Alcom-FT Technical Report Series
ALCOMFT-TR-02-58

Optimal Non-Preemptive Semi-Online Scheduling on Two Related Machines

Leah Epstein[1⋆] and Lene M. Favrholdt[2⋆⋆]

[1] School of Computer Science, The Interdisciplinary Center, Herzliya, Israel. lea@idc.ac.il.
[2] Department of Mathematics and Computer Science, University of Southern Denmark, Odense, Denmark.
lenem@imada.sdu.dk

Abstract. We consider the following non-preemptive semi-online scheduling problem. Jobs with non-increasing sizes arrive one by one to be scheduled on two uniformly related machines, with the goal of minimizing the makespan. We analyze both the overall competitive ratio, and the competitive ratio as a function of the speed ratio ($q \geq 1$) between the two machines. We show that the greedy algorithm LPT has optimal competitive ratio $\frac{1}{4}(1+\sqrt{17}) \approx 1.28$ overall, but does not have optimal competitive ratio for every value of q. We determine the intervals of q where LPT is an algorithm of optimal competitive ratio, and design different algorithms of optimal competitive ratio for the intervals where it fails to be the best algorithm. As a result, we give a tight analysis of the competitive ratio for every speed ratio.

1 Introduction

The Problem. In this paper we study non-preemptive semi-online scheduling on two uniformly related machines. In the model of uniformly related machines, each machine has a *speed* and each job has a *size* which is the time it takes to complete it on a machine with unit speed. The jobs arrive one by one in order of non-increasing sizes. Each job must be assigned to one of the machines without any knowledge of future jobs (except for a bound on their size that follows from the size of the current job). Since the jobs are known to have non-increasing sizes, the problem cannot be seen as on-line but semi-online. We study the non-preemptive case, where it is not allowed to split a job in more parts and run the various parts on different machines. The goal is to minimize the *makespan*, i.e., the latest completion time of any job.

The processing time of a job on a given machine is also called the *load* of the job on that machine. The load of a machine is the sum of the loads of the jobs assigned to it. Thus, the makespan is the maximum load of any machine.

Since we study the case of two machines, the important parameter is the speed ratio $q \geq 1$ between the two machines. Without loss of generality, we assume that the faster machine has speed 1, and the other machine has speed $\frac{1}{q}$. We denote the faster machine by M_1 and the other machine by M_q.

Preliminaries. The quality of a semi-online algorithm, similarly to on-line algorithms, is measured by the *competitive ratio* which is the worst case ratio of the cost (the makespan, in this paper) of the semi-online algorithm to the cost of an optimal off-line algorithm which knows the whole sequence in advance.

The semi-online algorithm under consideration as well as its makespan is denoted by SONL. Similarly, the optimal off-line algorithm as well as its makespan is denoted by OPT.

⋆ Research supported in part by the Israel Science Foundation, (grant no. 250/01)
⋆⋆ Supported in part by the Danish Natural Science Research Council (SNF) and in part by the Future and Emerging Technologies program of the EU under contract number IST-1999-14186 (ALCOM-FT).

Thus, the competitive ratio of an algorithm SONL is

$$C = \inf\{c \mid \mathrm{SONL} \leq c \cdot \mathrm{OPT}, \text{ for any input sequence}\}.$$

For any $c \geq C$, SONL is said to be *c-competitive.*

For the first k jobs of an input sequence, we denote the makespan of the semi-online algorithm by SONL_k and the makespan of the optimal off-line algorithm by OPT_k. The jobs in a sequence are denoted $J_1, J_2, \ldots, J_\ell$ and their sizes are denoted $p_1, p_2, \ldots, p_\ell$.

The greedy algorithm LPT (Longest Processing Time first) was originally designed by Graham [6] for off-line scheduling on identical machines. It sorts the jobs by non-increasing sizes and schedules them one by one on the least loaded machine. This algorithm also works for the semi-online version where the jobs arrive in order of non-increasing sizes. The natural extension for uniformly related machines is as follows:

Algorithm LPT: Assign each arriving job J (of size p) to the machine that would finish it first. Formally, for each machine i let L_i be its load before the arrival of J. The job J is assigned to the fastest machine i for which $L_i + \frac{p}{s_i}$ is minimized.

Previous Work. All previous study of this problem on non-identical machines involves a study of the LPT algorithm. For two machines, Mireault, Orlin and Vohra [7] give a complete analysis of LPT as a function of the speed ratio. They show that the interval $q \geq 1$ is partitioned into nine intervals, and introduce a function which gives the competitive ratio in each interval (they consider the off-line problem, so they do not use the term competitive ratio). Some properties of LPT were already shown earlier. Graham [6] shows that the exact approximation ratio of LPT is $\frac{7}{6}$ for two identical machines. Seiden, Sgall and Woeginger [8] show that this is tight, i.e., LPT has the best possible competitive ratio for the problem. For two related machines, [5] shows that for any speed ratio, the performance ratio of LPT is at most $\frac{1}{4}(1+\sqrt{17}) \approx 1.28$.

For m identical machines, Graham [6] shows that the exact approximation ratio of LPT is $\frac{4}{3} - \frac{1}{3m}$. For three machines, [8] gives a general lower bound of $\frac{1}{6}(1+\sqrt{37}) \approx 1.18$. For a general setting of m related machines, Friesen [4] shows that the overall approximation ratio of LPT is between 1.52 and $\frac{5}{3}$. Dobson [2] claims to improve the upper bound to $\frac{19}{12} \approx 1.58$. Unfortunately, his proof does not seem to be complete.

Our Results. In this paper we give the exact competitive ratio as a function of the speed ratio q for semi-online non-preemptive scheduling on two related machines with non-increasing job sizes (see Figure 1). The function involves 15 distinct intervals, as defined in Theorem 1.

In some of those intervals, we give general lower bounds which match the upper bounds in [7]. In those cases, LPT is an optimal semi-online algorithm. In the other intervals, we design new algorithms and prove matching general lower bounds. We show that, in terms of overall competitive ratio, $\frac{1}{4}(1+\sqrt{17})$ is the optimal competitive ratio achieved at $q = \frac{1}{4}(1+\sqrt{17})$ by LPT.

It is interesting to examine our results in the view of the results for on-line scheduling on two related machines. Unlike our problem, for that problem, LPT is optimal for all $q \geq 1$. The overall competitive ratio is ϕ. For $q \leq \phi$ the competitive ratio is $1 + \frac{q}{q+1}$ and the competitive ratio for $q \geq \phi$ is $1 + \frac{1}{q}$. There are only two distinct intervals and the worst competitive ratio is achieved at ϕ. Surprisingly, for both problems, the highest competitive ratio is equal to the value of q for which it is achieved. The upper bounds, as well as the overall lower bound are given in [1], the other lower bounds are given in [3].

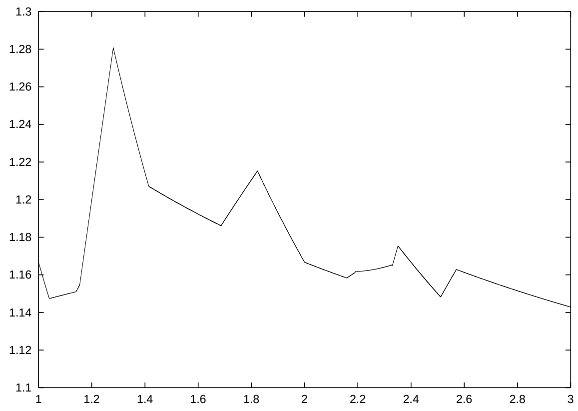

Fig. 1. The competitive ratio as a function of q

2 The Function

Theorem 1. *The optimal competitive ratio for semi-online scheduling on two related machines (with speed ratio $q \geq 1$) is described by the following function.*

$$C(q) = \begin{cases} C_1(q), & 1 \leq q \leq q_1 \approx 1.0401 \\ C_2(q), & q_1 \leq q \leq q_2 \approx 1.1410 \\ C_3(q), & q_2 \leq q \leq \sqrt{\frac{4}{3}} \approx 1.1547 \\ C_4(q), & \sqrt{\frac{4}{3}} \leq q \leq \frac{1}{4}(1+\sqrt{17}) \approx 1.2808 \\ C_5(q), & \frac{1}{4}(1+\sqrt{17}) \leq q \leq \sqrt{2} \approx 1.4142 \\ C_6(q), & \sqrt{2} \leq q \leq \frac{1}{4}(1+\sqrt{33}) \approx 1.6861 \\ C_7(q), & \frac{1}{4}(1+\sqrt{33}) \leq q \leq \frac{1}{2}(1+\sqrt{7}) \approx 1.8229 \end{cases} \qquad C(q) = \begin{cases} C_8(q), & \frac{1}{2}(1+\sqrt{7}) \leq q \leq 2 \\ C_9(q), & 2 \leq q \leq \frac{1}{2}(1+\sqrt{11}) \approx 2.1583 \\ C_{10}(q), & \frac{1}{2}(1+\sqrt{11}) < q \leq q_{10} \approx 2.1956 \\ C_{11}(q), & q_{10} \leq q \leq q_{11} \approx 2.3307 \\ C_{12}(q), & q_{11} \leq q \leq \frac{1}{4}(3+\sqrt{41}) \approx 2.3508 \\ C_{13}(q), & \frac{1}{4}(3+\sqrt{41}) \leq q \leq q_{13} \approx 2.5111 \\ C_{14}(q), & q_{13} \leq q \leq q_{14} \approx 2.5704 \\ C_{15}(q), & q \geq q_{14}, \end{cases}$$

$$C_1(q) = \frac{2}{3} + \frac{1}{2q}, \qquad C_2(q) = 1 + \frac{1}{2}\left(4q^2 + 4q - 1 - \sqrt{(4q^2+4q-1)^2 - 4q^2}\right),$$

$$C_3(q) = \frac{6q+4}{3q+6}, \qquad C_4(q) = q, \qquad C_5(q) = \frac{q+2}{2q}, \qquad C_6(q) = \frac{2q+3}{2q+2},$$

$$C_7(q) = \frac{2q+1}{q+2}, \qquad C_8(q) = \frac{2q+3}{3q}, \qquad C_9(q) = \frac{2q+3}{2q+2}, \qquad C_{10}(q) = \frac{3q+2}{2q+3},$$

$$C_{11}(q) = \frac{q^2 + 3 + \sqrt{q^4 - 6q^2 + 24q + 9}}{6q}, \qquad C_{12}(q) = \frac{q}{2}, \qquad C_{13}(q) = \frac{3}{4} + \frac{1}{q},$$

$$C_{14}(q) = 1 + \frac{q^2 + 2q - 2 - \sqrt{q^4 + 8q + 4}}{2q+4}, \qquad C_{15}(q) = \frac{2q+2}{2q+1},$$

q_1 *is the largest real root of* $84q^4 - 24q^3 - 80q^2 + 6q + 9$,
q_2 *is the largest real root of* $27q^4 + 48q^3 - 54q^2 - 48q + 8$,
q_{10} *is the smallest real root of* $3q^4 - 9q^3 - 8q^2 + 21q + 18$,
q_{11} *is the largest real root of* $q^3 - 2q - 8$,
q_{13} *is the largest real root of* $20q^4 - 39q^3 - 46q^2 + 32q + 32$,
q_{14} *is the largest real root of* $4q^5 + 2q^4 - 24q^3 - 23q^2 + 6q + 8$.

The proof of the upper bound is given in Section 5 and the proof of the lower bound in Section 6.

3 Analysis of the Overall Competitive Ratio

In this section we show that LPT is optimal if we are only interested in the overall competitive ratio. That is, the maximum competitive ratio achieved by any set of two uniformly related machines. One way to determine the overall competitive ratio would be to analyze each interval separately, and take the maximum. However, we use previous results to do it in a simpler way.

Theorem 2. *The overall optimal competitive ratio for non-preemptive scheduling on two related machines is* $\frac{1}{4}(1+\sqrt{17})$.

Proof. We use a result given already in [5]. That paper shows that LPT on two related machines has a competitive ratio of at most $\frac{1}{4}(1+\sqrt{17})$. Later in this paper we show that for $q = \frac{1}{4}(1+\sqrt{17})$, the competitive ratio of any semi-online algorithm is at least $\frac{1}{4}(1+\sqrt{17})$, which completes the proof. $\square$

The conclusion is that LPT is an optimal semi-online algorithm in the sense that its overall competitive ratio is best possible. However, analyzing the competitive ratio as a function of the speed ratio q gives the surprising result that LPT is not optimal for all q.

4 Properties and Assumptions

In this section we describe a few facts and assumptions used in the upper bound analysis.

We assume without loss of generality that OPT $= 1$. Thus, the ratio $\frac{\text{SONL}}{\text{OPT}} = \text{SONL}$. For a given input sequence, we denote the total size of jobs by P and the size of the last job by p_ℓ. Note that $P \leq 1 + \frac{1}{q} = \frac{q+1}{q}$ always holds, since the total size of jobs scheduled by OPT is at most 1 on M_1 and at most $\frac{1}{q}$ on M_q.

In the upper bound analysis, we consider worst case sequences of each possible length separately. Thus, we can assume that the makespan is determined by the last (and hence the smallest) job of the sequence, since otherwise it is handled by the analysis of shorter sequences.

The following lemma appears in [7]. For completeness, we prove it here as well.

Lemma 1. *For a given input sequence and a given semi-online algorithm SONL, assume that the last job J_ℓ is scheduled according to the LPT rule and that $SONL > SONL_{\ell-1}$. Then*

$$SONL \leq 1 + \frac{q}{q+1} p_\ell.$$

Proof. Let $P_q^{\ell-1}$ and $P_1^{\ell-1}$ denote the total size of jobs assigned to the slow and the fast machine, respectively, just before the last job is assigned. Note that $P = P_q^{\ell-1} + P_1^{\ell-1} + p_\ell$. Since the last job is assigned by the LPT rule, and by the assumption that the last job determines the makespan, the final makespan is $\min\{q(P_q^{\ell-1}+p_\ell), P_1^{\ell-1}+p_\ell\}$. This is bounded by the convex combination

$$\frac{1}{q+1} q(P_q^{\ell-1}+p_\ell) + \frac{q}{q+1}(P_1^{\ell-1}+p_\ell) = \frac{q}{q+1}(P_q^{\ell-1}+P_1^{\ell-1}+2p_\ell) = \frac{q}{q+1}(P+p_\ell) \leq 1 + \frac{q}{q+1} p_\ell.$$

$\square$

We will sometimes use Lemma 1 in the following form.

Corollary 1. *For a given input sequence and a given semi-online algorithm SONL, assume that the last job J_ℓ is scheduled according to the LPT rule and that $SONL > SONL_{\ell-1}$.*

$$\text{If } SONL > C, \text{ then } p_\ell > (C-1)\frac{q+1}{q} .$$

Lemma 2. *For a given input sequence and a given semi-online algorithm, assume that J_ℓ is scheduled according to the LPT rule and that $SONL > SONL_{\ell-1}$. Let $k \in \mathbb{N}$.*

$$\text{If } SONL > 1 + \frac{1}{(k+1)(q+1)}, \text{ then OPT runs at most } k \text{ jobs on } M_q.$$
$$\text{If } SONL > 1 + \frac{q}{(k+1)(q+1)}, \text{ then OPT runs at most } k \text{ jobs on } M_1.$$

Proof. If OPT schedules at least $k+1$ jobs on M_1, the smallest of these k jobs has size at most $\frac{1}{k+1}$. Thus, $p_\ell \le \frac{1}{k+1}$. However, using Lemma 1, we get SONL $\le 1 + \frac{q}{q+1}\frac{1}{k+1}$. Similarly, if at least $k+1$ jobs are scheduled on M_q, $p_\ell \le \frac{1}{q(k+1)}$. Using Lemma 1, we get SONL $\le 1 + \frac{q}{q+1}\frac{1}{q(k+1)} = 1 + \frac{1}{(q+1)(k+1)}$. □

Lemma 3. *For a given input sequence and a given semi-online algorithm, assume that J_ℓ is scheduled according to the LPT rule and that $SONL > SONL_{\ell-1}$. Let $P_1^{\ell-1}$ and $P_q^{\ell-1}$ be the total size of jobs assigned to M_1 and M_q, respectively, just before the arrival of J_ℓ. Then,*

$$SONL = \min\{P_1^{\ell-1} + p_\ell,\ q(P_q^{\ell-1} + p_\ell)\}.$$

Proof. LPT schedules J_ℓ on a machine such that the resulting load on that machine is minimized. By the assumption that J_ℓ determines the makespan, the final makespan is equal to the load on the machine running J_ℓ. □

5 New Algorithms

In this section we present algorithms of optimal competitive ratio, for intervals where LPT is not optimal. We first mention all intervals where LPT is an optimal algorithm. In [7] the exact performance ratio of LPT is given. In all intervals where the lower bound in Section 6 matches the upper bound in [7], clearly LPT has optimal competitive ratio.

The names we use for the intervals are taken from the definition of the function. Hence, we deal with intervals 1–15. The intervals where LPT is optimal are as follows: The first interval is the point $q = 1$. For $q = 1$, it is known [6] that the competitive ratio of LPT is $\frac{7}{6}$ and that this is the best possible competitive ratio for any semi-online algorithm [8]. However, for $q = 1+\varepsilon$, for small $\varepsilon > 0$, this paper shows that LPT is not an optimal semi-online algorithm.

The other intervals where LPT is optimal are $\frac{1}{6}(1+\sqrt{37}) \le q \le q_9$ and $q \ge q_{14} \approx 2.57$, where $q_9 \approx 2.04$ is the largest real root of $4q^3 - 4q^2 - 10q + 3$.

This leaves the following intervals to deal with. Intervals 1–4 (not including $q = 1$ in interval 1, and interval 4 only up to $\frac{1}{6}(1+\sqrt{37})$) and intervals 9–14 (interval 9 starting only at q_9). We design three new algorithms *Slow-LPT, Balanced-LPT and Opposite-LPT*. Slow-LPT has optimal competitive ratio in the interval $1 < q < \frac{1}{6}(1+\sqrt{37})$. Balanced-LPT has optimal competitive ratio in the intervals $q_9 < q \le q_{10}$ and $\frac{1}{4}(3+\sqrt{41}) \le q < q_{14}$, and Opposite-LPT has optimal competitive ratio in the interval $q_{10} \le q \le \frac{1}{4}(3+\sqrt{41})$.

As can be seen in the next section, the most difficult sequences for the algorithms are quite short sequences (up to six jobs). For longer sequences, the last job is relatively small. Thus, as shown by Lemma 1, the algorithm benefits from the non-increasing order in this case. As may be seen from the definitions of the three new algorithms, the most difficult decision is either the decision for the second job, or the first and the third jobs.

5.1 The First Four Intervals

We design an algorithm Slow-LPT which has optimal competitive ratio in the interval $1 < q < \frac{1}{6}(1+\sqrt{37}) \approx 1.18$. Intuitively, the reason why LPT fails in this interval is that the slow machine is not much slower than the faster one. Since the fast machine does not dominate the slow machine so easily, it often makes sense to use the slow machine first, and keep the fast machine free for future jobs. The algorithm is actually optimal in the interval $1 \le q \le \frac{1}{4}(1+\sqrt{17}) \approx 1.28$, giving an alternative algorithm with optimal competitive ratio in the interval $\frac{1}{6}(1+\sqrt{37}) \le q \le \frac{1}{4}(1+\sqrt{17})$.

Algorithm Slow-LPT

Assign J_1 to M_q. Assign J_2 to M_1.
If $q(p_1+p_3) \le C(q)(p_2+p_3)$, assign J_3 to M_q, and otherwise to M_1.
Assign the rest of the jobs by the LPT rule.

To analyze the algorithm, first note that, for $i, j \in \{1,2,3,4\}$, $C_i(q) \le C_j(q)$ in interval j. Thus, in the first four intervals, $C(q) = \max\{C_1(q), C_2(q), C_3(q), C_4(q)\}$.

Note that $C_1(q) > 1 + \frac{q}{4q+4}$ for $q < \frac{1}{7}(1+\sqrt{43}) \approx 1.08$, and $C_3(q) > 1 + \frac{q}{4q+4}$ for $q > \frac{1}{9}(1+\sqrt{73}) \approx 1.06$. Thus, if Slow-LPT is not $C(q)$-competitive, then by Lemma 1, $p_\ell > \frac{1}{4}$. In this case, OPT runs at most 3 jobs on each machine. Hence, we need only consider sequences of up to 6 jobs. In the following, we analyze sequences of each length $\ell = 1, \ldots, 6$ separately. This allows us to assume that the last job, J_ℓ, determines the makespan, i.e., $\mathrm{SONL} > \mathrm{SONL}_{\ell-1}$, since if $\mathrm{SONL} = \mathrm{SONL}_{\ell-1}$, then $\frac{\mathrm{SONL}_{\ell-1}}{\mathrm{OPT}_{\ell-1}} \ge \frac{\mathrm{SONL}}{\mathrm{OPT}}$.

One job. Clearly, $\mathrm{OPT} = p_1$ and $\mathrm{SONL} = qp_1$. Thus, $\mathrm{SONL} \le q = C_4(q) \le C(q)$.

Two jobs. The makespan of Slow-LPT is determined by the first job, since $qp_1 \ge qp_2 \ge p_2$. Thus, since sequences of one job cannot break the ratio, sequences of two jobs cannot either.

Three jobs. Note that since OPT must assign at least two jobs on one machine, $\mathrm{OPT} \ge p_2+p_3$. Slow-LPT schedules J_3 on M_q, only if $q(p_1+p_3) \le C(q)(p_2+p_3)$. Thus, in this case the ratio is maintained, since $\mathrm{SONL} = q(p_1+p_3)$ by Lemma 3. If Slow-LPT schedules J_3 on M_1, $\mathrm{SONL} = p_2+p_3 \le \mathrm{OPT}$.

Four jobs. If Slow-LPT runs J_3 on M_q, J_4 will be scheduled on M_1 and will not determine the makespan, since $q(p_1+p_3) \ge q(p_2+p_4) \ge p_2+p_4$.

If Slow-LPT runs J_3 on M_1, $\mathrm{SONL} = \min\{q(p_1+p_4), p_2+p_3+p_4\}$. If OPT schedules J_1 and some other job on one machine, $\mathrm{OPT} \ge p_1+p_4$. In this case, $\mathrm{SONL} \le \frac{q(p_1+p_4)}{p_1+p_4} = q \le C(q)$. Otherwise, $\mathrm{OPT} \ge p_2+p_3+p_4 \ge \mathrm{SONL}$.

Five jobs. If Slow-LPT assigns J_3 to M_q, J_4 is assigned to M_1. After scheduling J_4 (and before scheduling J_5), the total size of jobs on M_1 is at most the total size on M_q, since $p_1+p_3 \geq p_2+p_4$. Thus, SONL $\leq \frac{1}{2}(p_1+p_2+p_3+p_4)+p_5 = \frac{1}{2}(P+p_5) \leq \frac{1}{2}(1+\frac{1}{q}+p_5)$. Having five jobs in the sequence, we know that OPT has three jobs on one of the machines. One job must be of size at most $\frac{1}{3}$. This gives $p_5 \leq \frac{1}{3}$ and thus SONL $\leq \frac{1}{2}(1+\frac{1}{q}+\frac{1}{3}) = \frac{2}{3}+\frac{1}{2q} = C_1(q)$. Hence, we assume that Slow-LPT assigns J_3 to M_1.

If Slow-LPT assigns J_4 to M_1, SONL $= \min\{q(p_1+p_5), p_2+p_3+p_4+p_5\}$, by Lemma 3. Since OPT runs at most three jobs on each machine, it runs J_1 with at least one other job. Thus, OPT $\geq p_1+p_5$, and SONL $\leq q(p_1+p_5) \leq q\,$OPT $\leq C(q)$. Hence, we assume that Slow-LPT assigns J_4 to M_q. This means that SONL $= \min\{q(p_1+p_4+p_5), p_2+p_3+p_5\}$.

If OPT runs J_1 with two more jobs, OPT $\geq p_1+p_4+p_5$. Hence, we will assume that OPT runs J_1 with exactly one other job. We split the remaining part of the analysis based on which job is combined with J_1.

Case 1: J_4 or J_5 runs with J_1 in the schedule of OPT. The machine not running J_1 is loaded by at least $p_2+p_3+p_5 \geq$ SONL.

Case 2: J_3 runs with J_1. In this case, $p_2+p_4+p_5 \leq 1$. Since $p_5 \leq p_4$, this implies $p_2+2p_5 \leq 1$. Thus, SONL $\leq p_2+p_3+p_5 \leq 2p_2+p_5 = 2(p_2+2p_5)-3p_5 \leq 2-3p_5$. Moreover, by Lemma 1, SONL $\leq 1+\frac{q}{q+1}p_5$. The two upper bounds on SONL are equal when $p_5 = \frac{q+1}{4q+3}$ and hence SONL $\leq 1+\frac{q}{4q+3}$. Solving $1+\frac{q}{4q+3} \leq C_2(q)$, we get $4q^2-2q-3 \leq 0$ and thus $q \leq \frac{1}{4}(1+\sqrt{13}) \approx 1.15$, which holds throughout the first two intervals. In the third and fourth intervals, $1+\frac{q}{4q+3} \leq C_3(q)$, as $9q^2-5q-6 \geq 0$ for $q \geq \frac{1}{18}(5+\sqrt{241}) \approx 1.14$.

Case 3: OPT runs J_1 with J_2. Let $p_1+p_2=\alpha$ and $p_3+p_4+p_5=\beta$. Then, by the way OPT schedules the jobs, either $\alpha \leq 1$ and $\beta \leq \frac{1}{q}$, or $\beta \leq 1$ and $\alpha \leq \frac{1}{q}$. Since J_3 was assigned to the fast machine by Slow-LPT, we know that $q(p_1+p_3) > C(q)(p_2+p_3)$. This can also be expressed as $q(\alpha-p_2+p_3) > C(q)(p_2+p_3)$, which is equivalent to $p_2 < \frac{q\alpha}{q+C(q)} + \frac{q-C(q)}{q+C(q)}p_3$. Thus,

$$\begin{aligned}\text{SONL} &< \frac{q\alpha}{q+C(q)} + \frac{q-C(q)+q+C(q)}{q+C(q)}p_3 + p_5 = \frac{q\alpha}{C(q)+q} + \frac{2q}{C(q)+q}p_3 + p_5 \\ &= \frac{q\alpha}{C(q)+q} + \frac{2q}{C(q)+q}(\beta-p_4-p_5) + p_5 = \frac{q\alpha+2q\beta}{C(q)+q} - \frac{2q}{C(q)+q}p_4 + \frac{C(q)-q}{C(q)+q}p_5.\end{aligned}$$

If $\alpha \leq \frac{1}{q}$ and $\beta \leq 1$, $q\alpha+2q\beta \leq 1+2q$. Otherwise, $q\alpha+2q\beta \leq q+2$. In both cases, $q\alpha+2q\beta \leq 1+2q$. Thus, since $p_4 \geq p_5$, we get SONL $\leq \frac{2q+1}{C(q)+q} + \frac{C(q)-3q}{C(q)+q}p_5$. Rewriting this upper bound as $\frac{2q+1-4qp_5}{C(q)+q}+p_5$, it is clear that it is maximized when $C(q)$ is minimized, since $p_5 \leq \frac{1}{3}$. Thus, using $C_2(q) \leq C(q)$, we get SONL $\leq \frac{2q+1}{C_2(q)+q} + \frac{C_2(q)-3q}{C_2(q)+q}p_5$. We show that this is at most $C_2(q)$. If $p_5 \leq (C_2(q)-1)\frac{q+1}{q}$, SONL $\leq C_2(q)$ by Corollary 1. Thus, assume that $p_5 > (C_2(q)-1)\frac{q+1}{q}$. The term $C_2(q)-3q$ in the upper bound on SONL is negative and we get SONL $< \frac{2q+1}{C_2(q)+q} + \frac{(C_2(q)-3q)(C_2(q)-1)(q+1)}{(C_2(q)+q)q}$. Substituting $C_2(q) = 1+\frac{1}{2}\left(4q^2+4q-1+\sqrt{(4q^2+4q-1)^2-4q^2}\right)$, we get SONL $< C_2(q)$.

Six jobs. Note that, if SONL $> C(q)$, OPT runs exactly three jobs on each machine. Since among J_1, J_2, and J_3, at least two run on one machine, this gives OPT $\geq p_2+p_3+p_6$.

If Slow-LPT runs J_3 on M_q, $q(p_1+p_3) \leq C(q)(p_2+p_3)$, by the definition of the algorithm. Furthermore, J_4 and J_5 are assigned to M_q, since $q(p_1+p_3) \geq p_2+p_4$. Thus, by Lemma 3, SONL $\leq q(p_1+p_3+p_6) \leq C(q)(p_2+p_3)+qp_6 \leq C(q)(p_2+p_3+p_6) \leq C(q)\cdot$ OPT. Hence, we assume that Slow-LPT runs J_3 on M_1.

We next consider the possibilities for J_4. If SONL assigns J_4 to M_1, $p_2+p_3+p_4 \leq q(p_1+p_4)$. Thus, $3p_4 - qp_4 \leq qp_1$. Since OPT schedules J_1 on a machine with two other jobs, $p_1+p_5+p_6 \leq$ OPT, and $p_1+2p_6 \leq 1$. Now, $(3-q)p_6 \leq (3-q)p_4 \leq qp_1 \leq q(1-2p_6)$. Solving for p_6 yields $p_6 \leq \frac{q}{3+q}$. Using Lemma 1 we have SONL $\leq 1+\frac{q^2}{(3+q)(q+1)}$, which is smaller than $C_2(q)$ in the first three intervals and smaller than $C_4(q)$ in the fourth interval. Hence, we assume that Slow-LPT assigns J_4 to M_q.

If J_5 is also assigned to M_q, SONL $\leq p_2+p_3+p_6 \leq$ OPT. Otherwise, SONL $\leq \min\{q(p_1+p_4+p_6), p_2+p_3+p_5+p_6\}$. If OPT does not run J_1 with both J_5 and J_6, OPT $\geq p_1+p_4+p_6$ and SONL $\leq q\cdot$ OPT $\leq C(q)\cdot$ OPT. Hence, we assume that OPT runs J_1, J_5, and J_6 on one machine.

Case 1: OPT runs J_1, J_5, and J_6 on M_q. Note that SONL $\leq q(p_1+p_4+p_6) = q(p_1+p_5+p_6)+q(p_4-p_5)$. From the structure of OPT, $p_1+p_5+p_6 \leq \frac{1}{q}$ and $p_2+p_3+p_4 \leq 1$. Thus, $p_4 \leq \frac{1}{3}$. Recall that, by Lemma 1, we can assume $p_5 \geq p_6 > \frac{1}{4}$. We have SONL $\leq 1+q(p_4-p_5) \leq 1+\frac{q}{12}$. For $q \geq \frac{12}{11} \approx 1.09$, $1+\frac{q}{12} \leq q \leq C(q)$. Otherwise, $1+\frac{q}{12} \leq \frac{2}{3}+\frac{1}{2q} = C_1(q)$ holds for $q \leq \sqrt{10}-2 \approx 1.16$.

Case 2: OPT runs J_1, J_5, and J_6 on M_1. In this case, SONL $\leq \min\{q(1+p_4-p_5), \frac{1}{q} - p_4+p_5+p_6\}$ and $p_4 \leq \frac{1}{3q}$. Let $\alpha = p_4 - p_5$. Then, $p_6 \leq p_5 \leq -\alpha + p_4 \leq -\alpha + \frac{1}{3q}$. We get SONL $\leq \min\{q(1+\alpha), \frac{1}{q} - 2\alpha + \frac{1}{3q}\}$. The two terms are equal for $\alpha = \alpha_1 = \frac{4-3q^2}{3q(q+2)}$ (α_1 is positive for $q \leq \sqrt{4/3}$) and SONL $\leq \frac{6q+4}{3q+6} \leq C(q)$. Note that for $q \geq \sqrt{4/3}$, we get SONL $\leq \frac{1}{q} - 2\alpha + \frac{1}{3q} \leq \frac{4}{3q} \leq q \leq C(q)$.

5.2 Intervals 9 and 10, 13 and 14

In intervals 9–14, only sequences of five jobs and less can be slightly problematic, unlike the intervals 1–4, where sequences of six jobs had to be considered. Both algorithms for intervals 9–14 have a special rule for the second job only. In this section, we consider the algorithm Balanced-LPT, which tries to assign the second job to M_q.

Algorithm Balanced-LPT

Assign J_1 to M_1.
If $qp_2 > C(q)(p_1+p_2)$, assign J_2 to M_1, and otherwise to M_q.
Assign the rest of the jobs by the LPT rule.

Intervals 9 and 10. Recall that $C_9(q) = \frac{2q+3}{2q+2}$ and $C_{10}(q) = \frac{3q+2}{2q+3}$, and note that, in intervals 9 and 10, $C(q) = \max\{C_9(q), C_{10}(q)\}$.

By Lemma 2, for any sequence such that SONL $> C(q)$, OPT runs at most one job on M_q, since $C(q) \geq C_9(q) = \frac{2q+3}{2q+2} = 1+\frac{1}{2(q+1)}$. Similarly, OPT runs at most 4 jobs on M_1, since $\frac{1}{2(q+1)} \geq \frac{q}{5(q+1)}$ for $q \leq 2.5$. This means that, we need only consider sequences of at most five jobs. If an optimal off-line algorithm does not run any jobs on M_q, Balanced-LPT will not break the ratio, so we will assume that OPT schedules exactly one job on M_q and at most four on M_1.

In intervals 9 and 10, Balanced-LPT always assigns J_2 to M_q, since $C(q)(p_1+p_2) \geq C_9(q)(p_1+p_2) \geq 2C_9(q)p_2 \geq qp_2$ for $q \leq \frac{1+\sqrt{13}}{2} \approx 2.30$. This shows that sequences of at most two jobs cannot break the ratio.

By Lemma 3, if there are at least three jobs, SONL $\leq P - p_2$. If OPT does not run J_1 on M_q, we get OPT $\geq P - p_2$. This leaves only the case where OPT runs J_1 on M_q and all other jobs on M_1 to consider.

Three jobs. By the assumption that OPT runs J_1 on M_q and all other jobs on M_1, OPT $= \max\{qp_1, p_2+p_3\}$. By Lemma 3, SONL $\leq p_1+p_3 \leq 2p_1 \leq qp_1 \leq$ OPT.

Four jobs. Since OPT runs J_1 on M_q and J_2, J_3, and J_4 on M_1, $p_1 \leq \frac{1}{q}$ and $p_2+p_3+p_4 \leq 1$. Combining the latter inequality with $p_2 \geq p_3 \geq p_4$ gives $p_3+p_4 \leq \frac{2}{3}$. Thus, using Lemma 3, we have SONL $\leq p_1+p_3+p_4 \leq \frac{1}{q}+\frac{2}{3} = \frac{2q+3}{3q} \leq \frac{2q+3}{2q+2} = C_9(q)$ for $q \geq 2$.

Five jobs. If Balanced-LPT assigns at least one of the jobs J_3 and J_4 to the slow machine, SONL $\leq p_1+p_3+p_5 = P-p_2-p_4 \leq 1+\frac{1}{q}-2p_5$. By Corollary 1, SONL $> C_9(q)$ implies SONL $< 1+\frac{1}{q}-2(C_9(q)-1)\frac{q+1}{q} = 1+\frac{1}{q}-\frac{2(q+1)}{q(2q+2)} = 1$, which is a contradiction.

Otherwise, SONL $\leq q(p_2+p_5)$. Since OPT runs J_2, J_3, J_4, and J_5 on the fast machine, $p_3+p_4+p_5 \leq 1-p_2$. Thus, $p_5 \leq \frac{1}{3}(1-p_2)$, and SONL $\leq q(p_2+p_5) \leq \frac{q}{3}(1+2p_2)$. Furthermore, SONL $\leq P-p_2 \leq 1+\frac{1}{q}-p_2$, so $p_2 \leq 1+\frac{1}{q}-$ SONL. Thus, if SONL $> C_{10}(q)$, we get $p_2 < 1+\frac{1}{q}-\frac{3q+2}{2q+3} = \frac{1}{q}-\frac{q-1}{2q+3}$, and SONL $\leq \frac{q}{3}(1+2p_2) < \frac{q}{3}(1+\frac{2}{q}-\frac{2q-2}{2q+3}) = \frac{3q+2}{2q+3} = C_{10}(q)$, which is a contradiction.

Note that since the analysis is valid for all of interval 9, this means that Balanced-LPT has optimal competitive ratio for $2 \leq q \leq q_9 \approx 2.04$, as well as LPT.

Intervals 13 and 14. Recall that $C_{13}(q) = \frac{3}{4}+\frac{1}{q}$ and $C_{14}(q) = 1+\frac{q^2+2q-2-\sqrt{q^4+8q+4}}{2q+4}$, and note that, in intervals 13 and 14, $C(q) = \max\{C_{13}(q), C_{14}(q)\}$.

By Lemma 2, for any sequence such that SONL $> C(q)$, OPT runs at most one job on M_q and at most 4 jobs on M_1, since $C_{13}(q) > 1+\frac{1}{2q+2}$ for $q < \frac{1}{2}(1+\sqrt{17}) \approx 2.56$ and $C_{13}(q) > 1+\frac{q}{5q+5}$ for $q < \frac{1}{6}(5+\sqrt{105}) \approx 2.54$. $C_{14}(q) > 1+\frac{1}{2q+2}$ for $q \geq 2.591$ and $C_{14}(q) > 1+\frac{q}{5q+5}$ for $q \geq 2.49$.

Two jobs. If Slow-LPT assigns J_2 to M_1, $qp_2 > p_1+p_2$ by the definition of the algorithm, so OPT $= p_1+p_2 =$ SONL. Otherwise, by Lemma 3 and the definition of the algorithm, SONL $= qp_2 \leq C(q)(p_1+p_2)$, so SONL $\leq C(q)$OPT, since OPT $\geq \min\{qp_2, p_1+p_2\}$.

Consider the first job J_i assigned by Balanced-LPT to M_q. If J_2 runs on M_1, J_3 runs on M_q, since $qp_3 < 3p_3 \leq p_1+p_2+p_3$. Thus, $J_i = J_2$ or $J_i = J_3$. Let P_1 be the total size of jobs that OPT assigns to M_1. By Lemma 3, SONL $\leq P_1+\frac{1}{q}-p_i$, if the sequence has at least $i+1$ jobs. Assuming SONL $> C(q) \geq C_{13}(q) = \frac{3}{4}+\frac{1}{q}$, this gives

$$p_i < P_1 - \frac{3}{4} \leq \frac{1}{4}P_1, \tag{1}$$

since $P_1 \leq 1$. Let J_k be the largest job that OPT runs on M_1. Since OPT runs at most 4 jobs on M_1, $p_k \geq \frac{1}{4}P_1 > p_i$, and hence J_i arrives later than J_k. Thus, if Balanced-LPT runs J_2 on

M_q, OPT puts J_1 on M_1. But then, by Lemma 3, SONL $\leq P - p_1 \leq$ OPT. Therefore, we will assume that Balanced-LPT runs J_1 and J_2 on M_1 and J_3 on M_q.

Three jobs. By Lemma 3, SONL $= qp_3$. If OPT runs all jobs on M_1, OPT $= p_1 + p_2 + p_3 \geq 3p_3 > qp_3$. Otherwise OPT $\geq qp_3$.

Four jobs. By the definition of Balanced-LPT and the assumption that J_2 is assigned to M_1, $qp_2 > C(q)(p_1 + p_2)$ or, equivalently, $p_1 < (\frac{q}{C(q)} - 1)p_2$. Thus, SONL $\leq p_1 + p_2 + p_4 \leq (\frac{q}{C(q)} - 1 + 1)p_2 + p_4 = \frac{q}{C(q)}p_2 + p_4$.

If OPT runs both J_1 and J_2 on M_1, OPT $\geq p_1+p_2+p_4 \geq$ SONL. Otherwise, OPT runs J_1 or J_2 on M_q, meaning that $p_2 \leq \frac{1}{q}$. Furthermore, using (1), we get $p_4 \leq p_3 < \frac{1}{4}$. Substituting these upper bounds on p_2 and p_4 in the upper bound on SONL, we get SONL $< \frac{1}{C(q)} + \frac{1}{4}$. For $C(q) = C_{13}(q)$, this implies $q > \frac{1}{5}(5 + \sqrt{65}) \approx 2.61$ and for $C(q) = C_{14}(q)$, we get $q \leq 2.45$. Both are contradictions.

Five jobs. We show that, if SONL $> C(q)$, Balanced-LPT assigns J_4 to the fast machine. If J_4 is run on M_q, Lemma 3 gives that SONL $\leq \frac{q+1}{q} - p_3 - p_4 \leq \frac{q+1}{q} - 2p_5$. Using Corollary 1, we get SONL $< \frac{q+1}{q} - 2(C(q) - 1)\frac{q+1}{q} = \frac{q+1}{q}(3 - 2C(q))$. For $C(q) = C_{13}(q)$, this is smaller than $C(q)$ as long as $3q^2 - 6q - 8 \geq 0$, which is true when $q \leq \frac{1}{3}(3 + \sqrt{33}) \approx 2.92$. For $C(q) = C_{14}(q)$, the upper bound is smaller than $C(q)$ whenever q is at least the largest root of $9q^4 - 9q^3 - 28q^2 - 5q + 6$. This root is smaller than 2.37. Thus, we assume that Balanced-LPT puts J_1, J_2, and J_4 on the fast machine. Therefore, if SONL $> C(q)$, the following three inequalities must hold. The third inequality holds by Corollary 1.

$$q(p_3 + p_5) > C(q) \tag{2}$$

$$p_1 + p_2 + p_4 + p_5 > C(q) \tag{3}$$

$$p_5 > (C(q) - 1)\frac{q+1}{q} \tag{4}$$

Furthermore, since OPT runs J_2, J_3, J_4, and J_5 on M_1,

$$p_2 + p_3 + p_4 + p_5 \leq 1 \tag{5}$$

Since we assume that Balanced-LPT runs J_2 on M_1, by the definition of the algorithm,

$$p_1 + p_2 < \frac{q}{C(q)}p_2 \tag{6}$$

By (5), (2), and (4), we get

$$p_2 \;\leq\; 1 - (p_3 + p_5) - p_4 \;<\; 1 - \frac{C(q)}{q} - (C(q) - 1)\frac{q+1}{q} \;=\; \frac{2q+1}{q} - \frac{q+2}{q}C(q) \tag{7}$$

Rearranging (3) and substituting the upper bound on $p_1 + p_2$ of (6), we get

$$p_4 + p_5 > C(q) - \frac{q}{C(q)}p_2 \tag{8}$$

Now, using (5), (2), and $p_4 \geq p_5$, we get $1 - p_2 - p_4 - p_5 \geq p_3 > \frac{1}{q}C(q) - p_5 \geq \frac{1}{q}C(q) - \frac{1}{2}(p_4 + p_5)$, which gives $\frac{1}{2}(p_4 + p_5) + p_2 < 1 - \frac{1}{q}C(q)$. Substituting (8), we have $\frac{1}{2}C(q) + (1 - \frac{q}{2C(q)})p_2 < 1 - \frac{1}{q}C(q)$. Since $C(q) \geq \frac{q}{2}$ in intervals 13 and 14, the term $1 - \frac{q}{2C(q)}$ is non-positive, and (7) gives $\frac{1}{2}C(q) + \left(1 - \frac{q}{2C(q)}\right)\left(2q + 1 - (q+2)C(q)\right)\frac{1}{q} < 1 - \frac{1}{q}C(q)$. Simplifying this expression yields $(q+2)C^2(q) - (q^2 + 4q + 2)C(q) + (2q^2 + q) > 0$. This is true for $C(q) < 1 + \frac{q^2+2q-2-\sqrt{q^4+8q+4}}{2q+4} = C_{14}(q)$ and for $C(q) > 1 + \frac{q^2+2q-2+\sqrt{q^4+8q+4}}{2q+4}$ which is greater than 2 for $q \geq 2$. Both possibilities are contradictions, the latter because $C(q) \leq \frac{1}{4}(1 + \sqrt{17}) \approx 1.28$.

5.3 Intervals 11 and 12

When assigning the second job, the following algorithm tries to do the opposite of LPT. If $qp_2 < p_1 + p_2$, LPT puts J_2 on M_q, so Opposite-LPT puts J_2 on M_1, unless $p_1 + p_2 > C(q)qp_2$. Similarly, if $qp_2 \geq p_1 + p_2$, Opposite-LPT puts J_2 on M_q, unless $qp_2 > C(q)(p_1 + p_2)$.

Algorithm Opposite-LPT

Assign J_1 to M_1.

Assign J_2 to M_1 if one of the following holds:

$qp_2 < p_1 + p_2 \leq C(q)\, qp_2$ *or* $qp_2 > C(q)(p_1 + p_2)$.

Otherwise, assign J_2 to M_q.

Assign the rest of the jobs by the LPT rule.

Recall that $C_{11}(q) = \frac{1}{6q}(q^2 + 3 + \sqrt{q^4 - 6q^2 + 24q + 1})$ and $C_{12}(q) = \frac{q}{2}$, and note that in intervals 11 and 12, $C(q) = \max\{C_{11}(q), C_{12}(q)\}$.

By Lemma 2, if SONL $> C(q)$, OPT runs at most four jobs on M_1, since, in intervals 11 and 12, $C(q) \geq C_{11}(q) > 1 + \frac{q}{5q+5}$ for $q > 0$, and at most one job on M_q, since $C_{11}(q) > 1 + \frac{1}{2q+2}$ for $q \geq 2.02$.

Two jobs. If OPT runs two jobs on M_1, then OPT $= p_1 + p_2$. Otherwise, OPT $\geq qp_2$. Thus, OPT $\geq \min\{p_1 + p_2, qp_2\}$. Assume first that Opposite-LPT schedules J_2 on M_1. If $p_1 + p_2 \leq qp_2$, then OPT $= p_1 + p_2 =$ SONL. If $p_1 + p_2 > qp_2$, then, by the definition of the algorithm, SONL $= p_1 + p_2 \leq C(q)\, qp_2$. Assume now that Opposite-LPT assigns J_2 to M_q. Then, by Lemma 3 and the definition of the algorithm, SONL $= qp_2 \leq C(q)(p_1 + p_2)$.

Three jobs. If J_2 is run on M_q by Opposite-LPT, the proof is the same as for intervals 9 and 10. Otherwise, SONL $= qp_3$, since $p_1 + p_2 + p_3 \geq 3p_3 > qp_3$. However, OPT $\geq \min\{qp_3, p_1 + p_2 + p_3\} = qp_3 =$ SONL.

Four jobs. If J_2 is scheduled on M_q by Opposite-LPT, we use the same reasoning as in intervals 9 and 10 to get SONL $\leq \frac{2}{3} + \frac{1}{q}$, which is smaller than $C(q)$ in intervals 11 and 12. Thus, we assume that J_2 runs on M_1. There are two possible reasons for this. One possibility is that $qp_2 > C(q)(p_1 + p_2)$. However, this is equivalent to $p_2 > \frac{C(q)}{q - C(q)}p_1$. Using $C(q) \geq \frac{q}{2}$, we get $p_2 > p_1$, which is a contradiction. This leaves only the second possibility, $qp_2 \leq p_1 + p_2$, which is equivalent to $p_2 \leq \frac{p_1}{q-1}$.

Since $p_1 + p_2 + p_3 \geq 3p_3 > qp_3$, Opposite-LPT runs J_3 on M_q.

We analyze the three possibilities for which job is run on M_q by OPT.

Case 1: OPT runs J_3 or J_4 on M_q. In this case, $\text{OPT} \geq p_1 + p_2 + p_4$. By Lemma 3, $\text{SONL} \leq p_1 + p_2 + p_4 \leq \text{OPT}$.

Case 2: OPT runs J_1 on M_q. This gives $p_2 + p_3 + p_4 \leq 1$ and $p_1 \leq \frac{1}{q}$. Thus, $\text{SONL} \leq p_1 + p_2 + p_4 \leq p_1 + p_2 + \frac{1-p_2}{2} = p_1 + \frac{1}{2}p_2 + \frac{1}{2} \leq p_1 + \frac{1}{2} \cdot \frac{p_1}{q-1} + \frac{1}{2} = \frac{2q-1}{2q-2}p_1 + \frac{1}{2} \leq \frac{2q-1}{2q(q-1)} + \frac{1}{2}$, which is smaller than $C_{11}(q)$ in interval 11 and smaller than $C_{12}(q)$ in interval 12.

Case 3: OPT runs J_2 on M_q. In this case, $p_2 \leq \frac{1}{q}$ and $p_1 + p_3 + p_4 \leq 1$. Similar reasoning as in Case 2 gives $p_1 + p_2 + p_4 \leq \frac{q+1}{2(q-1)}p_1 + \frac{1}{2}$. Since $q(p_3 + p_4) \leq q(1 - p_1)$, $\text{SONL} \leq \min\{q(1-p_1), p_1+p_2+p_4\}$. Assuming $p_1+p_2+p_4 > C(q)$ yields $\frac{q+1}{2(q-1)}p_1+\frac{1}{2} > C(q)$, leading to $\text{SONL} \leq q(1 - p_1) \leq q(1 - \frac{(C(q)-\frac{1}{2})2(q-1)}{q+1})$. This can be simplified to $C(q) < \frac{2q^2}{2q^2-q+1}$. For $C(q) = C_{11}(q)$, this holds only for $q < 0$, and for $C(q) = C_{12}(q)$, this holds only for $q < 2.28 < q_{11}$.

Five jobs. We split the proof according to where Opposite-LPT schedules J_2.

Case 1: Opposite-LPT puts J_2 on M_1. As in the case of four jobs, J_3 runs on M_q, if $\text{SONL} > C(q)$. We now show that, if $\text{SONL} > C(q)$, then $q(p_3 + p_4) > p_1 + p_2 + p_4$, and J_4 is run on M_1. Using Corollary 1, we have that, if $\text{SONL} > C(q)$, then $p_5 \geq (C(q) - 1)\frac{q+1}{q}$ and hence $p_3 + p_5 \geq 2p_5 > 2(C(q) - 1)\frac{q+1}{q}$. Thus, $p_1 + p_2 + p_4 \leq \frac{q+1}{q} - (p_3 + p_5) < \frac{q+1}{q} - 2(C(q)-1)\frac{q+1}{q} = (3-2C(q))\frac{q+1}{q}$, and $q(p_3+p_4) \geq q(p_3+p_5) > 2(C(q)-1)(q+1)$. Since $2(C(q) - 1)$ is larger than $\frac{3-2C(q)}{q}$ in intervals 11 and 12, this means that J_4 is run on M_1, and $\text{SONL} \leq q(p_3 + p_5)$.

Assuming $\text{SONL} > C(q)$, we get $p_3 + p_5 > \frac{C(q)}{q} \geq \frac{1}{2}$, as $C(q) \geq \frac{q}{2}$. Thus, $p_2 + p_4 \geq p_3 + p_5 > \frac{1}{2}$ and $p_2 + p_3 + p_4 + p_5 > 1$, which is a contradiction since OPT runs 4 jobs on M_1.

Case 2: Opposite-LPT puts J_2 on M_q. By Lemma 3, $\text{SONL} \leq p_1 + p_3 + p_4 + p_5$. This means that if OPT runs J_1 on M_1, $\text{OPT} \geq p_1+p_3+p_4+p_5 \geq \text{SONL}$. Thus, we assume that OPT runs J_1 on M_q. We split the proof in subcases according to what caused Opposite-LPT to put J_2 on M_q.

Case 2.1: $p_1 + p_2 > C(q)\, qp_2$. In this case, $p_2 < \frac{1}{qC(q)-1}p_1 \leq \frac{1}{qC(q)-1}\frac{1}{q}$. By the analysis in Case 1, Opposite-LPT runs J_3 and J_4 on M_1. Thus, $\text{SONL} \leq q(p_2 + p_5) \leq q(p_2 + \frac{1-p_2}{3}) = \frac{2q}{3}p_2 + \frac{q}{3} < \frac{2}{3qC(q)-3} + \frac{q}{3}$, which is smaller than $C(q)$ in intervals 11 and 12.

Case 2.2: $p_1 + p_2 \leq qp_2$. In this case $p_2 \geq \frac{p_1}{q-1}$. Thus, $\text{SONL} \leq p_1 + p_3 + p_4 + p_5 = P - p_2 \leq 1 + \frac{1}{q} - \frac{p_1}{q-1} \leq 1 + \frac{1}{q} - \frac{1}{q^2-q} = \frac{q^2-2}{q^2-q}$, which is smaller than both $C_{11}(q)$ and $C_{12}(q)$ for $q > 1$.

6 Matching Lower Bounds

In this section we present job sequences that prove the lower bounds matching the upper bounds of Section 5 or, in the intervals where LPT is optimal, the bounds of LPT as given in [7].

For all of the intervals 4–15, q is at least as large as the competitive ratio (the competitive ratio is bounded by $\frac{1}{4}(1 + \sqrt{17}) \approx 1.28$), so any algorithm scheduling the first job on M_q has a competitive ratio of at least $C(q)$. Thus, when proving lower bounds for these intervals, it can be assumed that the algorithm schedules the first job on M_1. In the intervals 1–3 we take care of both possibilities (the first job may be assigned to either machine). In all sequences, unless otherwise mentioned, jobs are scaled so that if the sequence is completed, OPT $= 1$. All sequences have between three and six jobs, most of them have exactly five jobs.

Interval 1 ($1 \leq q \leq q_1 \approx 1.04$): $C_1(q) = \frac{2}{3} + \frac{1}{2q}$. The sequence consists of five jobs with sizes $p_1 = p_2 = \frac{1}{2q}$, $p_3 = p_4 = p_5 = \frac{1}{3}$.

The schedule of OPT_2 is achieved by running one job on each machine. This gives $\mathrm{OPT}_2 = \frac{1}{2}$. If SONL schedules both jobs on M_1, $\mathrm{SONL}_2 = \frac{1}{q} = \frac{2}{q}\,\mathrm{OPT}_2 > C_1(q)\,\mathrm{OPT}_2$. Thus, assume that SONL schedules one of the first two jobs on M_q.

For the complete sequence, OPT runs the first two jobs on M_q and the other jobs on M_1. If SONL assigns two of the last three jobs to M_1, SONL $\geq \frac{1}{2q} + \frac{2}{3}$, and if SONL assigns two of the last three jobs to M_q, SONL $\geq \frac{1}{2} + \frac{2}{3}q \geq \frac{1}{2q} + \frac{2}{3}$.

In the next two intervals, we build the sequence assuming that J_1 is assigned to the slow machine by the algorithm, since algorithms that put the first job on M_1, are not $C(q)$-competitive in these two intervals. The sequence proving this is based on a sequence from [7]. The definition of the sequence (as a function of p_1) is $p_2 = \frac{3+2q-2q^2}{2q^2+q}p_1$, $p_3 = p_4 = p_5 = \frac{q+1}{2q+1}p_1$. Note that for this sequence, the makespan of OPT is not necessarily 1.

For $1 \leq q \leq 1 + \sqrt{7}$, $0 \leq \frac{3+2q-2q^2}{2q^2+q} \leq 1$, so the sequence is well-defined and $p_2 \leq p_1$. Furthermore, since $q \leq \frac{1}{6}(1 + \sqrt{37}) \approx 1.18$, $p_3 \leq p_2$.

Since $\frac{3+2q-2q^2}{2q^2+q} \leq \frac{1}{q}$ for $q \geq 1$, $\mathrm{OPT}_2 = p_1$. If SONL puts J_2 on M_1, $\frac{\mathrm{SONL}_2}{\mathrm{OPT}_2} = \frac{p_1+p_2}{p_1} = \frac{3q+3}{2q^2+q} > \frac{3q+3}{2q^2+2q} = \frac{3}{2q} \geq \frac{3}{2}\sqrt{\frac{3}{4}} > 1.299 > C(q)$. Otherwise, SONL $\geq q(p_2 + 2p_3) = p_1 + 2p_3 = \frac{4q+3}{2q+1}p_1$. The optimal makespan is, however, $\frac{3q+3}{2q+1}p_1$, achieved by scheduling the first two jobs on M_q and the last three on M_1. This gives a ratio of $\frac{4q+3}{3q+3}$, which is larger than $C(q)$ in intervals 2 and 3.

Interval 2 ($q_1 \leq q \leq q_2 \approx 1.14$): $C_2(q) = 1 + \frac{1}{2}\left(4q^2 + 4q - 1 - \sqrt{(4q^2 + 4q - 1)^2 - 4q^2}\right)$. The sequence consists of five jobs with sizes $p_1 = \frac{1}{q} - \frac{2q+1}{q+1}p_5$, $p_2 = \frac{2q+1}{q+1}p_5$, $p_3 = 1 - 2p_5$, $p_4 = p_5 = \frac{q+1}{2q}(4q^2 + 4q - 1 - \sqrt{(4q^2 + 4q - 1)^2 - 4q^2})$.

The first job is larger than the second job, since $\frac{1}{q} > \frac{4q+2}{q+1}p_5$ for $q < \frac{1}{4}(1 + \sqrt{13}) \approx 1.15$. This is equivalent to $p_5 < \frac{q+1}{4q^2+2q}$ for $q < \frac{1}{4}(1 + \sqrt{13})$. Since $\frac{q+1}{4q^2+2q} \leq \frac{1}{3}$ for $q \geq 1$, this also implies $p_3 \geq p_4$. Finally, $p_2 \geq p_3$, since $p_5 \geq \frac{q+1}{4q+3}$ for $q \leq \frac{1}{4}(1 + \sqrt{13})$.

If the algorithm assigns J_2 to M_q, $\mathrm{SONL}_2 = 1$. However, we show that $\mathrm{OPT}_2 \leq \frac{2}{3}$. Clearly, $\mathrm{OPT}_2 \leq \max\{p_1, qp_2\}$. By the proof that $p_1 \geq p_2$, $p_2 \leq \frac{1}{2q}$. Thus, $qp_2 \leq \frac{1}{2}$. Furthermore, $p_3 \leq \frac{1}{2}$, and $p_5 \geq \frac{1}{2}(1 - p_3) \geq \frac{1}{4}$. Now, $p_1 = \frac{1}{q} - \frac{2q+1}{q+1}p_5 \leq \frac{1}{q} - \frac{2q+1}{4q+4}\frac{2}{3}$ for all $q \geq 1$. Thus, to be $C(q)$-competitive, the algorithm must put J_2 on M_1.

If the algorithm assigns the third job to M_q, $\mathrm{SONL}_3 \geq (p_1 + p_3)q = 1 - \frac{2q^2+q}{q+1}p_5 + q - \frac{2q^2+2q}{q+1}p_5 = q + 1 - \frac{4q^2+3q}{q+1}p_5$. On the other hand, $\mathrm{OPT}_3 \leq \max\{qp_1, p_2 + p_3\} = \max\{1 - \frac{2q^2+1}{q+1}p_5, 1 - \frac{1}{q+1}p_5\} = 1 - \frac{1}{q+1}p_5$. Thus, in this case $\mathrm{SONL}_3 \leq C(q)\mathrm{OPT}_3$, if and only if $q + 1 - \frac{4q^2+3q}{q+1}p_5 \leq C(q)(1 - \frac{1}{q+1}p_5)$. Note that $p_5 = \frac{1}{q}(C(q) - 1)(q + 1)$. Substituting this in the inequality, we get $1 + \frac{1}{2}\left(4q^2 + 4q - 1 - \sqrt{(4q^2 + 4q - 1)^2 - 4q^2}\right) \leq C(q) \leq 1 + \frac{1}{2}\left(4q^2 + 4q - 1 + \sqrt{(4q^2 + 4q - 1)^2 - 4q^2}\right) = C_2(q)$. Thus, if the algorithm is better than $C_2(q)$-competitive in this interval, it assigns the third job to M_1.

Now, consider the whole sequence. OPT runs J_1 and J_2 on M_q and the other jobs on M_1. If SONL runs J_4 and J_5 on M_q, SONL $\geq (p_1 + 2p_5)q = (\frac{1}{q} - \frac{2q+1}{q+1}p_5 + \frac{2q+2}{q+1}p_5)q = 1 + \frac{q}{q+1}p_5 = C_2(q)$. Otherwise, SONL $\geq (p_2 + p_3 + p_5) = \frac{2q+1}{q+1}p_5 + 1 - \frac{q+1}{q+1}p_5 = 1 + \frac{q}{q+1}p_5 = C_2(q)$.

Interval 3 ($q_2 \leq q \leq \sqrt{4/3} \approx 1.15$): $C_3(q) = \frac{6q+4}{3q+6}$. This is the only case of a lower bound sequence that consists of six jobs. Let $\beta = \frac{4-3q^2}{3q(q+2)}$. In this interval, $\beta > 0$. The sequence is $p_1 = 1 - 2p_5$, $p_2 = p_3 = p_4 = \frac{1}{3q}$, $p_5 = p_6 = \frac{1}{3q} - \beta$.

The first job is larger than the second job, because $p_1 = 1 - 2p_5 > 1 - 2p_2 > p_2$, since $p_2 < \frac{1}{3}$.

Recall that we can assume that the algorithm schedules J_1 on M_q. We now show that we can also assume that among the following three jobs, exactly one is scheduled on M_q. First note that $\mathrm{OPT}_4 \leq \max\{p_1 + p_2, q(p_3 + p_4)\} = \max\{1 - 2(\frac{1}{3q} - \beta) + \frac{1}{3q}, \frac{2}{3}\} = \max\{1 - \frac{1}{3q} + 2\beta, \frac{2}{3}\} = 1 - \frac{1}{3q} + 2\beta = \frac{3q(q+2)-(q+2)+2(4-3q^2)}{3q(q+2)} = \frac{6+5q-3q^2}{3q(q+2)}$. Now, if SONL puts all three jobs J_2, J_3, J_4 on M_1, $\mathrm{SONL}_4 \geq \frac{1}{q} = \frac{6+5q-3q^2}{3q+6}\,\mathrm{OPT}_4$, which is larger than $C_3(q)\,\mathrm{OPT}_4$ for $q < \frac{1}{6}(5 + \sqrt{97}) \approx 2.47$. If SONL puts two of the jobs J_2, J_3, and J_4 on M_q, $\mathrm{SONL}_4 \geq q(p_1 + p_3 + p_4) = q(1 - 2(\frac{1}{3q} - \beta) + \frac{2}{3q}) = q(1 + 2\beta) > 1$, yielding an even worse ratio.

If J_5 and J_6 are both scheduled on M_1, SONL $\geq p_3 + p_4 + p_5 + p_6 = \frac{2}{3q} + \frac{2}{3q} - 2\beta = \frac{4(q+2)-2(4-3q^2)}{3q(q+2)} = \frac{6q+4}{3q+6}$. Otherwise, SONL $\geq q(p_1 + p_4 + p_5) = q(1 - 2p_5 + p_4 + p_5) = q(1 + p_4 - p_5) = q(1 + \beta) = \frac{6q+4}{3q+6}$.

For intervals 4–9, we use sequences given in [7] as negative examples for LPT. We show that those sequences are in fact lower bound sequences for any semi-online algorithm. In intervals 4 and 9, the proof holds for the entire interval, even though LPT is not optimal in the complete interval.

Interval 4 ($\sqrt{4/3} \leq q \leq \frac{1}{4}(1 + \sqrt{17}) \approx 1.28$): $C_4(q) = q$. The sequence consists of three jobs with sizes $p_1 = \frac{1}{q}$, $p_2 = p_3 = \frac{1}{2}$. OPT is achieved by running J_1 on M_q and the other jobs on M_1. We consider two cases. If both of the last two jobs are assigned to the slow machine, then SONL $\geq q(\frac{1}{2} + \frac{1}{2}) = q$. If at least one such job is assigned to the fast machine then SONL $\geq \frac{1}{q} + \frac{1}{2} = \frac{q+2}{2q}$. Since in this interval $2q^2 - q - 2 \leq 0$, in this case SONL $\geq q$ also holds.

Interval 5 ($\frac{1}{4}(1 + \sqrt{17}) \leq q \leq \sqrt{2} \approx 1.41$): $C_5(q) = \frac{q+2}{2q}$. The sequence (and also the optimal off-line schedule) is the same as in the previous interval. Only, in this interval, $2q^2 - q - 2 \geq 0$ and hence the lower bound is $\frac{q+2}{2q}$.

Interval 6 ($\sqrt{2} \leq q \leq \frac{1}{4}(1 + \sqrt{33}) \approx 1.69$): $C_6(q) = \frac{2q+3}{2q+2}$. The sequence consists of four jobs with sizes $p_1 = \frac{2q^2+q-2}{2q(q+1)}$, $p_2 = \frac{q+2}{2q(q+1)}$, $p_3 = p_4 = \frac{1}{2q}$. Since $q \geq \sqrt{2}$, $p_2 \leq p_1$.It is easy to see that $p_3 \leq p_2$ for all $q \geq 1$.

OPT_2 is achieved by running J_1 on M_1 and J_2 on the slow machine. The fast machine is loaded by qp_2 and, since $q^2 - q - 2 \geq 0$ for $q \leq 2$, we have $qp_2 \geq p_1$ and $\mathrm{OPT}_2 = \frac{q+2}{2q+2}$. If the algorithm assigns J_2 to M_1, we get $\mathrm{SONL}_2 = p_1 + p_2 = \frac{2q^2+2q}{2q(q+1)} = 1$. This gives a ratio of $\frac{2q+2}{q+2}$. For all $q \geq 1$, $\frac{2q+2}{q+2} \geq \frac{2q+3}{2q+2}$ holds and thus it is not possible to run both jobs on M_1 without violating the competitive ratio for the subsequence J_1, J_2. Thus, it can be assumed that SONL schedules the second job on M_q.

For the complete sequence, OPT is achieved by running the first two jobs on M_1, and the last two jobs on M_q. For the semi-online algorithm, if at least one of the jobs J_3 and J_4 runs on M_q then SONL $\geq q(p_2+p_3) = \frac{2q+3}{2q+2}$. Otherwise, all jobs but J_2 run on M_1 and SONL $\geq p_1+p_3+p_4 = \frac{2q^2+3q}{2q(q+1)} = \frac{2q+3}{2q+2}$. In both cases the lower bound on the competitive ratio is achieved.

Interval 7 ($\frac{1}{4}(1+\sqrt{33}) \leq q \leq \frac{1}{2}(1+\sqrt{7}) \approx 1.82$): $C_7(q) = \frac{2q+1}{q+2}$. The sequence consists of four jobs with sizes $p_1 = \frac{q+2}{q(q+2)}$, $p_2 = \frac{2+2q-q^2}{q(q+2)}$, $p_3 = p_4 = \frac{q^2-1}{q(q+2)}$. Since $q \geq 1$, $p_1 \geq p_2$. Moreover, $p_2 \geq p_3$, since $2q^2-2q-3 \leq 0$ holds for $q \leq \frac{1}{2}(1+\sqrt{7})$.

As in the previous interval, OPT_2 is achieved by running J_1 on M_1 and J_2 on M_q, and $\text{OPT}_2 = \max(p_1, qp_2)$. Again, $qp_2 \geq p_1$, since $q+2 \leq q(2+2q-q^2)$ for $1 \leq q \leq 2$. The semi-online algorithm must behave the same in order not to violate the competitive ratio. Comparing $\frac{p_1+p_2}{qp_2}$ to $\frac{2q+1}{q+2}$ we get that the violation happens when $2q^4-4q^3-5q^2+8q+8 \geq 0$. This function is always positive in this interval.

For the complete sequence, OPT runs J_1 on M_q, and all other jobs on M_1. If the semi-online algorithm runs at least one more job than p_2 on M_q, it gives a load of at least $q(p_2+p_3) = \frac{2q+1}{q+2}$. Otherwise, the load on M_1 is at least $p_1+p_3+p_4 = \frac{2q^2+q}{q(q+2)} = \frac{2q+1}{q+2}$. In both cases, the lower bound on the competitive ratio is achieved.

Interval 8 ($\frac{1}{2}(1+\sqrt{7}) \leq q \leq 2$): $C_8(q) = \frac{2q+3}{3q}$. The sequence consists of four jobs with sizes $p_1 = \frac{1}{q}$, $p_2 = p_3 = p_4 = \frac{1}{3}$. OPT runs J_1 on M_q, and the other jobs on M_1. If at least two jobs run on M_q, SONL $\geq \frac{2q}{3}$. For $q \geq \frac{1}{2}(1+\sqrt{7})$, $\frac{2q}{3} \geq \frac{2q+3}{3q}$. Otherwise, SONL $\geq \frac{1}{q}+\frac{2}{3} = \frac{2q+3}{3q}$. In both cases the lower bound on the competitive ratio is achieved.

Interval 9 ($2 \leq q \leq \frac{1}{2}(1+\sqrt{11}) \approx 2.16$): $C_9(q) = \frac{2q+3}{2q+2}$. The sequence is the same as in interval 6. The condition for this sequence to be non-increasing was $q \geq \sqrt{2}$, which clearly holds in this interval too.

Again, we show that the semi-online algorithm assigns J_2 to M_q, and then the proof for the complete sequence is identical to the proof for interval 6. Note that since $q \geq 2$, OPT_2 still assigns J_1 to M_1 and J_2 to M_q, but $p_1 \geq qp_2$, and hence $\text{OPT}_2 = p_1$. If SONL assigns J_2 to M_1, then $\text{SONL}_2 = p_1+p_2$. Since $\frac{p_1+p_2}{p_1} = 1+\frac{q+2}{2q^2+q-2}$ and $\frac{q+2}{2q^2+q-2} \geq \frac{1}{2q+2}$ for all positive q, SONL must schedule J_2 on M_q not to break the competitive ratio. As mentioned before, from this point on, the proof is identical to the proof for interval 6.

Interval 10 ($\frac{1}{2}(1+\sqrt{11}) < q \leq q_{10} \approx 2.20$): $C_{10}(q) = \frac{3q+2}{2q+3}$. The sequence consists of five jobs: $p_1 = \frac{1}{q}$, $p_2 = \frac{-q^2+3q+3}{2q^2+3q}$, $p_3 = p_4 = p_5 = \frac{q^2-1}{2q^2+3q}$. The sequence is non-increasing, since $p_1 \geq p_2$ for all $q \geq 1$ and $p_2 \geq p_3$ for $q \leq \frac{1}{4}(3+\sqrt{41})$.

Consider the subsequence J_1, J_2. Since $q^3-3q^2-q+3 \leq 0$ for $1 \leq q \leq 3$, $\text{OPT}_2 = \max\{p_1, qp_2\} = qp_2$. If SONL puts J_2 on M_1, $\text{SONL}_2 = \frac{-q^2+5q+6}{2q^2+3q}$. This is larger than $C_{10}(q)qp_2$ for $q \leq q_{10}$. Thus, any $C(q)$-competitive algorithm must run J_2 on M_q.

For the whole sequence, we consider two cases. If J_2 is the only job to be put on M_q, SONL $\geq p_1+3p_3 = \frac{2q+3+3q^2-3}{2q^2+3q}\frac{3q+2}{2q+3} = C_{10}(q)$. If J_2 is not alone on M_q, SONL $\geq q(p_2+p_3) = q\frac{3q-q^2+3+q^2-1}{2q^2+3q} = C_{10}(q)$. Since $qp_1 = p_2+3p_3 = 1$, this proves the lower bound.

Interval 11 ($q_{10} \le q \le q_{11} \approx 2.33$): $C_{11}(q) = \frac{1}{6q}(q^2+3+\sqrt{q^4-6q^2+24q+9})$. The sequence consists of five jobs: $p_1 = \frac{1}{q}$, $p_2 = \frac{1}{4q^2}(3-q^2+\sqrt{q^4-6q^2+24q+9})$, $p_3 = p_4 = p_5 = \frac{1}{3}(1-p_2)$.

To show that the sequence is valid we need to show $p_1 \ge p_2$ and $p_2 \ge p_3$. The first is true for all $q \ge \sqrt{7}-1 \approx 1.65$. The second results in the inequality $q^3-2q-8 \le 0$ which is true for all $q \le q_{11}$.

Consider the subsequence J_1, J_2. Since $qp_2 \ge \frac{1}{q}$ is equivalent to $q^2-3q-1 \le 0$, which holds for $q \le \frac{1}{2}(3+\sqrt{13})$, OPT $= qp_2$. If SONL puts J_2 on fast, SONL $= p_1+p_2$ which is exactly equal to $C_{10}(q)$. Thus, assume that SONL puts J_2 on M_q. If at least one other job is put on M_q too, SONL $\ge q(p_2+\frac{1-p_2}{3}) = \frac{q}{3}(2p_2+1)$, which equals $C_{10}(q)$. Otherwise, SONL $= p_1+1-p_2 = \frac{1}{q}+1-p_2$, which is at least $C_{10}(q)$, when q is at least q_{10} and at most the largest root of $3q^4-9q^3-8q^2+21q+18$ which is approximately 2.76.

Intervals 12 and 13 ($q_{11} \le q \le \frac{1}{4}(3+\sqrt{41}) \approx 2.35$ and $\frac{1}{4}(3+\sqrt{41}) \le q \le q_{13} \approx 2.5111$): $C_{12}(q) = \frac{q}{2}$, $C_{13}(q) = \frac{3}{4}+\frac{1}{q}$. The sequence for both intervals consists of five jobs: $p_1 = \frac{1}{q}$, $p_2 = p_3 = p_4 = p_5 = \frac{1}{4}$.

Since the algorithm runs J_1 on M_1, we may consider only the complete sequence. OPT runs J_1 on M_q, and all other jobs on M_1. The algorithm needs to either run at least three of the other jobs on M_1, or at least two of them on M_q. In the first case, SONL $\ge \frac{1}{q}+\frac{3}{4}$. In the second case SONL $\ge \frac{q}{2}$. The competitive ratio is at least $\min\{\frac{1}{q}+\frac{3}{4}, \frac{q}{2}\}$. In interval 12, $\frac{q}{2} \le \frac{3}{4}+\frac{1}{q}$, and the competitive ratio is at least $C_{12}(q)$. In interval 13, $\frac{3}{4}+\frac{1}{q} \le \frac{q}{2}$, and the competitive ratio is at least $C_{13}(q)$.

Interval 14 ($q_{13} \le q \le q_{14} \approx 2.57$): $C_{14}(q) = 1+\frac{q^2+2q-2-\sqrt{q^4+8q+4}}{2q+4}$. The sequence consists of five jobs: $p_1 = \frac{1}{q}$, $p_2 = 1-\frac{1}{q}-\frac{q+2}{q+1}p_5$, $p_3 = \frac{1}{q}-\frac{q}{q+1}p_5$, $p_4 = p_5 = \frac{q+1}{2q(q+2)}\big(q^2+2q-2-\sqrt{q^4+8q+4}\big)$. The sequence is valid, since $p_1 \ge p_2 \ge p_3$ for $q \ge 2$, $p_3 \ge p_4$ for $q \le q_{14}$, and $p_5 \ge 0$ for $q \ge 2$.

Consider the subsequence J_1, J_2. Any $C(q)$-competitive algorithm assigns J_2 to M_1, since $\frac{qp_2}{p_1+p_2} = C_{14}(q)$.

Consider now the subsequence J_1, J_2, J_3. $\mathrm{OPT}_3 \le \max\{p_1+p_2, qp_3\} = p_1+p_2$, since $p_1+p_2 > qp_3$ for any $q > 2$. Any $C(q)$-competitive algorithm puts J_3 on M_q, since $\frac{p_1+p_2+p_3}{p_1+p_2} \ge C_{14}(q)$ for any $q \le 2.8$.

For the whole sequence, OPT runs J_1 on M_q and the remaining four jobs on M_1. For the semi-online algorithm there are two cases. If both J_4 and J_5 run on M_1, we get SONL $\ge p_1+p_2+2p_5 = 1+\frac{q}{q+1}p_5 = C_{14}(q)$. Otherwise, SONL $\ge q(p_3+p_5) = q(\frac{1}{q}+\frac{1}{q+1}p_5) = C_{14}(q)$.

Interval 15 ($q \ge q_{14}$): $C_{15}(q) = \frac{2q+2}{2q+1}$. For this whole interval, LPT is optimal. However, using the the negative example in [7] does not directly yield the desired general bound. We use an adaptation of that sequence.

For $q \ge 1+\sqrt{3}$, the adapted sequence consists of the five jobs $p_1 = \frac{2q^2-2q-3}{2q^2+q}$, $p_2 = \frac{1}{q}$, $p_3 = p_4 = p_5 = \frac{q+1}{2q^2+q}$. Note that $p_1 \ge p_2$ for $q \ge 1+\sqrt{3}$ and $p_2 \ge p_3$ for $q \ge \frac{1}{4}(3+\sqrt{41})$. We show that, any algorithm with a competitive ratio of $C_{15}(q)$ must schedule J_2 on M_1. After two jobs, $\mathrm{OPT}_2 \le p_1+p_2 = \frac{2q^2-2}{2q^2+q}$. If SONL runs J_2 on M_q, $\mathrm{SONL}_2 \ge 1$. In this case, the competitive ratio is at least $\frac{2q^2+q}{2q^2-2} \ge \frac{2q+2}{2q+1}$ (since $(2q+1)^2 \ge 2q(2q+2) \ge (2q-\frac{2}{q})(2q+2)$). Thus, when considering the whole sequence, we can assume that SONL schedules J_1 and J_2

on M_1. OPT is achieved by running J_2 on M_q and all other jobs on M_1. There are two cases. If SONL schedules at least two jobs on M_q, then SONL $\geq \frac{2q(q+1)}{q(2q+1)} = \frac{2q+2}{2q+1}$. Otherwise, two extra jobs (apart from J_1 and J_2) run on M_1, which gives SONL $\geq \frac{2q^2-2}{2q^2+q} + \frac{2(q+1)}{2q^2+q} = \frac{2q^2+2q}{2q^2+q} = \frac{2q+2}{2q+1}$ as needed.

For $q < 1+\sqrt{3}$ the above sequence does not apply. This follows from the fact that $p_1 \geq p_2$ requires $2q^2 - 2q - 3 \geq 2q + 1$ which holds only for $q \geq 1 + \sqrt{3}$. We switch the order of the first two jobs in the sequence and use the sequence $p'_1 = p_2$, $p'_2 = p_1$, $p'_3 = p'_4 = p'_5 = p_3$.

We show that for $q \geq q_{14}$, J'_2 must be scheduled on M_1. After that, the sequence behaves the same way as the sequence for $q \geq 1 + \sqrt{3}$. OPT$_2$ is again at most $\frac{2q^2-2}{2q^2+q}$. However, if J'_2 runs on M_q, SONL$_2 = \frac{2q^2-2q-3}{2q+1}$. This violates the competitive ratio when $\frac{2q^2-2q-3}{2q^2-2} \geq \frac{2q+2}{2q^2+q}$. This is true when q is at least the largest root of $4q^4 - 6q^3 - 12q^2 + q + 4$, which is also the largest root of $4q^5 + 2q^4 - 24q^3 - 23q^2 + 6q + 8$.

7 Conclusion

We have given a complete analysis of deterministic semi-online algorithms for two related machines and non-increasing job sizes. It is left as an open problem to analyze the behavior of randomized algorithms for two machines. For a general setting of m machines, it should be difficult to give a complete analysis depending on the speeds. However, it is intriguing to close the open question: what is the best overall competitive ratio for m machines?

References

1. Y. Cho and S. Sahni. Bounds for List Schedules on Uniform Processors. *SIAM Journal on Computing*, 9(1):91–103, 1980.
2. G. Dobson. Scheduling Independent Tasks on Uniform Processors. *SIAM Journal on Computing*, 13(4):705–716, 1984.
3. L. Epstein, J. Noga, S. S. Seiden, J. Sgall, and G. J. Woeginger. Randomized Online Scheduling on Two Uniform Machines. *Journal of Scheduling*, 4(2):71–92, 2001.
4. D. K. Friesen. Tighter Bounds for LPT Scheduling on Uniform Processors. *SIAM Journal on Computing*, 16(3):554–560, 1987.
5. T. Gonzalez, O. H. Ibarra, and S. Sahni. Bounds for LPT Schedules on Uniform Processors. *SIAM Journal on Computing*, 6(1):155–166, 1977.
6. R. L. Graham. Bounds on Multiprocessing Timing Anomalies. *SIAM J. Appl. Math*, 17:416–429, 1969.
7. P. Mireault, J. B. Orlin, and R. V. Vohra. A Parametric Worst Case Analysis of the LPT Heuristic for Two Uniform Machines. *Operations Research*, 45:116–125, 1997.
8. S. Seiden, J. Sgall, and G. J. Woeginger. Semi-Online Scheduling with Decreasing Job Sizes. *Operations Research Letters*, 27(5):215–221, 2000.
9. J. Sgall. On-Line Scheduling. In *A. Fiat and G. J. Woeginger, editors,* Online Algorithms: The State of the Art, volume 1442 of *LNCS*, pages 196–231. Springer-Verlag, 1998.