ELSEVIER

Discrete Applied Mathematics 94 (1999) 181–191

DISCRETE APPLIED MATHEMATICS

Approximation algorithms for partitioning small items in unequal bins to minimize the total size

P. Dell'Olmo[a], M. Grazia Speranza[b,*]

[a] *Department of Comp. Sc., Sys. and Prod., University of Rome 'Tor Vergata' Via della Ricerca Scientifica, I-00133 Rome, Italy*

[b] *Department of Quantitative Methods, University of Brescia, Contrada S. Chiara, 48/b - I-25122 Brescia, Italy*

Received 28 November 1996; revised 29 January 1998; accepted 8 May 1998

Abstract

A set of items has to be assigned to a set of bins with different sizes. If necessary the size of each bin can be extended. The objective is to minimize the total size, i.e. the sum of the sizes of the bins. In this paper we study both the off-line case and the on-line variant of this problem under the assumption that each item is smaller than any bin. For the former case, when all times are known in advance, we analyze the worst-case performance of the longest processing time heuristic and prove a bound of $2(2-\sqrt{2})$. For the on-line case, where each incoming item has to be assigned immediately to a bin and the assignment cannot be changed later, we give a lower bound of $\frac{7}{6}$ on the worst-case relative error of any on-line algorithm with respect to the off-line problem and we show that a list scheduling algorithm, which assigns the incoming item to the bin with biggest idle space, has a worst-case performance ratio equal to $\frac{5}{4}$. This bound is shown to be tight.

Keywords: Bin-packing; Multiprocessor scheduling; Approximation; On-line algorithms; Worst-case performance

1. Introduction

In the variable sized bin-packing problem, a list of items and an inexhaustible set of unequal bins are given and one is asked to pack the items into the minimum number of bins. Bins have different, but fixed, sizes, i.e. the total size of the items packed in any single bin does not have to exceed the bin size (see for example [2,3,5]).

In the problem we investigate here, the number of available bins is limited but their sizes are *extendible*, that is, the total size of the items packed into a single bin can exceed it, if necessary. One is asked to minimize the total size of the bins.

* Corresponding author.

0166-218X/99/$ - see front matter
PII: S0166-218X(99)00020-7

This particular bin-packing problem arises in a variety of scheduling and storage-allocation problems. As a storage-allocation problem, our model may be applied in any case in which extra space can be obtained from a fixed set of locations (bins) of different size at a given cost and we are asked to minimize the total cost. As a scheduling model consider a problem with two dedicated machines where two types of tasks have to be executed. Tasks of the first type require both machines for some time, and then only one machine for an additional processing time. Tasks of the second type require execution on the second machine only. The objective function is the minimization of the makespan. Another application of the same model is the scheduling of duties within workers' regular time. In case of need, the workers can make some overtime work (exceeding regular time). The problem is that of assigning duties to workers in such a way that the total overtime work is minimized. It is easy to be seen with a reduction from 3-partition that this problem is strongly NP-hard. Therefore, approximation algorithms should be applied which can find near-optimal solutions in reasonable time. The special case with equal bins has been recently investigated in [1], where the authors study the (off-line) problem, and in [6] where the on-line variant of the problem is considered.

In this paper, we analyze the general packing problem with unequal bins, under the assumption that each item is smaller than any bin, in both the off-line case, in which all items are known in advance, and the on-line variant, in which unknown items arriving one-by-one must be assigned immediately to a bin and the assignment cannot be changed later.

For the off-line problem, we will investigate the worst-case performance of the well-known *longest processing time* (LPT) algorithm. This heuristic was introduced by Graham [4] as an approximation to the multiprocessor scheduling problem with the objective of minimizing the makespan. LPT sorts the items in non-increasing order of their processing times into a list and assigns at each iteration the first item of the list to a bin with biggest idle space. The time complexity of LPT is $O(n(\log m + \log n))$, where m denotes the number of bins and n the number of items. In Section 3, we will show that it yields a worst-case performance of $2(2-\sqrt{2})$.

The on-line variant is examined is Section 4, where we first show a lower bound of $\frac{7}{6}$ on the worst-case relative error of any on-line algorithm, with respect to the off-line optimum. Secondly, we analyze the performance of the *list scheduling* (LS) algorithm, which assigns the incoming item to the bin with biggest idle space. The time complexity of LS is $O(n \log m)$. We show that the LS algorithm has a worst-case error equal to $\frac{5}{4}$. This bound is shown to be tight.

2. Notation and problem definition

We are given m bins $B_1,\ldots,B_m$, with size $b_1,\ldots,b_m$. The items to be assigned to the bins constitute a set $\mathcal{T}$ of n elements of length t_i. Since an item is characterized by its length, we will identify it with its length t_i. It is assumed that

$$t_i \leqslant \min_{1\leqslant j\leqslant m} b_j, \quad i=1,\ldots,n. \tag{1}$$

The *load* ℓ_j of a bin B_j is just the length of the items contained in B_j. The *size* of a bin B_j is defined by $\max\{\ell_j, b_j\}$. The objective is to minimize the total size of the bins.

Consider the solution given by any heuristic algorithm. In general, some of the bins are covered with possibly an item which partially exceeds the original size of the bin, while some other bins still have idle space. Given a heuristic solution, we denote by: T_k the set of items which exceed a bin; $k = |T_k|$ the number of items which exceed bins (that is the number of exceeded bins); T_δ the set of items which do not exceed a bin and *assigned after* an item has exceeded a bin; $\delta = |T_\delta|$; $T_{n-k-\delta}$ the set of items *assigned before* an item exceeds a bin. Note that $\mathcal{T} = T_k \cup T_\delta \cup T_{n-k-\delta}$. Moreover, let us denote by: r_j the idle space of an exceeded bin B_j, before the last item has been assigned to B_j; e_j the exceeding part of an exceeded bin B_j; s_j the idle space of a non-exceeded bin B_j.

We renumber the bins so that the first k bins are exceeded. Observe that in this analysis we can assume $k < m$, otherwise it can be easily verified that both the proposed algorithms find an optimal solution of value equal to $\sum_{i=1}^{n} t_i$. This assumption will be implicitly used in some proofs of the following sections.

3. An approximation result for the LPT heuristic

In this section, the worst-case behavior of LPT for our problem will be analyzed. We will prove a performance ratio strictly less than $2(2-\sqrt{2})$.

Let us denote with $\mathcal{LPT}(I)$ the value of the heuristic solution for instance I, and $\mathcal{O}(I)$ the value of the optimal solution for instance I.

Lemma 1. *The first m items assigned by the heuristic LPT do not exceed any bin*; i.e. $n-k-\delta \geqslant m$.

Proof. As the heuristic LPT assigns the largest item to the largest idle space, when the first exceeding item t_l is assigned, no idle space greater than or equal to t_l is available. This implies that, by assumption (1), i.e. $b_j \geqslant t_l$, $j = 1, \ldots, m$, in that step of the algorithm there is no empty bin. Hence, $n-k-\delta$, that is the number of items assigned before the first item t_l has exceeded a bin, is greater than or equal to m. □

Remark 1. As the heuristic LPT assigns items to the largest idle space it holds

$$\frac{\sum_{j=1}^{k} r_j}{k} \geqslant \min_{j=1,\ldots,k} r_j \geqslant \max_{j=k+1,\ldots,m} s_j \geqslant \frac{\sum_{j=k+1}^{m} s_j}{m-k},$$

then

$$(m-k)\sum_{j=1}^{k} r_j \geqslant k \sum_{j=k+1}^{m} s_j. \tag{2}$$

Theorem 1. *For all problem instances I,*

$$\frac{\mathscr{LPT}(I)}{\mathscr{C}(I)} \leqslant 1 + \frac{(m-k)k}{(n-\delta)m}.$$

Proof. Three different cases will be examined separately.

Case (a):

$$\sum_{j=1}^{m} b_j = \sum_{i=1}^{n} t_i.$$

In this case: $\sum_{j=1}^{k} e_j = \sum_{j=k+1}^{m} s_j$. Then, due to (2)

$$(m-k)\sum_{j=1}^{k} r_j \geqslant k \sum_{j=1}^{k} e_j.$$

Adding $(m-k)\sum_{j=1}^{k} e_j$ to both right and left members of the above inequality, we obtain

$$(m-k)\sum_{j=1}^{k} (r_j + e_j) \geqslant m \sum_{j=1}^{k} e_j.$$

As $(r_j + e_j)$ is the length of the item which exceeds bin B_j, the above inequality can be rewritten as

$$(m-k) \sum_{t_i \in T_k} t_i \geqslant m \sum_{j=1}^{k} e_j. \tag{3}$$

The exceeding items are smaller than or equal to the items assigned before the first item exceeds a bin; therefore

$$\frac{\sum_{t_i \in T_k} t_i}{k} \leqslant \max_{t_i \in T_k} t_i \leqslant \min_{t_i \in T_{n-k-\delta}} t_i \leqslant \frac{\sum_{t_i \in T_{n-k-\delta}} t_i}{n-k-\delta}.$$

Hence

$$(n-k-\delta) \sum_{t_i \in T_k} t_i \leqslant k \sum_{t_i \in T_{n-k-\delta}} t_i.$$

Adding $k(\sum_{t_i \in T_k} t_i + \sum_{t_i \in T_\delta} t_i)$ to both members of the previous inequality, we obtain

$$(n-\delta) \sum_{t_i \in T_k} t_i + k \sum_{t_i \in T_\delta} t_i \leqslant k \sum_{i=1}^{n} t_i,$$

$$(n-\delta) \sum_{t_i \in T_k} t_i \leqslant k \sum_{i=1}^{n} t_i. \tag{4}$$

By combining (3) with (4), it follows that

$$\sum_{i=1}^{n} t_i \geqslant \frac{n-\delta}{k} \sum_{t_i \in T_k} t_i \geqslant \frac{(n-\delta)m}{(m-k)k} \sum_{j=1}^{k} e_j,$$

then

$$\mathscr{LPT}(I) = \sum_{j=1}^{m} b_j + \sum_{j=1}^{k} e_j \leqslant \sum_{j=1}^{m} b_j + \frac{(m-k)k}{(n-\delta)m} \sum_{i=1}^{n} t_i, \tag{5}$$

$$\mathscr{C}(I) \geqslant \sum_{j=1}^{m} b_j.$$

$$\frac{\mathscr{LPT}(I)}{\mathscr{C}(I)} \leqslant 1 + \frac{(m-k)k}{(n-\delta)m},$$

due to the assumption of the *Case* (a): $\sum_{j=1}^{m} b_j = \sum_{i=1}^{n} t_i$.

Case (b):

$$\sum_{j=1}^{m} b_j > \sum_{i=1}^{n} t_i.$$

In this case, we can consider a new set $\tilde{\mathscr{T}}$, such that $\forall \tilde{t}_i \in \tilde{\mathscr{T}}$ the following four conditions are satisfied:

(i) $\tilde{t}_i \leqslant \min_{t_i \in \mathscr{T}} t_i$;
(ii) $\sum_{j=1}^{m} b_j = \sum_{t_i \in \mathscr{T}} t_i + \sum_{\tilde{t}_i \in \tilde{\mathscr{T}}} \tilde{t}_i$;
(iii) $\tilde{t}_i$ does not exceed any bin in the heuristic solution;
(iv) $\tilde{t}_i$ is smaller than any idle space in the optimal solution.

Under the above conditions, we have

$$\hat{n} = |\mathscr{T} \cup \tilde{\mathscr{T}}|, \quad \hat{\delta} = |\mathscr{T}_\delta \cup \tilde{\mathscr{T}}_\delta| = |\mathscr{T}_\delta \cup \tilde{\mathscr{T}}|,$$

hence

$$\hat{n} \geqslant n, \quad \hat{\delta} = \delta + (\hat{n} - n) \geqslant \delta, \quad \hat{n} - \hat{\delta} = n - \delta,$$

$$\frac{(m-k)k}{(\hat{n}-\hat{\delta})m} = \frac{(m-k)k}{(n-\delta)m}.$$

If $\hat{I}$ denotes the new instance obtained by the same bins of I and the new set of items $\mathscr{T} \cup \tilde{\mathscr{T}}$, from (iv) it follows that

$$\mathscr{C}(\hat{I}) = \mathscr{C}(I).$$

Moreover, by (5) obtained for the *Case* (a):

$$\mathscr{LPT}(I) = \mathscr{LPT}(\hat{I}) \leqslant \sum_{j=1}^{m} b_j + \frac{(m-k)k}{(\hat{n}-\hat{\delta})m} \left(\sum_{t_i \in \mathscr{T}} t_i + \sum_{\tilde{t}_i \in \tilde{\mathscr{T}}} \tilde{t}_i \right)$$

$$= \sum_{j=1}^{m} b_j + \frac{(m-k)k}{(n-\delta)m} \left(\sum_{t_i \in \mathscr{T}} t_i + \sum_{\tilde{t}_i \in \tilde{\mathscr{T}}} \tilde{t}_i \right).$$

For (ii):

$$\frac{\sum_{t_i \in \mathscr{T}} t_i + \sum_{\tilde{t}_i \in \tilde{\mathscr{T}}} \tilde{t}_i}{\sum_{j=1}^{m} b_j} = 1,$$

hence

$$\frac{\mathscr{LPT}(I)}{\mathscr{O}(I)} = \frac{\mathscr{LPT}(\hat{I})}{\mathscr{O}(\hat{I})} \leqslant \frac{\sum_{j=1}^{m} b_j + \left[\frac{(m-k)k}{(n-\delta)m}\right] \left(\sum_{t_i \in \mathscr{T}} t_i + \sum_{\tilde{t}_i \in \tilde{\mathscr{T}}} \tilde{t}_i\right)}{\sum_{j=1}^{m} b_j}$$

$$= 1 + \frac{(m-k)k}{(n-\delta)m}.$$

Case (c):

$$\sum_{j=1}^{m} b_j < \sum_{i=1}^{n} t_i.$$

Let us consider a new instance I' obtained by the same bins of I and the biggest items $t'_i \in \mathscr{T}$, whose sum is equal to $\sum_{j=1}^{m} b_j$ (if necessary, cut an item in two parts and consider only one of them). Let $\mathscr{T}'$ be the new set of items. As the new instance falls in *Case* (a), due to (5)

$$\mathscr{LPT}(I') \leqslant \sum_{j=1}^{m} b_j \left(1 + \frac{(m-k)k}{(n-\delta)m}\right)$$

and

$$\mathscr{LPT}(I) \leqslant \mathscr{LPT}(I') + \left(\sum_{i=1}^{n} t_i - \sum_{j=1}^{m} b_j\right).$$

Obviously

$$\mathscr{O}(I) \geqslant \sum_{i=1}^{n} t_i,$$

thus

$$\frac{\mathscr{LPT}(I)}{\mathscr{O}(I)} \leqslant \frac{\mathscr{LPT}(I') + \left(\sum_{i=1}^{n} t_i - \sum_{j=1}^{m} b_j\right)}{\sum_{i=1}^{n} t_i}$$

$$\leqslant \frac{\sum_{j=1}^{m} b_j \left[\frac{1+(m-k)k}{(n-\delta)m}\right] + \left(\sum_{i=1}^{n} t_i - \sum_{j=1}^{m} b_j\right)}{\sum_{i-1}^{n} t_i}$$

$$= \frac{\sum_{j=1}^{m} b_j \frac{(m-k)k}{(n-\delta)m} + \sum_{i=1}^{n} t_i}{\sum_{i=1}^{n} t_i}$$

$$< \frac{\sum_{i=1}^{n} t_i \frac{(m-k)k}{(n-\delta)m} + \sum_{i=1}^{n} t_i}{\sum_{i=1}^{n} t_i}$$

$$= 1 + \frac{(m-k)k}{(n-\delta)m}. \qquad \square$$

Theorem 2. *For all problem instances I,*

$$\frac{\mathscr{LPT}(I)}{\mathscr{C}(I)} < 2(2-\sqrt{2}).$$

Proof. As

$$n \geqslant m+k+\delta, \quad n-\delta \geqslant m+k,$$

it follows that

$$\frac{(m-k)k}{(n-\delta)m} \leqslant \frac{(m-k)k}{(m+k)m}.$$

We directly verify that

$$\frac{(m-k)k}{(m+k)m} \leqslant 3-2\sqrt{2}, \quad \forall m,k \in Z^+.$$

$$km-k^2 \leqslant (3-2\sqrt{2})(m^2+km),$$

$$(3-2\sqrt{2})m^2+2(1-\sqrt{2})km+k^2 \geqslant 0,$$

as

$$(3-2\sqrt{2})m^2+2(1-\sqrt{2})km+k^2 = [(1-\sqrt{2})m+k]^2.$$

Thus

$$\frac{\mathscr{LPT}(I)}{\mathscr{C}(I)} \leqslant 1+\frac{(m-k)k}{(n-\delta)m} < 2(2-\sqrt{2}),$$

where the strict inequality is due to the irrational value. □

Remark 2. Note that

$$2(2-\sqrt{2}) \approx 1.171573,$$

$$1.1\bar{6} \approx \tfrac{7}{6} < 2(2-\sqrt{2}) < \tfrac{6}{5} = 1.2$$

and $2(2-\sqrt{2})$ is the smallest upper bound for the parametric bound obtained from Theorem 1. This can be easily seen from the proof of Theorem 2. On the other hand, the parametric bound of Theorem 1 is obtained by using $\sum_{j=1}^{m} b_j$ as lower bound for the optimum value, which is probably not sufficient to obtain the tight bound.

We can build an instance I such that the bound $\mathscr{LPT}(I)/\sum_{j=1}^{m} b_j$ is very close to $2(2-\sqrt{2})$.

Example 1. Take 12 bins, each with length l and 17 items, each with length $\frac{12}{17}l$. The heuristic algorithm assigns the first 12 items to different bins and finally 5 items exceed 5 bins. The value of the heuristic is

$$\mathscr{LPT}(I) = 7l + 5\left(2\frac{12}{17}l\right) = \frac{239}{17}l,$$

while: $\sum_{j=1}^{m} b_j = 12l$. Thus the ratio:

$$\frac{\mathscr{LPT}(I)}{\sum_{j=1}^{m} b_j} = \frac{239}{12 \times 17} = 1.171569,$$

which is very close to $2(2-\sqrt{2})$. However, in this instance the optimum is not $\sum_{j=1}^{m} b_j$ and the heuristic finds the optimum solution.

Our conjecture is that the tight bound is $\frac{8}{7}$, as obtained by the instance with two bins of lengths $b_1 = 4$, $b_2 = 3$, and three items of sizes $t_1 = 3$, and $t_2 = t_3 = 2$.

4. Worst-case analysis for on-line list scheduling algorithm

In this section, we consider the on-line assignment problem, where each incoming item has to be assigned immediately to a bin and the assignment cannot be changed later. The objective is still the minimization of the sum of the sizes of the bins. Denoting with $\mathscr{H}(I)$ the value of the objective function, obtained by an on-line algorithm on the problem instance I, we show a lower bound of $\frac{7}{6}$ on the worst-case relative error of any $\mathscr{H}(I)$, with respect to the off-line problem. Also we show that the list scheduling algorithm, which assigns the incoming item to the bin with biggest idle space, has a worst-case error equal to $\frac{5}{4}$.

In order to show the performance of any on-line algorithm, we apply the same reasoning presented in [6] and consider an instance I given by 2 bins both of length 1. Suppose two items arrive of length $\frac{1}{3}$. If an algorithm H assigns them to the first bin, then two items, both of length $\frac{2}{3}$ may arrive. In this case:

$$\frac{\mathscr{H}(I)}{\mathscr{O}(I)} = \frac{7}{6}.$$

If the algorithm H assigns the first two items to different bins, then 2 items of length $\frac{1}{3}$ and 1 may arrive. In this case:

$$\frac{\mathscr{H}(I)}{\mathscr{O}(I)} = \frac{7}{6}.$$

Remark 3. No on-line algorithm H can have a worst-case performance better than $\frac{7}{6}$.

Let us denote with $\mathscr{LS}(I)$ the value of the solution obtained by the LS algorithm, which assigns the incoming item to the bin with largest idle space.

Theorem 3. *For all problem instances I,*

$$\frac{\mathscr{LS}(I)}{\mathscr{O}(I)} \leqslant \frac{5}{4}.$$

Moreover, the bound is tight.

Proof. Let the bins of the solution, obtained by LS, be numbered so that the exceeded bins are the first k bins of I.

We first show that:

$$\frac{\mathscr{LS}(I)}{\mathscr{C}(I)} \leqslant 1 + \frac{(m-k)k}{m^2}.$$

Two different cases will be examined separately.

Case (a):

$$\sum_{j=1}^{m} b_j \geqslant \sum_{i=1}^{n} t_i.$$

In this case $\sum_{j=k+1}^{m} s_j \geqslant \sum_{j=1}^{k} e_j$ and, by operating exactly as in Case (a) of Theorem 1, one obtains inequality (3). Then, the value obtained by LS is

$$\mathscr{LS}(I) = \sum_{j=1}^{m} b_j + \sum_{j=1}^{k} e_j \leqslant \sum_{j=1}^{m} b_j + \frac{m-k}{m} \sum_{t_i \in \mathscr{T}_k} t_i.$$

Denoting

$$t_{\max} = \max_{1 \leqslant i \leqslant n} t_i, \quad b_{\min} = \min_{1 \leqslant j \leqslant m} b_j,$$

from the assumption (1) we obtain:

$$\mathscr{LS}(I) \leqslant \sum_{j=1}^{m} b_j + \frac{(m-k)k}{m} t_{\max} \leqslant \sum_{j=1}^{m} b_j + \frac{(m-k)k}{m} b_{\min}. \tag{6}$$

As $\mathscr{C}(I) \geqslant \sum_{j=1}^{m} b_j$, we obtain:

$$\frac{\mathscr{LS}(I)}{\mathscr{C}(I)} \leqslant \frac{\sum_{j=1}^{m} b_j + [(m-k)k/m]b_{\min}}{\sum_{j=1}^{m} b_j} \leqslant 1 + \frac{[(m-k)k/m]b_{\min}}{\sum_{j=1}^{m} b_j}$$

$$\leqslant 1 + \frac{[(m-k)k/m]b_{\min}}{mb_{\min}} = 1 + \frac{(m-k)k}{m^2}.$$

Case (b):

$$\sum_{j=1}^{m} b_j < \sum_{i=1}^{n} t_i.$$

Let us consider a new instance I' obtained by the same bins of I and the biggest items $t'_i \in \mathscr{T}$, whose sum is equal to $\sum_{j=1}^{m} b_j$ (if necessary, cut an item into two parts and consider only one of them). Let $\mathscr{T}'$ be the new set of items. As the new instance falls in *Case* (a), from (6):

$$\mathscr{LS}(I') \leqslant \sum_{j=1}^{m} b_j + \frac{(m-k)k}{m} t_{\max}.$$

Moreover,

$$\mathscr{LS}(I) \leqslant \mathscr{LS}(I') + \left(\sum_{i=1}^{n} t_i - \sum_{j=1}^{m} b_j\right) \leqslant \frac{(m-k)k}{m} t_{\max} + \sum_{i=1}^{n} t_i,$$

$$\mathscr{O}(I) \geqslant \sum_{i=1}^{n} t_i > \sum_{j=1}^{m} b_j,$$

then

$$\frac{\mathscr{LS}(I)}{\mathscr{O}(I)} \leqslant \frac{[(m-k)k/m]t_{\max}}{\sum_{j=1}^{m} b_j} + \frac{\sum_{i=1}^{n} t_i}{\sum_{i=1}^{n} t_i} \leqslant 1 + \frac{(m-k)k}{m^2}.$$

We directly verify that

$$\frac{(m-k)k}{m^2} \leqslant \frac{1}{4}, \quad \forall m, k \in Z^+.$$

$$4(mk - k^2) \leqslant m^2,$$

$$m^2 - 4mk + 4k^2 \geqslant 0, \quad \forall m, k \in Z^+,$$

as

$$m^2 - 4mk + 4k^2 = (m - 2k)^2.$$

Thus

$$\frac{\mathscr{LS}(I)}{\mathscr{O}(I)} \leqslant \frac{5}{4}.$$

In order to show the bound is tight, consider, for example, the instance with two bins of length $b_1 = b_2 = 2$ and three items $t_1 = t_2 = 1$, $t_3 = 2$ arriving in the order $t_1 \prec t_2 \prec t_3$. □

5. Conclusions

In this paper we studied the problem of partitioning items in a set of unequal bins under the assumption that each item is smaller than any bin. The objective is to minimize the total size, i.e. the sum of the sizes of the bins considering that the size of each bin, if necessary, can be extended. We examined both the off-line case and the on-line variant of this problem. For the former case, when all items are known in advance, we analyzed the worst-case performance of the longest processing time heuristic and proved a bound of $2(2 - \sqrt{2})$. Our conjecture is that the tight bound is $\frac{8}{7}$.

For the on-line case, where each unknown incoming item has to be assigned immediately to a bin and the assignment cannot be changed later, we gave a lower bound of $\frac{7}{6}$ on the worst-case relative error of any on-line algorithm with respect to the off-line optimum. Moreover, we analyzed the performance of a list scheduling algorithm, which

assigns the incoming item to the bin with biggest idle space, proving a worst-case error equal to $\frac{5}{4}$. This bound was shown to be tight.

We conclude by observing that other heuristics which perform well in the standard bin packing problem may do no better than LPT. This is the case of the best fit descending (BFD) algorithm in which items are ordered by size, like in LPT, but an item is assigned to the fullest bin it fits in. Example 1 shows that, if the sum of the bins is used as lower bound on the optimum, the bound on the worst-case performance of LPT cannot be reduced by the BFD. For the on-line case, a best fit (BF) algorithm assigns the incoming item to the fullest bin it fits in. In this case, a variant of the example given at the end of the proof of Theorem 3, with $t_1 = 1$, $t_2 = 1 + \varepsilon$, $t_3 = 2$ shows that BF cannot do better than LS in the worst-case.

References

[1] P. Dell'Olmo, H. Kellerer, M.G. Speranza, Zs. Tuza, A 13/12 approximation algorithm for bin packing with extendable bins, IPL 65 (1998) 229–233.

[2] D.K. Friesen, F.S. Kuhl, Analysis of a hybrid algorithm for packing unequal bins, SIAM J. Comput. 17 (1988) 23–40.

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

[4] R.L. Graham, Bounds on multiprocessing timing anomalies, SIAM J. Appl. Math. 17 (1969) 263–269.

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

[6] M.G. Speranza, Zs. Tuza, On-line approximation algorithms for partitioning items in extendable bins, 1996, to appear in Annals of Operational Research.