ELSEVIER

Theoretical Computer Science 262 (2001) 437–458

Theoretical Computer Science

www.elsevier.com/locate/tcs

Preemptive multiprocessor scheduling with rejection ☆

Steven S. Seiden

Department of Computer Science, 298 Coates Hall, Louisiana State University, Baton Rouge, LA 70803, USA

Received 23 February 1998; revised 6 April 2000; accepted 27 June 2000
Communicated by G. Ausiello

Abstract

The problem of online multiprocessor scheduling with rejection was introduced by Bartal et al. (SIAM J. Discrete Math. 13(1) (2000) 64–78). They show that for this problem the competitive ratio is $1 + \phi \approx 2.61803$, where ϕ is the golden ratio. A modified model of multiprocessor scheduling with rejection is presented where preemption is allowed. For this model, it is shown that better performance is possible. An online algorithm which is $(4 + \sqrt{10})/3 < 2.38743$-competitive is presented. We prove that the competitive ratio of any online algorithm is at least 2.12457. We say that an algorithm schedules obliviously if the accepted jobs are scheduled without knowledge of the rejection penalties. We also show a lower bound of 2.33246 on the competitive ratio of any online algorithm which schedules obliviously. As a subroutine in our algorithm, we use a new optimal online algorithm for preemptive scheduling without rejection. This algorithm never achieves a larger makespan than that of the previously known algorithm of Chen et al. (Oper. Res. Lett. 18(3) (1995) 127–131), and achieves a smaller makespan for some inputs.

Keywords: Analysis of algorithms; Scheduling; Online algorithms

1. Introduction

Consider the following problem: we have m machines. Jobs arrive periodically and are scheduled or rejected at a given penalty. The processing time (size) of each job is known when it arrives, and each job is performed equally well on any machine. Scheduling occurs *online*; each job is irrevocably assigned to a machine or rejected before the next job arrives. By the *load* of a machine, we mean the sum of the sizes of the jobs assigned to that machine. The *makespan* is the maximum machine load. The cost of a schedule is the makespan plus the sum of the penalties for rejected jobs.

☆ This research was done while the author was a post-doc at the Technical University, Graz and at the Max-Planck-Institute for Computer Science. It has been supported by the START program Y43-MAT of the Austrian Ministry of Science and by the EU ESPRIT LTR Project N. 20244 (ALCOM-IT), WP 3.2.

0304-3975/01/$ - see front matter
PII: S0304-3975(00)00288-7

Competitive analysis is a type of worst-case analysis where the performance of an online algorithm is compared to that of the optimal offline algorithm. This approach to analyzing online problems was initiated by Sleator and Tarjan, who used it to analyze the List Update problem [16]. The term competitive analysis originated in [12]. For a given job sequence σ let $\text{cost}_A(\sigma)$ be the cost incurred by an algorithm A on σ. Let $\text{cost}(\sigma)$ be the cost of the optimal offline schedule for σ. A scheduling algorithm A is *c-competitive* if

$$\text{cost}_A(\sigma) \leqslant c\,\text{cost}(\sigma),$$

for all job sequences σ. The infimum over all c such A is c-competitive is called the *competitive ratio* of A. The goal is to find an algorithm with minimal competitive ratio. We call this problem *Multiprocessor Scheduling with Rejection* or MSR for short [4].

We also consider the situation where the accepted jobs are preemptively scheduled. With preemption, a job may be scheduled on multiple machines. A *time slot* is a non-empty real interval $(t,u]$ where $t \geqslant 0$. When each job arrives it is rejected or assigned time slots on one or more machines. The sum of the sizes of these time slots must equal the size of the job. Further, if a job is assigned to time slots $(t_1,u_1]$, $(t_2,u_2],\ldots,(t_i,u_i]$, then $u_j \leqslant t_{j+1}$ for $j=1,\ldots,i-1$. No two jobs may have overlapping time slots on the same machine. The makespan is the last finishing time of any job. Again, the cost of a schedule is the makespan plus the sum of the penalties for rejected jobs. We call this problem *Preemptive Multiprocessor Scheduling with Rejection* or PMSR for short.

The study of multiprocessor online scheduling was initiated by Graham who showed an algorithm called LIST which is $(2-1/m)$-competitive. Since Graham's seminal work, many researchers have investigated this problem [1–3, 5, 6, 9–11, 14, 15]. The competitive ratio is known to lie in the interval $[1.852, 1.923]$ for large m [1].

Preemptive multiprocessor online scheduling was introduced by Chen et al. [7]. They present a $\beta(m)$-competitive algorithm and a matching lower bound where

$$\beta(m) = \frac{\theta^m}{\theta^m - 1}, \qquad \theta = \frac{m}{m-1}.$$

Note that $\beta(m)$ converges to $\mathrm{e}/(\mathrm{e}-1) \approx 1.58197$ as m goes to infinity.

MSR was first studied by Bartal et al. [4]. For $m=2$, they show a ϕ-competitive algorithm, called REJECT PENALTY, where $\phi = (1+\sqrt{5})/2 \approx 1.61803$ is the golden ratio. For $m>2$, they present an algorithm, called REJECT TOTAL PENALTY, which achieves a competitive ratio of

$$1 + \frac{1 - \frac{2}{m} + \sqrt{5 - \frac{8}{m} + \frac{4}{m^2}}}{2}. \tag{1}$$

This approaches $1+\phi$ as m goes to infinity. Bartal et al. [4] further show a lower bound of c for MSR where c is a solution of

$$c^{m-1} + \cdots + c^0 = c^m. \tag{2}$$

Table 1
Upper bound results

m	γ	α	c
2	1	$\phi - 1$	ϕ
3	27/38	29/38	2
4	0.696772	0.751091	2.09948
5	0.689232	0.744497	2.15804
6	0.684465	0.740337	2.19665
7	0.681178	0.737473	2.22403
8	0.678774	0.735381	2.24446
9	0.676939	0.733785	2.26029
10	0.675492	0.732528	2.27292
∞	2/3	$(\sqrt{10}-1)/3$	$(4+\sqrt{10})/3$

This lower bound also applies to PMSR. For $m=2$ this implies that the competitive ratio is at least ϕ; REJECT PENALTY is optimal for $m=2$ with or without preemption. As m grows, the solution to (2) approaches 2. Thus a lower bound of 2 holds for PMSR. Bartal et al. [4] further show a lower bound which approaches $1+\phi$ as m grows. However, this lower bound does not carry over to the preemptive case.

Offline scheduling with rejection has been studied Bartal et al. [4] and Engels et al. [8].

The algorithms of Bartal et al. [4] consist of two parts, a *rejection scheme*, which decides which jobs are rejected, and a *scheduling algorithm*, which assigns accepted jobs to a machine. In [4] the given rejection schemes are combined with Graham's LIST algorithm. In fact, the rejection schemes given there can be combined with any scheduling algorithm. By combining them with algorithms for different scheduling models, we get new algorithms for scheduling with rejection. We do this for the preemptive scheduling model. However, the technique we use is general and potentially could be used in many different scheduling models.

The rejection scheme that we present is a generalization of the REJECT TOTAL PENALTY scheme of [4]. We combine this scheme with a new preemptive scheduling algorithm, based on that of Chen et al. [7]. What results is a family of algorithms with two real parameters, α and γ. We show that for a specific choice of α and γ, we get an algorithm which is $(4+\sqrt{10})/3 < 2.38743$-competitive for all $m \geqslant 2$. For small m, we derive values of α and γ which result in better bounds, summarized in Table 1. Further, we show that for this family of algorithms a lower bound of 2.38518 holds. Therefore, our algorithm achieves a competitive ratio within 3×10^{-3} of the best possible bound achievable within this family.

We show that the competitive ratio of any online PMSR algorithm is at least 2.12457. We also show a lower bound for an interesting class of PMSR algorithms: We say that an algorithm *schedules obliviously* if the accepted jobs are scheduled without knowledge of the rejection penalties. We show a lower bound of 2.33246 on the competitive ratio of any such algorithm. The optimal online algorithm for MSR schedules obliviously

[4]. It would, therefore, be curious if some algorithm which beats this lower bound exists.

Finally, we note that our preemptive algorithm is of independent interest. It achieves the same competitive ratio as that of Chen et al. [7], but outperforms their algorithm on certain inputs. In fact, we show an example were the makespan of their algorithm is 10% more than that of our algorithm. The makespan of our algorithm is never larger than that of theirs.

2. A rejection scheme

Let p_j and q_j be the processing time and rejection penalty for job j.

A generalization of the REJECT TOTAL PENALTY scheme of Bartal et al. [4] is presented in Fig. 1. Our scheme is different from that of Bartal et al. [4] in that they fix $\gamma = 1$. Combining this scheme with a preemptive algorithm presented in the next section we get the first algorithm for PMSR.

For $3 \leqslant m \leqslant 10$, we use the values of α and γ given in Table 1. These values approximate the solution to the system of equations:

$$\begin{aligned} c &= \frac{\beta(m)}{\gamma} = 1 + \alpha + \frac{m-2}{m}\gamma, \\ \alpha &= \frac{\gamma(2-m) + \sqrt{(m-2)^2\gamma^2 - 4\gamma m + 4m^2}}{2m}. \end{aligned} \tag{3}$$

As we shall see in Section 5, this choice of parameters is the best possible one for our algorithm.

For general m, we would like to use the values given by (3), however the general form of the solution is too unwieldy. We therefore choose

$$\gamma = \frac{2}{3}, \quad \alpha = \frac{2 - m + \sqrt{10m^2 - 10m + 4}}{3m}. \tag{4}$$

Initialize $T \leftarrow 0$.

For each job j:
  If $q_j \leq \frac{\gamma}{m} \cdot p_j$ then:
    Reject j.
  Else if $T + q_j \leq \alpha \cdot p_j$ then:
    Reject j.
    $T \leftarrow T + q_j$.
  Else:
    Accept j.

Fig. 1. A slight variation on the REJECT TOTAL PENALTY scheme.

We show that this yields a competitive ratio of

$$\max\left\{\frac{3\beta(m)}{2}, 1+\alpha+\frac{m-2}{m}\gamma\right\} \leqslant \frac{4+\sqrt{10}}{3} < 2.38743,$$

for all $m \geqslant 2$

3. A new preemptive algorithm

We consider a slightly modified version of the PREEMPTIVE algorithm of Chen et al. [7].

First we need a few definitions. Let J be the set of all jobs in the input. For any $I \subset J$ define

$$P_I = \sum_{j\in I} p_j, \quad Q_I = \sum_{j\in I} q_j, \quad M_I = \begin{cases} 0, & \text{if } I = \emptyset, \\ \max_{j\in I} p_j, & \text{otherwise.} \end{cases}$$

At any point in the schedule, let the loads on the machines be $L_1 \leqslant L_2 \leqslant \cdots \leqslant L_m$. Define

$$f(x,y) = \begin{cases} \frac{\beta(m)}{m}x, & \text{if } \frac{1}{m}x \geqslant y, \\ \frac{m-\beta(m)}{m-1}y + \frac{\beta(m)-1}{m-1}x, & \text{otherwise.} \end{cases}$$

Note that f is continuous and non-decreasing in both its parameters.

The algorithm, which we call MODIFIED PREEMPTIVE, schedules each job maintaining the invariants

$$L_m \leqslant f(P_J, M_J), \tag{5}$$

$$\sum_{i=1}^{\ell} L_i \leqslant \frac{\theta^\ell - 1}{\theta^m - 1} P_J \quad \text{for } 1 \leqslant \ell \leqslant m. \tag{6}$$

Our invariants differ from those of [7] in that Chen et al. only require that

$$L_m \leqslant \beta(m) \max\left\{\tfrac{1}{m}P_J, M_J\right\}.$$

Clearly, if the algorithm is well defined, if it can really maintain these invariants, then the following lemma holds:

Lemma 1. MODIFIED PREEMPTIVE *has makespan at most* $f(P_J, M_J)$ *for any sequence with job set* J.

We show that the algorithm is well defined in the appendix. As a corollary, note that this demonstrates that the algorithm is $\beta(m)$-competitive since

$$f(P_J, M_J) \leqslant \beta(m) \max\left\{\tfrac{1}{m}P_J, M_J\right\} \leqslant \beta(m)\,\mathrm{cost}(\sigma),$$

for all $0 \leqslant M_J \leqslant P_J$. We explain how each job is scheduled. Essentially, this is unchanged from [7], but we recapitulate for the reader's convenience. When a new job arrives, it is given a time slot on each machine. On the most loaded machine this time

slot is $(L_m, f(P_J, M_J)]$. On each other machine $i<m$ the slot is $(L_i, L_{i+1}]$. The job is allotted to time slots greedily, starting with the slot on L_m, next using the slot on L_{m-1}, etc. We define k to be the index of the least loaded machine on which some portion of the job is scheduled. The time slots on machines $L_1,\ldots,L_{k-1}$ are unused. Possibly, the slot on machine k is partially used. The slots on $L_{k+1},\ldots,L_m$ are fully used.

Although PREEMPTIVE is optimal in terms of the competitive ratio, MODIFIED PREEMPTIVE sometimes achieves a better makespan. Essentially, PREEMPTIVE maintains L_m as large as possible, given the restriction of $\beta(m)$-competitiveness. MODIFIED PREEMPTIVE does the opposite.

This can be seen in the following example for $m=2$: Consider two jobs of size 3 followed by a job of size 18. After the first two jobs, both algorithms have $L_2=4$ and $L_1=2$. However, PREEMPTIVE puts the entire third job on the more heavily loaded machine as it maintains $L_2 \leqslant \beta(2)\max\{18, 24/2\} = \frac{4}{3}18 = 24$. Therefore its makespan is 22. MODIFIED PREEMPTIVE, on the other hand, effectively puts the entire third job on the least loaded machine since $f(24,18)=20$. Its makespan is 20. PREEMPTIVE is 10% worse than MODIFIED PREEMPTIVE.

We further show that MODIFIED PREEMPTIVE is never outperformed by PREEMPTIVE:

Theorem 2. *There is no input for which the makespan of* MODIFIED PREEMPTIVE *is more than the makespan of* PREEMPTIVE.

Proof. Let σ be any job sequence. We show by induction that the theorem is true after each job is scheduled. Clearly it holds when no jobs have been scheduled. Let J_i be the set consisting of the first i jobs in σ. After i jobs are scheduled, let the loads in MODIFIED PREEMPTIVE's schedule be $A_1^i \leqslant A_2^i \leqslant \cdots \leqslant A_m^i$ while the loads in PREEMPTIVE's are $B_1^i \leqslant B_2^i \leqslant \cdots \leqslant B_m^i$. Consider the scheduling of the ith job. By the inductive hypothesis we have $B_m^{i-1} \geqslant A_m^{i-1}$. If $B_m^i = \beta(m)\max\{(1/m)P_{J_i}, M_{J_i}\}$ then

$$A_m^i \leqslant f(P_{J_i}, M_{J_i}) \leqslant \beta(m)\max\left\{\tfrac{1}{m}P_{J_i}, M_{J_i}\right\} = B_m^i.$$

If $B_m^i < \beta(m)\max\{(1/m)P_{J_i}, M_{J_i}\}$ then the ith job is scheduled entirely on the most loaded machine by PREEMPTIVE. I.e. $B_m^i = B_m^{i-1} + p_i \geqslant A_m^{i-1} + p_i \geqslant A_m^i$. □

4. An upper bound for REJECT TOTAL PENALTY

Let R_{opt} be the set of jobs rejected by the optimal offline algorithm. Let R_0 be the set of jobs with $q_j \leqslant \gamma p_j/m$. Let R_1 be the set of jobs rejected by the algorithm. Note that by the definition of the algorithm we have $R_0 \subset R_1$. We partition J into six subsets as follows:

$$X = J - (R_{\text{opt}} \cup R_1), \quad Y = R_{\text{opt}} - R_1,$$
$$Z = R_{\text{opt}} \cap (R_1 - R_0), \quad U = R_{\text{opt}} \cap R_0,$$
$$V = R_0 - R_{\text{opt}}, \quad W = (R_1 - R_0) - R_{\text{opt}}.$$

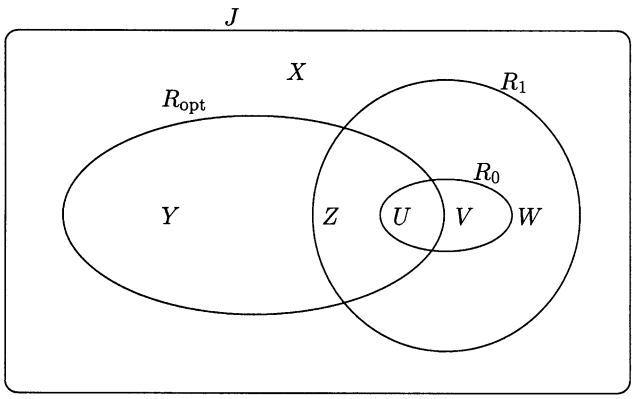

Fig. 2. Relationships among J, U, V, W, X, Y and Z.

The relationships among these sets are illustrated in Fig. 2. Abusing notation, define $M_1 = M_{J-R_1}$ and $M_{\text{opt}} = M_{J-R_{\text{opt}}}$. By Lemma 1, the algorithm's cost is at most

$$f(P_{J-R_1}, M_{J-R_1}) + Q_{R_1} = f(P_X + P_Y, M_1) + Q_W + Q_Z + Q_V + Q_U. \tag{7}$$

Note that we have encapsulated all important information about our scheduling algorithm in the function f. If we were to change scheduling algorithms or models we need merely change f. Thus our analysis is quite general.

Since the makespan of any preemptive schedule is at least the average load, and also at least the size of the largest job, the optimal offline cost is at least

$$\max\left\{\tfrac{1}{m}P_{J-R_{\text{opt}}}, M_{J-R_{\text{opt}}}\right\} + Q_{R_{\text{opt}}} = o + Q_U + Q_Z + Q_Y,$$

where o is subject to

$$o \geqslant \tfrac{1}{m}(P_W + P_X + P_V), \quad o \geqslant M_{\text{opt}}. \tag{8}$$

Without loss of generality, we assume that the optimal cost is 1, i.e.

$$o + Q_U + Q_Z + Q_Y = 1. \tag{9}$$

From the preceding definitions and the definition of the algorithm we get

$$\begin{aligned} &P_W \geqslant M_W, \quad P_Y \geqslant M_Y, \quad P_X \geqslant M_X, \\ &Q_W \geqslant \tfrac{\gamma}{m}P_W, \; Q_Z \geqslant \tfrac{\gamma}{m}P_Z, \; Q_Y \geqslant \tfrac{\gamma}{m}P_Y, \\ &Q_U \leqslant \tfrac{\gamma}{m}P_U, \; Q_V \leqslant \tfrac{\gamma}{m}P_V, \\ &M_{\text{opt}} \geqslant M_X, \; M_{\text{opt}} \geqslant M_W. \end{aligned} \tag{10}$$

We show further constraints on the values of our variables in the following two lemmas:

Lemma 3. *For all job sets J,*

$$Q_W \leqslant \alpha M_W. \tag{11}$$

Proof. Let j be the last job in W to be rejected. At the time it was rejected

$$T + q_j \leqslant \alpha p_j$$

held. Since it was the last job in W to be rejected we have $Q_W \leqslant T + q_j$. Certainly we have $p_j \leqslant M_W$. □

Lemma 4. *For all job sets J,*

$$Q_Z + Q_W + Q_Y \geqslant \alpha M_Y. \tag{12}$$

Proof. There exists a job $j \in Y$ with $p_j = M_Y$. At the time j was accepted

$$T + q_j \geqslant \alpha p_j = \alpha M_Y$$

held. Note that $Q_Z + Q_W \geqslant T$ and that $Q_Y \geqslant q_j$. □

To complete the analysis, we split into 3 cases, based on the relative values of M_1, M_X and M_Y. The first is

$$M_1 \geqslant \tfrac{1}{m}(P_X + P_Y),$$
$$M_1 = M_X. \tag{13}$$

The second is

$$M_1 \geqslant \tfrac{1}{m}(P_X + P_Y),$$
$$M_1 = M_Y. \tag{14}$$

The third and final case is

$$M_1 < \tfrac{1}{m}(P_X + P_Y). \tag{15}$$

The analysis of each case involves bounding the value of a linear program. If we show that all three linear programs have value at most c, then we have shown that the algorithm is c-competitive. In each linear program, we seek to maximize (7) subject to (8)–(12) and the conditions of the case. I.e. either (13), (14) or (15). We call these three linear programs $\mathbf{L}_1$, $\mathbf{L}_2$ and $\mathbf{L}_3$, respectively. For any fixed m, we can bound the values of $\mathbf{L}_1$, $\mathbf{L}_2$ and $\mathbf{L}_3$ by way of a computer program. Using the values of γ and α given in Table 1 we have done exactly that for $3 \leqslant m \leqslant 10$. This method of analysis has proven to be quite useful in that it allowed the author to conjecture the general form of the solution for large m.

For large m, we need a different approach. We fix a value γ and show that it works well for all m. The analysis is completed by showing the following three lemmas:

Lemma 5. *For α and γ as defined by* (4) $\mathbf{L}_1$ *has value at most* $(1/3m)(4m - 2 + \sqrt{10m^2 - 10m + 4})$ *for all* $m \geqslant 2$.

Lemma 6. *For* α *and* γ *as defined by* (4) $\mathbf{L}_2$ *has value at most* $(1/3m)(4m-2+\sqrt{10m^2-10m+4})$ *for all* $m \geqslant 2$.

Lemma 7. *For* α *and* γ *as defined by* (4) $\mathbf{L}_3$ *has value at most* $3\beta(m)/2$ *for all* $m \geqslant 2$.

The proofs are purely algebraic, and are given in the appendix.

5. A lower bound for REJECT TOTAL PENALTY

We show that the preceding analysis of REJECT TOTAL PENALTY is tight, and that $\gamma = 2/3$ is a good choice for large m.

Lemma 8. *If* REJECT TOTAL PENALTY *is c-competitive then*

$$c \geqslant 1 + \max\left\{\alpha, \frac{\gamma}{m}\right\} + \frac{m-2}{m}\gamma.$$

Proof. Consider the sequence consisting of $m-2$ jobs of size 1 with penalty γ/m, followed by a job of size 1 with penalty $\max\{\alpha, \gamma/m\}$ and a job of size 1 and infinite penalty. REJECT TOTAL PENALTY rejects the first $m-1$ jobs and schedules the last. It pays $1 + \max\{\alpha, \gamma/m\} + (m-2)\gamma/m$. The adversary schedules all jobs by placing each on a machine, and so his cost is 1. □

Lemma 9. *For* $\alpha \leqslant \gamma/m$ *if* REJECT TOTAL PENALTY *is c-competitive then*

$$c \geqslant (1 + \sqrt{4m-3})/2.$$

Proof. Obviously, if $\alpha \leqslant \gamma/m$ then m/γ is a lower bound on the competitive ratio. From the previous lemma, so is $1 + (m-1)\gamma/m$. The minimum of the two bounds is achieved by picking γ to satisfy $1 + (m-1)\gamma/m = m/\gamma$, from which the stated bound follows. □

Lemma 10. *For* $\alpha > \gamma/m$ *if* REJECT TOTAL PENALTY *is c-competitive then*

$$c \geqslant (m - \gamma + m\alpha)/m\alpha.$$

Proof. Consider a sequence consisting of two jobs, the first of size 1 with penalty $\gamma/m + \varepsilon$ the second of size x with penalty $x\gamma/m + \varepsilon$, where

$$x = \gamma/(m\alpha - \gamma).$$

Note that the choice of x implies that

$$\tfrac{\gamma}{m} + \tfrac{\gamma}{m}x + 2\varepsilon > \alpha x.$$

Therefore, REJECT TOTAL PENALTY rejects the first job and accepts the second. REJECT TOTAL PENALTY pays $\gamma/m+\varepsilon+x$. The adversary rejects both jobs and pays $\gamma/m+x\gamma/m+2\varepsilon$. We have

$$\frac{\gamma/m+\varepsilon+x}{\gamma/m+x\gamma/m+2\varepsilon}=\frac{m\gamma-\gamma^2+m\alpha\gamma+\varepsilon m(\alpha m-\gamma)}{m\alpha\gamma+2\varepsilon m(\alpha m-\gamma)},$$

which can be made arbitrarily close to the desired bound by choosing sufficiently small ε. □

Lemma 11. *On a sequence of $k\geqslant m$ jobs of size* 1, MODIFIED PREEMPTIVE *achieves a makespan of $f(k,1)$*

The proof is given in the appendix.

Lemma 12. *If* REJECT TOTAL PENALTY *is c-competitive then*

$$c\geqslant\beta(m)/\gamma.$$

Proof. Consider a sequence consisting of n jobs of size 1 with penalty $\gamma/m+\varepsilon$ where $0<\varepsilon<\alpha-\gamma/m$. We define

$$i=\left\lfloor\frac{\alpha}{\gamma/m+\varepsilon}\right\rfloor.$$

We have $k(\gamma/m+\varepsilon)>\alpha$ for all $k>i$. And so the first i jobs are rejected, while the remaining $n-i$ jobs are accepted. We pick n and ε so that $n-i>m$. By Lemma 11 the makespan achieved by MODIFIED PREEMPTIVE is $f(n-i,1)$. The cost to REJECT TOTAL PENALTY is therefore $i(\gamma/m+\varepsilon)+f(n-i,1)$. The adversary rejects all jobs and pays $n(\gamma/m+\varepsilon)$. Note that

$$\begin{aligned}\frac{i(\gamma/m+\varepsilon)+f(n-i,1)}{n(\gamma/m+\varepsilon)}&=\frac{i\gamma+im\varepsilon+mf(n-i,1)}{n\gamma+nm\varepsilon}\\&=\frac{i\gamma+im\varepsilon+(n-i)\beta(m)}{n\gamma+nm\varepsilon}\end{aligned}$$

can be made arbitrarily close to $\beta(m)/\gamma$ by choosing a large enough value for n and small enough value for ε. □

Theorem 13. *For $m\geqslant 3$, if* REJECT TOTAL PENALTY *is c-competitive then*

$$c\geqslant\min_{\gamma}\max\left\{\frac{\beta(m)}{\gamma},\frac{(m-2)\gamma+2m+\sqrt{(m-2)^2\gamma^2-4\gamma m+4m^2}}{2m}\right\}.$$

Proof. First we consider $\alpha>\gamma/m$. From Lemma 12 we have $c\geqslant\beta(m)/\gamma$. From Lemmas 8 and 10 we have $c\geqslant\max\{1+\alpha+(m-2)\gamma/m,\ (m-\gamma+m\alpha)/(m\alpha)\}$. The first lower bound increases with α, while the second decreases for $\gamma<m$. Clearly, the

algorithm is m-competitive for $\gamma \geqslant m$. For $m \geqslant 3$ the minimum of this expression over $\alpha > 0$ is achieved at

$$\alpha = \frac{\gamma(2-m) + \sqrt{(m-2)^2\gamma^2 - 4\gamma m + 4m^2}}{2m}.$$

Substituting this value for α gives the bound in the hypothesis of the theorem.

Now consider $\alpha \leqslant \gamma/m$. One can easily verify that the bound in Lemma 9 is greater than the bound in the hypothesis for $m \geqslant 3$. □

Note that this implies that the competitive ratio of REJECT TOTAL PENALTY for arbitrarily large m is at least

$$\min_{\gamma} \max \left\{ 1 + \frac{\gamma + \sqrt{4+\gamma^2}}{2}, \frac{\mathrm{e}}{\gamma(\mathrm{e}-1)} \right\} > 2.38518.$$

The correct choice of γ can be found by setting the two expressions in the above maximum equal, and solving the resulting equation. By picking $\gamma = 2/3$ we achieve a competitive ratio which exceeds the best possible by less than 3×10^{-3} for large m.

6. General lower bounds

In this section, we show lower bounds on the PMSR problem in general. We prove a lower bound of 2.12457 for the competitive ratio of any online algorithm.

We are able to a show a stronger lower bound for a broad class of PMSR algorithms. We say that an algorithm *schedules obliviously* if the accepted jobs are scheduled without knowledge of the rejection part of the problem. I.e. for each job, the algorithm first decides whether it is accepted or rejected. If the job is accepted, it is given to an online scheduling algorithm, which learns only the sizes of the jobs given.

Any algorithm which is to beat this lower bound must somehow integrate the rejection and scheduling processes. The existence of such an algorithm would be somewhat surprising since no such interaction is required in the MSR problem; in that case the optimal online algorithm schedules obliviously [4].

It mostly likely is still not intuitively clear to the reader why choosing $\gamma < 1$ helps our algorithm. As our final result, we show that $\gamma < 1$ is in fact necessary for the improved performance of REJECT TOTAL PENALTY.

The following two lemmas from the literature will prove useful:

Lemma 14 (McNaughton [13]). *The optimal offline preemptive makespan for any set of jobs J is exactly* $\max\{(1/m)P_J, M_j\}$.

Lemma 15 (Chen et al. [7]). *For any preemptive online scheduling algorithm A, there exists a sequence σ such that* $\mathrm{cost}_A(\sigma) \geqslant \beta(m)\mathrm{cost}(\sigma)$.

We also need the following lemma, which is adapted from [4]:

Lemma 16. *On a sequence beginning with k jobs of penalties $x/c, x/c^2, \ldots, x/c^k$ all of size x, any online algorithm which accepts one of the first k jobs is no better than c-competitive.*

Proof. Suppose the algorithm accepts one of the first k jobs. Let i be the index of the first such job. The adversary rejects the first i jobs and the competitive ratio is at least

$$\left(x + \sum_{j=1}^{i-1} \frac{x}{c^j} \right) \Big/ \sum_{j=1}^{i} \frac{x}{c^j} = c. \qquad \square$$

In all of the proofs that follow, m is a power of 2 and $\lg m = \log_2 m$. The main result of this section is:

Theorem 17. *If an algorithm for PMSR is c-competitive for all m then $c \geqslant \zeta > 2.12457$ where ζ is the solution to*

$$\zeta = \frac{(2\zeta + 1)(\zeta - 1)}{2(\zeta^2 - \zeta - 1)}.$$

Proof. Define

$$x = \frac{2\zeta(\zeta - 2)}{\zeta - 1} \approx 0.47068.$$

Consider the following input sequence:

(1) $\lg m$ jobs with penalties $1/\zeta, 1/\zeta^2, \ldots, 1/\zeta^{\lg m}$,
(2) $k = m - \lg m - 1$ jobs with penalty x/k, and
(3) k jobs with infinite penalty.

All jobs have size 1. If the algorithm accepts one of the first $\lg m$ jobs then its competitive ratio is at least ζ by Lemma 16. So suppose the first $\lg m$ jobs are all rejected. Suppose that the algorithm rejects yk of the jobs with penalty x/k. If $y \geqslant \frac{1}{2}$ then only one of the infinite penalty jobs is given. The algorithm's cost is at least

$$\begin{aligned} 1 + \frac{x}{2} + \sum_{j=1}^{\lg m} \frac{1}{\zeta^j} &= 1 + \frac{x}{2} + \frac{1}{\zeta - 1} - \frac{1}{(\zeta - 1)\zeta^{\lg m}} \\ &\geqslant 1 + \frac{x}{2} + \frac{1}{\zeta - 1} - \frac{1}{m} \\ &= \zeta - \frac{1}{m}. \end{aligned}$$

The adversary schedules all jobs and pays 1 and therefore the competitive ratio is $\zeta - o(1)$.

If $y < \frac{1}{2}$ then the algorithm is given the infinite penalty jobs, one at a time. These jobs cannot be rejected. If all jobs are given, a total of $(1 - y)k + k = (2 - y)k$ jobs are accepted.

Let u_i be the completion time in the algorithm's schedule for the ith infinite penalty job. Let u_0 be the maximum completion time over the set of accepted x/k penalty jobs. Define $U_i = \max_{0 \leqslant j \leqslant i} u_j$. Note that $U_1 \leqslant U_2 \leqslant \cdots \leqslant U_k$. We have

$$\sum_{i=1}^{k} U_i \geqslant (2-y)m - \mathrm{O}(\lg m),$$

which can be seen as follows: For $1 \leqslant i \leqslant k$, after time U_{k-i}, at most i jobs remain and so at most i machines are busy at any point in $(U_{k-i}, U_{k-i+1}]$. Let T be the total amount of processing that is completed by the algorithm. The desired inequality now follows:

$$\begin{aligned}(2-y)k &\leqslant T \\ &\leqslant \sum_{i=1}^{k-1} i(U_{k-i+1} - U_{k-i}) + mU_1 \\ &= \sum_{i=1}^{k} U_i + (m-k)U_1 \\ &= \sum_{i=1}^{k} U_i + \mathrm{O}(\lg m).\end{aligned}$$

We consider the competitive ratio achieved by the algorithm after each of the infinite penalty jobs is scheduled. The adversary's solution when i of the infinite penalty jobs are given is to reject $i-1$ of the jobs with penalty x/k, and schedule all others. Since the total number of jobs is $m+i-1$, the adversary schedules m jobs and has makespan 1. Therefore, for all $1 \leqslant i \leqslant m$ we must have

$$U_i + xy + \sum_{j=1}^{\lg m} \frac{1}{\zeta^j} \leqslant c\left(1 + \frac{x(i-1)}{k}\right).$$

Summing both sides we get

$$\sum_{i=1}^{k}\left(U_i + xy + \sum_{j=1}^{\lg m} \frac{1}{\zeta^j}\right) \leqslant \sum_{i=1}^{k} c\left(1 + \frac{x(i-1)}{k}\right) \leqslant c\left(k + \frac{xk}{2}\right).$$

We also have

$$\begin{aligned}\sum_{i=1}^{k}\left(U_i + xy + \sum_{j=1}^{\lg m} \frac{1}{\zeta^j}\right) &= \sum_{i=1}^{k} U_i + kxy + k\sum_{j=1}^{\lg m} \frac{1}{\zeta^j} \\ &\geqslant \sum_{i=1}^{k} U_i + kxy + \frac{k}{\zeta - 1} - \frac{k}{m} \\ &= k(2-y) + kxy + \frac{k}{\zeta - 1} - \mathrm{O}(\lg m) \\ &= 2k - ky(1-x) + \frac{k}{\zeta - 1} - \mathrm{O}(\lg m) \\ &\geqslant \frac{k(3+x)}{2} + \frac{k}{\zeta - 1} - \mathrm{O}(\lg m).\end{aligned}$$

Combining these facts, we find that

$$\begin{aligned} c &\geqslant \frac{\frac{3}{2}+\frac{x}{2}+\frac{1}{\zeta-1}-\mathrm{O}(\lg m)/k}{1+\frac{x}{2}} \\ &\geqslant \frac{\frac{3}{2}+\frac{x}{2}+\frac{1}{\zeta-1}}{1+\frac{x}{2}}-\mathrm{O}\left(\frac{\lg m}{m}\right) \\ &= \frac{(2\zeta+1)(\zeta-1)}{2(\zeta^2-\zeta-1)}-\mathrm{O}\left(\frac{\lg m}{m}\right) \\ &= \zeta-\mathrm{O}\left(\frac{\lg m}{m}\right). \qquad \square \end{aligned}$$

We now consider algorithms which schedule obliviously:

Theorem 18. *If an algorithm which schedules obliviously is c-competitive for all m then*

$$c \geqslant \chi = \frac{2\mathrm{e}-1+\sqrt{4\mathrm{e}^2-8\mathrm{e}+5}}{2(\mathrm{e}-1)} > 2.33246.$$

Proof. Note that $\chi = 1/(\chi-1)+\mathrm{e}/(\mathrm{e}-1)$. Suppose the algorithm uses online preemptive scheduling algorithm A to schedule jobs. By Lemma 15, the exists a sequence σ such that makespan of A on σ is at least $\beta(m)\mathrm{cost}(\sigma)$. Further, we assume without loss of generality that $\mathrm{cost}(\sigma)=1$ since any sequence can be re-scaled. Consider the following input sequence: $\lg m$ jobs with penalties $y/\chi, y/\chi^2, \ldots, y/\chi^{\lg m}$ all of size $y = m/(m-\lg m)$ followed the jobs in σ which are all given infinite penalty. If the algorithm accepts one of the first $\lg m$ jobs then its competitive ratio is at least χ by Lemma 16. Suppose the first $\lg m$ jobs are all rejected. The algorithm must schedule the remaining jobs. By definition of σ, the makespan of the algorithm on these jobs is at least $\beta(m)$. The total cost to the algorithm is

$$\beta(m)+\sum_{j=1}^{\lg m}\frac{1}{\chi^j} = \beta(m)+\frac{1}{\chi-1}-\frac{1}{(\chi-1)\chi^{\lg m}} \geqslant \beta(m)+\frac{1}{\chi-1}-\frac{1}{m}.$$

Using the fact that $\theta^{m-1} \leqslant \mathrm{e}$ we find that

$$\begin{aligned} \beta(m) &= 1+\frac{1}{\theta^m-1} \geqslant 1+\frac{1}{\mathrm{e}\theta-1} = \frac{\mathrm{e}}{\mathrm{e}-1}\left(1-\frac{1}{(\mathrm{e}-1)m+1}\right) \\ &\geqslant \frac{\mathrm{e}}{\mathrm{e}-1}-\frac{1}{m}. \end{aligned}$$

Therefore, the algorithm's cost is at least

$$\frac{\mathrm{e}}{\mathrm{e}-1}+\frac{1}{\chi-1}-\frac{2}{m} = \chi-\frac{2}{m}.$$

On the other hand, the adversary schedules all jobs and by Theorem 18 pays at most y, since the largest job is of size y and the total load is at most $m + y \lg m = my$. The competitive ratio is at least

$$\frac{m - \lg m}{m}\left(\chi - \frac{2}{m}\right) \geqslant \chi - \frac{\chi \lg m + 2}{m} = \chi - \mathrm{O}\left(\frac{\lg m}{m}\right). \qquad \square$$

We say that an algorithm A has *absolute threshold* x if A rejects all jobs with $q_j \leqslant (x/m)p_j$. REJECT TOTAL PENALTY has absolute threshold γ. The following theorem shows that any algorithm with absolute threshold $x \geqslant 1$ has competitive ratio at least $1 + \phi$. The theorem is applicable to almost any imaginable model of scheduling with rejection. The only fact used about the scheduling problem is that the makespan is at least the largest processing time.

Theorem 19. *If algorithm A has absolute threshold x and is c-competitive for all m then*

$$c \geqslant \psi = 1 + \frac{x + \sqrt{4 + x^2}}{2}.$$

Proof. Note that $\psi = 1 + x + 1/(\psi - 1)$. Consider the following input sequence: $\lg m$ jobs with penalties $1/\psi$, $1/\psi^2, \ldots, 1/\psi^{\lg m}$ followed 1 job with infinite penalty and $k = m - \lg m - 1$ jobs with penalty x/m. All jobs have size 1. If the algorithm accepts one of the first $\lg m$ jobs then its competitive ratio is at least ψ by Lemma 16. So suppose the first $\lg m$ jobs are all rejected. The algorithm must schedule the job with infinite penalty, and reject the remaining jobs since it has absolute threshold x. The adversary schedules all jobs and pays 1. The competitive ratio is

$$\begin{aligned} 1 + \frac{k}{m}x + \sum_{j=1}^{\lg m} \frac{1}{\psi^j} &= 1 + x + \frac{1}{\psi - 1} - \frac{\lg m + 1}{m}x - \frac{1}{(\psi - 1)\psi^{\lg m}} \\ &\geqslant 1 + x + \frac{1}{\psi - 1} - \frac{\lg m + 1}{m}x - \frac{1}{m} \\ &= \psi - \mathrm{O}\left(\frac{\lg m}{m}\right). \qquad \square \end{aligned}$$

7. Conclusions

We have shown how rejection can be incorporated into preemptive multiprocessor scheduling. Our results imply that an online algorithm can improve its performance by using preemption. Our algorithm is the natural generalization of the algorithm of Bartal et al. [4] for non-preemptive scheduling, which has been shown to optimal.

We have also shown a general lower bound of 2.12457 and a lower bound of 2.33246 for any algorithm which schedules obliviously. There are two possibilities, both of which would be of interest:

(1) The optimal algorithm schedules obliviously. In that case our algorithm is reasonably close to optimal.
(2) The optimal algorithm integrates scheduling and rejection. This would be in sharp contrast with the situation for MSR.

We have also presented a new optimal online algorithm for preemptive scheduling. This algorithm outperforms the previously known optimal algorithm on certain inputs where the worst-case competitive ratio is not achieved.

When rejection is not allowed, scheduling with preemption is much easier and more well understood than scheduling without preemption. Our surprising conclusion is that the opposite situation seem to hold when rejection is introduced.

The obvious problem which remains is to improve either the lower or upper bounds given here. It would seem that the lower bound of 2.12457 for PMSR is weak, but we have been unable to improve it.

Another open problem is to consider online scheduling with rejection in other models, perhaps online versions of some of the problems studied in [8].

Appendix A

A.1. Proofs of Lemmas 1 and 11

We now prove that (5) and (6) are maintained when each job is scheduled. The proof is by induction.

Recall that $\theta = m/(m-1)$ and $\beta(m) = \theta^m/(\theta^m - 1)$.

Certainly (5) and (6) hold before any jobs arrive, and so the basis holds.

For the inductive step, we consider the arrival of a new job j of size s. We assume without loss of generality that the sum of the sizes of previous jobs is 1. Let x be the size of the largest of the previous jobs. Define $x' = \max\{x, s\}$.

Essentially, the following lemma is a modification of Lemma 2 of [7].

Lemma A.1. *Job j can be accommodated by the allocated time slots.*

Proof. Note that the union of the allocated time slots is $(L_1, f(P_J, M_J)]$. So we need to show that $L_1 + s \leqslant f(P_J, M_J) = f(1+s, x')$. We have two cases. In the first $(1+s)/m \geqslant x'$ and we must show

$$L_1 + s \leqslant \frac{\beta(m)}{m}(1+s),$$

which is true if and only if $\beta(m) - mL_1 - (m - \beta(m))s \geqslant 0$. By the inductive hypothesis we have $L_1 \leqslant (\theta - 1)/(\theta^m - 1) = (\beta(m) - 1)/(m-1)$. Further note that $(1+s)/m \geqslant x' \geqslant s$ implies $s \leqslant 1/(m-1)$. We therefore have

$$\beta(m) - \frac{m(\beta(m)-1)}{m-1} - \frac{(m-\beta(m))}{m-1} = 0.$$

In the second case we show that

$$L_1 + s \leqslant \frac{m-\beta(m)}{m-1}x' + \frac{\beta(m)-1}{m-1}(1+s),$$

which is equivalent to

$$0 \leqslant \frac{\beta(m)-m}{m-1}(x'-s) + \frac{\beta(m)-1}{m-1} - L_1.$$

By the inductive hypothesis we have $L_1 \leqslant (\theta-1)/(\theta^m-1) = (\beta(m)-1)/(m-1)$ and so it is sufficient to show that

$$0 \leqslant \frac{m-\beta(m)}{m-1}(x'-s).$$

Noting that $m \geqslant 2 \geqslant \beta(m)$ and $x' \geqslant s$ we are done. □

The next lemma is an adaptation of Lemma 3 of [7]:

Lemma A.2. *Invariants* (5) *and* (6) *hold after job j is scheduled.*

Proof. Let L_i be the load of machine i before the job, and L'_i be the load after. First note that $L'_m \leqslant f(1+s,x')$. Second, note that the load of a machine $i<k$ does not increase. Therefore

$$\sum_{i=1}^{\ell} L'_i = \sum_{i=1}^{\ell} L_i \leqslant \frac{\theta^\ell - 1}{\theta^m - 1} \leqslant \frac{\theta^\ell - 1}{\theta^m - 1}(1+s),$$

for $\ell < k$. We now show that

$$\sum_{i=\ell+1}^{m} L'_i \geqslant \frac{\theta^m - \theta^\ell}{\theta^m - 1}(1+s),$$

for $k \leqslant \ell \leqslant m$, which is equivalent to what we want to show. When $\ell = m$ this obviously holds. For $\ell < m$ we note that

$$\begin{aligned}\sum_{i=\ell+1}^{m} L'_i &= \sum_{i=\ell+1}^{m-1} L_{i+1} + f(1+s,x') = \sum_{i=\ell+2}^{m} L_i + f(1+s,x') \\ &\geqslant \frac{\theta^m - \theta^{\ell+1}}{\theta^m - 1} + f(1+s,x'),\end{aligned}$$

where the last step is by the inductive hypothesis. To finish, we show

$$\frac{\theta^m - \theta^{\ell+1}}{\theta^m - 1} + f(1+s,x') - \frac{\theta^m - \theta^\ell}{\theta^m - 1}(1+s) \geqslant 0.$$

First consider the case where $(1+s)/m \geqslant x'$. We have

$$\begin{aligned}
&\frac{\theta^m-\theta^{\ell+1}}{\theta^m-1}+f(1+s,x')-\frac{\theta^m-\theta^{\ell}}{\theta^m-1}(1+s)\\
&\quad=\frac{\theta^m-\theta^{\ell+1}}{\theta^m-1}+\frac{\theta^m}{m(\theta^m-1)}(1+s)-\frac{\theta^m-\theta^{\ell}}{\theta^m-1}(1+s)\\
&\quad=\frac{(1+s-ms)\theta^m+(m+ms-\theta m)\theta^{\ell}}{\theta^m-1}\\
&\quad=\frac{(1+s-ms)\theta^m+(ms-\theta)\theta^{\ell}}{\theta^m-1}.
\end{aligned}$$

Note that $(1+s)/m \geqslant x' \geqslant s$ implies $s \leqslant 1/(m-1)$ and therefore $ms-\theta \leqslant 0$. Therefore it is sufficient to show that

$$\frac{(1+s-ms)\theta^m+(ms-\theta)\theta^{m-1}}{\theta^m-1} \geqslant 0.$$

Replacing θ with $m/(m-1)$ in the left-hand side above yields zero exactly. Now consider the case where $(1+s)/m < x'$. We have

$$\begin{aligned}
&\frac{\theta^m-\theta^{\ell+1}}{\theta^m-1}+f(1+s,x')-\frac{\theta^m-\theta^{\ell}}{\theta^m-1}(1+s)\\
&\quad\geqslant\frac{\theta^m-\theta^{\ell+1}}{\theta^m-1}+f(1+s,(1+s)/m)-\frac{\theta^m-\theta^{\ell}}{\theta^m-1}(1+s)\\
&\quad=\frac{\theta^m-\theta^{\ell+1}}{\theta^m-1}+\frac{\theta^m}{m(\theta^m-1)}(1+s)-\frac{\theta^m-\theta^{\ell}}{\theta^m-1}(1+s),
\end{aligned}$$

which is the same as in the first case. The second step above follows from the fact that $f(1+s,x')$ is increasing in x'. □

The preceding lemma completes the induction.

To prove Lemma 11, note that $L_m \leqslant f(P_J,M_J)=f(k,1)=k\beta(m)/m$ since $k \geqslant m$. Further we have

$$\sum_{i=1}^{m-1} L_i \leqslant \frac{\theta^{m-1}-1}{\theta^m-1}P_J=\frac{\theta^{m-1}-1}{\theta^m-1}k.$$

Since

$$\frac{\beta(m)}{m}k+\frac{\theta^{m-1}-1}{\theta^m-1}k=\left(\frac{\theta^m}{m(\theta^m-1)}+\frac{\theta^{m-1}-1}{\theta^m-1}\right)k=k,$$

and $\sum_{i=1}^{m} L_i = k$ we must have $L_m = f(k,1)$.

A.2. Proofs of Lemmas 5, 6 and 7

First we need the following bounds:

Lemma A.3. *For $\gamma = \frac{2}{3}$ with α as defined by (3) we have*

$$\alpha \geqslant \frac{\gamma}{m}, \tag{A.1}$$

$$\alpha \leqslant 1, \tag{A.2}$$

$$0 \geqslant m + \gamma - \alpha m - \gamma m, \tag{A.3}$$

$$0 = \gamma - 2\alpha\gamma - m + \alpha^2 m + \alpha\gamma m, \tag{A.4}$$

$$0 \geqslant \gamma - m + \beta(m)m - \gamma m, \tag{A.5}$$

$$0 \geqslant \gamma\alpha m + \beta(m)m^2 + 2\gamma^2 m - \gamma\alpha m^2 - \gamma^2 m^2 - m^2 - \gamma^2, \tag{A.6}$$

$$0 \geqslant \gamma\alpha m + \gamma\beta(m)m - \beta(m)m - \gamma\beta(m), \tag{A.7}$$

for all $m \geqslant 2$.

Proof. Bounds (A.1)–(A.4) can all be shown algebraically. Relation (A.5) is verified directly for $m = 2, \ldots, 7$. Using the fact that $\beta(m) \leqslant \mathrm{e}/(\mathrm{e}-1)$ we verify (A.5) for $m \geqslant 7$. Relation (A.6) is verified directly for $m = 2, 3$. For $m \geqslant 4$ we again use $\beta(m) \leqslant \mathrm{e}/(\mathrm{e}-1)$ to show (A.6) holds algebraically. Relation (A.7) is verified directly for $m = 2, \ldots, 5$. For $m \geqslant 6$ we have $\beta(m) \geqslant 3/2$. Using this fact we verify (A.7) algebraically. □

Proof of Lemma 5. Define $x = Q_U + Q_Z$. We bound the objective function above thus:

$$\begin{aligned} & f(P_X + P_Y, M_X) + Q_W + Q_Z + Q_V + Q_U \\ & \quad \leqslant f(P_X + mQ_Y/\gamma, M_X) + Q_W + x + \frac{\gamma}{m} P_V \\ & \quad = \frac{\beta(m)-1}{m-1} P_X + \frac{m(\beta(m)-1)}{\gamma(m-1)} Q_Y + \frac{m-\beta(m)}{m-1} M_X + Q_W + x + \frac{\gamma}{m} P_V. \end{aligned}$$

Now note that $o = 1 - x - Q_Y \geqslant (1/m)(P_X + P_W + P_V) \geqslant (1/m)(P_X + Q_W/\alpha + P_V)$ and thus $P_V \leqslant m - mx - mQ_Y - P_X - Q_W/\alpha$. Thus the objective function is at most

$$\begin{aligned} & \frac{m-\beta(m)}{m-1} M_X + \frac{\alpha m - \gamma}{\alpha m} Q_W + \frac{\gamma^2 - m + \beta(m)m - \gamma^2 m}{\gamma(m-1)} Q_Y \\ & \quad + \frac{\gamma - m + \beta(m)m - \gamma m}{(m-1)m} P_X + (1-\gamma)x + \gamma. \end{aligned}$$

Note that by (A.1) we have $\alpha m-\gamma\geqslant 0$ and that $o=1-x-Q_Y\geqslant M_W\geqslant Q_W/\alpha$ implies $Q_W\leqslant\alpha-\alpha x-\alpha Q_Y$. Therefore, we get

$$\frac{m-\beta(m)}{m-1}M_X+\frac{\alpha\gamma m+2\gamma^2 m-m^2+\beta(m)m^2-\alpha\gamma m^2-\gamma^2 m^2-\gamma^2}{\gamma(m-1)m}Q_Y$$
$$+\frac{\gamma-m+\beta(m)m-\gamma m}{(m-1)m}P_X+\frac{\gamma+m-\alpha m-\gamma m}{m}x+\alpha+\frac{\gamma(m-1)}{m}.$$

By (A.5) we have $\gamma-m+\beta(m)m-\gamma m\leqslant 0$. Also $P_X\geqslant M_X$ and we therefore get the bound

$$\frac{m-\gamma}{m}M_X+\frac{\alpha\gamma m+2\gamma^2 m-m^2+\beta(m)m^2-\alpha\gamma m^2-\gamma^2 m^2-\gamma^2}{\gamma(m-1)m}Q_Y$$
$$+\frac{\gamma+m-\alpha m-\gamma m}{m}x+\alpha+\frac{\gamma(m-1)}{m}.$$

By Lemma A.3, the coefficient of M_X is positive, while those of Q_Y and x are negative. We note that $o=1-x-Q_Y\geqslant M_X$. The objective function is maximized when $M_X=1$, $x=0$ and $Q_Y=0$. The maximum value achieved is

$$1+\alpha+\frac{\gamma(m-2)}{m}=\frac{4m-2+\sqrt{10m^2-10m+4}}{3m}.\qquad\square$$

Proof of Lemma 6. We bound the objective function above thus:

$$\begin{aligned}&f(P_X+P_Y,M_Y)+Q_W+Q_Z+Q_V+Q_U\\&\quad\leqslant f(P_X+mQ_Y/\gamma,M_Y)+Q_Z+Q_U+\alpha M_W+\frac{\gamma}{m}P_V\\&\quad=\frac{\beta(m)-1}{m-1}P_X+\frac{m(\beta(m)-1)}{\gamma(m-1)}Q_Y+\frac{m-\beta(m)}{m-1}M_Y+Q_Z+Q_U\\&\qquad+\alpha M_W+\frac{\gamma}{m}P_V.\end{aligned}$$

Now note that $o=1-Q_U-Q_Z-Q_Y\geqslant(1/m)(P_X+P_W+P_V)\geqslant(1/m)(P_X+M_W+P_V)$ and thus $P_V\leqslant m-mQ_U-mQ_Z-mQ_Y-P_X-M_W$. Thus the objective function is at most

$$\frac{\alpha m-\gamma}{m}M_W+\frac{m-\beta(m)}{m-1}M_Y+\frac{\gamma^2-m+\beta(m)m-\gamma^2 m}{\gamma(m-1)}Q_Y$$
$$+(1-\gamma)Q_U+(1-\gamma)Q_Z+\frac{\gamma-m+\beta(m)m-\gamma m}{(m-1)m}P_X+\gamma.$$

Note that $1-\gamma\geqslant 0$ and that $o=1-Q_Z-Q_Y-Q_U\geqslant M_W$ implies $Q_U\leqslant 1-Q_Z-Q_Y-M_W$. So we bound the objective function by

$$\frac{\alpha m+\gamma m-\gamma-m}{m}M_W+\frac{m-\beta(m)}{m-1}M_Y+\frac{\gamma-m+\beta(m)m-\gamma m}{\gamma(m-1)}Q_Y$$
$$+\frac{\gamma-m+\beta(m)m-\gamma m}{(m-1)m}P_X+1.$$

By (A.5) we have $\gamma - m + \beta(m)m - \gamma m \leqslant 0$. We also have $P_X \geqslant 0$ and $Q_Y \geqslant \gamma P_Y/m \geqslant \gamma M_Y/m$. Therefore it suffices to bound

$$\frac{m-\gamma}{m}M_Y + \frac{\alpha m + \gamma m - \gamma - m}{m}M_W + 1.$$

Now note that (A.3) implies $\alpha m + \gamma m - \gamma - m \geqslant 0$. We have $1 - Q_Z - Q_Y \geqslant 1 - Q_Z - Q_Y - Q_U = o \geqslant M_W$ which implies that $1 - Q_Z - M_W \geqslant Q_Y$. From (12) we have $Q_Z + Q_W + Q_Y \geqslant \alpha M_Y$ which implies $Q_Y \geqslant \alpha M_Y - Q_Z - Q_W$. We therefore have $1 - Q_Z - M_W \geqslant \alpha M_Y - Q_Z - Q_W$ or equivalently $1 - \alpha M_Y \geqslant M_W - Q_W$. Further, we have $Q_W \leqslant \alpha M_W$ and therefore $M_W \leqslant (1 - \alpha M_Y)/(1 - \alpha)$. The objective function is at most

$$\frac{\gamma - 2\alpha\gamma - m + \alpha^2 m + \alpha\gamma m}{(\alpha - 1)m}M_Y + \frac{\gamma(m-1)}{(1-\alpha)m}.$$

Finally, note by (A.4) the objective function is at most

$$\frac{\gamma(m-1)}{(1-\alpha)m} = \frac{4m - 2 + \sqrt{10m^2 - 10m + 4}}{3m}. \qquad \square$$

Proof of Lemma 7. Again define $x = Q_U + Q_Z$. We bound the objective function above thus:

$$\begin{aligned} & f(P_X + P_Y, M_1) + Q_W + Q_Z + Q_V + Q_U \\ & \quad \leqslant f(P_X + P_Y, M_1) + x + Q_W + \frac{\gamma}{m}P_V \\ & \quad = \frac{\beta(m)}{m}P_X + \frac{\beta(m)}{m}P_Y + x + Q_W + \frac{\gamma}{m}P_V. \end{aligned}$$

Now note that $1 - x - \gamma P_Y/m \geqslant 1 - x - Q_Y = o \geqslant (1/m)(P_X + P_W + P_V) \geqslant (1/m)(P_X + mQ_W/\gamma + P_V)$ and thus $P_V \leqslant m - mx - \gamma P_Y - P_X - mQ_W/\gamma$. Thus the objective function is at most

$$\gamma + \frac{\alpha m - \gamma}{\alpha m}Q_W + \frac{\beta(m) - \gamma}{m}P_X + \frac{\beta(m) - \gamma^2}{m}P_Y + (1 - \gamma)x.$$

Note that $\beta(m) - \gamma = \beta(m) - 2/3 \geqslant 0$. We have $0 \leqslant P_V \leqslant m - mx - \gamma P_Y - P_X - mQ_W/\gamma$ and so $P_X \leqslant m - mx - \gamma P_Y - mQ_W/\gamma$. This allows us to bound the objective function by

$$\frac{\beta(m)(1-\gamma)}{m}P_Y + \frac{\alpha m - \beta(m)}{\alpha m}Q_W + (1 - \beta(m))x + \beta(m).$$

Now note that $\beta(m)(1-\gamma) = \beta(m)/3$ and $1 - x - \gamma P_Y/m \geqslant 1 - x - Q_Y = o \geqslant M_W \geqslant Q_W/\alpha$ implies that $P_Y \leqslant m/\gamma - mx/\gamma - mQ_W/(\gamma\alpha)$. The objective function is at most

$$\frac{\gamma\alpha m + \gamma\beta(m)m - \beta(m)m - \gamma\beta(m)}{\gamma\alpha m}Q_W + \frac{\gamma - \beta(m)}{\gamma}x + \frac{\beta(m)}{\gamma}.$$

Note that $\gamma - \beta(m) = 2/3 - \beta(m) \leqslant 0$ and that by (A.7) the coefficient of Q_W is not positive. Therefore the objective function is at most

$$\beta(m)/\gamma = 3\beta(m)/2. \qquad \square$$

References

[1] S. Albers, Better bounds for online scheduling, SIAM J. Comput. 29 (2) (1999) 459–473.
[2] Y. Bartal, A. Fiat, H. Karloff, R. Vohra, New algorithms for an ancient scheduling problem, J. Comput. System Sci. 51 (3) (1995) 359–366.
[3] Y. Bartal, H. Karloff, Y. Rabani, A better lower bound for on-line scheduling, Inform. Process. Lett. 50 (3) (1994) 113–116.
[4] Y. Bartal, S. Leonardi, A. Marchetti-Spaccamela, J. Sgall, L. Stougie, Multiprocessor scheduling with rejection, SIAM J. Discrete Math. 13 (1) (2000) 64–78.
[5] B. Chen, A. van Vliet, G. Woeginger, A lower bound for randomized on-line scheduling algorithms, Inform. Process. Lett. 51 (5) (1994) 219–222.
[6] B. Chen, A. van Vliet, G. Woeginger, New lower and upper bounds for on-line scheduling, Oper. Res. Lett. 16 (4) (1994) 221–230.
[7] B. Chen, A. van Vliet, G. Woeginger, An optimal algorithm for preemptive on-line scheduling, Oper. Res. Lett. 18 (3) (1995) 127–131.
[8] D. Engels, D. Karger, S. Kolliopoulos, S. Sengupta, R. Uma, J. Wein, Techniques for scheduling with rejection, Proc. 6th European Symp. on Algorithms, August 1998, pp. 490–501.
[9] U. Faigle, W. Kern, G. Turàn, On the performance of on-line algorithms for partition problems, Acta Cybernet. 9 (2) (1989) 107–119.
[10] G. Galambos, G. Woeginger, An online scheduling heuristic with better worst case ratio than Graham's list scheduling, SIAM J. Comput. 22 (2) (1993) 349–355.
[11] D. Karger, S. Phillips, E. Torng, A better algorithm for an ancient scheduling problem, J. Algorithms 20 (2) (1996) 400–430.
[12] A. Karlin, M. Manasse, L. Rudolph, D. Sleator, Competitive snoopy caching, Algorithmica 3 (1) (1988) 79–119.
[13] R. McNaughton, Scheduling with deadlines and loss functions, Management Sci. 6 (1959) 1–12.
[14] S.S. Seiden, Randomized algorithms for that ancient scheduling problem, Proc. 5th Workshop on Algorithms and Data Structures, August 1997, pp. 210–223.
[15] J. Sgall, A lower bound for randomized on-line multiprocessor scheduling, Inform. Process. Lett. 63 (1) (1997) 51–55.
[16] D. Sleator, R. Tarjan, Amortized efficiency of list update and paging rules, Comm. ACM 28 (2) (1985) 202–208.