b_2/(b_1 + b_2), & \text{if } b_1 > 2b_2. \end{cases}$$

Proof: Consider the following cases:

Case 1. $b_1 \le 4b_2/3$. We use $A_2(\alpha)$, by setting $\alpha = b_2/3$. The proof is similar as the one of Theorem 5.

Case 2. $4b_2/3 < b_1 \le 2b_2$. We only need to consider the case that exactly one bin is heavy, the excess of the heavy bin is over $b_1 - b_2$, and the free space of the light bin is larger than $b_1 - b_2$. However, we will show this case is impossible. Before the last item a_n, which makes the excess over $b_1 - b_2$, is assigned, if B_2 is empty, then the excess is at most $b_1 - b_2$ after assigning a_n. Assume that B_2 is not empty before a_n is assigned. Note that B_1 has an free space larger than $b_1 - b_2$ before a_n is assigned. It means that the total size of items in B_2 is more than $2(b_1 - b_2)$. Then the idle space of bin B_2 is at most $b_2 - 2(b_1 - b_2) < (b_1 - b_2)$. In this case a_n is assigned to B_1. B_2 is a light bin and the idle space is less than $b_1 - b_2$. It is a contradiction.

Case 3. $b_1 > 2b_2$. It follows from Theorem 11. □

Final Remarks. There are many open questions. Although we have proved that the algorithm $A_2(\alpha)$ is optimal in terms of the overall competitive ratio, it is still open to find a best possible on-line algorithm for two bins in terms of the competitive ratio. It is interesting to design an algorithm with a better competitive ratio than LS algorithm for any number of bins. The problem without the assumption (1) becomes more difficult. We only proved the competitive ratio of LS for the two-bin case. Any improvement on the lower bounds is also of interest.

Acknowledgements

The authors would like to thank the referees and the Associate Editor for their helpful suggestions.

References

[1] N. Alon, Y. Azar, G.J. Woeginger, and T. Yadid. Approximation schemes for scheduling on parallel machines. *Journal of Scheduling* **1**:55-66, 1998.

[2] Y. Azar and O. Regev. On-line bin-stretching. *Theoretical Computer Science* **268**:17-41, 2001.

[3] E.G. Coffman and G.S. Lueker. Approximation algorithms for extensible bin packing. *Proceedings of the twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, 586-588, 2001.

[4] P. Dell'Olmo, H. Kellerer, and M.G. Speranza, Z. Tuza. A 13/12 approximation algorithm for bin packing with extendable bins. *Information Processing Letters* **65**:229-233, 1998.

[5] P. Dell'Olmo and M.G. Speranza. Approximation algorithms for partitioning small items in unequal bins to minimize the total size. *Discrete Applied Mathematics* **94**:181-191, 1999.

[6] L. Epstein. Bin stretching revisited. *Acta Informatica* **39**:97-117, 2003.

[7] M.R. Garey and D.S. Johnson. Computers and Intractability: A guide to the Theory of $\mathcal{NP}$-completeness. W.H.Freeman, San Francisco, 1979.

[8] R. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal* **45**:1563-1581, 1966.

[9] J. Sgall. On-line scheduling - A survey, Online Algorithms: The State of the Art, eds. A. Fiat and G. J. Woeginger. *Lecture Notes in Comput. Sci.* **1442**, 196-231, Springer-Verlag, 1998.

[10] M.G. Speranza and Z. Tuza. On-line approximation algorithms for scheduling tasks on identical machines with extendable working time. *Annals of Operations Research* **86**:494-506, 1999.

[11] G.J. Woeginger. When does a dynamic programming formulation guarantee the existence of an FPTAS? In *Proceedings of the tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 820-829, 1999.

[12] D. Ye and G. Zhang. On-line scheduling with extendable working time on a small number of machines. *Information Processing Letters* **85**:171-177, 2003.