CrossMark

# An efficient algorithm for semi-online multiprocessor scheduling with given total processing time

Hans Kellerer[1] · Vladimir Kotov[2] · Michaël Gabay[3]

**Abstract** We consider a semi-online multiprocessor scheduling problem with a given a set of identical machines and a sequence of jobs, the sum of whose processing times is known in advance. The jobs are to be assigned online to one of the machines and the objective is to minimize the makespan. The best known algorithm for this problem achieves a competitive ratio 1.6 (Cheng et al. in Theor Comput Sci 337:134–146, 2005). The best known lower bound is approximately 1.585 (Albers and Hellwig in Theor Comput Sci 443:1–9, 2012) if the number of machines tends to infinity. We present an elementary algorithm with competitive ratio equal to this lower bound. Thus, the algorithm is best possible if the number of machines tends to infinity.

**Keywords** Semi-online scheduling · Competitive analysis · Multiprocessor scheduling

✉ Hans Kellerer
hans.kellerer@uni-graz.at

Vladimir Kotov
kotovVM@bsu.by

Michaël Gabay
michael.gabay@g-scop.grenoble-inp.fr

[1] Institut für Statistik und Operations Research, Universität Graz, Universitätsstraße 15, 8010 Graz, Austria

[2] Faculty of Applied Mathematics and Computer Science, Belarusian State University, Nezavisimosti Ave. 4, 220030 Minsk, Belarus

[3] Grenoble-INP/UJF-Grenoble 1/CNRS, G-SCOP UMR5272, 38031 Grenoble, France

## 1 Introduction

The well-known classical multiprocessor scheduling problem is a fundamental and well-investigated scheduling problem both in the offline and the online setting. A set of $n$ independent jobs is to be processed on $m$ parallel, identical machines in order to minimize the makespan. The offline scenario of the problem is strongly NP-hard but can be approximated efficiently (see e.g., Hochbaum and Shmoys 1987 for a polynomial time approximation scheme).

In the online scenario each job must be immediately and irrevocably assigned to one of the machines without any knowledge on future jobs. This problem was first investigated by Graham (1966, 1969) who showed that the list scheduling algorithm has a performance ratio of exactly $2 - 1/m$. It is also shown that this algorithm is best possible for $m \leq 3$ (Faigle et al. 1989). A long list of improved algorithms has since been published. The best heuristic known for this problem is due to Fleischer and Wahl (2000). They designed an algorithm with competitive ratio smaller than 1.9201 when the number of machines tends to infinity. The best lower bound is 1.88 and is due to Rudin (2001).

Recent research has focused on scenarios between offline and online scenarios where the online constraint is relaxed but no full information on the input data is available. For a survey on recent advances we refer to the paper by Albers (2013). In this paper we consider the online multiprocessor scheduling with the additional assumption that the sum of processing times is given in advance. The resulting problem is denoted as the *known total processing time scheduling problem*.

A related semi-online problem has been introduced by Azar and Regev (2001), who labeled it as the *online bin*

*stretching problem.* A sequence of items is given and we know that these items *can* be packed into $m$ bins of unit size. The items are to be assigned online to the bins and the aim is to minimize the *stretching factor* of the bins, i.e., to stretch the sizes of the bins as least as possible such that all items fit in the bins. Thus, the bin stretching problem can be interpreted as a semi-online scheduling problem where, instead of the total processing time, even the value of the optimal makespan is known in advance. Consequently, any algorithm for the known total processing time scheduling problem works also for the bin stretching problem, i.e., it achieves at least the same competitive ratio. For the bin stretching problem, a sophisticated proof for an algorithm with stretching factor 1.625 was given by Azar and Regev (2001). Kellerer and Kotov (2013) proposed an algorithm with stretching factor $11/7 \approx 1.571$ by using techniques of grouping bins into batches. The latest algorithm is due to Böhm et al. (to be published). They give an algorithm for online bin stretching with a stretching factor of 1.5 for any number of bins. They also show a specialized algorithm for three bins with a stretching factor of $11/8 = 1.375$.

In a previous paper (Kellerer et al. 1997), among three models, the semi-online model in consideration is also investigated here and an algorithm with performance ratio $4/3$ for the known total processing time scheduling problem on two machines was given. This bound is best possible for $m = 2$. In Cheng et al. (2005) presented an algorithm with performance ratio 1.6 for the known total processing time scheduling problem for an arbitrary number of machines. Moreover, they established a lower bound of 1.5 for $m \geq 6$ machines. Angelelli et al. (2004) gave a deterministic algorithm with performance ratio $(1 + \sqrt{6})/2 \approx 1.725$ and showed a lower bound of 1.565. Recently, Albers and Hellwig (2012) developed an improved lower bound showing that no deterministic semi-online algorithm for the known total processing time scheduling problem can attain a competitive ratio smaller than approximately 1.585 when $m$ tends to infinity. Moreover, they give a simple algorithm with performance ratio 1.75.

In this paper we will present an algorithm with competitive ratio approximately 1.585, which is equal to the lower bound of Albers and Hellwig. Thus, our algorithm is best possible if $m$ tends to infinity. Our algorithm is an adaptation of the recently published algorithm for the bin stretching problem by Kellerer and Kotov (2013): jobs and machines are classified and the algorithm runs in two phases. In the first phase, jobs are assigned to the machines depending on jobs and machines classes. In the second phase we distinguish two cases depending on the structure of the machines after Phase 1 and for each case a separate algorithm is presented.

## 2 Problem definition and notation

We are given $m$ identical machines and a sequence of jobs $1, \ldots, n$ with processing times $p_1, \ldots, p_n$. Any of the jobs is to be assigned online to one of the machines. We assume that the sum $S = \sum_{j=1}^{n} p_j$ of the jobs processing times is given in advance. Without loss of generality, $S = m$. The objective is to minimize the makespan. Since our algorithm is strongly related to the bin stretching algorithm in Kellerer and Kotov (2013), we will usually speak of bins $B_1, \ldots, B_m$ instead of machines and of items with weights $p_j$ instead of jobs with processing times $p_j$.

Let $R$ be a set of items. The *weight* of $R$ is defined as $\sum_{j \in R} p_j$, and is denoted by $w(R)$. The *weight of a bin $B$* is defined as the total weight of all items assigned to $B$, and is denoted by $w(B)$. When we speak of *time $j$*, we mean the state of the system just before item $j$ is assigned.

Let $\alpha$ be the positive root of the function $f(x) = 4x^3 + 4x^2 - 2x - 1$. We will show that the competitive ratio of our algorithm is equal to $1 + \alpha \approx 1.58504$. Throughout the paper we will use several times the equation

$$\frac{1}{2\alpha} = (1 + \alpha)2\alpha - 1. \tag{1}$$

The items are divided into several classes. Items with weights in $(0, \alpha \approx 0.585]$ are called *small*, items in $(\alpha, \frac{1}{2\alpha} \approx 0.8546]$ are called *medium*, and items larger than $\frac{1}{2\alpha}$ are called *large*. Moreover, small items with weights less or equal than $\frac{\alpha}{2} \approx 0.2925$ are called *tiny*.

A bin $B$ with $w(B) \in (0, \frac{\alpha}{2} \approx 0.2925]$ is called *tiny*, for $w(B) \in (0, \alpha]$ it is called *small*, for $w(B) \in (\alpha, \frac{1}{2\alpha}]$ it is called *medium*, for $w(B) \in (\frac{1}{2\alpha}, 1]$ it is called *big*, and for $w(B) > 1$ it is called *huge*. Big and huge bins are also denoted as *large*. Notice that each tiny bin is small. If $B$ contains a large item, it is called a *large item bin*. The number of empty bins, tiny bins, small bins, medium bins, and big bins is abbreviated by $eB$, $tB$, $sB$, $mB$, and $bB$, respectively. When a bin is *closed*, no more items can be assigned to that bin. Otherwise, it is called *open*. If we want to specify the time $j$, we write $eB^j$, $w^j(B)$ instead of $eB$, $w(B)$, respectively. The definitions of item and bin classes are depicted in Tables 1 and 2.

We denote by $LB$ a lower bound for the optimal makespan which is recalculated throughout the algorithm. Since $S = m$, we have $LB \geq 1$. Each bin of the algorithm has a *capacity*

**Table 1** Item classes

| Item class | Tiny | Small | Medium | Large |
|---|---|---|---|---|
| Interval | $(0, \frac{\alpha}{2}]$ | $(0, \alpha]$ | $(\alpha, \frac{1}{2\alpha}]$ | $(\frac{1}{2\alpha}, \infty]$ |

**Table 2** Bin classes

| Bin class | Empty bin $eB$ | Tiny bin $tB$ | Small bin $sB$ | Medium bin $mB$ |
|---|---|---|---|---|
| Definition | $w(B) = 0$ | $0 < w(B) \leq \frac{\alpha}{2}$ | $0 < w(B) \leq \alpha$ | $\alpha < w(B) \leq \frac{1}{2\alpha}$ |
| Bin class | Big bin $bB$ | Huge bin | Large bin | Large item bin |
| Definition | $\frac{1}{2\alpha} < w(B) \leq 1$ | $1 < w(B)$ | $\frac{1}{2\alpha} < w(B)$ | $B$ contains a large item |

of $(1 + \alpha)LB$ and we say that item $j$ *fits* into a bin $B$, if $w(B) + p_j \leq (1 + \alpha)LB$.

## 3 Description of the algorithm and proof of the upper bound

In this section we will present the algorithm for the known total processing time scheduling problem with competitive ratio $1 + \alpha \approx 1.585$. This algorithm is split into two parts. The first part (called Phase 1) is an adaptation of Phase 1 for the bin stretching problem as described in Kellerer and Kotov (2013). It runs until the number of empty bins is around one third of the number of small bins.

Denote by $q_1, \ldots, q_j$ the weights of the items at time $j$ sorted in non-increasing order, i.e., $q_1 \geq q_2 \geq \cdots \geq q_j$. If $j < m + 1$, set $q_{j+1} = q_{j+2} = \cdots = q_{m+1} = 0$. Then, an obvious lower bound $LB$ for the optimal makespan is given by

$$LB = \max\{1, q_1, q_m + q_{m+1}\}. \tag{2}$$

We will use the following observation several times:

**Observation 1** *Let $B$ be a small or an empty bin. Then any item $j$ fits in bin $B$.*

*Proof* From $p_j \leq q_1 \leq LB$ follows with $LB \geq 1$ that $w(B) + p_j \leq \alpha + LB \leq (1 + \alpha)LB$. □

A formal description of Phase 1 of the algorithm is depicted in Fig. 1. Note that the numbering reflects the prior-

ity rules for the assignment of item $j$, i.e., a packing option is skipped if no bin exists which fulfills the related condition and the next option is checked.

Initially, the lower bound $LB$ is set to 1 and it is recalculated each time when a new item $j$ arrives. Moreover, all bins are open in the beginning.

The following lemma lists some simple properties of the bins after Phase 1. The proof of the lemma is a slight adaptation of the proof of Lemma 1 in Kellerer and Kotov (2013).

**Lemma 1** *During any time of Phase 1 of the algorithm, the following properties hold:*

(a) *The weights of all bins are smaller than or equal to $(1 + \alpha)LB$.*
(b) *Medium bins consist of exactly one medium item.*
(c) *There is at most one tiny bin, i.e., $tB \leq 1$.*
(d) *All large bins are large item bins.*
(e) *All closed bins are large item bins.*
(f) $sB = 0$ *or* $bB = 0$.
(g) $sB < 3eB$, *while Phase 1 does not stop.*

*Proof* In Step 1.1 items are only assigned if they fit in the bin. In the other steps item $j$ is assigned to an empty bin or to a small bin. By Observation 1 each item fits in a small or empty bin. This shows property (a).

Since medium bins can only be constructed by assigning a medium item to an empty bin, (b) follows.

**Fig. 1** Algorithmic description of Phase 1

<div style="border:1px solid">

**Phase 1 of the Algorithm**

Let $j$ be the current item to be assigned. Recalculate $LB$ according to (2).

1. $j$ is small:
   1.1 Put $j$ in a large item bin $B$ if $j$ fits in $B$, i.e., if $w(B) + p_j \leq (1 + \alpha)LB$.
   1.2 Put $j$ in a small bin $B$ if $B$ remains small, i.e., $w(B) + p_j \leq \alpha$.
   1.3 Put $j$ in an empty bin.
2. $j$ is medium:
   2.1 Put $j$ in an empty bin.
3. $j$ is large:
   3.1 Put $j$ in the small bin with largest weight.
   3.2 Put $j$ in an empty bin.

Stopping condition: If
$$sB = 3eB + \lambda, \quad 0 \leq \lambda \leq 3, \tag{3}$$
close all huge bins and goto Phase 2.

</div>

Given a tiny bin $B$ and a tiny item $j$, then $w(B) + p_j \leq \alpha$. Then, priority rule 1.2 guarantees that $tB \leq 1$ and (c) holds.

Large bins can only be constructed by assigning large items to small bins or empty bins. Assertion (d) follows. All closed bins are huge. Thus, (d) implies (e).

We prove assertions (f) and (g) by induction. Assume (f) is true at time $j$. If there are neither small bins nor big bins at time $j$, then $sB^{j+1} = 0$ or $bB^{j+1} = 0$.

Assume now that there are small bins, but no big bin at time $j$. If $bB^{j+1} = 0$, the assertion is true. If $bB^{j+1} = 1$, item $j$ is large and it must be assigned to a tiny bin since otherwise the new weight of the bin would be larger than 1. By definition, $j$ is assigned to a small bin with largest weight. Hence, the set of small bins at time $j$ consists of a single tiny bin by (c). Thus, $sB^{j+1} = 0$.

Finally, assume that there are big bins, but no small bin at time $j$. If $sB^{j+1} = 0$, the assertion is true. If $sB^{j+1} = 1$, item $j$ is small. By (d) each big bin is a large item bin. Because of $bB^j > 0$ there are always big bins in which the small item $j$ fits. According to the priority rules for small items, it cannot occur that $sB^{j+1} > 0$.

Assertion (g) is correct for $j = 1$, i.e., before any item is assigned. Assume it is true at times $1, \ldots, j$ and Phase 1 does not stop after assigning item $j - 1$, i.e., condition (3) does not hold. This means $sB^j < 3eB^j$ by the induction hypothesis. After the assignment of item $j$ the number $sB^j$ can increase by at most 1 and $eB^j$ can decrease by at most 1. Hence,

$$sB^{j+1} \leq sB^j + 1 \leq 3eB^j \leq 3(eB^{j+1} + 1) = 3eB^{j+1} + 3. \tag{4}$$

If the stopping condition (3) holds, there is nothing to prove. If the stopping condition (3) does not hold, we have either $sB^{j+1} > 3eB^{j+1} + 3$ or $sB^{j+1} < 3eB^{j+1}$. The first inequality is excluded by (4) and assertion (g) follows. □

Notice that Phase 1 of the algorithm is well defined: due to Lemma 1(g) Phase 1 is only running while $eB > 0$, hence items can be assigned to empty bins if necessary.

It is possible that all items are assigned already after Phase 1. Of course, in this case, the upper bound holds.

By Lemma 1, after Phase 1, we can only have empty bins, closed bins, and either only large item bins and medium bins or only small bins (including at most one tiny bin) and medium bins. We distinguish two cases for the algorithm.

First, consider the case $sB = 0$. The corresponding phase of the algorithm is denoted as Phase 2a. Note that from condition (3) we get $0 = 3eB + \lambda$ which implies that there are no empty bins. Without loss of generality, assume that the bins $B_1, \ldots, B_k$ are open after Phase 1 and the bins $B_{k+1}, \ldots, B_m$ are closed. Moreover, the open bins shall be sorted in non-increasing order of their weights, i.e., $w(B_1) \geq w(B_2) \geq \cdots \geq w(B_k)$. During Phase 2a any item is assigned to the bin with largest weight. If it does not fit, it is packed in the bin with the smallest weight. Afterward, the bin with the largest weight is closed. Throughout the algorithm the open bin with maximum weight is denoted by $B^{\max}$ and the open bin with minimum weight by $B^{\min}$, respectively. A formal description of Phase 2a of the algorithm is depicted in Fig. 2.

There is an index $r' \geq 1$ such that the bins $B_1, \ldots, B_{r'-1}$ correspond to the bins which are closed in Step 3 of Phase 2a. Moreover, there is an index $r$, $r' \leq r \leq k$, such that the Bins $B_{r+1}, \ldots, B_k$ correspond to the bins to which an item is assigned in Step 4 of Phase 2a. If Step 2 is executed, then bin $B_r$ is the last open bin to which an item is assigned. Thus, each of the bins $B_1, \ldots, B_{r'-1}$ corresponds to bin $B^{\max}$ when it is closed, and each of the bins $B_{r+1}, \ldots, B_k$ corresponds to bin $B^{\min}$ when it is closed, respectively.

**Lemma 2** *The following properties hold for Phase 2a.*

(a) *At the beginning of Phase 2a, each bin contains an item with weight greater than $\alpha$.*
(b) *At least one large item is assigned to each of the bins $B_{r+1}, \ldots, B_k$ in Step 4 of Phase 2a. In particular, at the end of Phase 2a, all bins $B_{r+1}, \ldots, B_m$ contain a large item.*
(c) *The weights of all closed bins are greater than 1.*
(d) *At the end of Phase 2a, all bins have weights smaller than or equal to $(1 + \alpha)LB$.*

**Fig. 2** Algorithmic description of Phase 2 with no small bins

<div style="border:1px solid">

**Phase 2a of the Algorithm:** $sB = 0$

Initialization: Set $s = k$ (number of open bins).

Iteration $j$:

1. Let $j$ be the current item to be assigned. Recalculate $LB$ according to (2).
2. If $s = 1$, assign the remaining items to the last open bin and stop.
3. Put $j$ in bin $B^{\max}$ if $j$ fits in it. If $j$ is put in bin $B^{\max}$ and $w(B^{\max}) + p_j > 1$, close bin $B^{\max}$ and set $s = s - 1$.
4. If $j$ does not fit in $B^{\max}$, put $j$ in bin $B^{\min}$, close bin $B^{\min}$ and set $s = s - 1$.

</div>

*Proof* Consider the beginning of Phase 2a when $sB = 0$ holds after Phase 1.

By Lemma 1(b), (d) all $m$ bins contain at least one medium or large item at the beginning of Phase 2. This shows (a).

Since a bin $B^{\max}$ is closed if its weight is greater than 1, the weight of any open bin $B^{\max}$ is at most 1. Thus, any small item $i$ fits in the open $B^{\max}$. Now consider an item $i$ which is assigned to one of the bins $B_{r+1}, \ldots, B_k$ during Phase 2a. Consequently, $p_i > \alpha$. By (a) there are $m + 1$ items with weight greater than $\alpha$ and we conclude with (2) that $LB \geq q_m + q_{m+1} \geq 2\alpha$. Since $i$ does not fit in $B^{\max}$, we get with (1)

$$1 + p_i \geq w^i(B^{\max}) + p_i > (1 + \alpha)LB$$
$$\geq (1 + \alpha)2\alpha = 1 + \frac{1}{2\alpha}.$$

This implies $p_i > \frac{1}{2\alpha}$, i.e., item $i$ is large. Hence all bins $B_{r+1}, \ldots, B_k$ contain a large item and by definition of the algorithm only one item is put in these bins during Phase 2a. Recall that by Lemma 1(e) each bin $B_{k+1}, \ldots, B_m$ contains a large item at the beginning of Phase 2a. This proves assertion (b).

Bins closed at the end of Phase 1 and bins closed in Step 3, serving as $B^{\max}$, are closed when their weights are greater than 1. Bins closed in Step 4, i.e., bins $B_{r+1}, \ldots, B_k$, contain an item with weight greater than $\alpha$ and, due to (b), a large item. Thus, their weights are greater than $\alpha + 1/(2\alpha) > 1$. This shows assertion (c).

Consider finally assertion (d). The bins $B_{r+1}, \ldots, B_k$ are closed in Step 4. Let $B_\lambda, r + 1 \leq \lambda \leq k$, be closed by assigning item $j$ to it. Due to (b) item $j$ is large. Bins $B_{\lambda+1}, \ldots, B_m$ are closed before $B_\lambda$ and contain a large item at time $j$ because of (b). Bin $B_\lambda$ contains a medium or large item $x$ with weight $p_x$ at time $j$. Recall that each bin contains a medium or large item and bins were sorted in non-increasing order at the beginning of Phase 2a. Moreover, by Lemma 1(b) each medium bin consists of a single medium item at the beginning of Phase 2b.

Thus, if $x$ is medium, we conclude that all bins $B_\mu$ with $1 \leq \mu < \lambda$, contain an item with weight not smaller than $p_x$ at time $j$. Hence, together with item $j$ the weights of at least $m + 1$ items are not smaller than $p_x$. This implies $LB \geq 2p_x$. If $x$ is large, there are together with item $j$ at least $m + 1$ large items. This implies $LB > 1/\alpha$. Thus, we have shown

$$LB \geq \min\left\{2p_x, \frac{1}{\alpha}\right\}. \tag{5}$$

If $x$ is medium, we have by Lemma 1(b) that $p_x = w^j(B_\lambda)$ and $p_x \leq \frac{1}{2\alpha}$. By (5) we get $p_x \leq LB/2$. Together with (1) and $LB \geq p_j$ we conclude that

$$p_j + w^j(B_\lambda) = p_j + p_x \leq LB + \frac{LB}{2} = \frac{3}{2}LB.$$

If $x$ is large, (5) implies that $LB \geq 1/\alpha$. We obtain from $LB \geq p_j$ that

$$p_j + w^j(B^{\max}) \leq LB + 1 = \left(1 + \frac{1}{LB}\right)LB \leq (1 + \alpha)LB.$$

This shows that the weights of all bins $B_{r+1}, \ldots, B_k$ are smaller than or equal to $(1 + \alpha)LB$.

Recall that bins $B_1, \ldots, B_{r'-1}$ are closed in Step 3. By definition the weights of all bins closed in Step 3 are not larger than $(1 + \alpha)LB$. By Lemma 1(a) this is also true for bins $B_{k+1}, \ldots, B_m$.

It remains to prove (d) for bins $B'_r, \ldots, B_r$. If $r' < r$, all these bins have weight at most one. If $r = r'$, bin $B_r$ is the last open bin and the remaining items are assigned to it in Step 2. Thus, by (c) the weights of all other bins are greater than 1. Since the total weight sum is equal to $m$, $w(B_r) \leq 1$ at the end of Phase 2a. This completes the proof of our Lemma. $\square$

Now we treat Phase 2b. It deals with the case where there are no big bins, i.e., the open bins are small (including at most one tiny bin), medium, or empty. Without loss of generality we may assume that there is at least one small bin because $sB = 0$ is considered in Phase 2a. By condition (3) there are $4k + \lambda$ empty or small bins after Phase 1 for some integer $k \geq 0$. Among these $4k + \lambda$ bins there are $k$ empty bins and the remaining $3k + \lambda$ are small. These bins are now partitioned into $k$ so-called *four-batches* $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_k$. Each four-batch $\mathcal{B}$ consists of four bins $B_1, B_2, B_3, B_4$ where bins $B_1, B_2, B_3$ are small, and the fourth bin $B_4$ is an empty bin. If $\lambda > 0$, there is an additional batch $\mathcal{B}_{k+1}$ of at most three non-empty bins. By Lemma 1(c) there is at most one tiny bin. This possibly existing tiny bin shall correspond to bin $B_1$ of four-batch $\mathcal{B}_1$. Therefore, we may assume that all other small bins are not tiny and have weight at least $\alpha/2$. Set $k' = k + 1$, if $\lambda > 0$ and $k' = k$, otherwise.

The medium bins are denoted by $M_1, M_2, \ldots, M_\ell$ and shall be sorted in non-increasing order of their weights at the beginning of Phase 2b, i.e., $w(M_1) \geq w(M_2) \geq \cdots \geq w(M_\ell)$. We call them briefly $M$-*bins*. Throughout Phase 2b denote the open $M$-bin with maximum weight by $M^{\max}$, the open $M$-bin with second largest weight by $M^2$ and the open $M$-bin with smallest weight by $M^{\min}$, respectively. If there are only two $M$-bins, bins $M^2$ and $M^{\min}$ are identical.

For the assignment of items to batches, we will always use *First Fit*, i.e., an item is assigned to the bin with smallest index in a batch in which it fits. The algorithm tries to assign an item $j$ to the bins in the following order: $M^{\max}$, $M^2$, an open batch with smallest index if $p_j \leq (1 + \alpha)/2$, an open batch with largest index if $p_j > (1 + \alpha)/2$, $M^{\min}$.

**Fig. 3** Algorithmic description of Phase 2 with no big bins

---

**Phase 2b of the Algorithm:** $bB = 0$

Initialization: Set $\mu = \ell$ (number of $M$-bins) and $\tau = k'$ (number of open batches).

Iteration $j$:

1. Let $j$ be the current item to be assigned. Recalculate $LB$ according to (2).
2. $\mu \geq 1$, $j$ fits in $M^{\max}$: assign $j$ to $M^{\max}$. If $w^j(M^{\max}) + p_j \geq 1$, close $M^{\max}$ and set $\mu = \mu - 1$.
3. $\mu \geq 2$, $p_j \leq \beta$: assign $j$ to $M^2$, close $M^2$ and set $\mu = \mu - 1$.
4. $\tau \geq 2$, $p_j \leq (1+\alpha)/2$: let $\mathcal{B}_i$ be the open batch with smallest index. Assign $j$ to $\mathcal{B}_i$ if it fits. Otherwise, close batch $\mathcal{B}_i$ and set $\tau = \tau - 1$. If still $\tau \geq 2$, assign $j$ to $\mathcal{B}_{i+1}$.
5. $\tau \geq 2$, $p_j > (1+\alpha)/2$: let $\mathcal{B}_i$ be the open batch with largest index. Assign $j$ to $\mathcal{B}_i$ if it fits. Otherwise, close batch $\mathcal{B}_i$ and set $\tau = \tau - 1$. If still $\tau \geq 2$, assign $j$ to $\mathcal{B}_{i-1}$.
6. $\tau = 1$: assign $j$ to the last open batch if it fits.
7. $\mu \geq 2$: assign $j$ to $M^{\min}$, close $M^{\min}$ and set $\mu = \mu - 1$.

---

Let $\beta = 1 + \alpha - \frac{1}{2\alpha} \approx 0.7304$. It can be easily seen that

$$\alpha = \frac{1}{4\alpha\beta}. \tag{6}$$

A formal description of Phase 2b of the algorithm is depicted in Fig. 3.

Steps 4 and 5 of Phase 2b ensure that there is an $r \leq k'$ such that only items with weights smaller than or equal to $(1 + \alpha)/2$ are assigned to batches $\mathcal{B}_1, \ldots, \mathcal{B}_{r-1}$ and only items with weights greater than $(1 + \alpha)/2$ are assigned to batches $\mathcal{B}_{r+1}, \ldots, \mathcal{B}_{k'}$. If there is a last open batch to which items are assigned in Step 6, then it is batch $\mathcal{B}_r$. Note that batch $\mathcal{B}_r$ is the only batch in which possibly items of all weights can be assigned.

The priority rules of Phase 2b are based on the following ideas: An item which is not put into bins $M^{\max}$ and $M^2$, has weight greater than $\beta$. Hence, while $\mu \geq 2$ only items greater than $\beta$ are assigned to batches $\mathcal{B}_1, \ldots, \mathcal{B}_{k'}$. It can be shown that when in Step 7 an item is assigned to $M^{\min}$, each bin of the batches contains an item greater than $\beta$ and all batches except from batch $\mathcal{B}_r$ are closed. This yields an improvement of the lower bound and guarantees that any item fits when assigned during Step 7 (see Lemma 5).

In order to assure that the small bins in batches $\mathcal{B}_{r+1}, \ldots, \mathcal{B}_{k'}$ have weight greater than $(1 + \alpha)/2$, the possible tiny bin is put in batch $\mathcal{B}_1$. The structure of the batches (three small bins and one empty bin) and the separation into batches $\mathcal{B}_1, \ldots, \mathcal{B}_{r-1}$ and $\mathcal{B}_{r+1}, \ldots, \mathcal{B}_{k'}$, guarantee that these batches have each average weight greater than 1 when they are closed (see Lemma 4). Altogether, we will show that the average weight of all closed bins is greater than 1 and any item which is assigned to a bin, fits in that bin. By Corollaries 1 and 2, Phase 2b can only fail when an item cannot be assigned in Steps 1 to 7. But this means that we have at most one open $M$-bin and the only open batch is batch $\mathcal{B}_r$. This case is treated in Theorem 1.

The following lemmas contain several properties which hold for Phase 2b. We start with a simple technical lemma

concerning the weights of the bins in one of the batches $\mathcal{B}_1, \ldots, \mathcal{B}_{k'}$.

**Lemma 3** (a) *If an item does not fit in the two bins $B_1$, $B_2$ of a batch, then $w(B_1) + w(B_2) \geq \frac{3}{2}\alpha + 1 \approx 1.878$.*

(b) *If an item does not fit in the three bins $B_1$, $B_2$, $B_3$ of a batch, then $w(B_1) + w(B_2) + w(B_3) \geq \frac{9}{4}\alpha + \frac{3}{2} \approx 2.816$.*

*Proof* By Observation 1 at least one item can be assigned to each bin of a given batch without exceeding the capacity. The first item which does not fit in bin $B_1$ is put in bin $B_2$ by First Fit. Since bin $B_2$ is not tiny, the weight of $B_2$ is greater than $\alpha/2$ at the beginning of Phase 2b, we get $w(B_1) + w(B_2) > 1 + \alpha + \alpha/2 = \frac{3}{2}\alpha + 1$. This shows (a).

By (a) the weight of at least one of the bins $B_1$ and $B_2$ is greater than $\frac{1}{2}(\frac{3}{2}\alpha + 1)$. We denote this bin by $B_1'$. The other bin shall be denoted by $B_2'$. The first item which does not fit in $B_1'$ and $B_2'$, is assigned to $B_3$. Since the weight of $B_3$ is greater than $\alpha/2$ at the beginning of Phase 2b, we get $w(B_1') + w(B_2') + w(B_3) > \frac{1}{2}\left(\frac{3}{2}\alpha + 1\right) + (1+\alpha) + \alpha/2 = \frac{9}{4}\alpha + \frac{3}{2}$. $\qquad\square$

The next lemma shows that in average all closed bins have weight at least one.

**Lemma 4** (a) *All bins closed in Steps 2 and 3 have weight at least one.*

(b) *The average weight of each of the closed batches $\mathcal{B}_1, \ldots, \mathcal{B}_{r-1}$ is greater than 1.*

(c) *The average weight of each of the closed batches $\mathcal{B}_{r+1}, \ldots, \mathcal{B}_{k'}$ is greater than 1.*

(d) *The weights of all bins closed in Step 7 are greater than 1.*

*Proof* By definition all bins closed in Step 2 have weight at least 1. Note that an $M$-bin never exceeds a weight of 1 unless it is closed. This follows from the fact that $M$-bins are closed because their weight is at least 1 (Step 1) or because an item is assigned to it (Steps 3 and 7). If bin $M^2$ is closed in Step 3, then $j$ did not fit in $M^{\max}$, i.e., $M^{\max}$ is open and

$w(M^{\max}) < 1$. Hence, $p_j > \alpha$ and $w^j(M^2) + p_j > \alpha + \alpha > 1$ This shows (a).

Each batch $\mathcal{B}_i$, $i \leq r - 1$, contains four bins $B_1, \ldots, B_4$. Let $B_1'$ the bin with largest weight among $B_1$ and $B_2$, and $B_2'$ the other bin. According to Lemma 3(a) $w(B_1') > \frac{3}{4}\alpha + \frac{1}{2}$. Since the weights of the items put into batch $\mathcal{B}_i$ are at most $(1 + \alpha)/2$, at least two items $x_1$, $x_2$ are put in the empty bin $B_4$. Both of them did not fit in $B_2'$ and $B_3$ which implies $w(B_2') + p_{x_1} > 1 + \alpha$ and $w(B_3) + p_{x_2} > 1 + \alpha$. We get

$$w(B_1') + w(B_2') + w(B_3) + w(B_4) > \frac{3}{4}\alpha + \frac{1}{2} + 2(1+\alpha) > 4.$$

Thus, (b) holds.

Only items with weight greater than $(1 + \alpha)/2$ are put in batches $\mathcal{B}_i$, $i \geq r + 1$. If $i \leq k$, $\mathcal{B}_i$ consists of four bins $B_1$, $B_2$, $B_3$, $B_4$. Since batches $\mathcal{B}_i$, $i \geq r + 1$, contain no tiny bin, the weights of the bins $B_1$, $B_2$, $B_3$ are all greater than $\alpha/2$. By Observation 1, an item fits in each of the four bins. We obtain

$$w(B_1) + w(B_2) + w(B_3) + w(B_4) > 3 \cdot \frac{\alpha}{2} + 4(1+\alpha)/2 > 4.$$

If $k' = k + 1$, $\mathcal{B}_{k'}$ consists of $\lambda$ small bins, $1 \leq \lambda \leq 3$, with weight greater than $\alpha/2$ each. Thus, after assigning items with weights greater than $(1 + \alpha)/2$ the weights of each of these bins are at least $\alpha/2 + (1+\alpha)/2 = \alpha + 1/2 > 1$. This shows (c).

Since $p_j > \beta = 1 + \alpha - \frac{1}{2\alpha}$, we have $w^j(M^{\min}) + p_j > \alpha + \beta > 1$ and (d) is shown. □

**Corollary 1** *The average weight of all closed bins is greater than 1.*

*Proof* The assertion follows directly from Lemma 4. □

**Lemma 5** *Suppose $\mu \geq 2$ and assume item $j$ is not assigned during Steps 2 to 6. Then, the following properties hold:*

(a) *All batches except from batch $\mathcal{B}_r$ are closed.*
(b) *Each bin belonging to a batch contains an item with weight greater than $\beta$.*
(c) *$LB \geq \min\{2w^j(M^{\min}), 2\beta\}$.*
(d) *Item $j$ fits in $M^{\min}$.*

*Proof* Assertion (a) follows directly from the structure of the algorithm.

As a consequence of Step 3 of Phase 2b, while $\mu \geq 2$, the weights of all items assigned to the batches during Phase 2b are greater than or equal to $\beta$ and (b) holds.

Since $j$ is not assigned in Steps 4 to 6 and by Observation 1 each bin of the batches contains an item whose weight is greater than $\beta$. By Lemma 1(e) all bins which are closed

after Phase 1 contain a large item, which is greater than $\beta$ by definition.

Let $M^{\min} = M_{\ell'}$, $\ell' \leq \ell$, at time $j$. All of the bins $M_{\ell'+1} \ldots, M_\ell$ contain an item greater than $\beta$ at time $j$. These items have been assigned in earlier iterations when these bins served as $M^{\min}$.

At time $j$ bin $M_{\ell'}$ is $M^{\min}$ and contains a single medium item $i$ and bins $M_1, \ldots, M_{\ell'-1}$ contain medium items which are not smaller than $w^j(M^{\min}) = p_i$. If $w^j(M^{\min}) \geq \beta$, there are $m + 1$ items greater than $\beta$ including item $j$. Otherwise, if $w^j(M^{\min}) \leq \beta$, there are $m + 1$ items greater than $w^j(M^{\min})$. By applying (2) with $LB \geq q_m + q_{m+1}$, (c) follows.

In order to show that $j$ fits into $M^{\min}$ we distinguish two cases. If $w^j(M^{\min}) \geq \beta$, we have $LB \geq 2\beta$. Hence, using (6) follows

$$p_j + w^j(M^{\min}) \leq LB + \frac{1}{2\alpha} \leq LB + \frac{1}{2\alpha}\frac{LB}{2\beta} = (1+\alpha)LB.$$

If $w^j(M^{\min}) \leq \beta$, we have $LB \geq 2w^j(M^{\min})$ and we get

$$p_j + w^j(M^{\min}) \leq LB + \frac{LB}{2} \leq (1+\alpha)LB.$$

This shows (d). □

We are now able to show that during Phase 2b no bin has weight exceeding $(1 + \alpha)LB$.

**Corollary 2** *When an item is assigned to a bin during Phase 2b, it always fits in the bin.*

*Proof* In Steps 2 and 6 an item is only assigned to a bin if it fits. We have to consider Steps 3, 4, 5, and 7. Assume item $j$ is assigned to bin $M^2$ in Step 3 of the algorithm. We have $w^j(M^2) + p_j \leq \frac{1}{2\alpha} + \beta = \frac{1}{2\alpha} + 1 + \alpha - \frac{1}{2\alpha} = 1 + \alpha$.

Consider the case that $j$ is assigned to batch $\mathcal{B}_{i+1}$ in Step 4. Since $\tau \geq 2$, $\mathcal{B}_{i+1} \neq \mathcal{B}_r$ and $j$ is the first item assigned to $\mathcal{B}_{i+1}$ during Phase 2b. Since each bin $B$ of $\mathcal{B}_{i+1}$ is small or empty, $j$ fits in $B$ because of Observation 1. Step 5 is analogous to Step 4. Because of Lemma 5(d) every item assigned in Step 7 fits in $M^{\min}$. □

We are ready to prove our main theorem.

**Theorem 1** *The presented algorithm is $(1 + \alpha)$-competitive for the known total processing time scheduling problem. Moreover, the bound is best possible if the number of machines tends to infinity.*

*Proof* In order to prove the theorem, we have to show that (i) all items are assigned to bins and that (ii) all items fit into the bins. We will prove these properties with respect to the Phases 1, 2a, and 2b.

Recall that Phase 1 is well defined, i.e., due to Lemma 1(g) items can be assigned to empty bins and (i) holds. By Lemma 1(a) the weights of all bins closed in Phase 1 are at most $(1 + \alpha)LB$, i.e., (ii) holds.

By definition all items are assigned in Phase 2a. In particular, if there is only one open bin left in Phase 2a, all remaining items are packed in this bin. By Lemma 2(d), the weights of all bins are at most $(1 + \alpha)LB$. This implies (i) and (ii) for Phase 2a.

Corollary 2 shows all assigned items fit into the corresponding bins during Phase 2b which is equivalent to (ii). Assume that (i) is not true for Phase 2b. Hence, there must be an item which cannot be assigned to any of the bins, i.e., the algorithm stops without assigning the item. Let this item be denoted by $j$. Item $j$ together with the remaining items arriving after $j$, shall be denoted by $R_j$.

While $\mu \geq 2$, each item which cannot be assigned during Steps 2–6 of Phase 2b, is automatically assigned in Step 7 to an $M$-bin. Hence, $\mu \leq 1$, if the algorithm stops. While $\tau \geq 2$, each item which cannot be assigned during Steps 2 or 3 of Phase 2b, is assigned either in Step 4 or in Step 5 to a batch. Consequently, we must have $\mu \leq 1$ and $\tau \leq 1$ at time $j$.

By definition and Lemma 5(a) only bin $M^{\max}$ and batch $\mathcal{B}_r$ are possibly not closed at time $j$.

Consider $\tau = 0$. Since batch $\mathcal{B}_r$ is never closed in Step 6 of Phase 2b, the only possibility is that there are no batches at the beginning of Phase 2b. This implies that there are no small bins at the beginning of Phase 2b. But this case is considered in Phase 2a. Hence, we assume in the following that $\tau = 1$.

Consider $\mu = 0$. We will show that the items of $R_j$ fit in batch $\mathcal{B}_r$ which is a contradiction to the assumption that the algorithm stops at time $j$. If $\mathcal{B}_r$ consists of four bins $B_1, \ldots, B_4$, we have by Lemma 3(b) that $w^j(B_1)+w^j(B_2)+w^j(B_3) \geq \frac{9}{4}\alpha + \frac{3}{2} \approx 2.816$. Because of Corollary 1 the average weight of all closed bins is greater than 1. Therefore, $w(R_j) \leq 1.2$ and the items of $R_j$ fit in $B_4$.

The proofs for less than four bins are analogous: If $\mathcal{B}_r$ consists of three bins, we have by Lemma 3(a) that $w^j(B_1)+w^j(B_2) \geq \frac{3}{2}\alpha + 1 \approx 1.878$. By Corollary 1 we get that $w(R_j)+w^j(B_3) \leq 1.2$ and the items of $R_j$ fit in $B_3$. For two bins we have again by Lemma 3(a) that $w^j(B_1)+w^j(B_2) \geq \frac{3}{2}\alpha + 1$. By Corollary 1 we get that $w(R_j) \leq 0.2$ and the items of $R_j$ fit in the bin with smaller weight. For one bin the assertion is obvious.

Finally, consider $\mu = 1$ and $\tau = 1$. Assume $\mathcal{B}_r$ consists of four bins $B_1, \ldots, B_4$. An item was put in $B_4$ since it did not fit in $B_3$. Thus, $w^j(B_3) + w^j(B_4) > 1 + \alpha$. Hence by Lemma 3(a) we have

$$w^j(B_1) + w^j(B_2) + w^j(B_3) + w^j(B_4)$$
$$> \frac{3}{2}\alpha + 1 + 1 + \alpha \approx 3.46.$$

Corollary 1 implies that $w(R_j) < 0.5$ and the items of $R_j$ fit in $M^{\max}$. For two and three bins in $\mathcal{B}_r$ the assertion follows with Lemma 3(a) and Lemma 3(b), respectively. Assume $\mathcal{B}_r$ consists of a single bin $B_1$. By Observation 1 the weights of the bins $B_1$ and $M^{\max}$ are each greater than $\alpha$ with total weight being equal to $2\alpha + \delta$ for some $\delta \geq 0$. Thus, the total weight of the items in $R_j$ is at most $2 - 2\alpha - \delta$ and they can be assigned to the bin with smaller weight since

$$(2\alpha + \delta)/2 + 2 - 2\alpha - \delta = 2 - \alpha - \delta/2 < 1 + \alpha.$$

This completes the proof of our main theorem. □

## References

Albers, S. (2013). Recent advances for a classical scheduling problem, In *Automata, languages and processing*, Lecture Notes in Computer Science (Vol. 7966, pp. 4–14). Berlin: Springer.

Albers, S., & Hellwig, M. (2012). Semi-online scheduling revisited. *Theoretical Computer Science*, *443*, 1–9.

Angelelli, E., Nagy, A. B., Speranza, M. G., & Tuza, Z. (2004). The on-line multiprocessor scheduling problem with known sum of the tasks. *Journal of Scheduling*, *7*, 421–428.

Azar, Y., & Regev, O. (2001). On-line bin-stretching. *Theoretical Computer Science*, *268*, 17–41.

Böhm, M., Sgall, J., van Stee, R., & Veselý, P. (2015). Better algorithms for online bin stretching. In *Approximation and Online Algorithms*, Lecture Notes of Computer Science (Vol. 8952).

Cheng, T. C. E., Kellerer, H., & Kotov, V. (2005). Semi-on-line multiprocessor scheduling with given total processing time. *Theoretical Computer Science*, *337*, 134–146.

Faigle, U., Kern, W., & Turan, G. (1989). On the performance of on-line algorithms for partition problems. *Acta Cybernetica*, *9*, 107–119.

Fleischer, R., & Wahl, M. (2000). On-line scheduling revisited. *Journal of Scheduling*, *3*, 343–353.

Graham, R. L. (1966). Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, *45*, 1563–1581.

Graham, R. L. (1969). Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, *17*, 263–269.

Hochbaum, D. S., & Shmoys, D. B. (1987). Using dual approximation algorithms for scheduling problems: theoretical and practical results. *Journal of the ACM*, *34*, 144–162.

Kellerer, H., & Kotov, V. (2013). An efficient algorithm for bin stretching. *Operations Research Letters*, *41*, 343–346.

Kellerer, H., Kotov, V., Speranza, M. G., & Tuza, Z. (1997). Semi on-line algorithms for the partition problem. *Operations Research Letters*, *21*, 235–242.

Rudin, III J. F. (2001). Improved bounds for the on-line scheduling problem. Ph.D. thesis, University of Texas