Available online at www.sciencedirect.com

SCIENCE DIRECT®

European Journal of Operational Research 147 (2003) 365–372

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Discrete Optimization

Algorithms for the variable sized bin packing problem

Jangha Kang, Sungsoo Park [*]

Department of Industrial Engineering, Korea Advanced Institute of Science and Technology, 373-1 Guseong-Dong Yuseong-Gu, Daejon 305-701, South Korea

Received 4 April 2001; accepted 15 February 2002

Abstract

In this paper, we consider the variable sized bin packing problem where the objective is to minimize the total cost of used bins when the cost of unit size of each bin does not increase as the bin size increases. Two greedy algorithms are described, and analyzed in three special cases: (a) the sizes of items and bins are divisible, respectively, (b) only the sizes of bins are divisible, and (c) the sizes of bins are not divisible. Here, we say that a list of numbers $a_1, a_2, \dots, a_m$ are divisible when a_j exactly divides a_{j-1}, for each $1 < j \leqslant m$. In the case of (a), the algorithms give optimal solutions, and in the case of (b), each algorithm gives a solution whose value is less than $\frac{11}{9}C(B^*) + 4\frac{11}{9}$, where $C(B^*)$ is the optimal value. In the case of (c), each algorithm gives a solution whose value is less than $\frac{3}{2}C(B^*) + 1$.

Keywords: Algorithm; Analysis of algorithm; Combinatorial problem; Packing

1. Introduction

Consider a set of items, a set of bin types, and infinite number of bins for each bin type. Let L be the list of n items and $w_j \in \mathbb{R}_+$ be the size of jth item, where $\mathbb{R}_+$ denotes the set of non-negative real numbers. Let T be a list of m types of bins. Let $c_i \in \mathbb{R}_+$ and $b_i \in \mathbb{R}_+$ be the cost and the size of ith type of bin, respectively. Without loss of generality, we assume that $b_1 \geqslant w_1$ and that L and T are sorted as $w_1 \geqslant w_2 \geqslant \cdots \geqslant w_n$ and $b_1 \geqslant b_2 \geqslant \cdots \geqslant b_m$.

The problem we investigate is assigning each item to one bin to minimize the total cost of used bins when the cost of unit size of each bin does not increase as the bin size increases, that is, the costs and the sizes of bins satisfy

$$\frac{c_{i_1}}{b_{i_1}} \leqslant \frac{c_{i_2}}{b_{i_2}} \quad \text{for all } 1 \leqslant i_1 < i_2 \leqslant m, \tag{1}$$

where $b_1 \geqslant b_2 \geqslant \cdots \geqslant b_m$. We call these monotonicity constraints and assume that all the costs and the sizes of bins satisfy these constraints from now on. Note that the objective of our problem is a generalization of minimizing the total size. This problem is referred to as "the variable sized bin packing problem".

To solve the classical bin packing problem, in which bins are of a single given size, Vance [8] proposed exact algorithm which is based on linear programming techniques that Gilmore and Gomory [4] had suggested. However, the

[*] Corresponding author. Tel.: +82-42-869-3121; fax: +82-42-869-3110.
E-mail address: sspark@kaist.ac.kr (S. Park).

algorithm takes too much time to be used practically. Therefore, heuristic algorithms for the classical bin packing problem have been studied extensively. Especially, the most well-known algorithms are First-Fit Decreasing (FFD) and Best-Fit Decreasing (BFD). Johnson et al. [6] showed that FFD and BFD guarantee asymptotic worst-case performance bounds of 11/9. For an excellent survey of heuristic algorithms for the classical bin packing problem, see [1]. By the way, only a few results have appeared concerning this more general problem in which bins do not need to be of a single given size. Moreover, the objectives of the problems that have been concerned are just to minimize the total bin space. Friesen and Langston [3] proposed three algorithms and showed that they guarantee asymptotic worst-case performance bounds of 2, 3/2, and 4/3, respectively. Murgolo [7] gave a fully polynomial time asymptotic approximation scheme for this problem. Coffman et al. [2] showed that FFD gives an optimal solution when $w_n|\cdots|w_1$ for the original bin packing problem, where for real numbers a and b, $b\,|\,a$ denotes that b exactly divides a. In addition, they extended the result to the variable sized bin packing problem in which the objective is to minimize the total bin size, and got the following result: a modified FFD algorithm gives an optimal solution when $w_n|\cdots|w_1$ and $w_j\,|\,b_i$ for all $i \in T$, $j \in L$ such that $w_j \leqslant b_i$. However, their result of extension is not correct. We show that with a counter example in Appendix A.

When general L and T are considered, it can be easily shown that this variable sized bin packing problem is NP-hard [3]. However, if L and T satisfy some divisibility constraints, this problem can be solved more easily.

In this paper, we consider two divisibility constraints as follows:

1. L has divisible item sizes, that is, w_{j+1} exactly divides w_j for all $j = 1,\ldots,n-1$,
2. T has divisible bin sizes, that is, b_{i+1} exactly divides b_i for all $i = 1,\ldots,m-1$.

We can find these divisibility constraints in Asynchronous Transfer Mode (ATM) Virtual Path (VP)-based leased line networks. In the networks, each demand is established by VP that has a pre-defined capacity, referred to as bandwidth, selected from a small set of alternatives. For example, there are seven alternatives 64 Kbps, 128 Kbps, 256 Kbps, 512 Kbps, 1.024 Mbps, 2.048 Mbps, and 45.056 Mbps in a network based on Synchronous Digital Hierarchy (SDH) transmission system. Note that the bandwidths have divisible sizes. In addition, there are two high-speed transmission rates (of link facilities) defined in SDH: STM-1(155.52 Mbps) and STM-4(622.08 Mbps) [5]. Note that the smaller transmission rate divides the larger one exactly. Furthermore, the installation costs of these link facilities satisfy the monotonicity constraints in general cases.

Throughout the paper, we use the following notation. $\mathbb{R}_+$ and $\mathbb{Z}_+$ are the set of non-negative real numbers and the set of non-negative integers, respectively. We let $\mathbb{Z}_+^m$ denotes a set of non-negative integer m-vectors. For every sets Π_i, $i = 1,\ldots,n$, we define $\bigcup_{i=v}^{w} \Pi_i \equiv \phi$ if $w < v$ and for real numbers $\tau_j, j = 1,\ldots,m$, we define $\sum_{j=v}^{w} \tau_j \equiv 0$ if $w < v$.

In the following section, we introduce our algorithms, named as "Iterative First-Fit Decreasing" and "Iterative Best-Fit Decreasing", respectively. We analyze the algorithms in three special cases: (a) the sizes of items and bins are divisible, respectively, (b) only the sizes of bins are divisible, and (c) the sizes of bins are not divisible. In Section 3, we show that each algorithm gives an optimal solution in the case (a). In Section 4, we also show that each of them gives a solution whose value is less than $\frac{11}{9}C(B^*) + 4\frac{11}{9}$, where $C(B^*)$ is the optimal value in the case (b). In Section 5, we show that each of them gives a solution whose value is less than $\frac{3}{2}C(B^*) + 1$ in the case of (c).

2. Algorithms

In this section, we introduce two algorithms based on two well-known algorithms, "First-Fit Decreasing" and "Best-Fit Decreasing", respectively. When the items are sorted as $w_1 \geqslant w_2 \geqslant \cdots \geqslant w_n$, the "First-Fit Decreasing Algorithm", FFD in short, considers them according to their order. In the first step, the algorithm initializes a bin with index 1. Then it assigns each item to the lowest indexed initialized bin into which it fits; only when the current item cannot fit into any initialized bin, a new bin is introduced with index $j+1$, where j is the number of initialized bins at hand. The "Best-Fit Decreasing", BFD in short, is a modification of FFD. It also considers the items in their order and assigns each item to the bin that has the smallest remaining capacity. Ties are broken by arbitrary selection. Assuming that $w_1 \leqslant b_1$, we introduce our greedy algorithms named "Iterative First-Fit Decreasing Algorithm", IFFD in short, and "Iterative Best-Fit Decreasing Algorithm", IBFD in short.

In IFFD (respectively, IBFD), we first allocate all the items to the largest size bins using FFD (respectively, BFD), and get a feasible solution. We then obtain another solution by repacking the items in the last bin of the solution to the next largest bins using FFD (respectively, BFD). Continuing this procedure until repacking of the items is impossible, we have some feasible solutions at hand. In this way, we can generally obtain a feasible solution for each type of bin. Then, the best solution among them is selected as the final solution with additional modification if necessary. The algorithms are formally stated as follows:

2.1. Algorithm: iterative first-fit decreasing (IFFD)

1. Allocate all the items into bins of type 1 by the first-fit decreasing manner. Let $B^1 = \langle B_1^1, B_2^1, \ldots, B_{k_1}^1 \rangle$ denote the ordered set of bins used in this step, and let C^1 denote the associated total cost $C(B^1)$. Let $l = 1$ and go to step 2.
2. If either $l = m$ or max $\{w_j : j \in B_{k_l}^l\} > b_{l+1}$, then go to step 4. Otherwise, let $l = l + 1$ and go to step 3.
3. Allocate all the items allocated in $B_{k_{l-1}}^{l-1}$ to bins of type l by the first-fit decreasing manner. Let $B^l = \langle B_1^l, B_2^l, \ldots, B_{k_l}^l \rangle$ denote the ordered set of bins used in this step, and let C^l denote $\sum_{i=1}^{l-1} C(B^i - \{B_{k_i}^i\}) + C(B^l)$. Go to step 2.
4. Let $i^* = \arg\min_{1 \leqslant i \leqslant l} C^i$.
5. For each used bin in $\bigcup_{l=1}^{i^*-1} (B^l - \{B_{k_l}^l\}) \bigcup B^{i^*}$, if the total size of the items in the bin is small enough to use a smaller sized bin, use the cheapest bin that can contain the items. Then, we let B^* denote the resulting set of bins.

Here, B^* is the solution obtained by the algorithm and $C(B^*)$ is the total cost of the solution. Note that there is no bin that can be replaced by smaller one in step 5 if $b_m | \cdots | b_1$. Step 5 is intended only for the problem instances in which the list of bin types does not have divisible bin sizes.

"Iterative Best-Fit Decreasing Algorithm" uses BFD instead of FFD. In addition, if multiple bins have the same smallest remaining capacities, it selects the smallest indexed one among them while it allocates the items by BFD manner.

3. Analysis for the case that *L* has divisible item sizes and *T* has divisible bin sizes

In the case that a list L of items has divisible item sizes and a list T of bin types has divisible bin sizes, if the costs and the sizes of bin types in T satisfy the monotonicity constraints, IFFD and IBFD give optimal solutions. In this section, we show this with some steps of proofs. One can easily understand that the resulting solution of IBFD is the same as that of IFFD when L has divisible item sizes and T has divisible bin sizes. Therefore, it is enough to show that IFFD gives an optimal solution.

For ease of analysis, we let $y = (y_1, y_2, \ldots, y_m)$ denote a feasible solution to the problem, where $y_i \in \mathbb{Z}_+$ denotes the number of used bins of type i in the solution. In addition, we let $M(L, T)$ denote the set of the feasible solutions to the problem, when a list L of items and a list T of bin types are considered.

Proposition 1. *Suppose a list T of bin types has divisible bin sizes. If $b_1 \leqslant \sum_{j=1}^n w_j$, then the optimal objective value of the problem* min $\{\sum_{i=2}^m c_i y_i : y \in M(L, T - \{1\})\}$ *is greater than or equal to the optimal objective value of the problem* min $\{\sum_{i=1}^m c_i y_i : y \in M(L, T) \cap \{y : y_1 \geqslant 1\}\}$.

To prove this proposition, we introduce the following lemma.

Lemma 1. *Suppose T has divisible bin sizes. Then, for any feasible solution $y \in M(L,T)$ such that $\sum_{i=2}^{m} b_i y_i \geqslant b_1$, there exist two positive integers: $s, 2 \leqslant s \leqslant m$, and y'_s, such that $\sum_{i=2}^{s-1} b_i y_i + b_s y'_s = b_1$ and $0 < y'_s \leqslant y_s$.*

Proof. We always can get s that satisfies two conditions: (i) $\sum_{i=2}^{s} b_i y_i \geqslant b_1$ and (ii) $\sum_{i=2}^{s-1} b_i y_i < b_1$. We can rewrite (i) as

$$\begin{aligned} &\sum_{i=2}^{s-1} b_i y_i + b_s y_s \geqslant b_1 \\ &\quad \Longleftrightarrow b_s y_s \geqslant b_1 - \sum_{i=2}^{s-1} b_i y_i \\ &\quad \Longleftrightarrow y_s \geqslant \frac{b_1}{b_s} - \sum_{i=2}^{s-1} \frac{b_i}{b_s} y_i. \end{aligned}$$

From (ii) and the fact that $b_m | \cdots | b_1$, the right-hand side is positive and integral. Let $y'_s = b_1/b_s - \sum_{i=2}^{s-1} (b_i/b_s) y_i$. □

Proof (Proposition 1). For each vector $y \in M(L, T - \{1\})$, from Lemma 1, we can get two integers, s and y'_s, which satisfy the two conditions: $\sum_{i=2}^{s-1} b_i y_i + b_s y'_s = b_1$ and $\sum_{i=2}^{s-1} c_i y_i + c_s y'_s \geqslant c_1$, where $0 < y'_s \leqslant y_s$ because b_i's and c_i's satisfy the monotonicity constraints. Consider a vector $y^* \in \mathbb{Z}_+^m$ such that $y_1^* = 1$, $y_k^* = 0$, for each $k \in [2, \ldots, s-1]$, $y_s^* = y_s - y'_s$, and $y_k^* = y_k$, for each $k \in [s+1, \ldots, m]$. We can see $y^* \in M(L,T)$ and $c^{\mathrm{T}} y^* \leqslant c^{\mathrm{T}} y$. □

From Proposition 1, the next corollary follows.

Corollary 1. *If T has divisible bin sizes, there exists an optimal solution $y \in M(L,T)$ such that $\sum_{i=s+1}^{m} b_i y_i < b_s$ and $\sum_{i=s+1}^{m} c_i y_i < c_s$, for all $1 \leqslant s < m$.*

Proof. We prove this corollary by constructing such a solution. First, we suppose that there is an optimal solution y which violates the first property. Let s be the smallest index which violates $\sum_{i=s+1}^{m} b_i y_i < b_s$. Since deletion of the items allocated to the bins not smaller than b_s does not change the packing of the remaining items, we can get another optimal solution by generating a bin packing of the remaining items without increasing the objective value as follows. We let L' and T' be the list of items allocated to the bins smaller than b_s and the set of bin types not greater than b_s ($\{s, \ldots, m\}$), respectively. Then, there is an optimal solution $\hat{y} \in M(L', T')$ using at least one bin of type s from Proposition 1. Moreover, the solution obtained from y by changing the packing of the items L' with $\hat{y}$ is also an optimal solution. Recursively using this procedure, we generate an optimal solution $\bar{y}$ satisfying the first property. Moreover, there exists an optimal solution which also satisfies the second property. We can show this with a similar way as follows: If $\bar{y}$ satisfies the first property but not the second, we select the smallest index s such that $\sum_{i=s+1}^{m} c_i \bar{y}_i \geqslant c_s$. Then, we can generate another optimal solution $\tilde{y}$ such that $\tilde{y}_i = \bar{y}_i$ for $i = 1, \ldots, s-1$, $\tilde{y}_s = \bar{y}_s + 1$, and $\tilde{y}_i = 0$ for $i = s+1, \ldots, n$, because $\sum_{i=s+1}^{m} b_i \bar{y}_i < b_s$ and b_i's and c_i's satisfy the monotonicity constraints. □

The following theorem is due to Coffman et al. [2].

Theorem 1. *When L has divisible item sizes, if only one type of bin is considered, FFD gives an optimal solution.*

Corollary 2. *If L has divisible item sizes and T has divisible bin sizes, there exists an optimal solution using k_1 or $k_1 - 1$ bins of type 1, where k_1 is the number of bins used in the first step of IFFD.*

Proof. There exists an optimal solution y' such that $\sum_{i=2}^{m} b_i y'_i < b_1$ from Corollary 1. Let $\tilde{L}$ be the set of items that are allocated to the bins of types in $T - \{1\}$. If $\tilde{L} \neq \phi$, $\tilde{L}$ can be packed into a bin of type 1, because $\sum_{i=2}^{m} b_i y'_i < b_1$. After packing $\tilde{L}$ into a bin of type 1, there are $y'_1 + 1$ bins of type 1. If $\tilde{L} = \phi$, there are y'_1 bins of type 1. After this packing, the number of type 1 bins is equal to k_1, the number of bins in an optimal solution in $M(L, \{1\})$ from Theorem 1. Therefore, we have that $y'_1 = k_1$ or $k_1 - 1$. □

Proposition 2. *Suppose L has divisible item sizes and T has divisible bin sizes, and we obtain a solution using IFFD. If $k_1 \geqslant 2$, there exists an optimal* *solution containing $\langle B_1^1, \ldots, B_{k_1-1}^1 \rangle$, where $\langle B_1^1, \ldots, B_{k_1}^1 \rangle$ is the ordered set of bins used in the first step of IFFD.*

To prove this proposition, we introduce a known fact due to Coffman et al. [2].

Fact 1. *If L has divisible item sizes, then any set of items S from L such that $w_s \leqslant w_j$ for $s \in S$ and $\sum_{s\in S} w_s \geqslant w_j$ for an item $j \in L$ contains a subset of items $\hat{S} \subseteq S$ such that $\sum_{s\in\hat{S}} w_s = w_j$.*

Proof (Proposition 2). We show that there exists an optimal solution containing a bin packed identically to the first bin, B_1^1 when $k_1 \geqslant 2$. Since the deletion of the items in that bin does not change the IFFD packing of the remaining items, the desired result follows immediately by induction on $B_1^1, \ldots, B_{k_1-1}^1$.

We show the existence of such an optimal solution by describing how to generate it. There exists an optimal packing containing at least $k_1 - 1$ bins of type 1 when $k_1 \geqslant 2$ from Corollary 2. Moreover, there exists an optimal solution in which item 1 is packed in a bin of type 1. We show this by constructing such a solution.

Consider an optimal solution in which bins of type 1 are used but they do not contain item 1. Note that there exists an optimal solution which uses type 1 bins when $k_1 \geqslant 2$. If the solution contains a bin of type 1 such that the items in it sum to at most w_1, we let B be such a bin and let L_1 be the set of items in B. Otherwise, all bins of type 1 contain items whose total size is larger than w_1. In this case, we arbitrarily select a bin of type 1 and let B denote it. From Fact 1, we always can find a set of items in B such that the total size of them is exactly equal to w_1. We let L_1 denote such a set of items. In the above two cases, we exchange the allocation of L_1 and item 1 to obtain another solution. Now B contains item 1 and such an exchange does not increase the objective value of the solution.

Next, we present the procedure which makes B identical to B_1^1. If B is different from B_1^1, we choose the item having the smallest index s among the items currently in B_1^1, but not in B. Note that the size of the largest item among the items in B, but not in B_1^1 is no larger than w_s because B_1^1 is the first bin obtained by FFD. If the total size of the items in B, the size of which individually is smaller than w_s, is also smaller than w_s, we let L_s denote the set of items. Otherwise, we can always find a subset L_s of items in B the sizes of which individually is smaller than w_s and which in total sum to exactly w_s from Fact 1. We can make B contain the item s by interchanging the allocation of item s and L_s without increasing the objective value. We do this procedure recursively until B contains all the items in B_1^1. Note that, after this procedure, B does not contain any item not in B_1^1 because B_1^1 is the first bin obtained by FFD, and therefore there is no room in B_1^1 to contain any other items. Hence, there exists an optimal solution containing a bin packed identically to the first IFFD bin. The proposition follows. □

When L has divisible item sizes and T has divisible bin sizes, $O^*(L,T) = C(B^1 - \{B_{k_1}^1\}) + O^* \times (B_{k_1}^1, T)$ by Proposition 2, where $O^*(L,T)$ denotes the cost of an optimal packing when L and T are considered. Since deletion of $B^1 - \{B_{k_1}^1\}$ does not change the IFFD packing of the remaining items, we have that $O^*(L_i, T_i) = C(B^i - \{B_{k_i}^i\}) + \min\{C(B_{k_i}^i), O^*(L_{i+1}, T_{i+1})\}$, $i = 1, \ldots, m-1$, where $L_1 = L$, $T_1 = T$, $T_{i+1} = \{i+1, \ldots, m\}$, L_{i+1} denotes the set of items in $B_{k_i}^i$, and B^i is the solution obtained by packing the items in L_i into the bins of type i by FFD manner. We can get the following theorem.

Theorem 2. *IFFD and IBFD give optimal solutions, when L has divisible item sizes and T has divisible bin sizes.*

4. Analysis for the case that only *T* has divisible bin sizes

In this section, we analyze the performances of IFFD and IBFD when only T has divisible bin sizes. When L has general item sizes, the bin packing problem is NP-hard, so it seems that simple greedy algorithms such as IFFD and IBFD cannot guarantee optimality. Therefore, it is reasonable to analyze the worst case performance bound of IFFD and IBFD.

From now on, for ease of analysis, we normalize the costs and the sizes of the bins so that $c_1 = b_1 = 1$. Considering only one type of bins, that is, $T = \{1\}$, let B_{FFD} and B_{BFD} denote the solutions obtained by FFD and BFD, respectively, and let B_1^* denote an optimal solution. When multiple bin types are considered, let B_{IFFD} and B_{IBFD} denote the solutions obtained by IFFD and IBFD, respectively, and let B^* denote an optimal solution.

Johnson et al. [6] showed that when only one type of bins are considered, FFD and BFD guarantee asymptotic worst-case performance bound of 11/9 by the following theorems.

Theorem 3. *There exists a list L of items such that* $C(B_{\text{FFD}}) = C(B_{\text{BFD}}) > \frac{11}{9}C(B_1^*) - 2$.

Theorem 4. *For all lists L of items,* $C(B_{\text{FFD}}) \leqslant \frac{11}{9}C(B_1^*) + 4$, *and* $C(B_{\text{BFD}}) \leqslant \frac{11}{9}C(B_1^*) + 4$.

The main result is the following.

Theorem 5. *For all L, if T has divisible bin sizes, then* $C(B_{\text{IFFD}}) < \frac{11}{9}C(B^*) + 4\frac{11}{9}$, *and* $C(B_{\text{IBFD}}) < \frac{11}{9}C(B^*) + 4\frac{11}{9}$.

Proof. The proofs of the both inequalities are the same. Therefore, we only show that $C(B_{\text{IFFD}}) < \frac{11}{9}C(B^*) + 4\frac{11}{9}$. We know that $C(B_{\text{FFD}}) \leqslant \frac{11}{9}C(B_1^*) + 4$ and that $C(B_{\text{IFFD}}) \leqslant C(B_{\text{FFD}})$. Moreover, $C(B_1^*) < C(B^*) + 1$ from Corollary 1 because $c_1 = b_1 = 1$. Therefore, we can conclude that $C(B_{\text{IFFD}}) \leqslant C(B_{\text{FFD}}) \leqslant \frac{11}{9}C(B_1^*) + 4 < \frac{11}{9}C(B^*) + 4\frac{11}{9}$. Therefore, $C(B_{\text{IFFD}}) < \frac{11}{9}C(B^*) + 4\frac{11}{9}$. □

When $T = \{1\}$, B_{IFFD} is identical to B_{FFD}. Therefore, there exists a problem instance such that $C(B_{\text{IFFD}}) = C(B_{\text{IBFD}}) > \frac{11}{9}C(B^*) - 2$ from Theorem 3. Therefore, the performance bound of 11/9 is tight.

5. Analysis for the general case

In this section, we analyze the performances of IFFD and IBFD for the general case, i.e. when T does not have divisible bin sizes. The main results about IFFD and IBFD are the following.

Theorem 6. $C(B_{\text{IFFD}}) < \frac{3}{2}C(B^*) + 1$ *and* $C(B_{\text{IBFD}}) < \frac{3}{2}C(B^*) + 1$ *for all L and T.*

Proof. Because the proofs for the two inequalities are the same, we just give a proof for the first inequality. Consider a solution $B = \langle B_1, \ldots, B_{k_1}\rangle$ obtained from B_{FFD} after performing the step (5) in IFFD. We have that $C(B) \geqslant C(B_{\text{IFFD}})$. Therefore, it is enough to show that $C(B) < \frac{3}{2}C(B^*) + 1$. We prove this by deriving contradiction. Friesen and Langston [3] also used a similar procedure to show that the performance bound of their second algorithm is $\frac{3}{2}$.

Without loss of generality we assume that $c_m < \cdots < c_1 = b_1 = 1$. Let $\theta(B_k)$, $b(B_k)$, and $c(B_k)$ be the total size of items packed into a bin B_k, the size of the bin, and the cost of the bin, respectively. Note that $\theta(B_k) \leqslant b(B_k) \leqslant c(B_k)$ and $C(B^*) \geqslant \sum_{k=1}^{k_1}\theta(B_k)$ from the monotonicity constraints.

Assume that there exist L and T such that $C(B) \geqslant \frac{3}{2}C(B^*) + 1$. Let k' be the largest index $k < k_1$ such that $\theta(B_k) < \frac{2}{3}c(B_k)$. Such a k' exists. Otherwise, $\sum_{k=1}^{k_1}\theta(B_k) + \frac{2}{3} > \frac{2}{3}\sum_{k=1}^{k_1}c(B_k)$ and it contradicts the assumption because $C(B^*) \geqslant \sum_{k=1}^{k_1}\theta(B_k)$. Because $k' < k_1$, we have that $\frac{1}{2} < \theta(B_{k'})$. Otherwise, any item in the $k'+1$th bin of B_{FFD} can be packed into the k'th bin. This contradicts that B is obtained from B_{FFD}. Hence, we have $\frac{1}{2} < \theta(B_{k'}) < \frac{2}{3}$ and $|B_{k'}| = 1$. Otherwise, that is, if $|B_{k'}| \geqslant 2$, $B_{k'}$ includes an item whose size is less than $\frac{1}{3}$, and this means that $\theta(B_k) > \frac{2}{3}$, for all $k < k'$ since B is obtained from B_{FFD}. Then, we have

$$C(B^*) \geqslant \sum_{k=1}^{k'-1}\theta(B_k) + \theta(B_{k'}) + \sum_{k=k'+1}^{k_1-1}\theta(B_k) + \theta(B_{k_1})$$

(since $\theta(B_{k'}) + \theta(B_{k_1}) > 1$)

$$> \frac{2}{3}(k'-1) + \frac{2}{3}\sum_{k=k'+1}^{k_1-1}c(B_k) + \left(\frac{2}{3} + \frac{1}{3}\right)$$

$$> \frac{2}{3}k' + \frac{2}{3}\sum_{k=k'+1}^{k_1-1}c(B_k) + \frac{1}{3}$$

$$> \frac{2}{3}\sum_{k=1}^{k'}c(B_k) + \frac{2}{3}\sum_{k=k'+1}^{k_1-1}c(B_k) + \frac{2}{3} - \frac{1}{3}$$

$$> \frac{2}{3}\sum_{k=1}^{k_1}c(B_k) - \frac{1}{3} = \frac{2}{3}C(B) - \frac{1}{3}.$$

This contradicts the assumption.

Because B_k's are obtained after performing the step (5) of IFFD, $c(B_k) \leqslant c_i$, for all i such that $b_i \geqslant \theta(B_{k'})$. We have that $\frac{3}{4} < c_i$ for each i such that $b_i \geqslant \theta(B_{k'})$ because $\frac{1}{2} < \theta(B_{k'}) < \frac{2}{3}c(B_{k'})$ and $\frac{2}{3}c(B_{k'}) \leqslant \frac{2}{3}c_i$. In addition, for each $k \leqslant k'$, B_k contains an item whose size is at least $\theta(B_{k'})$ and which cannot be packed in a bin of size 1 with any item from B_j, for each $j > k'$. Therefore, $B^* = B_1^* + B_2^*$, where B_1^* is an optimal packing of the items packed in $\langle B_1, \ldots, B_{k'} \rangle$ and B_2^* is an optimal packing of the items packed in $\langle B_{k'+1}, \ldots, B_{k_1} \rangle$. Finally, we have

$$\begin{aligned} C(B) &\leqslant k' + \frac{3}{2} \sum_{k=k'+1}^{k_1} \theta(B_k) + 1 \\ &< \frac{3}{2} \left[\frac{3}{4}k' + \sum_{k=k'+1}^{k_1} \theta(B_k) \right] + 1 \\ &\leqslant \frac{3}{2} \left[C(B_1^*) + C(B_2^*) \right] + 1 \\ &\leqslant \frac{3}{2} C(B^*) + 1. \end{aligned}$$

This contradicts the assumption. Therefore, $C(B) < \frac{3}{2}C(B^*) + 1$ for all L and T. The theorem follows from this. □

Consider the following problem instance. $T = \{1, 2\}$, $c_1 = b_1 = 1$, $c_2 = b_2 = \frac{1}{3} + \epsilon$, $L = \{1, \ldots, 2m\}$, $w_1 = w_2 = \cdots = \frac{1}{3} + \epsilon$, where ϵ is very small positive number. This shows that the asymptotic bound of 3/2 is tight. Even though L has divisible item sizes, the bound cannot be improved.

6. Conclusion

We proposed two algorithms, IFFD and IBFD, for the variable sized bin packing problem to minimize the cost. We considered bins whose sizes and costs satisfy monotonicity constraints. The algorithms give good solutions when the items and the bins have some divisibility constraints. First, when the sizes of items and the sizes of bins are divisible, the algorithms give optimal solutions. Second, when only the size of bins are divisible, the algorithms guarantee asymptotic worst-case performance bound 11/9, which is the same as the performance bound of FFD and BFD. It is known that the performance bounds of FFD and BFD decrease as the size of largest item decreases [1]. Therefore, we expect that the performance bounds of IFFD and IBFD also decrease as the size of largest item decreases. Finally, when the sizes of bins are not divisible, the algorithms guarantee asymptotic worst-case performance bound of 3/2.

Appendix A

Here, we show that the result of Coffman's extension to the variable sized bin packing problem is not correct. The modified First-Fit Decreasing algorithm [2] is described as follows.

First, pack items by FFD into bins of size b_1 until the remaining, unpacked items sum to less than b_1, or until all items are packed. If items remain and the largest exceeds b_2, the algorithm places the remaining items into a bin of size b_1 and then halts. Otherwise, the above procedure is applied recursively to the remaining items and bins of sizes $b_2, b_3, \ldots$

They insisted that this algorithm gives an optimal solution to the variable sized bin packing problem when $w_n | \cdots | w_1$ and $w_j | b_i$ for all $i \in T$, $j \in L$ such that $w_j \leqslant b_i$.

Now, we show that their algorithm cannot guarantee optimality with a counter example. Consider the following problem instance. There are two bin capacities, 6 and 4, and there are eight items of size 1. By the modified first-fit decreasing algorithm, we get a solution in which there are two bins used. One has size 6 and packs 6 items. The other has size 4 and packs 2 items. However, in the optimal solution, two bins of size 4 are used. Therefore, we can say that the modified first-fit decreasing algorithm cannot guarantee optimality.

References

[1] E.G. Coffman, M.R. Garey, D.S. Johnson, Approximation algorithms for bin packing – an updated survey, in: G. Ausiello, M. Lucertini, P. Serafini (Eds.), Algorithm Design for Computer System Design, Springer, Vienna, 1984, pp. 49–106.

[2] E.G. Coffman, M.R. Garey, D.S. Johnson, Bin packing with divisible item sizes, J. Complexity 3 (1987) 406–428.

[3] D.K. Friesen, M.A. Langston, Variable sized bin packing, SIAM J. Comput. 15 (1) (1986) 222–230.

[4] P.C. Gilmore, R.E. Gomory, A linear programming approach to the cutting stock problem, Oper. Res. 9 (1961) 849–859.

[5] ITU-T recommendation. E.735, Framework for traffic control and dimensioning in B-ISDN, 1997.

[6] D.S. Johnson, A. Demers, J.D. Ullman, M.R. Garey, R.L. Graham, Worst-case performance bounds for simple one-dimensional packing algorithms, SIAM J. Comput. 3 (4) (1974) 299–325.

[7] F.D. Murgolo, An efficient approximation scheme for variable-sized bin packing, SIAM J. Comput. 16 (1) (1987) 149–161.

[8] P.H. Vance, Branch-and-price algorithms for the one-dimensional cutting stock problem, Comput. Optimization Appl. 9 (3) (1998) 211–228.