Online Scheduling Revisited

Rudolf Fleischer\textsuperscript{1} \hspace{1cm} Michaela Wahl\textsuperscript{2}

\textsuperscript{1} University of Waterloo, Department of Computer Science  
email: rudolf@uwaterloo.ca

\textsuperscript{2} Max-Planck-Institut für Informatik, Saarbrücken  
email: altherr@mpi-sb.mpg.de

Abstract. We present a new online algorithm \textsc{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 $1 + \sqrt[2]{\frac{1 + \ln 2}{2}} < 1.9201$.

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 \textsc{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 $(2 - \frac{1}{m} - \varepsilon_m)$-competitive algorithm by  
Galambos and Woeginger [11, 5], with $\varepsilon_m \to 0$ 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}{m})$-competitive algorithm (for $m \geq 70$) by Bartal et al. [2]. Note that $2 − \frac{1}{m} \approx 1.986$. Not much later Karger et al. [16] proposed the algorithm CHASM, 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 \( \frac{\sqrt{m}}{\sqrt{e}} \approx 1.58 \) 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 $1 + \sqrt{\frac{1+\ln m}{2}} < 1.9201$ for $m \to \infty$. For $m \geq 64$ this beats the 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 the 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, \(l_j^t\) 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 \(\texttt{MR}\) is parameterized by the competitive ratio \(c\) we hope to achieve. We show in the next section that \(\texttt{MR}\) works well if we choose

\[
c \geq 1 + \sqrt{\frac{1 + \ln 2}{2}}.
\]

Let

\[
i = \left\lceil \frac{5c - 2c^2 - 1}{c} \cdot m \right\rceil - 1
\]

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

\[
k = 2i - m.
\]

We call \(\texttt{MR}\)’s schedule flat at time \(t\) if
\[ p_k^{-1} < \frac{2(c - 1)}{2c - 3} \cdot D_i^{-1}, \]

i.e., if before scheduling \( J^t \) the load of the \( k \)th 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 \( p^t + l_i^{-1} > c \cdot D^t \) then schedule \( J^t \) on the smallest machine \( M_i^{-1} \), else schedule it on \( M_i^{-1} \).

Note that \( l_i^{-1} \) is the load of the \( i \)th 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 \( m \to \infty \). \( \square \)

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

### 3 Proof of the Main Theorem

Let

\[ c = 1 + \sqrt{\frac{1+\ln 2}{2}} \approx 1.9201. \]

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 \( J^f \). Let

\[ a = p^f \]

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

\[ D = D^f \]

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 \quad \iff \quad a > c \cdot D - b \]

\((C2)\quad a + b > c \cdot a \quad \iff \quad a < \frac{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
\[ l_{k-1}^t < \frac{b}{b - d} \cdot D_{i+1}^t. \]

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 \( J^t \) were scheduled on \( M_{i}^{t-1} \) then by definition of MR
\[ a + b = p^t + l_{i}^{t-1} \leq c \cdot D^t = c \cdot D, \]
contradicting (C1). Therefore \( J^t \) is scheduled on the smallest machine \( M_{i}^{t-1} \).

In particular, all machines have load at least \( b \) before \( J^t \) 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 \( b - d > (c - \frac{2}{3}) \cdot D \).

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

**Proof.** Assume the schedule is steep at time \( t^f \). Then
\[ l_{k}^{t'-1} \geq \frac{2(c - 1)}{2c - 3} \cdot D_{i+1}^{t' - 1} \geq \frac{2(c - 1)}{2c - 3} \cdot b \]
and therefore
\[ D \geq \frac{k \cdot l_{k}^{t - 1} + (m - k) \cdot b}{m} \]
\[ \geq \frac{k \cdot \frac{2(c-1)}{2c-3} \cdot b + (m - k) \cdot b}{m} \]
\[ > \frac{m + \frac{k}{2c-3} \cdot (c - 1) \cdot D}{m} \approx 1.22 \cdot D, \]
a contradiction. \[\Box\]

For \( j = 1, \ldots, m + 1 \) let \( t_j \) be the time the \( j \)th filling job arrives. And for \( j = 1, \ldots, m \) let
\[ D_j = D^{t_j-1} \]
be the average load of machines \( M_{j-1}^{t_j-1}, \ldots, M_{m-1}^{t_j-1} \) (which are exactly the non-full machines at that time) just prior to filling the \( j \)th 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 \( J^{s+1}, J^{s+2}, \ldots, J^{m+1} \) are scheduled flatly. If \( s < i \) then we choose \( s = i \).

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

**Proof.** Consider one of these jobs at time \( t \). Since machine \( M_{i-1}^{t-1} \) is already full, \( J^t \) can only be filling if it is scheduled on \( M_{m-1}^{t-1} \). 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
\[ D^t \geq \frac{i \cdot l_{i-1}^{t-1} + b + (m - i - 1) \cdot (b - d)}{m} \]
if \( t < t^f \) because the first \( i \) machines have load at least \( l_{i-1}^{t-1} \), 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
\[ p^t > c \cdot D^t - l^t_i \]
\[ \geq c \cdot \left( \frac{(i+1) b + (m-i-1) \cdot (b-d)}{m} \right) - b \]
\[ = c \cdot b - \frac{m-i-1}{m} \cdot c \cdot d - b \]
\[ = 2(c - 1)^2 \cdot d + (4c - 2c^2 - 1) \cdot d \]
\[ = d. \]

The second inequality holds because \( \frac{c}{m} - 1 \approx 0.22 > 0 \) and \( l^t_i \geq b. \] □

Note that in the proof of the lemma in the worst case \( l^t_i = b \) 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 \( J^{i+1}, J^{i+2}, \ldots, J^m \) 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.** \( J^{i_1}, J^{i_2}, \ldots, J^{i_s} \) 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 \( J^{u+1}, J^{u+2}, \ldots, J^{i_s} \) are scheduled on the smallest machine because they are filling jobs. But that 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

\[ t^{i-1}_{k} \geq b \]
\[ = \frac{2(c - 1)}{2c - 3} \cdot (b - d) \]
\[ \geq \frac{2(c - 1)}{2c - 3} \cdot D_{i+1} \]
\[ \geq \frac{2(c - 1)}{2c - 3} \cdot D_{i+1}^{i-1}. \]
Hence $J^{i+k+1}, J^{i+k+2}, \ldots, J^{i+k}$ are also scheduled on the smallest machine.

This also implies $t_i^k \leq t_{i+1}^{k+1} \leq \cdots \leq t_{i+(m-i)}^{k+(m-i)} = t_m^i$. For $j = 1, \ldots, k$, $J^{j_i}$ is either scheduled on $M^j_{i-1}$ or on $M^j_{m-1}$. But

$$t_m^{j_i-1} \leq t_i^{j_i-1} \leq t_i^j \leq t_i^k \leq D_{i+1}^{j_i} < b - d$$

and hence $J^{i_1}, J^{i_2}, \ldots, J^{i_k}$ are scheduled on a half-empty machine.

This concludes the proof of the Main Theorem.

4 Proof of Lemma 4

Let $\alpha = \frac{b}{h}$. 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 $J^{i+1}, \ldots, J^{m}$ together already have a higher load than is possible. Therefore the assumption must be wrong. For $j = 1, \ldots, m - i$ let

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

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

Since $s = i$, $J^{i+1}, \ldots, J^{m}$ 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 $U'$ 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 $p > \omega D - b'$.

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. $D^{i+j} \geq \omega_j \cdot D$ 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., \( t^{t_{i+1}-1}_{i+1} = \ldots = t^{t_{i+1}-1}_i = b \). 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
\[
\omega_j' \geq \frac{(i+1) \cdot (m-i-1) \cdot \frac{b - d}{2(c-1)m} \cdot \alpha}{m} = \alpha \cdot \frac{(2c-3)m + i+1}{2(c-1)m} = \alpha \cdot \left( \frac{c-1}{c} - \frac{m-i-1}{2(c-1)m} \right) \cdot 1 + \frac{\alpha}{c} = \omega_j.
\]

For \( j > 1 \), if we assume that \( t^{t_{i+j}-1}_{i+1} = b = \alpha D \) then Observation 6 implies
\[
\omega_j' = \omega_{j-1}' + \frac{t^{t_{i+j}}}{mD} \\
\quad \geq \omega_{j-1}' + \frac{\omega_{j-1}' - \alpha}{m} \\
\quad = \omega_{j-1}' \cdot \left( 1 + \frac{\alpha}{m} \right) - \frac{\alpha}{m} \\
\quad = \left[ \alpha \cdot \left( \frac{c-1}{c} - \frac{m-i-1}{2(c-1)m} \right) \cdot \left( 1 + \frac{\alpha}{m} \right)^{j-2} + \frac{\alpha}{c} \right] \cdot \left( 1 + \frac{\alpha}{m} \right) - \frac{\alpha}{m} \\
\quad = \alpha \cdot \left( \frac{c-1}{c} - \frac{m-i-1}{2(c-1)m} \right) \cdot \left( 1 + \frac{\alpha}{m} \right)^{j-1} + \frac{\alpha}{c} = \omega_j.
\]

But is the assumption \( t^{t_{i+j}-1}_{i+1} = b \) 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. \( \Box \)

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

Proof. \( \alpha > c - 1 \) implies
\[ \omega_{m-i} = \alpha \cdot \left( \frac{c-1}{c} - \frac{m-i-1}{2(c-1)m} \right) \cdot \left( 1 + \frac{c}{m} \right)^{m-i-1} + \frac{\alpha}{c} \]
\[ = \alpha \cdot \left( \frac{c-1}{c} - \frac{1}{2(c-1)} \right) \cdot \left( 1 + \frac{c}{m} \right)^{m-i-1} + \frac{\alpha}{c} \]
\[ > \frac{(2(c-1)^2 - (2c^2 + 1 - 4c)) \left( 1 + \frac{c}{m} \right)^{2c^2 + 1 - i - 1} + 2(c-1)}{2c} \]
\[ = 1 + \frac{c^2 + 1 - i - 1 - 2}{2c}. \]

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 \( c \geq 1 + \sqrt{\frac{1 + \ln 2}{2}} \). \( \square \)

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 \( i = 40, k = 16, c = 1.9229 < 1.923 \) 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
\[
\frac{D^t}{D} \geq \frac{k \cdot l_k^{s-1} + (s - k) \cdot b + (m - s) \cdot (b - d)}{k \cdot \frac{2(c-1)}{2c-3} \cdot \frac{mD}{m-i} + (m - k) \cdot b - (m - s) \cdot d} \\
= \frac{\alpha}{m} \cdot \left( k \cdot \frac{2(c-1)}{2c-3} \cdot \left( 1 - \frac{mD}{m-s+1} \frac{1}{2(c-1)(m-i)} \right) + m - k \right) \\
= \frac{\alpha}{m} \cdot k \cdot \frac{2(c-1)}{2c-3} + m - k \\
= \alpha \cdot \left( 1 + \frac{k}{(2c-3) \cdot m} \right) \\
\geq 1.33 \alpha \\
\]
We call this term \( E_s \). If seen as a function of \( s \) then for \( m \to \infty \)

\[
E_m = \frac{\alpha}{m} \cdot \left( k \cdot \frac{2(c-1)}{2c-3} + m - k \right) \\
= \alpha \cdot \left( 1 + \frac{k}{(2c-3) \cdot m} \right) \\
\geq 1.33 \cdot \alpha \\
\]

and

\[
\omega_{m-i} = \alpha \cdot \left[ \left( \frac{c-1}{c} - \frac{m-i-1}{2(c-1)m} \right) \cdot e^{2c^2+1-4c} + \frac{1}{c} \right] \\
\leq 1.09 \cdot \alpha \\
\]
i.e., \( E_m \geq \omega_{m-i} \). Further we have \( E_{i+1} = \omega_i = \omega_{i+1-i} \). Since \( E_s \) is linearly growing in \( s \) and \( \omega_{s-i} \) 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 \( D^{t+i} \geq \omega_{s-i+j} \cdot D \) for \( j = 0, \ldots, m-s \). Note that the steepness condition only affects the load of the first \( k \) largest machines which never become the \( i \)th 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 \( i \)th 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


