Online Scheduling Revisited

Rudolf Fleischer[1] Michaela Wahl[2]

[1] University of Waterloo, Department of Computer Science
email: rudolf@uwaterloo.ca

[2] Max-Planck-Institut für Informatik, Saarbrücken
email: altherr@mpi-sb.mpg.de

Abstract. We present a new online algorithm, MR, for non-preemptive scheduling of jobs with known processing times on m identical machines which beats the best previous algorithm for $m \geq 64$. For $m \to \infty$ its competitive ratio approaches $1 + \sqrt{\frac{1+\ln 2}{2}} < 1.9201$.

1 Introduction

Scheduling problems are of great practical interest. However, even some of the simplest variants of the problem are not fully understood. In this paper, we study the classical problem of scheduling jobs online on m identical machines without preemption, i.e., the jobs arrive one at a time with known processing times and must immediately be scheduled on one of the machines, without knowledge of what jobs will come afterwards, or how many jobs are still to come. The goal is to achieve a small *makespan* which is the total processing time of all jobs scheduled on the most loaded machine. Since the jobs must be scheduled online we cannot expect to achieve the minimum makespan whose computation would require a priori knowledge of all jobs (even then computing the minimum makespan is difficult, i.e., NP-hard [12]). The quality of an online algorithm is therefore measured by how close it comes to that optimum [4, 10]. It is said to be *c-competitive* if its makespan is at most c times the optimal makespan for all possible job sequences.

Graham [14] showed some 30 years ago that the List algorithm which always puts the next job on the least loaded machine is exactly $(2 - \frac{1}{m})$-competitive. Only much later were better algorithms designed. The first improvement was a $(2 - \frac{1}{m} - \epsilon_m)$-competitive algorithm by Galambos and Woeginger [11, 5], with $\epsilon_m \to 0$ for $m \to \infty$. The first algorithm with competitive ratio approaching a value strictly smaller than 2 for large m was a

$(2 - \frac{1}{70})$-competitive algorithm (for $m \geq 70$) by Bartal et al. [2]. Note that $2 - \frac{1}{70} \approx 1.986$. Not much later Karger et al. [16] proposed the algorithm $\texttt{CHASM}_\alpha$ with competitive ratio somewhere between 1.9378 and 1.945. The last improvement so far was Albers' [1] 1.923-competitive algorithm `M2`.

On the other hand, the lower bound for the problem was raised in similarly small steps: Faigle et al. [9] showed a lower bound of 1.707 for $m \geq 4$ for any deterministic online algorithm. This was 'greatly' improved by Bartal et al. [3] who showed a lower bound of 1.837. In the next step Albers [1] raised the bar to 1.852. And recently Gormley et al. [13] applied exhaustive computer search to find a slight perturbation of Albers' hard request sequence which yields a lower bound of 1.85358.

Better bounds are known for a few special cases. For $m = 2$ and $m = 3$, Faigle et al. [9] gave a lower bound of $(2 - \frac{1}{m})$, i.e., `List` is optimal in these cases. And for $m = 4$, Chen et al. [7] proposed a 1.733-competitive algorithm. The best lower bound for randomized algorithms is $\frac{e}{e-1} \approx 1.58$ for large m, proved independently by Chen et al. [6] and Sgall [18], and Seiden [17] showed that at least for $m \leq 5$ randomized algorithms can beat the best deterministic lower bound. For scheduling with preemption the competitive ratio is exactly $\frac{e}{e-1}$ for $m \to \infty$ [8]. For more results on scheduling see the survey chapters in the recent books [4, 10, 15].

In this paper we propose another small improvement on the upper bound of the competitiveness of the online scheduling problem, decreasing it to $1 + \sqrt{\frac{1+\ln 2}{2}} < 1.9201$ for $m \to \infty$. For $m \geq 64$ we beat the best previous bound of 1.923 [1]. Our new algorithm, called `MR` (the authors' initials), tries to schedule jobs either on the least loaded machine as long as its load is relatively low, or on a certain machine with medium-high load if it can safely do this. This is explained in Section 2. In Sections 3 and 4 we establish the competitiveness of `MR`. The proof is quite straightforward and relatively simple (compared to proofs of previous algorithms). In particular, we give an intuitive explanation for our choice of the crucial parameters of `MR` optimizing its performance. And contrary to some of the recent papers [1, 16] we even derive a closed formula for the competitive ratio c.

The proof for the competitiveness of `MR` builds on three elementary lower bounds on the optimal makespan, namely the average total load, the size of the largest job, and twice the size of the $(m + 1)$th largest job (inequalities (C1)-(C3) in Section 3). The proofs for all previous algorithms use the same set of bounds (except for the proof of Graham's *List* algorithm which only

needs (C1) and (C2)). We conjecture that better upper bounds are possible if more lower bounds are added to this list (for example one involving the size of the $(2m+1)$th largest job), but currently we do not know how to do this.

2 The Algorithm

We assume that at each time step $t = 1, 2, 3, \ldots$ a new job J^t with processing time p^t (also called the *size* of the job) arrives which must be scheduled online on one of m identical machines. The *current schedule* at time t is an assignment of the first t jobs on the machines. The *load* of a machine at time t is the sum of the sizes of the jobs assigned to it in the current schedule. We assume that the machines are always ordered by non-increasing load, i.e., M_1^t is the machine with the highest load and M_m^t is the machine with the lowest load, after J^t was scheduled. The load of M_j^t at time t will be denoted by l_j^t, for $1 \leq j \leq m$. Thus, l_1^t is the makespan of the current schedule at time t. At time $t = 0$, all machines have load 0. For $j = 1, \ldots, m$ let D_j^t be the average load of machines $M_j^t, \ldots, M_m^t$ at time t, i.e.,

$$D_j^t = \frac{1}{m-j+1} \cdot \sum_{k=j}^{m} l_k^t \,.$$

Let

$$D^t = D_1^t$$

be the average load of all machines at time t (which is independent of the current schedule).

Our new algorithm `MR` is parameterized by the competitive ratio c we hope to achieve. We show in the next section that `MR` works well if we choose

$$c \geq 1 + \sqrt{\tfrac{1+\ln 2}{2}} \,.$$

Let

$$i = \left\lceil \tfrac{5c-2c^2-1}{c} \cdot m \right\rceil - 1$$

be a 'magic' number which will also be explained in the next section, and let

$$k = 2i - m \,.$$

We call `MR`'s schedule *flat* at time t if

$$l_k^{t-1} < \frac{2(c-1)}{2c-3} \cdot D_{i+1}^{t-1} ,$$

i.e., if before scheduling J^t the load of the kth largest machine is not much higher than the average load of the $m-i$ smallest machines. Otherwise we call it *steep*. We say J^t is *scheduled flatly (steeply)* if MR's schedule is flat (steep) at time t.

As the worst case example for Graham's List algorithm shows [14] one should try to avoid situations were all machines have approximately the same load. So we try to make flat schedules steeper again by scheduling new jobs on some medium-loaded machine, if possible. All previous algorithms [11, 2, 16, 1] are based on this basic idea, however they differ in their definitions of flatness and medium-loadedness.

Algorithm MR

Let p^t be the size of the next job J^t to be scheduled.
If the current schedule is steep or if $p^t + l_i^{t-1} > c \cdot D^t$ then schedule J^t on the smallest machine M_m^{t-1}, else schedule it on M_i^{t-1}.

Note that l_i^{t-1} is the load of the ith largest machine prior to scheduling J^t, and D^t is the average load of all machines after scheduling J^t.

Theorem 1 (Main theorem). *The competitive ratio of* MR *approaches* $1 + \sqrt{\frac{1+\ln 2}{2}} < 1.9201$ *for* $m \to \infty$. □

The proof of this theorem is given in the next section.

3 Proof of the Main Theorem

Let

$$c = 1 + \sqrt{\tfrac{1+\ln 2}{2}} \approx 1.9201 .$$

Then

$$i \approx 0.639 \cdot m ,$$

$$k \approx 0.278 \cdot m \ ,$$

and

$$\frac{2(c-1)}{2c-3} \approx 2.19 \ .$$

The following proof is done under the assumption that $m \to \infty$ but it can easily be adjusted to finite m, however at the cost of a slightly higher value of c (Lemmas 8 and 9 are the critical ones). Note that we can therefore ignore the rounding in the definition of i.

Consider an arbitrary sequence of jobs to be scheduled online. Let t^f be the number of jobs in this sequence. We may assume w.l.o.g. that MR's makespan is defined by the machine receiving the last job J^{t^f}. Let

$$a = p^{t^f}$$

be the size of J^{t^f} and let b be the load of the machine which receives J^{t^f} (not including p^{t^f}). Then MR's final makespan is $a + b$. Let

$$D = D^{t^f}$$

be the final average load of all machines. Furthermore, let P_{m+1} be the size of the $(m+1)$th largest job in the sequence; if $t^f \leq m$ then $P_{m+1} = 0$.

We know the following three lower bounds for the optimal makespan (the first two were introduced by Graham [14], the third seems to be first used by Galambos and Woeginger [11]): D, a, and $2P_{m+1}$. Thus, the following three inequalities must hold for MR to fail to be c-competitive.

$$(C1) \quad a + b > c \cdot D \qquad \Leftrightarrow \quad a > c \cdot D - b$$

$$(C2) \quad a + b > c \cdot a \qquad \Leftrightarrow \quad a < \tfrac{b}{c-1}$$

$$(C3) \quad a + b > c \cdot 2P_{m+1}$$

We assume from now that (C1) and (C2) hold. We will show that then (C3) cannot hold, thus proving the Main Theorem. Since $a + b < \frac{c}{c-1} \cdot b$ by (C2) it suffices to show that $P_{m+1} \geq d$ where

$$d = \frac{b}{2(c-1)} \ .$$

Note that

$$b - d = \frac{2c-3}{2(c-1)} \cdot b \,,$$

and the schedule is flat at time t if

$$l_k^{t-1} < \frac{b}{b-d} \cdot D_{i+1}^{t-1} \,.$$

We call jobs of size at least d *big*. We prove $P_{m+1} \geq d$ by showing that the last job is big and that each machine receives at least one big job before the last job arrives.

We call a machine *full* if its current load is at least b, *half-full* if its current load is at least $b-d$ but less than b, and *half-empty* if its current load is less than $b-d$. Note that MR does not know whether a machine is full or not before the end of the sequence. A *filling job* is a job scheduled on a non-full machine which is full afterwards, i.e., the job *fills* the machine. Slightly abusing this term, we also call the last job a filling job. In the following we are only interested in filling jobs.
If the last job J^{t^f} were scheduled on $M_i^{t^f-1}$ then by definition of MR

$$a + b = p^{t^f} + l_i^{t^f-1} \leq c \cdot D^{t^f} = c \cdot D \,,$$

contradicting (C1). Therefore J^{t^f} is scheduled on the smallest machine $M_m^{t^f-1}$. In particular, all machines have load at least b before J^{t^f} is scheduled. Therefore, there are exactly $m+1$ filling jobs, and we prove that each of them is big. Note that any filling job scheduled on a half-empty machine is big. (C1) and (C2) together imply

$$b > (c-1) \cdot D$$

justifying the name 'half-full' for machines with load at least $b-d > (c-\frac{3}{2}) \cdot D$.

Lemma 2. *The last job is scheduled flatly.*

Proof. Assume the schedule is steep at time t^f. Then

$$\begin{aligned} l_k^{t^f-1} &\geq \frac{2(c-1)}{2c-3} \cdot D_{i+1}^{t^f-1} \\ &\geq \frac{2(c-1)}{2c-3} \cdot b \end{aligned}$$

and therefore

$$
\begin{aligned}
D &\geq \frac{k \cdot l_k^{t^f-1} + (m-k) \cdot b}{m} \\
&\geq \frac{k \cdot \frac{2(c-1)}{2c-3} \cdot b + (m-k) \cdot b}{m} \\
&> \frac{m + \frac{k}{2c-3}}{m} \cdot (c-1) \cdot D \\
&\approx 1.22 \cdot D \ ,
\end{aligned}
$$

a contradiction. □

For $j = 1, \ldots, m+1$ let t_j be the time the jth filling job arrives. And for $j = 1, \ldots, m$ let

$$D_j = D_j^{t_j - 1}$$

be the average load of machines $M_j^{t_j-1}, \ldots, M_m^{t_j-1}$ (which are exactly the non-full machines at that time) just prior to filling the jth machine. Clearly, $D_{i+1} \leq \cdots \leq D_m < D$.

From Lemma 2 we conclude that there is a minimal index $s \leq m$ such that $J^{t_{s+1}}, J^{t_{s+2}}, \ldots, J^{t_{m+1}}$ are scheduled flatly. If $s < i$ then we choose $s = i$.

Lemma 3. *$J^{t_{s+1}}, J^{t_{s+2}}, \ldots, J^{t_{m+1}}$ are big.*

Proof. Consider one of these jobs at time t. Since machine M_i^{t-1} is already full, J^t can only be filling if it is scheduled on M_m^{t-1}. If this machine is half-empty then J^t must be big to fill it. If it is half-full then $D^t = D$ if $t = t^f$, or

$$D^t \geq \frac{i \cdot l_i^{t-1} + b + (m-i-1) \cdot (b-d)}{m}$$

if $t < t^f$ because the first i machines have load at least l_i^{t-1}, one machine just becomes full, and all other machines are half-full; if $t = t^f$ then all machines have load at least b. But `MR` will only schedule J^t on the smallest machine if

$$
\begin{aligned}
p^t &> c \cdot D^t - l_i^{t-1} \\
&\geq c \cdot \tfrac{(i+1)\cdot b + (m-i-1)\cdot(b-d)}{m} - b \\
&= c \cdot b - \tfrac{m-i-1}{m} \cdot c \cdot d - b \\
&= 2(c-1)^2 \cdot d + (4c - 2c^2 - 1) \cdot d \\
&= d\,.
\end{aligned}
$$

The second inequality holds because $\frac{ci}{m} - 1 \approx 0.22 > 0$ and $l_i^{t-1} \geq b$. □

Note that in the proof of the lemma in the worst case $l_i^{t-1} = b$ and then our choice of $i = \frac{5c-2c^2-1}{c} \cdot m - 1$ is the smallest possible to prove the claim. The next lemma shows that $J^{t_{s+1}}, J^{t_{s+2}}, \ldots, J^{t_m}$ are scheduled on half-empty machines. This will imply in Lemma 5 that the first s filling jobs are also big. Since the proof of Lemma 4 is more involved we give it later in Section 4.

Lemma 4. *$D_{i+1} < b - d$. And if $s > i$ then $D_s < b - d$.*

Proof. See Section 4. □

Lemma 5. *$J^{t_1}, J^{t_2}, \ldots, J^{t_s}$ are big.*

Proof. We show that all these jobs are scheduled on a half-empty machine. Hence they must be big. Let $u = \max\{i+1, s\}$.

If $s > i$ then $J^{t_{i+1}}, J^{t_{i+2}}, \ldots, J^{t_s}$ are scheduled on the smallest machine because they are filling jobs and M_i is already full. But the smallest machine is half-empty just before time t_s by Lemma 4, so it is half-empty at any time before t_s.

For $j = k+1, \ldots, i$, the schedule is steep at time t_j because

$$
\begin{aligned}
l_k^{t_j-1} &\geq b \\
&= \frac{2(c-1)}{2c-3} \cdot (b-d) \\
&\overset{(L4)}{>} \frac{2(c-1)}{2c-3} \cdot D_{i+1} \\
&\geq \frac{2(c-1)}{2c-3} \cdot D_{i+1}^{t_j-1}\,.
\end{aligned}
$$

Hence $J^{t_{k+1}}, J^{t_{k+2}}, \ldots, J^{t_i}$ are also scheduled on the smallest machine.

This also implies $l_i^{t_k} \leq l_{i+1}^{t_{k+1}} \leq \cdots \leq l_{i+(m-i)}^{t_{k+(m-i)}} = l_m^{t_i}$. For $j = 1, \ldots, k$, J^{t_j} is either scheduled on $M_i^{t_j-1}$ or on $M_m^{t_j-1}$. But

$$l_m^{t_j-1} \leq l_i^{t_j-1} \leq l_i^{t_j} \leq l_i^{t_k} \leq l_m^{t_i} \leq D_{i+1} \overset{(L4)}{<} b - d$$

and hence $J^{t_1}, J^{t_2}, \ldots, J^{t_k}$ are scheduled on a half-empty machine. □

This concludes the proof of the Main Theorem.

4 Proof of Lemma 4

Let $\alpha = \frac{b}{D}$. We first consider the case $s = i$.

We assume $D_{i+1} \geq b - d$ and show that this would imply that all the jobs $J^{t_{i+1}}, \ldots, J^{t_m}$ together already have a higher load than is possible. Therefore the assumption must be wrong. For $j = 1, \ldots, m - i$ let

$$\omega_j = \alpha \cdot \left(\tfrac{c-1}{c} - \tfrac{m-i-1}{2(c-1)m}\right) \cdot \left(1 + \tfrac{c}{m}\right)^{j-1} + \tfrac{\alpha}{c} \,.$$

We derive the contradiction by first showing that $\omega_j \cdot D$ is a lower bound for $D^{t_{i+j}}$, i.e., the average load after scheduling $J^{t_{i+j}}$, and then showing that $\omega_{m-i} > 1$.

Since $s = i$, $J^{t_{i+1}}, \ldots, J^{t_m}$ are scheduled flatly but are not scheduled on machine M_i (they are filling jobs). The following observation follows directly from the definition of `MR`.

Observation 6. *Let b' be the current load of M_i, let p be the size of the next job J, and let $\omega \cdot D$ be the average load before scheduling J for some $\omega \leq 1$. If the current schedule is flat and J cannot be scheduled on M_i then $p > c\omega D - b'$.* □

We even have the stronger bound $p > c \cdot (\omega D + \frac{p}{m}) - b'$ which would improve the lower bound on p by a factor of $\frac{1}{1-\frac{c}{m}}$, but under the assumption $m \to \infty$ this factor equals 1.

Lemma 7. *$D^{t_{i+j}} \geq \omega_j \cdot D$ for $j = 1, \ldots, m - i$.*

Proof. For $j = 1, \ldots, m-i$ let

$$\omega'_j = \frac{D^{t_{i+j}}}{D} .$$

We prove by induction on j that $\omega'_j \geq \omega_j$.

The proof of Lemma 3 shows that $J^{t_{i+1}}$ cannot have size smaller than d, and the smallest size is only possible if the first i machines are just at the border between half-full and full prior to scheduling $J^{t_{i+1}}$, i.e., $l_1^{t_{i+1}-1} = \ldots = l_i^{t_{i+1}-1} = b$. After scheduling $J^{t_{i+1}}$ we have $i+1$ machines of size at least b and $m-i-1$ machines of average size at least $b-d$. Therefore

$$\begin{aligned}
\omega'_1 &\geq \frac{(i+1)\cdot\alpha + (m-i-1)\cdot\frac{2c-3}{2(c-1)}\cdot\alpha}{m} \\
&= \alpha \cdot \frac{(2c-3)m+i+1}{2(c-1)m} \\
&= \alpha \cdot \left(\frac{c-1}{c} - \frac{m-i-1}{2(c-1)m}\right) \cdot 1 + \frac{\alpha}{c} \\
&= \omega_1 .
\end{aligned}$$

For $j > 1$, if we assume that $l_i^{t_{i+j}-1} = b = \alpha D$ then Observation 6 implies

$$\begin{aligned}
\omega'_j &= \omega'_{j-1} + \frac{p^{t_{i+j}}}{mD} \\
&\overset{(O6)}{\geq} \omega_{j-1} + \frac{c\omega_{j-1} - \alpha}{m} \\
&= \omega_{j-1} \cdot \left(1 + \frac{c}{m}\right) - \frac{\alpha}{m} \\
&= \left[\alpha \cdot \left(\frac{c-1}{c} - \frac{m-i-1}{2(c-1)m}\right) \cdot \left(1 + \frac{c}{m}\right)^{j-2} + \frac{\alpha}{c}\right] \cdot \left(1 + \frac{c}{m}\right) - \frac{\alpha}{m} \\
&= \alpha \cdot \left(\frac{c-1}{c} - \frac{m-i-1}{2(c-1)m}\right) \cdot \left(1 + \frac{c}{m}\right)^{j-1} + \frac{\alpha}{c} \\
&= \omega_j .
\end{aligned}$$

But is the assumption $l_i^{t_{i+j}-1} = b$ justified? Observation 6 seems to imply that the lower bound on the size of the next job is smaller if the load on machine M_i is higher. However, if we want to take advantage of that, i.e., decrease the lower bound on the next job by some $\epsilon > 0$ then we must first increase the load of M_i to at least $b + \epsilon$ (at some time prior to time $t_{i+j} - 1$), so the net effect of that action does not lead to a smaller average load than the lower bound obtained with our assumption. □

Lemma 8. $\omega_{m-i} > 1$.

Proof. $\alpha > c - 1$ implies

$$\begin{aligned}
\omega_{m-i} &= \alpha \cdot \left(\tfrac{c-1}{c} - \tfrac{m-i-1}{2(c-1)m} \right) \cdot \left(1 + \tfrac{c}{m}\right)^{m-i-1} + \tfrac{\alpha}{c} \\
&= \alpha \cdot \left(\tfrac{c-1}{c} - \tfrac{1 - \frac{5c-2c^2-1}{c}}{2(c-1)} \right) \cdot \left(1 + \tfrac{c}{m}\right)^{\left(1 - \frac{5c-2c^2-1}{c}\right) \cdot m} + \tfrac{\alpha}{c} \\
&> \frac{\left(2(c-1)^2 - (2c^2+1-4c)\right) \cdot \left(1+\frac{c}{m}\right)^{\frac{2c^2+1-4c}{c} \cdot m} + 2(c-1)}{2c} \\
&= 1 + \tfrac{e^{2c^2+1-4c} - 2}{2c} \, .
\end{aligned}$$

For the last equality we used our assumption $m \to \infty$ which implies $(1 + \frac{c}{m})^m = e^c$. Therefore, $\omega_{m-i} > 1$ if $c \geq 1 + \sqrt{\frac{1+\ln 2}{2}}$. □

This concludes the proof of the case $s = i$ for $m \to \infty$. For given finite m we can easily compute c such that the lemma holds. For example, if $m = 64$ then we can choose $i = 40$, $k = 16$, and $c = 1.9229 < 1.923$ which is the best previous bound [1] (Lemma 9 below is also true with these parameters).

Some tedious analysis shows that if $c = 1 + \sqrt{\frac{1+\ln 2}{2}}$ then the proof of Lemma 8 works *only* for $i = \frac{5c-2c^2-1}{c} \cdot m - 1$. If c is chosen bigger then there is some interval around this value of i which works fine, and if c is chosen smaller then no value of i works.

We now consider the case $s > i$. We assume $D_s \geq b - d$ and derive the same contradiction as in the previous case.

Lemma 9. *$D^{t_s} \geq \omega_{s-i} \cdot D$.*

Proof. Prior to scheduling J^{t_s} we have

$$D_{i+1}^{t_s-1} \geq \frac{(s-1-i) \cdot b + (m-s+1) \cdot (b-d)}{m-i} \, .$$

Since J^{t_s} is scheduled steeply we have

$$l_k^{t_s-1} \geq \frac{2(c-1)}{2c-3} \cdot D_{i+1}^{t_s-1}$$

and therefore

$$\begin{aligned}\frac{D^{t_s}}{D} &\geq \frac{k \cdot l_k^{t_s-1} + (s-k)\cdot b + (m-s)\cdot(b-d)}{mD} \\ &\geq \frac{k \cdot \frac{2(c-1)}{2c-3} \cdot \frac{(s-1-i)\cdot b + (m-s+1)\cdot(b-d)}{m-i} + (m-k)\cdot b - (m-s)\cdot d}{mD} \\ &= \frac{\alpha}{m} \cdot \left(k \cdot \frac{2(c-1)}{2c-3} \cdot \left(1 - \frac{m-s+1}{2(c-1)(m-i)} \right) + m - k - \frac{m-s}{2(c-1)} \right) .\end{aligned}$$

We call this term E_s. If seen as a function of s then for $m \to \infty$

$$\begin{aligned}E_m &= \frac{\alpha}{m} \cdot \left(k \cdot \frac{2(c-1)}{2c-3} + m - k \right) \\ &= \alpha \cdot \left(1 + \frac{k}{(2c-3)\cdot m} \right) \\ &\geq 1.33 \cdot \alpha\end{aligned}$$

and

$$\begin{aligned}\omega_{m-i} &= \alpha \cdot \left[\left(\frac{c-1}{c} - \frac{m-i-1}{2(c-1)m} \right) \cdot e^{2c^2+1-4c} + \frac{1}{c} \right] \\ &\leq 1.09 \cdot \alpha ,\end{aligned}$$

i.e., $E_m \geq \omega_{m-i}$. Further we have $E_{i+1} = \omega_1 = \omega_{i+1-i}$. Since E_s is linearly growing in s and ω_{s-i} is exponentially growing in s we can conclude that $E_s \geq \omega_{s-i}$ for $s = i+1, \ldots, m$. □

Now we can just proceed as in the proof of Lemma 7 proving $D^{t_{s+j}} \geq \omega_{s-i+j} \cdot D$ for $j = 0, \ldots, m-s$. Note that the steepness condition only affects the load of the first k largest machines none of which would ever become the ith largest machine in the worst case scenario (with minimal load increases) of the proof of Lemma 7. Therefore the assumption about the size of the ith largest machine in that proof is still justified. This concludes the proof of Lemma 4.

5 Acknowledgements

We thank Erik Demaine for suggesting the term 'half-empty'. We also thank an unknown referee whose detailed comments were very valuable. `Maple` was

also of great help.

References

1. S. Albers. Better bounds for online scheduling. In *Proceedings of the 29th ACM Symposium on the Theory of Computation (STOC'97)*, pages 130–139, 1997. To appear in *SIAM Journal on Computing*.

2. Y. Bartal, A. Fiat, H. Karloff, and R. Vohra. New algorithms for an ancient scheduling problem. *Journal of Computer and System Sciences*, 51:359–366, 1995.
   A preliminary version was published in *Proceedings of the 24th ACM Symposium on the Theory of Computation (STOC'92)*, pages 51–58, 1992.

3. Y. Bartal, H. Karloff, and Y. Rabani. A better lower bound for on-line scheduling. *Information Processing Letters*, 50:113–116, 1994.

4. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, England, 1998.

5. R. Chandrasekaran, B. Chen, G. Galambos, P. R. Narayanan, A. van Vliet, and G. J. Woeginger. A note on 'An on-line scheduling heuristic with better worst case ratio than Graham's list scheduling'. *SIAM Journal on Computing*, 26(3):870–872, 1997.

6. B. Chen, A. van Vliet, and G. J. Woeginger. A lower bound for randomized on-line scheduling algorithms. *Information Processing Letters*, 51:219–222, 1994.

7. B. Chen, A. van Vliet, and G. J. Woeginger. New lower and upper bounds for on-line scheduling. *Operations Research Letters*, 16:221–230, 1994.

8. B. Chen, A. van Vliet, and G. J. Woeginger. An optimal algorithm for preemptive on-line scheduling. *Operations Research Letters*, 18:300–306, 1994.
   A preliminary version was published in *Proceedings of the 2nd European Symposium on Algorithms (ESA'94)*. Springer Lecture Notes in Computer Science 855, pages 300–306, 1994.

9. U. Faigle, W. Kern, and G. Turán. On the performance of on-line algorithms for partition problems. *Acta Cybernetica*, 9:107–119, 1989/90.

10. A. Fiat and G. Woeginger, editors. *Online Algorithms — The State of the Art*. Springer Lecture Notes in Computer Science 1442. Springer-Verlag, Heidelberg, 1998.

11. G. Galambos and G. J. Woeginger. An on-line scheduling heuristic with better worst case ratio than Graham's list scheduling. *SIAM Journal on Computing*, 22(2):349–355, 1993.

12. M. R. Garey and D. S. Johnson. *Computers and Intractability — A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, New York, 1979.

13. T. Gormley, N. Reingold, E. Torng, and J. Westbrook. Generating adversaries for request-answer games. In *Proceedings of the 11th ACM-SIAM Symposium on Discrete Algorithms (SODA'00)*, pages 564–565, 2000.

14. R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

15. D. S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, Boston, MA, 1996.

16. D. R. Karger, S. J. Phillips, and E. Torng. A better algorithm for an ancient scheduling problem. *Journal of Algorithms*, 20(2):400–430, 1996.
A preliminary version was published in *Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms (SODA'94)*, pages 132–140, 1994.

17. S. Seiden. Randomized algorithms for that ancient scheduling problem. In *Proceedings of the 5th Workshop on Algorithms and Data Structures (WADS'97).* Springer Lecture Notes in Computer Science 1272, pages 210–223, 1997.

18. J. Sgall. A lower bound for randomized on-line multiprocessor scheduling. *Information Processing Letters*, 63:51–55, 1997.