Two semi-online scheduling problems on two uniform machines

C. T. Ng [a, *] Zhiyi Tan[b, †] Yong He[b] T. C. E. Cheng[a, ‡]

[a] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong SAR, P. R. China
[b] Department of Mathematics, State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, P. R. China

Abstract

This paper considers two semi-online scheduling problems, one with known optimal value and the other with known total sum, on two uniform machines with a machine speed ratio of $s \geq 1$. For the first problem, we provide an optimal algorithm for $s \in [\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{21}}{4}]$, and improved algorithms or/and lower bounds for $s \in [\frac{1+\sqrt{21}}{4}, \sqrt{3}]$, over which the optimal algorithm is unknown. As a result, the largest gap between the competitive ratio and the lower bound decreases to 0.02192. For the second problem, we also present algorithms and lower bounds for $s \geq 1$. The largest gap between the competitive ratio and the lower bound is 0.01762, and the length of the interval over which the optimal algorithm is unknown is 0.47382. Our algorithms and lower bounds for these two problems provide insights into their differences, which are unusual from the viewpoint of the known results on these two semi-online scheduling problems in the literature.

Mathematics Subject Classification (1991). 90B35, 90C27
Key words: analysis of algorithm, scheduling, semi-online, competitive ratio.

1 Introduction

In this paper we consider semi-online scheduling problems. We are given a sequence $\mathcal{J}$ of independent jobs with positive sizes $p_1, p_2, \ldots, p_n$, which must be non-preemptively scheduled onto m uniform machines $M_1, M_2, \cdots, M_m$. We identify the jobs with their sizes. Machine M_i has speed s_i. Without loss of generality, we assume $1 = s_1 \leq s_2 \leq \cdots \leq s_m$. If job p_j is assigned to machine M_i, then p_j/s_i time units are required to process this job. If all the machines have the same speed 1, they are called identical machines. The jobs arrive online over list, i.e., each job should be assigned to a machine before the next job is revealed. The goal is to minimize the makespan, which is the maximum completion time among the machines. Further we assume that the smallest makespan

[*] Supported by The Hong Kong Polytechnic University under grant number G-U060. Email address: lgtctng@polyu.edu.hk.

[†] Supported by the National Natural Science Foundation of China (10671177, 60021201) and Zhejiang Provincial Natural Science Foundation of China (Y607079). Email address: tanzy@zju.edu.cn.

[‡] Supported by The Hong Kong Polytechnic University under grant number G-U060. Email address: lgtcheng@polyu.edu.hk.

is achievable (i.e., the optimal value) for the sequence is known in advance. We call this problem the *semi-online scheduling problem with known optimal value*, and denoted it by $Pm|opt|C_{\max}$ if all the machines have the same speed 1, and by $Qm|opt|C_{\max}$ otherwise. Azar and Regev [2] gave an application of this problem in file allocation.

A closely related problem is the *semi-online scheduling problem with known total sum*, where the total sum of all the job sizes is known in advance [7]. We denote this problem by $Pm|sum|C_{\max}$ if all the machines have the same speed 1, and by $Qm|sum|C_{\max}$ otherwise. As the total sum of all the job sizes gives a trivial lower bound for the optimal value, the problem with known total sum may be viewed as a relaxation of the problem with known optimal value. But these two problems are clearly different, since, among other reasons, some jobs may have sizes greater than the average and hence the optimal makespan is still unknown.

The quality of the performance of an on-line or a semi-online algorithm is measured by its *competitive ratio*. For a job sequence $\mathcal{J}$ and an algorithm A, let $C^A(\mathcal{J})$ (or briefly C^A) denote the makespan produced by A and let $C^{OPT}(\mathcal{J})$ (or briefly C^{OPT}) denote the optimal makespan of the off-line version. Then the competitive ratio of A is defined as $R_A = \sup_{\mathcal{J}}\{\frac{C^A(\mathcal{J})}{C^{OPT}(\mathcal{J})}\}$. An online (semi-online) scheduling problem has a *lower bound* ρ if no online (semi-online) algorithm has a competitive ratio smaller than ρ. An online (semi-online) algorithm A is called *optimal* if its competitive ratio matches the lower bound of the problem.

Previous work: For the problem $Pm|opt|C_{\max}$, Azar and Regev [2] presented an algorithm with a competitive ratio of 13/8, which is a combination of two families of algorithms all with a competitive ratio of 5/3. They showed that a lower bound for the problem is at least 4/3. Furthermore, they presented an optimal algorithm with a competitive ratio of 4/3 when $m = 2$. If the information that all the jobs arrive in non-increasing sizes is known in advance for $P2|opt|C_{\max}$, Epstein [5] provided an optimal algorithm with a competitive ratio of 10/9.

For the problem $Q2|opt|C_{\max}$, Epstein [5] provided a comprehensive study of the competitive ratio as a function of s, where $s = s_2/s_1$ is the speed ratio of the two machines. She provided two algorithms $FAST$ and $SLOW$. Algorithm $FAST$ is for $s \in [1, \sqrt{2}]$ and $SLOW$ for $s \in [\sqrt{2}, \infty)$. The competitive ratios are as follows:

$$\begin{cases} \max\{s, \frac{2s+2}{2s+1}\}, & \text{for } 1 \le s \le \sqrt{2}, \\ \frac{s+2}{s+1}, & \text{for } s \ge \sqrt{2}, \end{cases} = \begin{cases} \frac{2s+2}{2s+1}, & \text{for } 1 \le s \le \frac{1+\sqrt{17}}{4} \approx 1.28078, \\ s, & \text{for } \frac{1+\sqrt{17}}{4} \le s \le \sqrt{2} \approx 1.41421, \\ \frac{s+2}{s+1}, & \text{for } s \ge \sqrt{2}. \end{cases}$$

To evaluate the optimality of the algorithms, she further presented the following lower bounds for the problem

$$\begin{cases} \frac{3s+1}{3s}, & \text{for } 1 \le s \le q_1 \approx 1.12433, \\ (\frac{3}{4} + \frac{\sqrt{65}}{20})s, & \text{for } q_1 \le s \le \frac{1+\sqrt{65}}{8} \approx 1.13278, \\ \frac{2s+2}{2s+1}, & \text{for } \frac{1+\sqrt{65}}{8} \le s \le \frac{1+\sqrt{17}}{4}, \\ s, & \text{for } \frac{1+\sqrt{17}}{4} \le s \le \frac{1+\sqrt{3}}{2} \approx 1.36603, \\ \frac{2s+1}{2s}, & \text{for } \frac{1+\sqrt{3}}{2} \le s \le \frac{1+\sqrt{5}}{2} \approx 1.61803, \\ \frac{s+1}{2}, & \text{for } \frac{1+\sqrt{5}}{2} \le s \le \sqrt{3} \approx 1.73205, \\ \frac{s+2}{s+1}, & \text{for } s \ge \sqrt{3}, \end{cases}$$

where q_1 is a solution of the equation $36x^4 - 135x^3 + 45x^2 + 60x + 10 = 0$. Hence the algorithms are optimal in the intervals $[\frac{1+\sqrt{65}}{8}, \frac{1+\sqrt{3}}{2}] \cup [\sqrt{3}, \infty)$; the length of the interval over which the algorithms are not optimal is about 0.4987, and the largest gap between the competitive ratio and the lower bound is about 0.07295. Furthermore, the overall competitive ratio is $\sqrt{2}$ while the overall (highest) lower bound is $\frac{1+\sqrt{3}}{2}$.

There are several papers studying the semi-online problem with known total sum, which are extensions of the basic semi-online model presented by Kellerer et al. [7]. It is interesting to note that the results (e.g., competitive ratios, even optimal algorithms), which were obtained independently by different authors, resemble those mentioned above for the problem with known optimal value. For example, Kellerer et al. [7] gave an optimal algorithm with a competitive ratio of 4/3 for $P2|sum|C_{\max}$. If it is further known that the jobs arrive in non-increasing processing times for $P2|sum|C_{\max}$, Tan and He [9] presented an optimal algorithm with a competitive ratio of 10/9. For the general m machine case, an algorithm with a competitive ratio of 8/5 was provided by Cheng et al. [4], and a lower bound of 1.565 was given by Angelelli et al. [1]. For the problem $Q2|sum|C_{\max}$, Tan and He [8] presented an algorithm with an overall competitive ratio of $\sqrt{2}$ and a lower bound of $\frac{1+\sqrt{3}}{2}$ (the algorithm and its competitive ratio are independent of s). Moreover, it is trivial that the lower bounds for $Q2|opt|C_{\max}$ given in [5] and in a later section of this paper are also lower bounds for $Q2|sum|C_{\max}$. And it can be verified that algorithms $FAST$ and $SLOW$ retain the same competitive ratios for solving $Q2|sum|C_{\max}$.

Our results: In this paper we focus on the two uniform machines case. We assume $s \ge 1$, since the case of $s \le 1$ can be converted to the case of $s \ge 1$ by scaling the job sizes. The contribution of this paper consists of two parts.

(1) We improve the results of [5] by studying the problem $Q2|opt|C_{\max}$ for $s \in [\frac{1+\sqrt{3}}{2}, \sqrt{3}]$, over which the optimal algorithm is unknown. To obtain improved algorithms for the considered interval, we develop a new unified procedure. By properly choosing the parameters of the procedure, we obtain an algorithm $STATUS1$ with a competitive ratio of

$$\max\{\frac{2s+1}{2s}, \frac{6s+6}{4s+5}\} = \begin{cases} \frac{2s+1}{2s}, & \text{for } \frac{1+\sqrt{3}}{2} \le s \le \frac{1+\sqrt{21}}{4} \approx 1.39562, \\ \frac{6s+6}{4s+5}, & \text{for } \frac{1+\sqrt{21}}{4} \le s \le \frac{1+\sqrt{13}}{3} \approx 1.535187, \end{cases}$$

and an algorithm $STATUS2$ with a competitive ratio of

$$\max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\} = \begin{cases} \frac{12s+10}{9s+7}, & \text{for } \frac{1+\sqrt{13}}{3} \le s \le \frac{5+\sqrt{241}}{12} \approx 1.71035, \\ \frac{2s+3}{s+3}, & \text{for } \frac{5+\sqrt{241}}{12} \le s \le \sqrt{3}. \end{cases}$$

Furthermore, we present improved lower bounds for the interval $[\sqrt{2}, \frac{5+\sqrt{73}}{8} \approx 1.69300]$ as follows:

$$\begin{cases} \frac{3s+5}{2s+4}, & \text{for } \sqrt{2} \le s \le \frac{\sqrt{21}}{3} \approx 1.52753, \\ \frac{3s+3}{3s+1}, & \text{for } \frac{\sqrt{21}}{3} \le s \le \frac{5+\sqrt{193}}{12} \approx 1.57437, \\ \frac{4s+2}{2s+3}, & \text{for } \frac{5+\sqrt{193}}{12} \le s \le \frac{7+\sqrt{145}}{12} \approx 1.5868, \\ \frac{5s+2}{4s+1}, & \text{for } \frac{7+\sqrt{145}}{12} \le s \le \frac{9+\sqrt{193}}{14} \approx 1.63517, \\ \frac{7s+4}{7s}, & \text{for } \frac{9+\sqrt{193}}{14} \le s \le \frac{5}{3}, \\ \frac{7s+4}{4s+5}, & \text{for } \frac{5}{3} \le s \le \frac{5+\sqrt{73}}{8}. \end{cases}$$

Then we can conclude that $STATUS1$ is optimal for $s \in [\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{21}}{4}]$. By combining our algorithms with those in [5] for different intervals, we have an algorithm $COMBINE1$ for the whole interval $[1, \infty)$. Then the length of the interval over which $COMBINE1$ is not optimal decreases to 0.46914, and the largest gap between the competitive ratio and the lower bound decreases to $8/365 \approx 0.02192$. Besides these, the overall competitive ratio of $COMBINE1$ is $\frac{1+\sqrt{3}}{2}$, which matches the overall lower bound.

Figure 1 shows that the competitive ratios of the new algorithms and the new lower bounds for the interval $[\frac{1+\sqrt{3}}{2}, \sqrt{3}]$, compared with those given in [5].

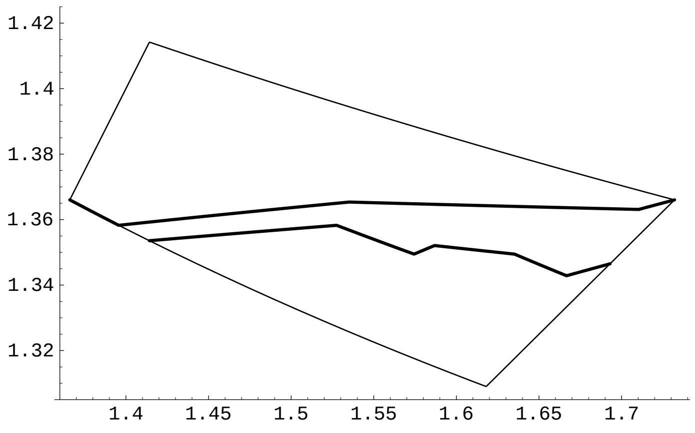

Figure 1: The upper and lower bounds for the interval $[\frac{1+\sqrt{3}}{2}, \sqrt{3}]$. The bold lines are the new upper and lower bounds.

(2) As mentioned above, from the relevant literature on the two problems $Q2|opt|C_{\max}$ and $Q2|sum|C_{\max}$, we know that information on the total sum seems strong enough to get the same competitive ratio as that of the problem with known optimal value; even the algorithms resemble or are almost the same. Moreover, it can also be shown that algorithms $FAST$ and $SLOW$ retain the same competitive ratios when they are used to solve $Q2|sum|C_{\max}$. Hence it is interesting to

identify the differences between these two problems. In this paper we make an attempt to address this question. We provide job sequences to show that algorithms $STATUS1$ and $STATUS2$ for $Q2|opt|C_{\max}$ no longer attain their competitive ratios in solving $Q2|sum|C_{\max}$. We then provide new algorithms $STATUS3$ and $STATUS4$ for the interval $[\frac{1+\sqrt{3}}{2}, \sqrt{3}]$ with competitive ratios of

$$\begin{cases} \frac{2s+1}{2s}, & \text{for } \frac{1+\sqrt{3}}{2} \le s \le q_2 \approx 1.3915, \\ \frac{s+\sqrt{29s^2+59s+30}}{4s+5}, & \text{for } q_2 \le s \le q_3 \approx 1.5062, \\ \frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s}, & \text{for } q_3 \le s \le q_4 \approx 1.6932 \\ \frac{\sqrt{4s^4+8s^3+s^2+4}+2s^2+3s+2}{2(s^2+3s)}, & \text{for } q_4 \le s \le \sqrt{3}, \end{cases}$$

where q_2, q_3 and q_4 are the roots of the following equations, respectively:

$$\begin{aligned} \frac{2s+1}{2s} &= \frac{s+\sqrt{29s^2+59s+30}}{4s+5}, \\ \frac{s+\sqrt{29s^2+59s+30}}{4s+5} &= \frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s}, \\ \frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s} &= \frac{\sqrt{4s^4+8s^3+s^2+4}+2s^2+3s+2}{2(s^2+3s)}. \end{aligned}$$

Algorithms $STATUS3$ and $STATUS4$ utilize the new procedure proposed in this paper, with different parameters.

The provided sequences also illustrate that algorithms $FAST$, $SLOW$, $STATUS1$ and $STATUS2$ cannot guarantee competitive ratios smaller than those of $STATUS3$ and $STATUS4$. Hence we can conclude that algorithms $STATUS3$ and $STATUS4$ are necessary for $Q2|sum|C_{\max}$. Furthermore, we give a lower bound for the problem $Q2|sum|C_{\max}$ for $s \in [q_5 \approx 1.62803, \sqrt{3}]$, which is strictly larger than that for $Q2|opt|C_{\max}$, where q_5 is a value of s in the following group of equations:

$$\frac{\sqrt{s^2x^2+4s(s+1-x)}+sx}{2s} = \frac{5s+2}{4s+1} = \frac{s(2s+2-x)}{(s+1)(x+2)}.$$

Based on the above results, we may conclude that the optimal algorithms for these two problems should be different for some interval of s.

By combining algorithms $FAST$, $SLOW$, $STATUS3$ and $STATUS4$, we have an algorithm $COMBINE2$ for the whole interval $[1, \infty)$. The algorithm is optimal for $s \in [\frac{1+\sqrt{65}}{8}, q_2] \cup [\sqrt{3}, \infty)$. The length of the interval over which $COMBINE2$ is not optimal is about 0.47328, and the largest gap between the competitive ratio and the lower bound is about 0.01762, which is even smaller than 0.02192, the gap existing for $Q2|opt|C_{\max}$. The overall competitive ratio of the algorithm is 1.3692, which is achieved at q_3 and only 0.0032 larger than the overall lower bound, $\frac{1+\sqrt{3}}{2}$. Figure 2 shows the differences between the competitive ratios and lower bounds of the two problems in the interval $[q_2, \sqrt{3}]$.

The structure of the paper is as follows. In Section 2 we give descriptions of our algorithms for the two problems. In Section 3 we prove the competitive ratios of the algorithms. In Section 4 we present lower bounds for the two problems.

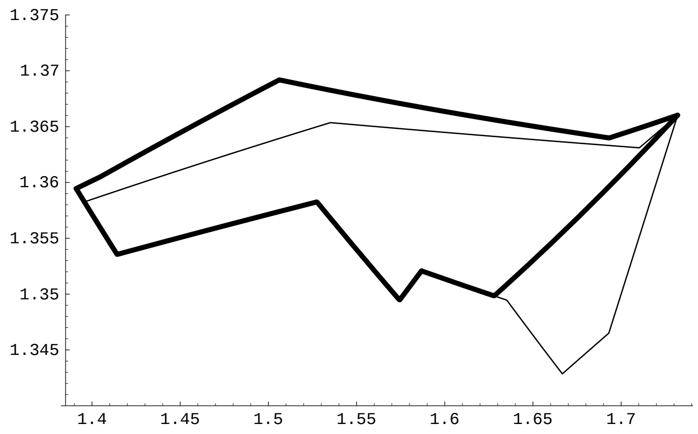

Figure 2: The differences between the upper and lower bounds for the two problems in the interval $[q_2, \sqrt{3}]$. The bold lines are for $Q2|sum|C_{\max}$.

2 Descriptions of algorithms

In this section we give descriptions of our algorithms. We first introduce some notation and definitions. In the remainder of this paper, without loss of generality, we assume that $C^{OPT} = 1$ when studying $Q2|opt|C_{\max}$, and the total sum is $1 + s$ when studying $Q2|sum|C_{\max}$. We define the *current load* of a machine as the total size of all the jobs currently assigned to it. Let p be a newly-arrived job, and let T_i be the current load of M_i right before p arrives, $i = 1, 2$. Let $r > 1$ and $t > 0$ be two parameters that will be specified later. In fact, r will be the desired competitive ratios of our algorithms. We call the process that assigns jobs one by one by an algorithm as a *scheduling process.*

Definition 2.1 *If $T_1 < 1 + s - rs$ and $T_1 + p \in [1 + s - rs, r]$, we say that a scheduling process is in Normal Stopping Status 1 ($NSS1$ for short). If $T_2 < 1 + s - r$ and $T_2 + p \in [1 + s - r, rs]$, we say that a scheduling process is in Normal Stopping Status 2 ($NSS2$ for short).*

Definition 2.2 *If (1) $T_1 < 1 + s - rs$ and $T_1 + p > r$ and (2) $T_2 < 1 + s - r$ and $T_2 + p > rs$, we say that a scheduling process is in Abnormal Stopping Status ($ANSS$ for short).*

Let $NSS = \{NSS1, NSS2\}$ and $SS = NSS \cup \{ANSS\}$. Remember that if a scheduling process is in the status of $NSS1$ and $NSS2$, it is impossible that it is in the status of $ANSS$, and vice versa.

Definition 2.3 *If $T_2 < 5 + 6s - 4r - 4rs - t$ and $T_2 + p \in [5 + 6s - 4r - 4rs - t, 2rs - 2s - 1]$, we say that a scheduling process is in Transition Status 1 ($TS1$ for short). If $T_1 < 3 + 4s - 2r - 3rs$ and $T_1 + p \in [3 + 4s - 2r - 3rs, t]$, we say that a scheduling process is in Transition Status 2 ($TS2$ for short). If $T_2 < 2 + 2s - 2r - rs$ and $T_2 + p \in [2 + 2s - 2r - rs, (r - 1)s]$, we say that a scheduling process is in Transition Status 3 ($TS3$ for short).*

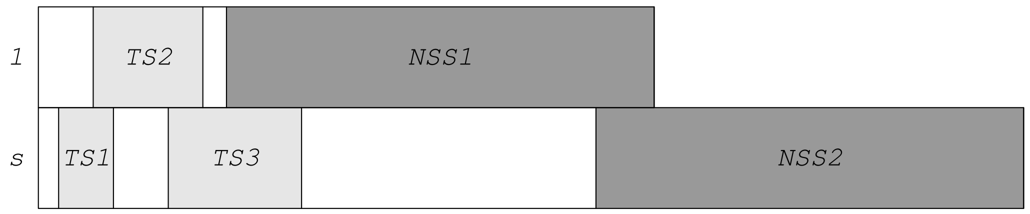

Figure 3: Normal Stopping Status and Transition Status.

Note that a scheduling process may be in more than one status simultaneously. For example, let r satisfy $r > \frac{2s+1}{2s}$ and $r > \frac{4s+3}{2s+4}$, and let the first job $p_1 = \frac{1}{2}$. Then clearly $1 + s - rs < p_1 < r$ and $2 + 2s - 2r - rs < p_1 < (r-1)s$ hold, and $T_1 = T_2 = 0$ right before an algorithm assigns p_1. Hence, according to Definitions 2.1 and 2.3, the scheduling process can be in both $TS3$ and $NSS1$ status. In such a case, we will stipulate in our algorithms that it is in $NSS1$ rather than $TS3$, which determines the assignment of the current jobs and those that come later (see the procedure defined below). Figure 3 shows which status a scheduling process may be simultaneously in. In general, we will always stipulate in our algorithms that the scheduling process is in SS rather than TS, TSj rather than TSi, $j > i$, and $NSS1$ rather than $NSS2$.

We now describe an assignment procedure with parameters r and t. It assigns jobs in a way that a schedule process ends in Stopping Status, which guarantees that the yielded solution has the desired competitive ratio. The schedule process may be first in Transition Status (for example, TSi). If so, the assignment procedure assigns the later-coming jobs such that it will be in the next Transition Status (i.e., TSj with $j > i$) or Stopping Status. In the following, let TS be a subset of $\{TS1, TS2, TS3\}$.

Assignment Procedure $AP(r, t, TS, M_i)$

While there exists at least one unassigned job, and p is the first such job, **we do**:

1. (NSS **Rule**) (1.1) If assigning job p to M_1 makes the scheduling process in $NSS1$, then assign p to M_1, and all the remaining jobs to M_2. Stop.

   (1.2) If assigning job p to M_2 makes the scheduling process in $NSS2$, then assign p to M_2, and assign all the remaining jobs to M_1. Stop.

2. ($ANSS$ **Rule**) If assigning job p makes the scheduling process in $ANSS$, assign p to the machine that can complete it earlier, and assign all the remaining jobs to another machine. Stop.

3. (TS **Rule**) If assigning job p to a machine makes the scheduling process in $TSi \in TS$, then assign it to this machine, and assign the later-coming jobs to another machine until the scheduling process in $TSj \in TS$, $j > i$, or one of SS.

4. If assigning p to a machine cannot make the scheduling process in any of SS and TS, assign it to M_i.

The above Assignment Procedure originated from He, Kellerer and Kotov [6], and Burkard, He and Kellerer [3]. They designed a procedure with parameters for solving the offline problems $P2||C_{\max}$ and $Q2||C_{\max}$. By a combination of three such procedures with different parameter values, they obtained a linear time algorithm with a worst-case ratio 12/11 for the former problem, and a linear time algorithm with a worst-case ratio 7/6 for the latter problem. Furthermore, an essentially similar procedure can be used to obtain algorithms for $Q2|sum|C_{\max}$ [8] and $Q2|opt|C_{\max}$ [5] (see below). Note that in that procedure, the authors introduced Stopping Status, although not explicitly mentioned; however they did not introduce Transition Status. Our above assignment procedure substantially extends that procedure. In fact, the algorithm of Tan and He [8] for $Q2|sum|C_{\max}$, SUM, and the algorithms of Epstein [5] for $Q2|opt|C_{\max}$, $FAST$ and $SLOW$, can be re-stated through the new procedure as follows. Here the notation $t = \infty$ means that the value of t is unnecessary, and the notion $TS = \emptyset$ means that the TS rule is deleted in the above Assignment Procedure.

Algorithm SUM [8]: Call $AP(\sqrt{2}, \infty, \emptyset, M_1)$.

Algorithm $SLOW$ [5]: Call $AP(\frac{s+2}{s+1}, \infty, \emptyset, M_1)$.

Algorithm $FAST$ [5]:

1. Let $r = \frac{2s+2}{2s+1}$ if $s \in [1, \frac{1+\sqrt{17}}{4}]$, and $r = s$ if $s \in [\frac{1+\sqrt{17}}{4}, \sqrt{2}]$. Let $t = \infty$ and $TS = \emptyset$.

2. Call $AP(r, t, TS, M_2)$.

Tan and He [8] proved that SUM has a competitive ratio of $\sqrt{2}$ for $Q2|sum|C_{\max}$ for any $s \geq 1$. Epstein [5] proved that for $Q2|opt|C_{\max}$, $FAST$ has a competitive ratio of $\frac{2s+2}{2s+1}$ for $s \in [1, \frac{1+\sqrt{17}}{4}]$, and s for $s \in [\frac{1+\sqrt{17}}{4}, \sqrt{2}]$, and $SLOW$ has a competitive ratio of $\frac{s+2}{s+1}$ for $s \geq \sqrt{2}$.

Now we give the descriptions of our improved algorithms. $STATUS1$ and $STATUS2$ are designed for $Q2|opt|C_{\max}$, while $STATUS3$ and $STATUS4$ are designed for $Q2|sum|C_{\max}$. Note that $STATUS1$ and $STATUS3$ use $TS2$ and $TS3$, together with SS; and $STATUS2$ and $STATUS4$ use all the status we have defined above.

Algorithm $STATUS1$:

1. Let $r = \max\{\frac{2s+1}{2s}, \frac{6s+6}{4s+5}\}$, $t = r - 1$ and $TS = \{TS2, TS3\}$.

2. Call $AP(r, t, TS, M_1)$.

Algorithm $STATUS2$:

1. Let $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\}$, $t = r - 1$ and $TS = \{TS1, TS2, TS3\}$.

2. Call $AP(r, t, TS, M_2)$.

Algorithm $STATUS3$:

1. Let $r = \frac{2s+1}{2s}$ if $s \in [\frac{1+\sqrt{3}}{2}, q_2]$, and $r = \frac{s+\sqrt{29s^2+59s+30}}{4s+5}$ if $s \in [q_2, q_3]$. Let $t = r - \frac{s+1}{r+1}$ and $TS = \{TS2, TS3\}$.

2. Call $AP(r, t, TS, M_1)$.

Algorithm $STATUS4$:

1. Let $r = \frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s}$ if $s \in [q_3, q_4]$, and $r = \frac{\sqrt{4s^4+8s^3+s^2+4}+2s^2+3s+2}{2(s^2+3s)}$ if $s \in [q_4, \sqrt{3}]$. Let $t = r - \frac{1+s-r}{rs-1}$ and $TS = \{TS1, TS2, TS3\}$.

2. Call $AP(r, t, TS, M_2)$.

Note that in the above description of the algorithms, q_2, q_3 and q_4 are defined in Section 1, and $\frac{s+\sqrt{29s^2+59s+30}}{4s+5}$, $\frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s}$ and $\frac{\sqrt{4s^4+8s^3+s^2+4}+2s^2+3s+2}{2(s^2+3s)}$ are the solutions of the following equations regarding r:

$$\begin{aligned} 5+6s-4r-4rs &= r - \tfrac{s+1}{r+1}, \\ 9+12s-6r-9rs &= r - \tfrac{1+s-r}{rs-1}, \\ 2+2s-2r-rs &= r - \tfrac{1+s-r}{rs-1}, \end{aligned}$$

respectively.

We first show that, after all the jobs have been assigned by Assignment Procedure, the scheduling process must be in SS. Otherwise, we have $T_1 + p_n < 1+s-rs < 1$ and $T_2 + p_n < 1+s-r < s$. Hence there exists a solution such that the makespan is less than 1, and the total sum of sizes is less than $1+s$, which violates the assumption that $C^{OPT} = 1$ for $Q2|opt|C_{\max}$ or $T = 1+s$ for $Q2|sum|C_{\max}$.

Lemma 2.1 *If the scheduling process of algorithm A ends in NSS, we have $\frac{C^A}{C^{OPT}} \le r$. If the scheduling process ends in $ANSS$ by assigning p, we have $C^A = \min\{T_1 + p, \frac{T_2+p}{s}\}$.*

Proof. If the scheduling process ends in $NSS1$, then $1+s-rs \le T_1 + p \le r$. It follows that the completion time of M_1 is less than r. Furthermore, the load of M_2 after assigning all the jobs is less than $\sum_{i=1}^n p_i - (T_1+p) \le (1+s) - (1+s-rs) = rs$. It follows that the completion time of M_1 is less than r, too. Hence we have $\frac{C^A}{C^{OPT}} \le r$. The case that the scheduling process ends in $NSS2$ can be proved similarly.

Suppose that the scheduling process ends in $ANSS$. (1) If $T_1 + p \le \frac{T_2+p}{s}$, then p is assigned to machine M_1 and all the remaining jobs to M_2 by the rule of Assignment Procedure. From $T_1 + p > r$, we know that the load of M_2 after assigning all the jobs is less than $\sum_{i=1}^n p_i - (T_1+p) \le (1+s) - r < rs$. It follows that $C^A = T_1 + p = \min\{T_1 + p, \frac{T_2+p}{s}\}$. (2) If $T_1 + p > \frac{T_2+p}{s}$, the result can be obtained similarly. $\square$

In fact, we will see that for $Q2|opt|C_{\max}$, the algorithms must end in NSS, resulting in the desired competitive ratios. However, for $Q2|sum|C_{\max}$, the algorithms may end in $ANSS$, but it can be proved that the desired competitive ratios are still valid because of the assignment rule in the algorithms. From this point, we can obtain some insights into the algorithms for these two problems.

3 Competitive ratios of algorithms

3.1 Algorithms for $Q2|opt|C_{\max}$

Theorem 3.1 *For $s \in [\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{13}}{3}]$, $STATUS1$ has a competitive ratio of $r = \max\{\frac{2s+1}{2s}, \frac{6s+6}{4s+5}\}$.*

Proof. It is easy to verify that for $s \in [\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{13}}{3}]$, the values of r and t defined in the algorithm satisfy the following group of inequalities

$$\begin{aligned} 0 < 2+2s-2r-rs < (r-1)s < 1+s-r < rs, \\ 0 < 3+4s-2r-3rs < t < 1+s-rs < r. \end{aligned} \tag{1}$$

Hence all SS and TS are well-defined. If the scheduling process of $STATUS1$ ends in NSS, it follows that $\frac{C^A}{C^{OPT}} \leq r$ (due to Lemma 2.1). Hence we suppose that the scheduling process ends in $ANSS$. Noting that $TS = \{TS2, TS3\}$ in algorithm $STATUS1$, we distinguish three cases to get a contradiction as follows:

Case 1 The scheduling process is first in $TS3$ before it ends in $ANSS$.

On the arrival of job p, the scheduling process is in $TS3$, i.e.,

$$2+2s-2r-rs \leq T_2 + p \leq (r-1)s. \tag{2}$$

By the TS Rule, the later-coming jobs are assigned to M_1 until the scheduling process is in another status. Since the new status is not $NSS1$ according to the hypothesis, there exists a job, denoted by p_a, such that the current load of M_1 is increased from less than $1+s-rs$ to greater than r if p_a is assigned to M_1. Hence $p_a > r-(1+s-rs) = rs+r-s-1$. Combining it with the first inequality of (2), we obtain $T_2+p+p_a \geq 1+s-r$. On the other hand, $C^{OPT} = 1$ implies $p_a \leq s$. Combining it with the second inequality of (2), we then have $T_2+p+p_a \leq rs$. Hence, we have $T_2+p+p_a \in [1+s-r, rs]$, which implies that the new status of the scheduling process must be $NSS2$ by assigning p_a to M_2, a contradiction.

Case 2 The scheduling process is first in $TS2$ before it ends in the status $ANSS$.

On the arrival of job p, the scheduling process is in $TS3$, i.e.,

$$3+4s-2r-3rs \leq T_1 + p \leq r-1. \tag{3}$$

By the TS Rule, the later-coming jobs are assigned to M_2 until the scheduling process is in another status. If the next status is $TS3$, then a similar argument as that in Case 1 can reach the conclusion. Otherwise, as no job makes the scheduling process in $TS3$, there exists a job, denoted by p_b, such that the current load of M_2 is increased from less than $2+2s-2r-rs$ to greater than $(r-1)s$ if p_b is assigned to M_2. Hence $p_b > (r-1)s-(2+2s-2r-rs) = 2rs+2r-3s-2$. Combining it with the first inequality of (3), we obtain $T_1+p+p_b > 1+s-rs$. If $T_1+p+p_b \le r$, i.e., $T_1+p+p_b \in [1+s-rs, r]$, the scheduling process is in $NSS1$ by assigning p_b to M_1. Hence $T_1+p+p_b > r$. Substituting the second inequality of (3) into it, we have $p_b > 1$.

p_b and the later-coming jobs are assigned to M_2 by the TS Rule. Since the scheduling process is never in $NSS2$, similarly there exists a job, denoted by p_c, such that $p_c > rs-(1+s-r) = (r-1)(s+1)$. Combining it with the first inequality of (3), we obtain

$$T_1+p+p_c \ge (3+4s-2r-3rs)+(r-1)(s+1) = 2+3s-r-2rs > 1+s-rs,$$

where the last inequality is from $r = \max\{\frac{2s+1}{2s}, \frac{6s+6}{4s+5}\} < \frac{2s+1}{s+1}$. To avoid the situation that the scheduling process is in $NSS1$ by assigning p_c to M_1, $T_1+p+p_c > r$ must hold. We thus have $p_c > 1$ because of the second inequality of (3). Now we have two jobs with sizes of greater than 1. Since $s < 2$, we obtain $C^{OPT} > 1$, a contradiction.

Case 3 The scheduling process is never in $TS2$ or $TS3$ before it ends in $ANSS$.

According to the algorithm, jobs are always assigned to M_1 if the scheduling process is never in SS and TS. Denote by p_d the first job that forces the load of M_1 to exceed $3+4s-2r-2rs$. If $p_d \le (r-1)-(3+4s-2r-3rs) = 3rs+3r-4s-4$, assigning p_d to M_1 makes the new load of M_1 lie in $[3+4s-2r-3rs, r-1]$, which implies that the scheduling process is in $TS2$, contradicting our assumption. Hence we obtain $p_d > 3rs+3r-4s-4$. Combining it with $r = \max\{\frac{2s+1}{2s}, \frac{6s+6}{4s+5}\} \ge \frac{6s+6}{4s+5}$, we obtain

$$p_d > 2+2s-2r-rs. \tag{4}$$

As there is no job on M_2 yet, if $p_d \le (r-1)s$, clearly assigning p_d to M_2 makes the scheduling process in $TS3$, which again contradicts our assumption. So we obtain $p_d > (r-1)s$. Combining it with $r = \max\{\frac{2s+1}{2s}, \frac{6s+6}{4s+5}\} \ge \frac{2s+1}{2s}$, we have $p_d > 1+s-rs$. To avoid the situation that the scheduling process ends in $NSS1$ by assigning p_d to M_1, the load of machine M_1 would be increased from less than $3+4s-2r-3rs$ to greater than r, which implies $p_d > r-(3+4s-2r-3rs) = 3rs+3r-4s-3$. Combining it with $r = \max\{\frac{2s+1}{2s}, \frac{6s+6}{4s+5}\} \ge \frac{5s+4}{3s+4}$, we have $p_d > 1+s-r$. On the other hand, $C^{OPT} = 1$ implies $p_d \le s < rs$. Now we have $1+s-r < p_d < rs$, then by assigning p_d to M_2 the scheduling process is in $NSS2$, a contradiction. The proof is complete. □

Theorem 3.2 *For $s \in [\frac{1+\sqrt{13}}{3}, \sqrt{3}]$, $STATUS2$ has a competitive ratio $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\}$.*

Proof. It is easy to verify that for $s \in [\frac{1+\sqrt{13}}{3}, \sqrt{3}]$, the values of r and t defined in algorithm $STATUS2$ satisfy (1) and

$$0 < 5+6s-4r-4rs-t < 2rs-2s-1 < 2+2s-2r-rs. \tag{5}$$

So all the status are well-defined. Similar to the proof of Theorem 3.1, we show the result by contradiction. We still suppose that the scheduling process of $STATUS2$ ends in $ANSS$. Noting that $TS = \{TS1, TS2, TS3\}$ in algorithm $STATUS2$, we distinguish four cases. The first two cases, together with their proofs, are the same as the corresponding parts in the proof of Theorem 3.1 and omitted.

Case 3 The scheduling process is first in $TS1$ before it ends in $ANSS$.

On the arrival of job p, the scheduling process is in $TS1$, i.e.,

$$6 + 6s - 5r - 4rs = 5 + 6s - 4r - 4rs - t \leq T_2 + p \leq 2rs - 2s - 1. \tag{6}$$

By the TS Rule, the later-coming jobs are assigned to M_1 until the scheduling process is in the next status. If the next status is $TS2$, then the same argument as that in Case 2 of Theorem 3.1 can reach the conclusion. Otherwise, as no job makes the scheduling process in $TS2$, there exists a job, denoted by p_e, such that the current load of M_1 is increased from less than $3+4s-2r-3rs$ to greater than $r-1$ if p_e is assigned to M_1. Hence $p_e > (r-1)-(3+4s-2r-3rs) = 3rs+3r-4s-4$. If $p_e \leq 1 + s - rs$, by (6), we obtain

$$\begin{aligned} 2 + 2s - 2r - rs &= (6 + 6s - 5r - 4rs) + (3rs + 3r - 4s - 4) \leq T_2 + p + p_e \\ &\leq (2rs - 2s - 1) + (1 + s - rs) = (r - 1)s. \end{aligned}$$

Hence assigning p_e to M_2 makes the scheduling process in $TS3$, and thus the proof of Case 1 of Theorem 3.1 can reach the conclusion. Therefore we assume $p_e > 1+s-rs$. To avoid the situation that the scheduling process is in $NSS1$ by assigning p_e to M_1, we have $p_e > r-(3+4s-2r-3rs) = 3rs + 3r - 4s - 3$. Combining it with the first inequality of (6), we have

$$T_2 + p + p_e > (6 + 6s - 5r - 4rs) + (3rs + 3r - 4s - 3) > 1 + s - r,$$

where the last inequality is from $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\} < \frac{s+2}{s+1}$. If further $T_2 + p + p_e \leq rs$, the scheduling process is in $NSS2$ by assigning p_e to M_2. Hence $T_2 + p + p_e > rs$. Combining it with the second inequality of (6), we obtain $p_e > rs - (2rs - 2s - 1) > s$, which contradicts $C^{OPT} = 1$.

Case 4 The scheduling process is never in TS before it ends in $ANSS$.

According to the algorithm, jobs are always assigned to M_2 if the scheduling process is never in SS and TS. Denote by p_f the first job that forces the load of M_2 to exceed $6 + 6s - 5r - 4rs$. We next prove $p_f > s$, which implies $C^{OPT} > 1$. This contradiction will complete the proof of Theorem 3.2.

In fact, if $p_f \leq (2rs - 2s - 1) - (6 + 6s - 5r - 4rs) = 6rs + 5r - 8s - 7$, assigning p_f to M_2 makes the new load of M_2 lie in $[6 + 6s - 5r - 4rs, 2rs - 2s - 1]$, which implies the scheduling process is in $TS1$, contradicting our assumption. Hence $p_f > 6rs + 5r - 8s - 7$. Combining it with $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\} \geq \frac{12s+10}{9s+7}$, we obtain $p_f > 3 + 4s - 2r - 3rs$. As there is no job on M_1 yet, if $p_f \leq t = r - 1$, assigning p_f to M_2 makes the scheduling process in $TS2$, contradicting our assumption. Hence we have $p_f > r - 1$. Combining it with $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\} \geq \frac{2s+3}{s+3}$, we have $p_f > 2 + 2s - 2r - rs$. If $p_f \leq (r-1)s$, assigning p_f to M_1 makes the scheduling process in $TS3$, which again contradicts our assumption. Therefore, we have $p_f > (r-1)s$. Combining it with $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\} \geq \frac{2s+1}{2s}$, we obtain $p_f > 1 + s - rs$. To avoid the situation that the scheduling process is in $NSS1$ by assigning p_f to M_1, $p_f > r$ must hold. Combing it with $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\} \geq \frac{s+1}{2}$, we get $p_f > 1 + s - r$. Recall that the current load of M_2 is no greater than $6 + 6s - 5r - 4rs$. If $p_f < rs - (6 + 6s - 5r - 4rs) = 5rs + 5r - 6s - 6$, assigning p_f to M_2 makes the scheduling process in $NSS2$. Hence we get $p_f > 5rs + 5r - 6s - 6$. Combining it with $r = \max\{\frac{12s+10}{9s+7}, \frac{2s+3}{s+3}\} \geq \frac{7s+6}{5s+5}$, we get $p_f > s$. Thus the proof is completed. $\square$

By combining our algorithms with those of Epstein [5], we have an algorithm $COMBINE1$ for the whole interval $[1, \infty)$ as follows:

$$\begin{cases} FAST, & \text{for } 1 \leq s \leq \frac{1+\sqrt{3}}{2}, \\ STATUS1, & \text{for } \frac{1+\sqrt{3}}{2} \leq s \leq \frac{1+\sqrt{13}}{3}, \\ STATUS2, & \text{for } \frac{1+\sqrt{13}}{3} \leq s \leq \sqrt{3}, \\ SLOW, & \text{for } s \geq \sqrt{3}. \end{cases}$$

The overall competitive ratio of $COMBINE1$ is $\frac{1+\sqrt{3}}{2}$, which matches the overall lower bound. Algorithm $COMBINE1$ can be viewed as an optimal algorithm in the sense that it yields an overall competitive ratio.

3.2 Algorithms for $Q2|sum|C_{\max}$

As mentioned before, $Q2|sum|C_{\max}$ is a relaxation of $Q2|opt|C_{\max}$, so any algorithm for the latter problem must have a competitive ratio of no greater than one if it can be used to solve the former problem. The next Theorem 3.3 states that $FAST$ and $SLOW$ retain the same competitive ratios for any $s \geq 1$, while the next Examples 1-2 state the contrary when $s \in [\frac{1+\sqrt{3}}{2}, \sqrt{3}]$, which show that algorithms $STATUS1$ and $STATUS2$ may achieve worse competitive ratios when applying them directly to the former problem. Hence new algorithms, such as $STATUS3$ and $STATUS4$, are necessary for this interval. Further, Theorems 3.4-3.5, together with the sequences given in the proof of Theorem 3.3, show that $STATUS3$ and $STATUS4$ have smaller competitive ratios than those of $FAST$ and $SLOW$ for the considered interval.

Combining these with the lower bound for $Q2|sum|C_{\max}$ presented in the next section, which is not valid for $Q2|opt|C_{\max}$, we conclude that these two problems are indeed different.

Theorem 3.3 *For $Q2|sum|C_{\max}$, if $s \in [1, \frac{1+\sqrt{3}}{2}]$, $FAST$ has a competitive ratio of*

$$\max\{s, \frac{2s+2}{2s+1}\} = \begin{cases} \frac{2s+2}{2s+1}, & for \quad s \in [1, \frac{1+\sqrt{17}}{4}], \\ s, & for \quad s \in [\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{17}}{4}]; \end{cases}$$

and if $s \in [\sqrt{3}, \infty)$, $SLOW$ has a competitive ratio of $\frac{s+2}{s+1}$.

Proof. By an easy modification of the proofs in [5] we can get the result. The proofs are quite similar to those of Theorems 3.4-3.5 and are omitted. The following sequences show that the algorithms cannot have competitive ratios smaller than those claimed in the theorem:

For $1 \le s \le \frac{1+\sqrt{17}}{4}$, let $\mathcal{J} = \{p_1 = \frac{2s^2+s-1}{2s+1}, p_2 = 1, p_3 = \frac{1}{2s+1}\}$. Then $FAST$ assigns p_1 to M_2, p_2 and p_3 to M_1. We have $C^{FAST} = \frac{2s+2}{2s+1}$, $C^{OPT} = 1$, and thus $\frac{C^{FAST}}{C^{OPT}} = \frac{2s+2}{2s+1}$.

For $\frac{1+\sqrt{17}}{4} \le s \le \sqrt{2}$, let $\mathcal{J} = \{p_1 = 1, p_2 = s\}$. Then $FAST$ assigns p_1 to M_2, p_2 to M_1. We have $C^{FAST} = s$, $C^{OPT} = 1$, and thus $\frac{C^{FAST}}{C^{OPT}} = s$.

For $s \ge \sqrt{2}$, let $\mathcal{J} = \{p_1 = \frac{1}{s+1}, p_2 = s, p_3 = \frac{s}{s+1}\}$. Then $SLOW$ assigns p_1 to M_1, and p_2 and p_3 to M_2. We have $C^{FAST} = \frac{s+2}{s+1}$, $C^{OPT} = 1$, and thus $\frac{C^{SLOW}}{C^{OPT}} = \frac{s+2}{s+1}$. □

Example 1: Let $s = \frac{3}{2} \in [\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{13}}{3}]$, where $STATUS1$ is used to solve $Q2|opt|C_{\max}$. The competitive ratio of $STATUS1$ is $\frac{15}{11}$ for $s = \frac{3}{2}$. Let $\mathcal{J} = \{p_1 = \frac{3}{22} - \epsilon, p_2 = \frac{5}{22} - \epsilon, p_3 = 1 + 3\epsilon, p_4 = \frac{25}{22} - \epsilon\}$, where ϵ is a sufficiently small positive number. In an optimal schedule, p_3 is assigned to M_1 and the remaining jobs are assigned to M_2. We thus have $C^{OPT} = 1 + \epsilon$. According to $STATUS1$, p_1 and p_2 are assigned to M_1 and the scheduling process is in $TS2$. As $p_1 + p_2 + p_3 = \frac{15}{11} + \epsilon$, the scheduling process cannot be in $NSS1$, p_3 is thus assigned to M_2 by the TS rule. p_4 makes the scheduling process end in $ANSS$ by assigning it to a machine such that $C^{STATUS1} = \min\{p_1 + p_2 + p_4, \frac{p_3+p_4}{s}\} = \frac{47}{33} + \frac{4}{3}\epsilon$. It follows that $\frac{C^{STATUS1}}{C^{OPT}} \to \frac{47}{33} > \frac{15}{11}$ when $\epsilon \to 0$. We conclude that $STATUS1$ cannot retain the same competitive ratio for both problems. Furthermore, as $\frac{47}{33} \approx 1.42424$ is even larger than the competitive ratio of $\frac{3+7\sqrt{15}}{22} \approx 1.36868$ of $STATUS3$ when $s = \frac{3}{2}$ (see Theorem 3.4), we conclude that $STATUS3$ is definitely better than $STATUS1$ when both are used to solve the same problem $Q2|sum|C_{\max}$.

Example 2 Let $s = \frac{12}{7} \in [\frac{1+\sqrt{13}}{3}, \sqrt{3}]$, where $STATUS2$ is used to solve $Q2|opt|C_{\max}$. The competitive ratio of $STATUS2$ is $\frac{15}{11}$ for $s = \frac{12}{7}$. Let $\mathcal{J} = \{p_1 = \frac{28}{77} - \epsilon, p_2 = 1 + 2\epsilon, p_3 = \frac{104}{77} - \epsilon\}$, where ϵ is a sufficiently small positive number. In an optimal schedule, p_3 is assigned to M_1 and the remaining jobs are assigned to M_2. We have $C^{OPT} = 1 + \epsilon$. On the other hand, it can be easily verified that $STATUS2$ assigns p_1 to M_1, and p_2 and p_3 to M_2. We have $C^{STATUS2} = \frac{181}{132} + \epsilon$. Thus $\frac{C^{STATUS2}}{C^{OPT}} \to \frac{181}{132} > \frac{15}{11}$ when $\epsilon \to 0$. We conclude that $STATUS2$ cannot retain the same competitive ratio for both problems. Furthermore, as $\frac{181}{132} \approx 1.37121$ is even larger than the competitive ratio 1.36507 of $STATUS4$ when $s = \frac{12}{7}$ (see Theorem 3.5), we conclude that $STATUS4$ is definitely better than $STATUS2$ when both are used to solve the same problem $Q2|sum|C_{\max}$.

The above examples can be easily extended to other values of s.

Theorem 3.4 *For $s \in [\frac{1+\sqrt{3}}{2}, q_2]$, $STATUS3$ has a competitive ratio of*

$$r = \begin{cases} \frac{2s+1}{2s}, & \text{for } \frac{1+\sqrt{3}}{2} \le s \le q_2, \\ \frac{s+\sqrt{29s^2+59s+30}}{4s+5}, & \text{for } q_2 \le s \le q_3. \end{cases}$$

Proof. Similarly we can show that all the status used in $STATUS3$ are well-defined. We will again prove the result by contradiction. Suppose that there exists a sequence satisfying $\frac{C^{STATUS3}}{C^{OPT}} > r$. By Lemma 2.1, we only need to consider the case that the scheduling process ends in $ANSS$.

Case 1 The scheduling process is first in $TS3$ before it ends in $ANSS$.

On the arrival of job p, the scheduling process is in $TS3$, i.e., $2+2s-2r-rs \le T_2+p \le (r-1)s$. By the same argument as that in the proof of Case 1 of Theorem 3.1, we know that there exists a job p_a such that $p_a > rs+r-s-1$ and $T_2+p+p_a \ge 1+s-r$. Since the scheduling process ends in the status $ANSS$ instead of $NSS2$, we have $T_2+p+p_a \ge rs$. Combining it with $T_2+p \le (r-1)s$, we have $p_a > s$. As $T_2+p+p_a \ge rs$ and $T_1+p_a \ge p_a > s > r$, p_a makes the scheduling process end in $ANSS$. Noting that the current load of machine M_2 is T_2+p (not T_2) when assigning p_a, by Lemma 2.1, we have

$$\frac{C^{STATUS3}}{C^{OPT}} \le \frac{\frac{T_2+p+p_a}{s}}{\frac{p_a}{s}} = \frac{T_2+p+p_a}{p_a} = \frac{T_2+p}{p_a}+1 \le r-1+1 = r.$$

Case 2 The scheduling process is first in $TS2$ before it ends in $ANSS$.

Assume that assigning job p makes the scheduling process in $TS2$, i.e., $3+4s-2r-3rs \le T_1+p \le t$. Using an argument analogous to that in the proof of Case 2 of Theorem 3.1, we know that there exists a job satisfying $p_b > r-(T_1+p) \ge r-t > 1$. Furthermore, there exists another job p_c such that $p_c > r-(T_1+p) \ge r-t > 1$ and all the jobs coming later than p_b but earlier than p_c, together with p_b, are assigned to M_2. Note that $\sum_{j=b}^{c} p_j \ge p_b+p_c > 2 > s > 1+s-r$. If $\sum_{j=b}^{c} p_j \le rs$, assigning p_c to M_2 makes the scheduling process end in $NSS2$, which contradicts our assumption. Thus we have $\sum_{j=b}^{c} p_j > rs$. Combing it with $T_1+p+p_c > r$, we know that assigning p_c makes the scheduling process end in $ANSS$. By Lemma 2.1, we get $C^{STATUS3} \le T_1+p+p_c$.

We next show $C^{OPT} \ge \min\{p_b,\ p_c\}$. In fact, if p_b and p_c are processed on the same machine, we have $C^{OPT} \ge \frac{p_b+p_c}{s} > \min\{p_b,\ p_c\}$ (due to $p_b, p_c > 1$ and $s < 2$). Otherwise, at least one of them is assigned on M_1, and we also have $C^{OPT} \ge \min\{p_b,\ p_c\}$.

If $C^{OPT} \ge p_b$, we have

$$\frac{C^{STATUS3}}{C^{OPT}} \le \frac{T_1+p+p_c}{p_b} \le \frac{\sum_{j=1}^{n} p_j - p_b}{p_b} = \frac{1+s-p_b}{p_b} = \frac{1+s}{p_b}-1 \le \frac{1+s}{r-t}-1 = r, \tag{7}$$

where the last equality is due to $t = r - \frac{s+1}{r+1}$. If $C^{OPT} \ge p_c$, we have

$$\frac{C^{STATUS3}}{C^{OPT}} \le \frac{T_1+p+p_c}{p_c} \le \frac{t+p_c}{p_c} = \frac{t}{p_c}+1 \le \frac{t}{r-t}+1 = \frac{r}{r-t} < r, \tag{8}$$

where the last inequality is due to $r-t > 1$.

Case 3 The scheduling process is never in $TS2$ or $TS3$ before it ends in $ANSS$.

By an argument analogous to that in the proof of Case 3 of Theorem 3.1, there exists a job p_d satisfying $p_d > t-(3+4s-2r-3rs)$. By the definitions of t and r, we have $p_d > t-(3+4s-2r-3rs) \ge 2+2s-2r-rs$, which is just inequality (4) in the proof of Theorem 3.1. By following the argument after inequality (4) in that proof, we obtain $p_d > rs$. Obviously, all the jobs earlier than p_d are assigned to M_1. Then we are confronted with $T_1+p_d > rs > r$ and $p_d > rs$, which means that p_d makes the scheduling process end in $ANSS$. By Lemma 2.1 we have $C^{STATUS3} = \frac{p_d}{s}$. On the other hand, $C^{OPT} \ge \frac{p_d}{s}$ trivially. We thus have $\frac{C^{STATUS3}}{C^{OPT}} = 1 < r$. □

Theorem 3.5 *For $s \in [q_3, \sqrt{3}]$, $STATUS4$ has a competitive ratio of*

$$r = \begin{cases} \frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s}, & \text{for } \ q_3 \le s \le q_4, \\ \frac{\sqrt{4s^4+8s^3+s^2+4}+2s^2+3s+2}{2(s^2+3s)}, & \text{for } \ q_4 \le s \le \sqrt{3}. \end{cases}$$

Proof. Similarly we can show that all the status used in $STATUS4$ are well-defined. We prove the result by contradiction, too. Hence we only need to consider the case that the scheduling process of $STATUS4$ ends in $ANSS$.

Case 1 The scheduling process is first in $TS3$ before it ends in $ANSS$.

Using the same argument as that in Case 1 of Theorem 3.4 can reach the conclusion.

Case 2 The scheduling process is first in $TS2$ before it ends in $ANSS$.

Using an argument analogous to that in the proof of Case 2 of Theorem 3.4, we know that there exists a job satisfying $p_b > r - (T_1 + p) \ge r - t > 1$, and a job p_c satisfying $p_c > r - (T_1 + p) \ge r - t > 1$. Moreover, assigning p_s makes the scheduling process end in $ANSS$. Noting that all the jobs arriving later than p_b but earlier than p_c, together with p_b, are assigned to M_2, we have $C^{STATUS4} = \min\{T_1 + p + p_c, \frac{\sum_{j=b}^{c} p_j}{s}\}$. Since $p_b > r - (T_1 + p)$, we have

$$C^{STATUS4} \le \frac{\sum_{j=b}^{c} p_j}{s} \le \frac{\sum_{j=1}^{n} p_j - (T_1 + p)}{s} = \frac{1 + s - (T_1 + p)}{s} \le \frac{1 + s - (r - p_b)}{s}. \tag{9}$$

On the other hand, we can obtain $C^{OPT} \ge \max\{p_b, p_c\}$ similarly. If $C^{OPT} \ge p_b$, from (9) we obtain

$$\frac{C^{STATUS4}}{C^{OPT}} \le \frac{1 + s - (r - p_b)}{sp_b} = \frac{1 + s - r}{sp_b} + \frac{1}{s} \le \frac{1 + s - r}{s(r - t)} + \frac{1}{s} = r, \tag{10}$$

where the second inequality is due to $p_b > r - t$, and the last equality is due to the definition of $t = r - \frac{1+s-r}{rs-1}$. If $C^{OPT} \ge p_c$, we get $\frac{C^{STATUS4}}{C^{OPT}} \le r$ in the same way as the proof of (8).

Case 3 The scheduling process is first in $TS1$ before it ends in $ANSS$.

On the arrival of job p, the scheduling process is in $TS1$, i.e.,

$$5 + 6s - 4r - 4rs - t \le T_2 + p \le 2rs - 2s - 1. \tag{11}$$

Using an argument analogous to that in the proof of Case 3 of Theorem 3.2, we find that there exists a job p_e such that the current load of M_1 is increased from $T_1 < 3 + 4s - 2r - 3rs$ to greater than t if p_e is assigned to M_1. Hence

$$p_e > t - (3 + 4s - 2r - 3rs). \tag{12}$$

We classify four subcases according to the value of p_e.

Subcase 3.1 $p_e \le 1 + s - rs$. From the second inequality of (11) and this subcase's assumption, we have $T_2 + p + p_e \le (r-1)s$. From the first inequality of (11) and (12), we have $2 + 2s - 2r - rs < T_2 + p + p_e$. These inequalities imply that the scheduling process is in $TS3$ by assigning p_e to M_2. Hence using the same argument as in Case 1 can complete the proof.

Subcase 3.2 $1 + s - rs < p_e \leq r - T_1$. We have $1 + s - rs < p_e \leq T_1 + p_e \leq r$. Hence the scheduling process can be in $NSS1$ by assigning p_e to M_1, which violates the hypothesis that the scheduling process ends in $ANSS$.

Subcase 3.3 $r - T_1 < p_e \leq rs - (T_2 + p)$. From $T_1 < 3 + 4s - 2r - 3rs$ and (11), we get

$$1+s-r < (5+6s-4r-4rs-t)+r-(3+4s-2r-3rs) \leq T_2+p+r-T_1 \leq T_2+p+p_e \leq rs.$$

Hence the scheduling process can be in $NSS2$ by assigning p_e to M_2, a contradiction again.

Subcase 3.4 $p_e > rs - (T_2 + p)$. By the second inequality of (11) and the definition of r, we have $T_2+p < (r-1)s$. Hence $p_e > s$. Now we have $T_2+p+p_e > rs$ and $T_2+p < (r-1)s < 1+s-r$, as well as $T_1 < 3 + 4s - 2r - 3rs < 1 + s - rs$ and $T_1 + p_e > r$, which states that p_e makes the scheduling process end in $ANSS$. By Lemma 2.1, we know

$$\frac{C^{STATUS4}}{C^{OPT}} \leq \frac{\frac{T_2+p+p_e}{s}}{\frac{p_e}{s}} \leq \frac{T_2+p}{p_e} + 1 \leq r - 1 + 1 = r.$$

Case 4 The scheduling process is never in one of TS before it ends in $ANSS$.

By an argument analogous to that in the proof of Case 4 of Theorem 3.2, there exists a job p_f such that the current load of M_2 is increased from $T_2 < 5+6s-4r-4rs-t$ to greater than $2rs-2s-1$ if p_f is assigned to M_2. Hence $p_f > (2rs - 2s - 1) - (5 + 6s - 4r - 4rs - t) = t + 6rs + 4r - 8s - 6$. By the definitions of t and r, we have $p_f > 3 + 4s - 2r - 3rs$ and $t \geq 2 + 2s - 2r - rs$. We classify five subcases according to the value of p_f.

Subcase 4.1 $p_f \leq t$. Since there is no job processed on M_1 yet, from $p_f > 3 + 4s - 2r - 3rs$ and $p_f \leq t$, we know that the scheduling process is in $TS2$ by assigning p_f to M_1. Hence using the same argument as in Case 2 can complete the proof.

Subcase 4.2 $t < p_f \leq (r-1)s - T_2$. By $t \geq 2 + 2s - 2r - rs$, we know that $2 + 2s - 2r - rs \leq T_2 + t \leq T_2 + p_f \leq (r-1)s$. Then the scheduling process is in $TS3$ by assigning p_f to M_2. Hence using the same argument as that in Case 1 can complete the proof.

Subcase 4.3 $(r-1)s - T_2 < p_f \leq r$. By $T_2 < 5 + 6s - 4r - 4rs - t$, and the definitions of t and r, we have

$$p_f > (r-1)s - T_2 > (r-1)s - (5 + 6s - 4r - 4rs - t) > 1 + s - rs.$$

As $1 + s - rs < p_f \leq r$, the scheduling process is in $NSS1$ by assigning p_f to M_1, which violates the hypothesis that the scheduling process ends in $ANSS$.

Subcase 4.4 $r < p_f \leq rs - T_2$. Then $T_2 + p_f \geq p_f > r > 1 + s - r$. Combining it with $T_2+p_f \leq rs$, we know that the scheduling process is in $NSS2$ by assigning p_f to M_2, a contradiction again.

Subcase 4.5 $p_f > rs-T_2$. By (1) and (5), we have $T_2 < 5+6s-4r-4rs-t < 2+2s-2r-rs < (r-1)s$, and thus $p_f > s$. Hence, similar to Subcase 3.4, assigning p_f to any machine makes the scheduling process end in $ANSS$. By Lemma 2.1, we have

$$\frac{C^{STATUS4}}{C^{OPT}} \le \frac{\frac{T_2+p_f}{s}}{\frac{p_f}{s}} \le \frac{T_2}{p_f}+1 < \frac{(r-1)s}{s}+1 \le r-1+1=r.$$

The proof is thus completed. $\square$

Based on Theorems 3.3-3.5, we have an algorithm $COMBINE2$ for the whole interval $[1,\infty)$ as follows:

$$\begin{cases} FAST, & \text{for } 1 \le s \le \frac{1+\sqrt{3}}{2}, \\ STATUS3, & \text{for } \frac{1+\sqrt{3}}{2} \le s \le q_3, \\ STATUS4, & \text{for } q_3 \le s \le \sqrt{3}, \\ SLOW, & \text{for } s \ge \sqrt{3}. \end{cases}$$

The overall competitive ratio of the algorithm is 1.3692, which is achieved at q_3 and only 0.0032 larger than the trivial overall lower bound of $\frac{1+\sqrt{3}}{2}$.

4 Lower bounds

This section considers lower bounds for $Q2|opt|C_{\max}$ and $Q2|sum|C_{\max}$. The proof will be completed by using an adversarial method. We will present a series of sequences and show that no semi-online algorithm can work well on all of them simultaneously, i.e., for any semi-online algorithm A, there always exists a sequence such that C^A/C^{OPT} is no less than our desired lower bound.

4.1 Lower bounds for $Q2|opt|C_{\max}$

This subsection focuses on the problem $Q2|opt|C_{\max}$. We present improved lower bounds for $s \in [\sqrt{2}, \frac{5+\sqrt{73}}{8}]$. All the sequences used in this subsection have the optimal value $C^{OPT}=1$ (and the total sum of sizes $1+s$), thus $C^A/C^{OPT}=C^A$. We prove the case $s \in [\sqrt{2}, \frac{\sqrt{21}}{3}]$ in detail. The remaining cases of $s \in [\frac{\sqrt{21}}{3}, \frac{5+\sqrt{73}}{8}]$ can be verified by essentially similar arguments, hence we sketch the proof by listing the schedules of algorithm A and the adversarial sequences for all the possible situations, which are given case by case in the ensuing Tables 1-3.

Theorem 4.1 *For $s \in [\sqrt{2}, \frac{\sqrt{21}}{3}]$, any semi-online algorithm A for $Q2|opt|C_{\max}$has a competitive ratio of at least $\frac{3s+5}{2s+4}$.*

Proof. Let $p_1 = \frac{3-s^2}{2s+4}$. We first consider the case that p_1 is assigned to M_1. Let $p_2 = \frac{s^2+s-2}{2s+4}$. If p_2 is also assigned to M_1, let the last two jobs be $p_3 = 1$ and $p_4 = \frac{2s^2+3s-1}{2s+4}$. Then we have

$$C^A \ge \begin{cases} p_1+p_2+p_3 = \frac{3s+5}{2s+4}, & \text{if } p_3 \text{ is assigned to } M_1, \\ \min\{p_1+p_2+p_4,\ \frac{p_3+p_4}{s}\} = \min\{s,\ \frac{2s^2+5s+3}{s(2s+4)}\} \ge \frac{3s+5}{2s+4}, & \text{otherwise.} \end{cases}$$

If p_2 is assigned to M_2, let $p_3 = \frac{s+1}{2s+4}$. If further p_3 is assigned to M_1, let the last two jobs be $p_4 = p_5 = \frac{s^2+2s+1}{2s+4}$. We obtain

$$C^A \geq \begin{cases} p_1 + p_3 + p_4 = \frac{3s+5}{2s+4}, & \text{if } p_4 \text{ is assigned to } M_1, \\ \min\{p_1 + p_3 + p_5,\ \frac{p_2+p_4+p_5}{s}\} = \frac{3s+5}{2s+4}, & \text{otherwise.} \end{cases}$$

If p_3 is assigned to M_2, let the last two jobs be $p_4 = s$ are $p_5 = \frac{2}{2s+4}$. We also have

$$C^A \geq \min\{p_1 + p_4,\ \frac{p_2 + p_3 + p_4}{s}\} = \min\{\frac{s^2 + 4s + 3}{2s + 4},\ \frac{3s^2 + 6s + 1}{2s + 4}\} \geq \frac{3s + 5}{2s + 4}.$$

Now we consider the case that A assigns p_1 to M_2. In this case, let $p_2 = \frac{4+s-s^2}{2s+4}$. If p_2 is assigned to M_2, let the last two jobs be $p_3 = s$ and $p_4 = \frac{2s^2+s-3}{2s+4}$. We have

$$C^A \geq \min\{p_3,\ \frac{p_1 + p_2 + p_3}{s}\} = \min\{s,\ \frac{5s + 7}{s(2s + 4)}\} \geq \frac{3s + 5}{2s + 4}.$$

If p_2 is assigned to M_1, let $p_3 = \frac{s^2+2s+1}{2s+4}$ and $p_4 = \frac{3s^2+3s-4}{2s+4}$. We have

$$C^A \geq \begin{cases} p_2 + p_3 = \frac{3s+5}{2s+4}, & \text{if } p_3 \text{ is assigned to } M_1, \\ \min\{p_1 + p_4,\ \frac{p_2+p_3+p_4}{s}\} = \min\{s,\ \frac{3s+5}{2s+4}\} \geq \frac{3s+5}{2s+4}, & \text{otherwise.} \end{cases}$$

We are done. □

Theorem 4.2 *For $s \in [\frac{\sqrt{21}}{3}, \frac{5+\sqrt{193}}{12}]$, any semi-online algorithm A for $Q2|opt|C_{\max}$ has a competitive ratio of at least $\frac{3s+3}{3s+1}$.*

Proof. Consider Table 1. It is easy to verify that all the values in the last column of the table are greater than or equal to $\frac{3s+3}{3s+1}$ for any $s \in [\frac{\sqrt{21}}{3}, \frac{5+\sqrt{193}}{12}]$. The theorem is thus proved. □

Theorem 4.3 *For $s \in [\frac{5+\sqrt{193}}{12}, \frac{5}{3}]$, any semi-online algorithm A for $Q2|opt|C_{\max}$ has a competitive ratio of at least*

$$\begin{cases} \frac{4s+2}{2s+3}, & \frac{5+\sqrt{193}}{12} \leq s \leq \frac{7+\sqrt{145}}{12}, \\ \frac{5s+2}{4s+1}, & \frac{7+\sqrt{145}}{12} \leq s \leq \frac{9+\sqrt{193}}{14}, \\ \frac{7s+4}{7s}, & \frac{9+\sqrt{193}}{14} \leq s \leq \frac{5}{3}. \end{cases}$$

Proof. Consider Table 2. If $s \in [\frac{5+\sqrt{193}}{12}, \frac{7+\sqrt{145}}{12}]$, we set

$$x = \frac{-2s^2 + 3s + 1}{2s + 3},\quad y = \frac{4s^2 - 4s - 1}{2s + 3},\quad z = \frac{2s^2 - s - 2}{2s + 3},\quad w = \frac{2}{2s + 3}.$$

Substituting these values into the expressions in the last column in Table 2, we have

$$\begin{aligned}
&\min\{1 + s - x, y + 1, s - x, \tfrac{1+s-y}{s}\} = \min\{\tfrac{4s^2+2s+2}{2s+3}, \tfrac{4s^2-2s+2}{2s+3}, \tfrac{4s^2-1}{2s+3}, \tfrac{-2s^2+9s+4}{s(2s+3)}\} = \tfrac{-2s^2+9s+4}{s(2s+3)}, \\
&\min\{1 + s - x - y, s, \tfrac{x+y+s}{s}, \tfrac{1+s}{s}\} = \min\{\tfrac{6s+3}{2s+3}, s, \tfrac{4s+2}{2s+3}, \tfrac{1+s}{s}\} = \tfrac{4s+2}{2s+3}, \\
&\min\{s + 1, x + z + 1, \tfrac{1+s-x-z}{s}, s\} = \min\{s + 1, \tfrac{4s+2}{2s+3}, \tfrac{2s^2+3s+4}{s(2s+3)}, s\} = \tfrac{4s+2}{2s+3}, \\
&\min\{1 + s - z, x + w + 1, s - z, \tfrac{1+s-x-w}{s}\} = \min\{\tfrac{6s+5}{2s+3}, \tfrac{-2s^2+5s+6}{2s+3}, \tfrac{4s+2}{2s+3}, \tfrac{4s+2}{2s+3}\} = \tfrac{4s+2}{2s+3}, \\
&\min\{1 + s - z - w, s + x, \tfrac{s+z+w}{s}, \tfrac{1+s-x}{s}\} = \min\{\tfrac{6s+3}{2s+3}, \tfrac{6s+1}{2s+3}, \tfrac{4s+2}{2s+3}, \tfrac{4s^2+2s+2}{s(2s+3)}\} = \tfrac{4s+2}{2s+3}.
\end{aligned}$$

Schedule by A		Adversary sequence	C^A
M_1	M_2		
$\{p_1, p_2, p_3, p_4\}$	$\emptyset$	$\{\frac{s-1}{3s+1}, \frac{3s^2-s-4}{3s+1}, 1, \frac{s+5}{3s+1}\}$	$s+1$
$\{p_1, p_2, p_3\}$	$\{p_4\}$		$\frac{3s^2+3s-4}{3s+1}$
$\{p_1, p_2, p_4\}$	$\{p_3\}$		s
$\{p_1, p_2\}$	$\{p_3, p_4\}$		$\frac{4s+6}{s(3s+1)}$
$\{p_1, p_3, p_4, p_5\}$	$\{p_2\}$	$\{\frac{s-1}{3s+1}, \frac{3s^2-s-4}{3s+1}, \frac{2}{3s+1}, \frac{2s+2}{3s+1}, \frac{2s+2}{3s+1}\}$	$\frac{5s+5}{3s+1}$
$\{p_1, p_3, p_4\}$	$\{p_2, p_5\}$		$\frac{3s+3}{3s+1}$
$\{p_1, p_3, p_5\}$	$\{p_2, p_4\}$		
$\{p_1, p_3\}$	$\{p_2, p_4, p_5\}$		
$\{p_1, p_4, p_5\}$	$\{p_2, p_3\}$	$\{\frac{s-1}{3s+1}, \frac{3s^2-s-4}{3s+1}, \frac{2}{3s+1}, s, \frac{-3s^2+3s+4}{3s+1}\}$	$\frac{5s+3}{3s+1}$
$\{p_1, p_4\}$	$\{p_2, p_3, p_5\}$		$\frac{3s^2+2s-1}{3s+1}$
$\{p_1, p_5\}$	$\{p_2, p_3, p_4\}$		$\frac{6s^2-2}{s(3s+1)}$
$\{p_1\}$	$\{p_2, p_3, p_4, p_5\}$		$\frac{3s^2+3s+2}{s(3s+1)}$
$\{p_2, p_3, p_4\}$	$\{p_1\}$	$\{\frac{s-1}{3s+1}, \frac{s+1}{3s+1}, \frac{2s+2}{3s+1}, \frac{3s^2-1}{3s+1}\}$	$\frac{3s^2+3s+2}{3s+1}$
$\{p_2, p_4\}$	$\{p_1, p_3\}$		s
$\{p_2, p_3\}$	$\{p_1, p_4\}$		$\frac{3s+3}{3s+1}$
$\{p_2\}$	$\{p_1, p_3, p_4\}$		
$\{p_3, p_4\}$	$\{p_1, p_2\}$	$\{\frac{s-1}{3s+1}, \frac{s+1}{3s+1}, s, \frac{s+1}{3s+1}\}$	$\frac{3s^2+2s+1}{3s+1}$
$\{p_3\}$	$\{p_1, p_2, p_4\}$		s
$\{p_4\}$	$\{p_1, p_2, p_3\}$		$\frac{3s+3}{3s+1}$
$\emptyset$	$\{p_1, p_2, p_3, p_4\}$		$\frac{s+1}{s}$

Table 1: The case $s \in [\frac{\sqrt{21}}{3}, \frac{5+\sqrt{193}}{12}]$ for Theorem 4.2

Schedule by A		Adversary sequence	C^A
M_1	M_2		
$\{p_2, p_3, p_4\}$	$\{p_1\}$	$\{x, y, 1, s-x-y\}$	$1+s-x$
$\{p_2, p_3\}$	$\{p_1, p_4\}$		$y+1$
$\{p_2, p_4\}$	$\{p_1, p_3\}$		$s-x$
$\{p_2\}$	$\{p_1, p_3, p_4\}$		$\frac{1+s-y}{s}$
$\{p_3, p_4\}$	$\{p_1, p_2\}$	$\{x, y, s, 1-x-y\}$	$1+s-x-y$
$\{p_3\}$	$\{p_1, p_2, p_4\}$		s
$\{p_4\}$	$\{p_1, p_2, p_3\}$		$\frac{x+y+s}{s}$
$\emptyset$	$\{p_1, p_2, p_3, p_4\}$		$\frac{1+s}{s}$
$\{p_1, p_2, p_3, p_4\}$	$\emptyset$	$\{x, z, 1, s-x-z\}$	$s+1$
$\{p_1, p_2, p_3\}$	$\{p_4\}$		$x+z+1$
$\{p_1, p_2\}$	$\{p_3, p_4\}$		$\frac{1+s-x-z}{s}$
$\{p_1, p_2, p_4\}$	$\{p_3\}$		s
$\{p_1, p_3, p_4, p_5\}$	$\{p_2\}$	$\{x, z, w, 1, s-x-w-z\}$	$1+s-z$
$\{p_1, p_3, p_4\}$	$\{p_2, p_5\}$		$x+w+1$
$\{p_1, p_3, p_5\}$	$\{p_2, p_4\}$		$s-z$
$\{p_1, p_3\}$	$\{p_2, p_4, p_5\}$		$\frac{1+s-x-w}{s}$
$\{p_1, p_4, p_5\}$	$\{p_2, p_3\}$	$\{x, z, w, s, 1-x-z-w\}$	$1+s-z-w$
$\{p_1, p_4\}$	$\{p_2, p_3, p_5\}$		$s+x$
$\{p_1, p_5\}$	$\{p_2, p_3, p_4\}$		$\frac{s+z+w}{s}$
$\{p_1\}$	$\{p_2, p_3, p_4, p_5\}$		$\frac{1+s-x}{s}$

Table 2: The case $\frac{5+\sqrt{193}}{12} \leq s \leq \frac{5}{3}$ for Theorem 4.3

$\{p_1, p_4, p_5, p_6\}$	$\{p_2, p_3\}$	$\{x, z, w, v, 1, s-x-z-w-v\}$	$1+s-z-w$
$\{p_1, p_4, p_5\}$	$\{p_2, p_3, p_6\}$		$x+v+1$
$\{p_1, p_4, p_6\}$	$\{p_2, p_3, p_5\}$		$s-z-w$
$\{p_1, p_4\}$	$\{p_2, p_3, p_5, p_6\}$		$\frac{1+s-x-v}{s}$
$\{p_1, p_5, p_6\}$	$\{p_2, p_3, p_4\}$	$\{x, z, w, v, s, 1-x-z-w-v\}$	$1+s-z-w-v$
$\{p_1, p_5\}$	$\{p_2, p_3, p_4, p_6\}$		$s+x$
$\{p_1, p_6\}$	$\{p_2, p_3, p_4, p_5\}$		$\frac{s+z+w+v}{s}$
$\{p_1\}$	$\{p_2, p_3, p_4, p_5, p_6\}$		$\frac{1+s-x}{s}$

Table 3: The case $\frac{5}{3} \leq s \leq \frac{5+\sqrt{73}}{8}$ for Theorem 4.4

Hence we have

$$C^A \geq \min\{\frac{-2s^2+9s+4}{s(2s+3)}, \frac{4s+2}{2s+3}\} = \frac{4s+2}{2s+3}.$$

Similarly, by setting

$$x = \frac{2s^2-2s-1}{4s+1}, \;\; y = \frac{-s^2+3s+1}{4s+1}, \;\; z = \frac{4s^2-4s-2}{4s+1}, \;\; w = \frac{-3s^2+5s+2}{4s+1}$$

if $s \in [\frac{7+\sqrt{145}}{12}, \frac{9+\sqrt{193}}{14}]$, and setting

$$x = \frac{1}{7}, \;\; y = \frac{3}{7}, \;\; z = w = \frac{2}{7}$$

if $s \in [\frac{9+\sqrt{193}}{14}, \frac{5}{3}]$, we can get the desired lower bounds for these intervals, too. □

Theorem 4.4 *For $s \in [\frac{5}{3}, \frac{5+\sqrt{73}}{8}]$, any semi-online algorithm A for $Q2|opt|C_{\max}$ has a competitive ratio of at least $\frac{7s+4}{4s+5}$.*

Proof. Replace the last four rows in Table 2 with all the rows in Table 3, and set

$$x = \frac{-2s^2+4s+1}{4s+5}, \;\; y = \frac{5s^2-5s-1}{4s+5}, \;\; z = w = \frac{2s^2-s-2}{4s+5}, v = \frac{-s^2+s+4}{4s+5}.$$

By a similar argument as that in the proof of Theorem 4.3, we can reach the conclusion. □

Combining Theorems 3.1-3.2 and 4.1-4.4, we have improved the known results for $Q2|opt|C_{\max}$ as follows: we have decreased the largest gap between the competitive ratio and the lower bound from 0.07295 to 0.02192, and the length of the interval over which the algorithm is not optimal from 0.4987 to 0.46814. Figure 4 shows the competitive ratios of algorithm $COMBINE1$ and the lower bounds for the problem.

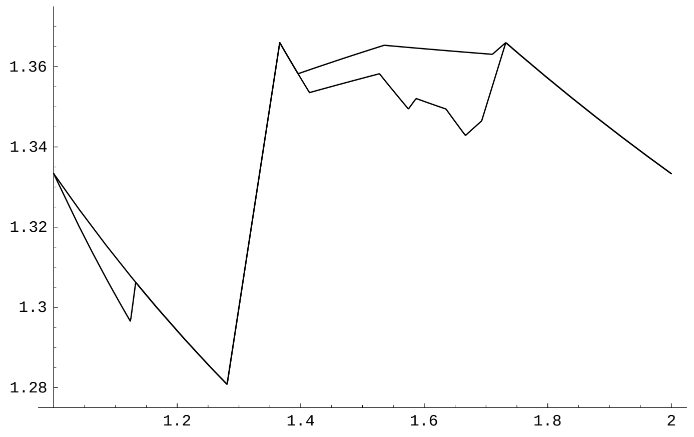

Figure 4: The competitive ratio of algorithm $COMBINE1$ and the lower bound.

4.2 Lower bound for $Q2|sum|C_{\max}$

Finally, we give a lower bound for $Q2|sum|C_{\max}$ in this subsection.

Theorem 4.5 *Any semi-online algorithm A for $Q2|sum|C_{\max}$ has a competitive ratio of at least*

$$\begin{cases} \frac{3s+1}{3s}, & \text{for } 1 \le s \le q_1 \approx 1.12433, \\ (\frac{3}{4}+\frac{\sqrt{65}}{20})s, & \text{for } q_1 \le s \le \frac{1+\sqrt{65}}{8} \approx 1.13278, \\ \frac{2s+2}{2s+1}, & \text{for } \frac{1+\sqrt{65}}{8} \le s \le \frac{1+\sqrt{17}}{4}, \\ s, & \text{for } \frac{1+\sqrt{17}}{4} \le s \le \frac{1+\sqrt{3}}{2} \approx 1.36603, \\ \frac{2s+1}{2s}, & \text{for } \frac{1+\sqrt{3}}{2} \le s \le \sqrt{2} \approx 1.41421, \\ \frac{3s+5}{2s+4}, & \text{for } \sqrt{2} \le s \le \frac{\sqrt{21}}{3} \approx 1.52753, \\ \frac{3s+3}{3s+1}, & \text{for } \frac{\sqrt{21}}{3} \le s \le \frac{5+\sqrt{193}}{12} \approx 1.57437, \\ \frac{4s+2}{2s+3}, & \text{for } \frac{5+\sqrt{193}}{12} \le s \le \frac{7+\sqrt{145}}{12} \approx 1.5868, \\ \frac{5s+2}{4s+1}, & \text{for } \frac{7+\sqrt{145}}{12} \le s \le \frac{9+\sqrt{193}}{14} \approx q_5, \\ c(s), & \text{for } q_5 \le s \le \sqrt{3}, \\ \frac{s+2}{s+1}, & \text{for } s \ge \sqrt{3}, \end{cases}$$

where $c(s) = \frac{\sqrt{s^2x^2+4s(s+1-x)}+sx}{2s}$, *$x$ is a root of the equation*

$$\frac{\sqrt{s^2x^2+4s(s+1-x)}+sx}{2s} = \frac{s(2s+2-x)}{(s+1)(x+2)},$$

and q_5 is defined in Section 1.

Proof. For $s \in [1, q_5] \cup [\sqrt{3}, \infty)$, the lower bound for $Q2|sum|C_{\max}$ is the same as that for the problem $Q2|opt|C_{\max}$. In fact, for $\sqrt{2} \le s \le q_5$, since all the sequences in the proof of Theorems 4.1-4.4 have the same total sum of sizes of $1+s$, we know that the lower bound remains valid. For $1 \le s \le \sqrt{2}$ and $s \ge \sqrt{3}$, the sequences used in [5] may have a total sum of less than $1+s$ although $C^{OPT} = 1$. If so, we can add a sufficient number of small jobs at the end of each such sequence such that the total sum becomes $1+s$ and $C^{OPT} = 1$ still holds, which suffices to get the same lower bound.

We consider the case of $s \in [q_5, \sqrt{3}]$ as follows: Let $y = \frac{\sqrt{s^2x^2+4s(s+1-x)}-sx}{2s} = c(s) - x$ be the positive root of the equation $x+y = \frac{1+s-x}{sy}$, and let $z = 1+\frac{sx-x^2}{s+2}$ be the root of $z+\frac{1+s-x-z}{2} = \frac{1+s-z}{s}$. It can be verified that for $s \in [q_5, \sqrt{3}]$, the following inequalities are satisfied:

$$x+y < s, \quad x+z < 1, \tag{13}$$

$$1+s-y > \frac{1+s-x}{s}, \tag{14}$$

$$y > 1, \tag{15}$$

$$\frac{x+z+s}{s} > \frac{s(2s+2-x)}{(s+1)(x+2)} = c(s), \tag{16}$$

$$x + z > s - 1. \tag{17}$$

Note that (13) guarantees that all the job sizes in the below sequence are positive.

Let $p_1 = x$. We first consider the case that A assigns p_1 to M_1. Then let $p_2 = y$. If p_2 is assigned to M_1, let $p_3 = 1$ and $p_4 = s - x - y$. We have $C^A \geq x + y$ and $C^{OPT} = 1$. It follows that $\frac{C^A}{C^{OPT}} \geq x + y = c(s)$. If p_2 is assigned to M_2, let $p_3 = 1 + s - x - y$. We have

$$C^A \geq \min\{p_1 + p_3, \frac{p_2 + p_3}{s}\} = \min\{1 + s - y, \ \frac{1 + s - x}{s}\} = \frac{1 + s - x}{s},$$

where the last equality is due to (14). On the other hand, the optimal makespan must be no greater than the makespan of the following feasible schedule: assign p_2 to M_1 and the remaining two jobs to M_2. It follows that

$$C^{OPT} \leq \max\{p_2, \frac{p_1 + p_3}{s}\} = \max\{y, \frac{1 + s - y}{s}\} = y,$$

where the last equality holds because of (15). Thus we have

$$\frac{C^A}{C^{OPT}} \geq \frac{1 + s - x}{sy} = x + y = c(s).$$

Now we consider the case that A assigns p_1 to M_2. Let $p_2 = z$. If p_2 is assigned to M_2, let $p_3 = s$ and $p_4 = 1 - x - z$. Since $\frac{x+z+s}{s} < \frac{s+1}{s} < s$ for $s \geq q_5$, we have

$$C^A \geq \min\{p_3, \frac{p_1 + p_2 + p_3}{s}\} = \min\{s, \frac{x + z + s}{s}\} = \frac{x + z + s}{s},$$

while $C^{OPT} = 1$ holds trivially. Hence, by (16) we have

$$\frac{C^A}{C^{OPT}} \geq \frac{x + z + s}{s} > c(s).$$

If p_2 is assigned to M_1, let $p_3 = p_4 = \frac{1+s-x-z}{2}$. We have

$$C^A = \min\{p_2 + p_3, \frac{p_1 + p_3 + p_4}{s}\} = \min\{z + \frac{1 + s - x - z}{2}, \ \frac{1 + s - z}{s}\} = \frac{1 + s - x + z}{2} = \frac{2s + 2 - x}{s + 2}.$$

On the other hand, the optimal makespan must be no greater than the makespan of the following feasible schedule: assign p_4 to M_1 and the remaining three jobs to M_2. It follows that

$$C^{OPT} \leq \max\{p_4, \frac{p_1 + p_2 + p_3}{s}\} = \frac{p_1 + p_2 + p_3}{s} = \frac{1}{s}\left(x + z + \frac{1 + s - x - z}{2}\right) = \frac{(s+1)(x+2)}{s(s+2)},$$

where the first equality holds, because $p_4 < 1$ and $p_1 + p_2 + p_3 > s$ (due to (17)). We thus have

$$\frac{C^A}{C^{OPT}} \geq \frac{s(2s + 2 - x)}{(s+1)(x+2)} = c(s).$$

□

Through Theorems 3.3-3.5 and 4.5, we conclude that for $Q2|sum|C_{\max}$ the largest gap between the competitive ratio of $COMBINE2$ and the lower bound is about 0.01762, and the length of the interval over which $COMBINE2$ is not optimal is about 0.47328. Figure 5 shows the competitive ratios of algorithm $COMBINE2$ and the lower bounds.

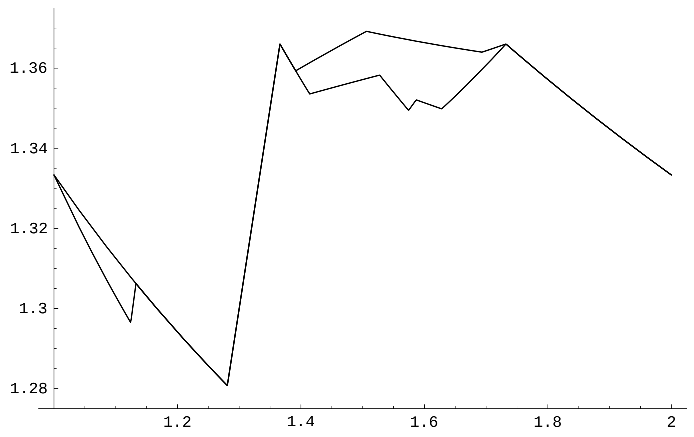

Figure 5: The competitive ratio of algorithm $COMBINE2$ and the lower bound.

Acknowledgement

We are thankful to an anonymous referee for their helpful comments on an earlier version of our paper.

References

[1] E. Angelelli, A. B. Nagy, M. G. Speranza, Z. Tuza, The on-line multiprocessor scheduling problem with known sum of the tasks, *Journal of Scheduling*, 7, 2004, 421-428.

[2] Y. Azar, O. Regev, On-line bin-stretching, *Theoretical Computer Science*, 168, 2001, 17-41.

[3] R. E. Burkard, Y. He, H. Kellerer, A linear compound algorithm for uniform machine scheduling, *Computing*, 61, 1998, 1–9.

[4] T. C. E. Cheng, H. Kellerer, V. Kotov, Semi-on-line multiprocessor scheduling with given total processing time, *Theoretical Computer Science*, 337, 2005, 134-146.

[5] L. Epstein, Bin stretching revisited, *Acta Informatica*, 39, 2003, 97-117.

[6] Y. He, H. Kellerer, V. Kotov, Linear compound algorithms for the partitioning problem, *Naval Research Logistics*, 47 2000, 593–601.

[7] H. Kellerer, V. Kotov, M. G. Speranza and Z. Tuza, Semi on-line algorithms for the partition problem, *Operations Research Letters*, 21, 1997, 235-242.

[8] Z. Y. Tan, Y. He, Semi-online scheduling on two uniform machines, *System Engineering-Theorey and Practice*, 21, 2001, 53-57. (in Chinese)

[9] Z. Y. Tan, Y. He, Semi-on-line problems on two identical machines with combined partial information, *Operations Research Letters*, **30**, 408-414, 2002.