Department of Applied Mathematics
Faculty of EEMCS

University of Twente
The Netherlands

P.O. Box 217
7500 AE Enschede
The Netherlands
Phone: +31-53-4893400
Fax: +31-53-4893114
Email: memo@math.utwente.nl
www.math.utwente.nl/publications

Memorandum No. 1808

Quality of move-optimal schedules for minimizing the vector norm of the workloads

T. Brueggemann and J.L. Hurink

July, 2006

ISSN 0169-2690

Quality of Move-Optimal Schedules for Minimizing the Vector Norm of the Workloads

Tobias Brueggemann [1,*], Johann L. Hurink [2].

Department of Applied Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

Abstract

We study the problem of minimizing the vector norm $||\cdot||_p$ of the workloads. We examine move-optimal assignments and prove a performance guarantee of

$$\frac{2^p-1}{p}\cdot\left(\frac{p-1}{2^p-2}\right)^{\frac{p-1}{p}},$$

for any integer $p>1$ and moreover, we show that this guarantee is tight.

Additionally, we consider assignments obtained by applying the LPT-heuristic of Graham (1969). We prove that an LPT-assignment has a performance guarantee of

$$\frac{3^p-2^p}{p}\cdot\left(\frac{p-1}{2\cdot 3^p-3\cdot 2^p}\right)^{\frac{p-1}{p}},$$

which reproves a result of Chandra and Wong (1975).

Key words: parallel machines, vector of workloads, vector norm, approximation, local search.
MSC2000: 90B35, 68W25.

* Corresponding author.
Email addresses: t.brueggemann@math.utwente.nl (Tobias Brueggemann), j.l.hurink@math.utwente.nl (Johann L. Hurink).

[1] supported by the Netherlands Organization for Scientific Research (NWO) grant 613.000.225 (Local Search with Exponential Neighborhoods).
[2] supported by BSIK grant 03018 (BRICKS: Basic Research in Informatics for Creating the Knowledge Society).

1 Introduction

We consider the problem of assigning n jobs $1, \ldots, n$ to m parallel identical machines. Each job j has a given processing time p_j. For a given assignment A, we denote by $M_i := \{j : A(j) = i\}$ the set of jobs processed by machine i. The workload L_i of a machine i is defined to be

$$L_i := \sum_{j \in M_i} p_j.$$

For the rest of this text, where not stated otherwise, let p be a real value with $p > 1$. Let $L = (L_1, \ldots, L_m)$. The objective value of an assignment A is calculated as

$$f_p(A) := ||L||_p = \left(\sum_{i=1}^{m} |L_i|^p \right)^{1/p}.$$

The task is to find an assignment with minimum objective value.

One way to receive approximative solutions is inspired by the idea of local search. One starts with an arbitrary assignment of jobs to machines and tries to move a job from its designated machine to another one. This move is applied if the objective value decreases. The process is iteratively repeated until no such job can be found. The assignment obtained at the end of this iterative improvement procedure we call a *move-optimal* assignment.

We prove the following theorem.

Theorem 1 *Let p be an integer with $p > 1$, A be a move-optimal assignment and A^* be an optimal assignment, then $f_p(A)/f_p(A^*) \leq R$ holds, with*

$$R := \frac{2^p - 1}{p} \cdot \left(\frac{p-1}{2^p - 2} \right)^{\frac{p-1}{p}}.$$

Moreover, this bound is tight.

Observe, that $\lim_{p \to \infty} R = 2$. We have the following values for several p and R:

p	2	3	4	10	100	1000
R	$\frac{3}{4}\sqrt{2} < 1.061$	$\frac{7}{9}\sqrt[3]{3} < 1.122$	1.182	1.447	1.892	1.985

Another possibility to receive approximative solutions is by using the LPT-algorithm of Graham [2]. For the LPT-assignments obtained by this algorithms, the following theorem gives a performance guarantee.

Theorem 2 *Let p be an integer with $p > 1$, A be a LPT-assignment and A^**

be an optimal assignment, then $f_p(A)/f_p(A^*) \leq T$ *holds, with*

$$T := \frac{3^p - 2^p}{p} \cdot \left(\frac{p-1}{2 \cdot 3^p - 3 \cdot 2^p}\right)^{\frac{p-1}{p}}.$$

This theorem was proven by Chandra and Wong [1] in a different way.

In Section 2 we prove a lower bound on $f_p(A)$ and introduce some properties that are useful to prove Theorem 1. In Section 3 we examine move-optimal assignments and derive the upper bound on the performance guarantee. In Section 4 we present the instances attaining this performance guarantee. In Section 5 we give the proof of Theorem 2, which uses ideas of the proof of Theorem 1.

2 Preliminaries and Lower Bound

Let A be an assignment. For the rest of this text we consider only scaled instances, so that

$$\sum_{j=1}^{n} p_j = \sum_{i=1}^{m} L_i = m. \tag{1}$$

This can be achieved without changing the ratio of $f_p(A)/f_p(A^*)$. The following lemma gives a lower bound on the objective value of any assignment.

Lemma 3 *Let A be an assignment of jobs to machines. Then*

$$f_p(A) \geq m^{1/p}.$$

PROOF. Let $q = \frac{p}{p-1}$. It holds $1/p + 1/q = 1$. Let $I = (1, \ldots, 1)$ be a vector of length m. Due to a special case of Hölder's inequality (see Hölder [3]),

$$||L||_p \cdot ||I||_q = \left(\sum_{i=1}^{m} L_i^p\right)^{1/p} \cdot ||I||_q \geq \sum_{i=1}^{m} L_i$$

holds. Since (1) and $||I||_q = m^{1/q}$, this simplifies to $f_p(A) \cdot m^{1/q} \geq m$. Thus, we have $f_p(A) \geq m^{1-1/q} = m^{1/p}$. $\square$

The following fact is needed further on. Let $f(x) = x^p$ with $p > 0$ and let a, b, h be values with $a > b$ and $h > 0$. Then

$$f(a+h) + f(b) > f(a) + f(b+h). \tag{2}$$

This holds since $f'(x)$ is strictly monotone increasing.

3 Move-Optimal Assignments

In the following, we consider a move-optimal assignment A with $L_1 \geq \ldots \geq L_m$. Let Δ_i denote the deviation of the workload of machine i to the minimal workload L_m, i.e. $\Delta_i = L_i - L_m$.

We use the move-optimality of A to yield a lower bound for the processing times of jobs.

Lemma 4 *Let A be a move-optimal assignment. For all jobs $j \in M_i$ we have $p_j \geq \Delta_i$.*

PROOF. Assume, for a job $j \in M_i$ we have $p_j < \Delta_i$. Consider the assignment A' arising from A by assigning job j to machine m instead of i. Let L and L' be the vectors of workloads corresponding to assignments A and A', respectively. By comparing the objective values of A and A', we get $f_p(A)^p - f_p(A')^p = L_i^p - (L_i')^p + L_m^p - (L_m')^p = (L_i' + p_j)^p - (L_i')^p + L_m^p - (L_m + p_j)^p$. By using $a := L_i' = L_i - p_j$, $b := L_m$ and $h := p_j$, we know due to (2), that $f_p(A)^p > f_p(A')^p$. This contradicts the move-optimality of assignment A. □

The next lemma shows that, if we can bound Δ_i in terms of L_m for all machines i, we obtain an upper bound for $f_p(A)/f_p(A^*)$, for integer values of p.

Lemma 5 *Let p be an integer with $p > 1$. If $\Delta_i < cL_m$ holds for some $c > 0$ and all $i = 1, \ldots, m$, then*

$$\frac{f_p(A)}{f_p(A^*)} \leq \frac{(1+c)^p - 1}{c \cdot p} \cdot \left(\frac{c \cdot (p-1)}{(1+c)^p - c - 1} \right)^{\frac{p-1}{p}}$$

holds.

PROOF. For the workloads we have:

$$\begin{aligned}
\sum_{i=1}^{m} (L_i)^p &= \sum_{i=1}^{m} (\Delta_i + L_m)^p = \sum_{i=1}^{m} \sum_{k=0}^{p} \binom{p}{k} \Delta_i^k \cdot L_m^{p-k} \\
&= mL_m^p + \sum_{i=1}^{m} \sum_{k=1}^{p} \binom{p}{k} \cdot \Delta_i^k \cdot L_m^{p-k} \\
&\leq mL_m^p + \sum_{i=1}^{m} \sum_{k=1}^{p} \binom{p}{k} c^{k-1} L_m^{k-1} \cdot \Delta_i \cdot L_m^{p-k} \\
&= mL_m^p + \frac{L_m^{p-1}}{c} \sum_{k=1}^{p} \binom{p}{k} c^k \sum_{i=1}^{m} \Delta_i,
\end{aligned}$$

and by using $\Delta_i = L_i - L_m$, this leads to

$$\begin{aligned}\sum_{i=1}^{m}(L_i)^p &\le mL_m^p + \frac{L_m^{p-1}}{c}\sum_{k=1}^{p}\binom{p}{k}c^k(m - mL_m)\\ &= mL_m^p + \frac{L_m^{p-1}}{c}(m - mL_m)\Big[(1+c)^p - 1\Big]\\ &= m\left(\frac{c+1-(1+c)^p}{c}L_m^p + \frac{(1+c)^p-1}{c}L_m^{p-1}\right).\end{aligned}$$

This becomes maximal for $L_m = \Omega$ with

$$\Omega := \frac{p-1}{p}\cdot\frac{(1+c)^p-1}{(1+c)^p-c-1}.$$

We obtain:

$$\begin{aligned}f_p(A)^p &= \sum_{i=1}^{m}(L_i)^p\\ &\le m\Omega^p\left[\frac{c+1-(1+c)^p}{c} + \frac{1}{\Omega}\cdot\frac{(1+c)^p-1}{c}\right]\\ &= m\Omega^p\left[\frac{c+1-(1+c)^p}{c} + \frac{p}{p-1}\cdot\frac{(1+c)^p-c-1}{(1+c)^p-1}\cdot\frac{(1+c)^p-1}{c}\right]\\ &= m\Omega^p\left[\frac{(p-1)\cdot[c+1-(1+c)^p] + p\cdot[(1+c)^p-c-1]}{c(p-1)}\right]\\ &= m\Omega^p\left[\frac{(1+c)^p-c-1}{c(p-1)}\right].\end{aligned}$$

By using Lemma 3, we get

$$\begin{aligned}\frac{f_p(A)}{f_p(A^*)} &\le \Omega\left[\frac{(1+c)^p-c-1}{c(p-1)}\right]^{1/p}\\ &= \frac{p-1}{p}\cdot\frac{(1+c)^p-1}{(1+c)^p-c-1}\cdot\left[\frac{(1+c)^p-c-1}{c(p-1)}\right]^{1/p}\\ &= \frac{(1+c)^p-1}{c\cdot p}\cdot\frac{c(p-1)}{(1+c)^p-c-1}\cdot\left[\frac{c(p-1)}{(1+c)^p-c-1}\right]^{-1/p}\\ &= \frac{(1+c)^p-1}{c\cdot p}\cdot\left(\frac{c(p-1)}{(1+c)^p-c-1}\right)^{\frac{p-1}{p}}.\end{aligned}$$ □

In order to give an upper bound on c so that $\Delta_i < cL_m$ for all $i = 1, \ldots, m$, in the following lemma we consider move-optimal assignments that assign many jobs to a single machine i, and show that the corresponding value Δ_i becomes small.

Lemma 6 *Let A be a move-optimal assignment and i be a machine with n_i assigned jobs. Then*

$$\Delta_i \leq \frac{1}{n_i - 1} L_m.$$

PROOF. Let j be a job with minimal processing time assigned to machine i. Then due to Lemma 4 holds

$$L_i - L_m = \Delta_i \leq p_j \leq \frac{1}{n_i} L_i.$$

This simplifies to

$$L_i \leq \frac{n_i}{n_i - 1} L_m.$$

Hence, we receive for Δ_i:

$$\Delta_i \leq \frac{1}{n_i} L_i \leq \frac{1}{n_i} \cdot \frac{n_i}{n_i - 1} L_m = \frac{1}{n_i - 1} L_m \qquad \square$$

While the previous lemma is suitable for many jobs assigned to a single machine, the next lemma concentrates on machines containing only one job.

Lemma 7 *Let I be an instance, A be a move-optimal assignment and A^* be an optimal assignment with $f_p(A)/f_p(A^*) > 1$. If there is only one job j assigned to a machine i with $L_i > 1$, then deleting job j and machine i strictly increases the ratio $f_p(A)/f_p(A^*)$.*

PROOF. It holds $p_j > 1$. We claim that job j is scheduled alone in any move-optimal assignment A'. Since A^* is also move-optimal, and by denoting with f' the objective values of the assignments restricted to the changed instance, this leads to

$$\frac{f'_p(A)^p}{f'_p(A^*)^p} = \frac{f_p(A)^p - p_j^p}{f_p(A^*)^p - p_j^p} > \frac{f_p(A)^p}{f_p(A^*)^p}.$$

In order to prove our claim, we assume to the contrary, that job j is not scheduled alone on machine i in a move-optimal assignment A'. Let j_0 be a job also scheduled on machine i. By using $p_j > 1 \geq L_m$, we obtain $p_{j_0} \leq L_i - p_j < L_i - L_m = \Delta_i$ contradicting Lemma 4. $\square$

Since we are interested in finding instances which maximize the right hand side of Lemma 5, by the previous Lemma 7 we only have to consider instances and move-optimal assignments such that

$$n_i \geq 2 \text{ for all machines } i \text{ with } L_i > 1.$$

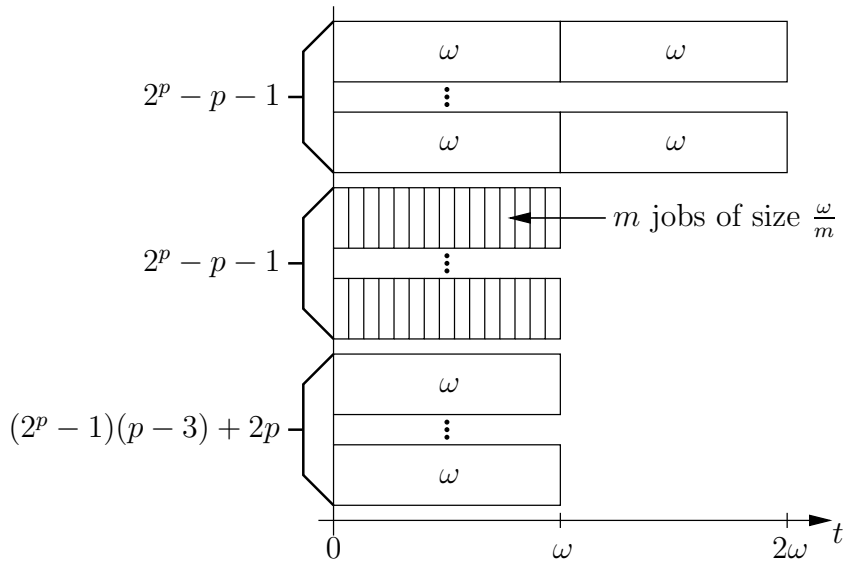

Fig. 1. Move-optimal assignment A

The next lemma proves the upper bound on the performance guarantee as given by Theorem 1.

Lemma 8 *Let p be an integer with $p > 1$. Let A be a move-optimal assignment and A^* be an optimal assignment, then $f_p(A)/f_p(A^*) \leq R$ holds.*

PROOF. In the case that $L_1 = 1$, it is $L_i = 1$ for all machines i and, thus $f_p(A)/f_p(A^*) = 1$.

On the other hand, consider $L_1 > 1$. Due to Lemma 7 we assume w.l.o.g. that at least 2 jobs are assigned to machine 1. With the help of Lemma 6 we obtain $\Delta_1 \leq L_m$. Moreover, it is $L_1 \geq L_i$ and thus, $L_m \geq \Delta_1 \geq \Delta_i$ for all machines $i = 1, \ldots, m$. Thus, we may choose $c = 1$ in Lemma 5, and the desired bound R is obtained. □

4 Worst-Case Instances

Let p be an integer with $p > 1$. Consider the following example with $m = (2^p - 1)(p - 1)$ machines. In the move-optimal assignment A we schedule on each of the first $2^p - p - 1$ machines 2 jobs with processing time ω, where

$$\omega := \frac{p-1}{p} \cdot \frac{2^p - 1}{2^p - 2}.$$

We assign to each of the next $2^p - p - 1$ machines m jobs of processing time ω/m. We schedule on each of the last $(2^p - 1)(p - 3) + 2p$ machines 1 jobs

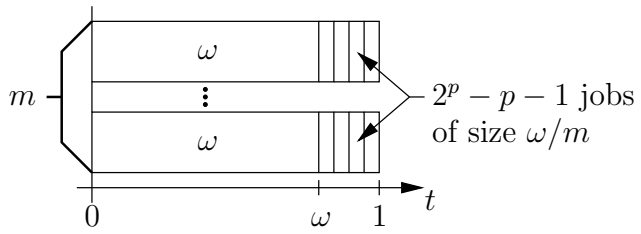

Fig. 2. Optimal assignment A^*

with processing time ω. For an illustration, see Figure 1. The objective value $f_p(A)$ is calculated as

$$\begin{aligned} f_p(A)^p &= (2^p - p - 1)(2\omega)^p + \omega^p \left[(2^p - 1)(p-2) + p\right] \\ &= \omega^p (2^p - 1)(2^p - 2). \end{aligned}$$

In the optimal assignment A^* we schedule on each of the m machines 1 job with processing time ω and $2^p - p - 1$ jobs with processing time ω/m For an illustration, see Figure 2. The objective value $f_p(A^*)$ is calculated as

$$\begin{aligned} f_p(A^*)^p &= m \left[\omega + \frac{\omega(2^p - p - 1)}{(2^p - 1)(p-1)}\right] \\ &= (2^p - 1)(p-1). \end{aligned}$$

Thus, for the ratio $f_p(A)/f_p(A^*)$ holds

$$\frac{f_p(A)}{f_p(A^*)} = \omega \left(\frac{2^p - 2}{p-1}\right)^{1/p} = \frac{2^p - 1}{p} \cdot \left(\frac{p-1}{2^p - 2}\right)^{\frac{p-1}{p}}.$$

This proves the tightness of the upper bound on the performance guarantee as given by Theorem 1.

5 LPT-Assignments

In the following part, we examine LPT-assignments and their performance guarantee on the problem of minimizing $f_p(A)$. We prove Theorem 2 by using properties of move-optimal assignments.

In the following, we assume that the jobs are ordered, so that $p_1 \geq \ldots \geq p_n$. For a given assignment A we define partial workloads of a machine i and any job j to be

$$L_{ij} := \sum_{\substack{k \in M_i \\ k \leq j}} p_k.$$

We denote with S_j the starting time of job j given by $S_j := L_{i,j-1}$. We re-order w.l.o.g. the machines, so that $L_1 \geq \ldots \geq L_m$. An LPT-assignment is

Algorithm: LPT

input: $p_1 \geq \ldots \geq p_n$.
output: an LPT-assignment A.
  for $j := 1$ **to** n **do**
  begin
    Determine i so that $L_{i,j-1} \leq L_{k,j-1}$ for all machines $k = 1, \ldots, m$;
    $A(j) := i$;
  end for;

Fig. 3. Algorithm LPT.

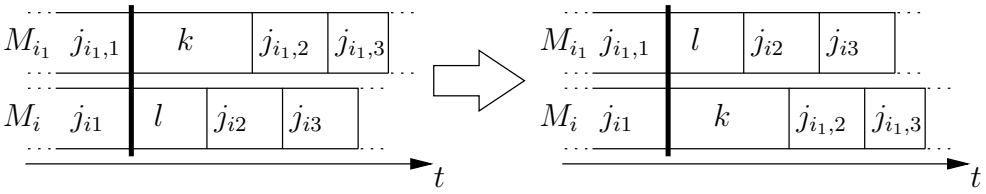

Fig. 4. Illustration of swapping schedules beginning with job k and l.

achieved by iteratively assigning the remaining job with largest processing time to a machine with minimal workload. The algorithm is given in Figure 3 and runs in $\mathcal{O}(n \log n)$ if the jobs are not sorted by their processing times and in $\mathcal{O}(n \log m)$ otherwise.

A direct consequence of this construction is that an assignment A is an *LPT-assignment*, if it allows a schedule for which the following hold:

- if for a pair of jobs j and k the relation $S_j > S_k$ holds, then: $p_j \leq p_k$,
- for every job j we have $S_j \leq L$ with $L := \min_{i=1}^{m} L_i$.

Because of the first property, we know that smaller jobs receive a larger starting time. Combining the first with the second property yields, that any LPT-assignment is also move-optimal and we may replace the second property by the move-optimality property.

Furthermore, we make the following observations. If we have two jobs k and l with $S = S_k = S_l$ we may swap the schedules from S onwards on the two machines jobs k and l are assigned to, without changing the objective value or any starting time of a job and without loosing the LPT property (see Figure 4).

In the following, we examine the effects of slightly changing the processing time of a job j_1. More precisely, we replace p_{j_1} by $p'_{j_1} = p_{j_1} - \varepsilon$ and all other processing times remain the same (i.e. $p'_j = p_j$ for all jobs $j \neq j_1$). To have better control and to loose not the LPT property, we have to ensure that the machine i job j is assigned to has a special schedule. More precisely, if on machine i a job k is scheduled after job j, and if some other job l has $S_l = S_k$, we must have $p_k \geq p_l$. Due to the above made observation, such a situation

can always be ensured. We denote such schedules as *clean with respect to job* j.

The next lemma shows, that the assignment A remains an LPT-assignment if ε is chosen sufficiently small. Moreover, the new objective value $f'_p(A)$ of assignment A in the changed instance is only a slight modification of the value $f_p(A)$.

Lemma 9 *Let j_1 be a job with $p_1 \geq \ldots \geq p_{j_1} > p_{j_1+1} \geq \ldots \geq p_n$ and let A be an LPT-assignment which is clean w.r.t. job j_1. For a sufficiently small $\varepsilon > 0$, the assignment A remains an LPT-assignment if the processing time of job j_1 is changed to $p'_{j_1} = p_{j_1} - \varepsilon$.*

The objective value $f'(A)$ in the changed instance is given by

$$f'_p(A)^p = f_p(A)^p + (L_{A(j_1)} - \varepsilon)^p - (L_{A(j_1)})^p.$$

Observe, that $f'_p(A)^p < f_p(A)^p$.

PROOF. Let $i_1 := A(j_1)$ be the machine job j_1 is assigned to. We define M as the set of jobs that follow job j_1 on machine i_1; i.e.

$$M := \{j \in M_{i_1} : \ j \geq j_1 + 1\}.$$

Let $\varepsilon > 0$ be so that the following hold:

(P1) $p_{j_1} - \varepsilon > \max\{p_{j_1+1}, 0\}$,
(P2) for all pairs (j, k) of jobs with $j \notin M$, $k \in M$ and $S_j < S_k$ we have $S_j \leq S_k - \varepsilon$,
(P3) for all jobs j with $S_j + p_j = L_{A(j)}$ and $S_j < L_{i_1}$ we have $S_j \leq L_{i_1} - \varepsilon$.

Because the assignment A is clean w.r.t. job j_1, an $\varepsilon > 0$ with the above given properties exists and the assignment A remains an LPT-assignment for the changed instance.

We now turn to the objective value of $f'_p(A)^p$. Since it is $L'_{A(j_1)} = L_{A(j_1)} - \varepsilon$, the statement of the lemma follows. □

The previous lemma shows, that we can decrease a processing time by some ε without loosing the LPT-property. As in Section 3, we are interested in an upper bound for the ratio $f_p(A)/f_p(A^*)$. Again, we scale the processing times and weights so that

$$\sum_{j=1}^{n} p_j = m,$$

without changing the value of $f_p(A)/f_p(A^*)$. Since an LPT-assignment is also move-optimal, we may use the results of Section 3, i.e. an upper bound on $f_p(A)/f_p(A^*)$ is given by Lemma 5. In the following, we look for an upper bound on c to get a best possible value for the upper bound in Lemma 5. In order to do so, the next lemma shows, that we can leave out of consideration certain instances.

Lemma 10 *Let I be an instance, A be an LPT-assignment and A^* be an optimal assignment with $f_p(A)/f_p(A^*) > 1$. If $L_1^{A^*} \geq L_1^A$, then there exists an instance I' and a corresponding LPT-assignment B with*

$$\frac{f'_p(B)}{f'_p(B^*)} > \frac{f_p(A)}{f_p(A^*)},$$

where f' denotes the objective function and B^* an optimal assignment for the instance I'.

PROOF. Let $j_1 \in M_1^{A^*}$ be a job processed by machine 1. We assume w.l.o.g. that A is clean w.r.t. job j_1. Furthermore, by exchanging indices with job j_1, we assume w.l.o.g. that $p_1 \geq \ldots \geq p_{j_1} > p_{j_1+1} \geq \ldots \geq p_n$ holds. Due to Lemma 9, there exists an $\varepsilon > 0$ such that in the changed instance with $p'_j := p_j$ for all j$\neq j_1$ and $p'_{j_1} := p_{j_1} - \varepsilon$, the assignment A remains an LPT-assignment and for the objective value of A in the changed instance holds

$$f'_p(A)^p = f_p(A)^p + (L_{A(j_1)} - \varepsilon)^p - (L_{A(j_1)})^p.$$

For assignment A^* we receive in the changed instance a workload $L'_1 = L_1^{A^*} - \varepsilon$. The objective value $f'_p(A^*)$ of assignment A^* in the changed instance thus calculates as

$$f'_p(A^*)^p = f_p(A^*)^p + (L_1^{A^*} - \varepsilon)^p - (L_1^{A^*})^p.$$

Furthermore, let B^* be the optimal assignment for the changed instance. This implies $f'_p(A^*) \geq f'_p(B^*)$. Therefore, we receive for the ratio of the objective values of assignments A and A^*:

$$\frac{f'_p(A)^p}{f'_p(B^*)^p} \geq \frac{f'_p(A)^p}{f'_p(A^*)^p} = \frac{f_p(A)^p + (L_{A(j_1)} - \varepsilon)^p - (L_{A(j_1)})^p}{f_p(A^*)^p + (L_1^{A^*} - \varepsilon)^p - (L_1^{A^*})^p} > \frac{f_p(A)^p}{f_p(A^*)^p},$$

where the last inequality is due to $(L_1^{A^*})^p - (L_1^{A^*} - \varepsilon)^p \geq (L_{A(j_1)})^p - (L_{A(j_1)} - \varepsilon)^p$, $f_p(A) > f_p(A^*)$ and (2). $\square$

The next lemma is crucial for Theorem 2.

Lemma 11 *Let A be an LPT-assignment and A^* an optimal assignment with $L_1^A > L_1^{A^*}$. Then*

$$\Delta_i \leq \frac{1}{2} L_m \text{ for all machines } i = 1, \ldots, m.$$

PROOF. At first we show that machine 1 contains at least 2 jobs. For this, assume that $n_1 = 1$ and let j be the only job assigned to machine 1. Then $L_1 = p_j \geq L_i$ for all $i = 1, \ldots, m$. For the machine i with maximal workload according to the optimal assignment A^* then holds $L_i^{A^*} \geq p_j = L_1^A$ which is a contradiction. Therefore, machine 1 contains at least 2 jobs.

Since an LPT-assignment A is also move-optimal, we know by Lemma 6 that if many jobs are assigned to a single machine i, the value Δ_i becomes small:

$$\Delta_i \leq \frac{1}{n_i - 1} L_m.$$

Since $\Delta_1 \geq \ldots \geq \Delta_m$, we only have to consider the case $n_1 = 2$. Let j_1 and j_2 be the two jobs assigned to machine 1 according to assignment A. We assume w.l.o.g. that $p_{j_1} \geq p_{j_2}$. Moreover, let p and x be values, so that $p = p_{j_1}$ and $xp = p_{j_2}$. Hence, the workload of machine 1 equals $L_1 = (1 + x)p$, where x takes values out of the interval $x \in]0, 1]$.

Since A is an LPT-assignment, all other machines $i \neq 1$ must have a workload of at least $L_i \geq p$. Therefore, $\Delta_i \leq xp \leq xL_m$ for all machines i and for $x \in]0, \frac{1}{2}]$ the statement follows.

Assume now that $x > \frac{1}{2}$. Let I_1 be the set of machines containing a job with processing time larger than p. Such a job is called a *big job*. Observe, that in the optimal assignment A^* no two big jobs may be assigned to the same machine. Since otherwise, we receive a makespan of at least $2p \geq L_1^A$, contradicting $L_1^A > L_1^{A^*}$.

Let $I_2 = \{1, \ldots, m\} \setminus I_1$ be the set of machines containing no big job. If $I_2 = \emptyset$, this again leads to a contradiction, since in this case we have at least m big jobs and the job j_2 with $p_{j_2} = xp$ leading to an optimal assignment A^* with makespan at least $(1 + x)p = L_1^A$, contradicting $L_1^A > L_1^{A^*}$. Thus, $I_2 \neq \emptyset$ and $x < 1$.

Since A is an LPT-assignment, all machines $i \in I_2$ contain at least 2 jobs with processing time at least xp. We call a job j with $p > p_j \geq xp$ a *blocking job*. Observe, that there are at least $2|I_2| + 1$ blocking jobs. In the optimal assignment A^* no blocking job can be combined with one or more of the $|I_1|$ big jobs. Because if we do so, we receive a makespan of at least L_1^A, contradicting $L_1^A > L_1^{A^*}$.

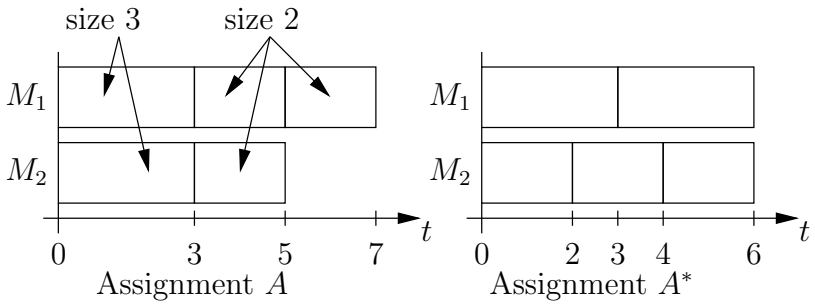

Fig. 5. Lower bound example on performance guarantee of LPT-assignments.

Hence, in order to receive an optimal assignment with $L_1^A > L_1^{A^*}$, we have to assign at least once, 3 blocking jobs to a single machine. Doing so, yields a makespan of at least $3xp$ which is greater or equal to L_1^A for $x > \frac{1}{2}$. Hence, this again contradicts $L_1^A > L_1^{A^*}$. Therefore, for $x > \frac{1}{2}$ there exists no optimal assignment with $L_1^A > L_1^{A^*}$. $\square$

In order to give an upper bound on the performance guarantee $f_p(A)/f_p(A^*)$, due to Lemma 10 we only have to consider instances where for the makespan of A and A^* holds $L_1^A > L_1^{A^*}$. Lemma 11 then guarantees, that

$$\Delta_i \leq \frac{1}{2} L_m \text{ for all machines } i = 1, \ldots, m.$$

Lemma 5 then yields the upper bound of Theorem 2.

The following lower bound example on the performance guarantee of LPT-assignments was given by Chandra and Wong [1] for the problem of minimizing $f_2(A)^2$. This example consists of 2 machines and 5 jobs. There are 2 jobs with processing times equal to 3 and 3 jobs with processing times equal to 2. The LPT-assignment shown in Figure 5 has an objective value of $f_p(A) = (7^p + 5^p)^{1/p}$. The optimal assignment A^* again has balanced machines yielding an objective value of $f_p(A^*) = 6\sqrt[p]{2}$. This together yields a ratio of

$$\frac{f_p(A)}{f_p(A^*)} = \frac{(7^p + 5^p)^{1/p}}{6\sqrt[p]{2}}.$$

Note, that for $p \to \infty$ this tends to 7/6, which equals the performance guarantee of LPT-assignments for 2 machines for the problem of minimizing the makespan, as shown by Graham [2].

If we change the processing times and weights of the three smaller jobs to a value slightly greater than 2 we can increase the lower bound a bit, but this has to be carried out for fixed values of p since for optimizing the lower bound we have to find zero crossings of some polynomial of degree depending on p.

Acknowledgments

The authors thank Paul S. Bonsma for his helpful comments.

References

[1] A. K. Chandra and C. K. Wong. Worst-case analysis of a placement algorithm related to storage allocation. *SIAM Journal on Computing*, 4(3):249–263, 1975.

[2] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17:416–429, 1969.

[3] O. Hölder. Über einen Mittelwertsatz. *Nachrichten von der Königl. Gesellschaft der Wissenschaften und der Georg-Augusts-Universitt zu Göttingen*, 1889:38–47, 1889.