Online and Semi-online Scheduling*

Zhiyi Tan and An Zhang

Contents

*Supported by the National Natural Science Foundation of China (10971191, 11271324), Zhejiang Provincial Natural Science Foundation of China (LR12A01001), and Fundamental Research Funds for the Central Universities.

Z. Tan (✉)
Department of Mathematics, Zhejiang University, Hangzhou, People's Republic of China
e-mail: tanzy@zju.edu.cn

A. Zhang
Institute of Operational Research and Cybernetics, Hangzhou Dianzi University, Hangzhou, People's Republic of China
e-mail: anzhanghz@yahoo.com.cn

P.M. Pardalos et al. (eds.), *Handbook of Combinatorial Optimization*,
DOI 10.1007/978-1-4419-7997-1_2, © Springer Science+Business Media New York 2013

Abstract

During the last two decades, online and semi-online scheduling problems have received considerable research interest. This chapter provides definitions of several online paradigms, including online over list, online over time, and semi-online. This chapter presents details of the basic online algorithms and referencing other significant results.

1 Introduction

1.1 Scheduling

Scheduling is the earliest studied branch in combinatorial optimization and also one of the problems with the most fruitful results achieved. Early scheduling problems are motivated from manufacturing systems. Later, more and more scheduling problems are formalized in the areas of information science, supply chain management, public management, and so on. In the following, we summarize the basic concepts and notations which will be used later. For more detailed introduction on scheduling theory, please refer to [21, 36, 117, 143].

Scheduling problems can be generally described as follows. There is a sequence $\mathcal{J}$ of n jobs $J_1, J_2, \cdots, J_n$, which need to be processed on a set $\mathcal{M}$ of m machines $M_1, M_2, \cdots, M_m$. A *schedule* is an assignment for each job to one or more time intervals of one or more machines. Schedules that satisfy various requirements of the problem are called *feasible*. Scheduling problems are specified by the machine environment, the job characteristics, and an optimality criterion.

There are two basic types of machine systems: single-stage system and multi-stage system. In a single-stage system, each job requires one operation, while in multiple-stage system the jobs may require several operations at different stages. There are also two classical single-stage systems: single machine and parallel machines. In a single machine system, job J_j must be processed on the unique machine M_1 with a *processing time* $p_j, j = 1, 2, \cdots, n$. In a parallel machines system, each job can be processed by *one* of the m machines. The time needed for job J_j to be processed on M_i is $p_{ji}, j = 1, \cdots, n, i = 1, \cdots, m$. If $p_{ji} = p_j$ for any $i = 1, \cdots, m$ and $j = 1, \cdots, n$, then the machine system is called *identical*. If there exist $s_i, i = 1, \cdots, m$ such that $\frac{p_{ji_1}}{p_{ji_2}} = \frac{s_{i_2}}{s_{i_1}}$ for any $j = 1, \cdots, n$, then the machine system is called *uniform*, and s_i is called the *speed* of $M_i, i = 1, \cdots, m$. W.l.o.g., we assume $s_1 \geq s_2 \geq \cdots \geq s_m = 1$ and let $p_j = p_{jm}$, which is the normalized *processing time* of job J_j. In the case of two uniform machines, it is of more convenience for discussion to replace the speed parameters s_1 and $s_2 (= 1)$ by the *speed ratio* $s = \frac{s_1}{s_2}$, and both are equivalent. Parallel machines system that does not fall into above categories is called *unrelated*. In a multi-stage system, each operation of a job has a separate processing time, and different operations of the same job cannot be scheduled at the same time. The multi-stage system is usually distinguished into *open shop*, when different operations of the same job may be scheduled in any order; *flow shop*, if the order of the operations is fixed and same for all the jobs; and *job shop*, if the order of the operations is fixed and possibly different for each job.

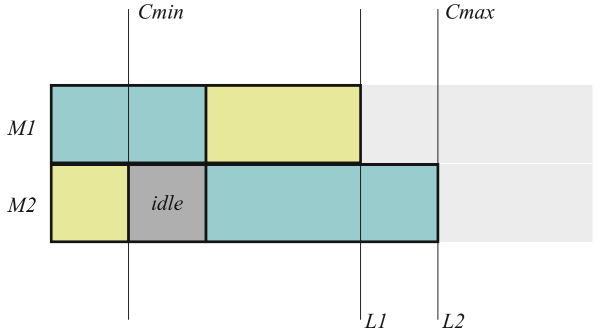
Apart from the processing time of the jobs, the processing mode can also be classified into the job characteristics. Some scheduling models allow *preemption* of jobs, that is, the processing of a job may be interrupted and resumed later on possibly a different machine. If each job can be arbitrarily split between the machines and parts of the same job can run on different machines in parallel, this model is called *fractional assignment*. There is another possibility allowing jobs to *restart*, which means that a processing job can be stopped and restarted later to finish the full processing time.

Given a schedule σ, let $C_j(\sigma)$ (or C_j if there is no confusing) be the completion time of J_j, $j = 1, 2, \cdots, n$. Some commonly used objective functions include:
The makespan: $\max_{1 \le j \le n} C_j(\sigma)$.
The total completion time: $\sum_{j=1}^{n} C_j(\sigma)$,
The total weighted completion time: $\sum_{j=1}^{n} w_j C_j(\sigma)$, where w_j is the weight of J_j.
All these objectives are for minimization problems. There is another criteria which maximizes the continuous period of time when all machines are busy. Such criteria have applications in the sequencing of maintenance actions for modular gas turbine aircraft engines [49] and fairly allocation of indivisible goods [15]. Problems with this objective are often called *machine covering*.

The period when one machine is not assigned to any job during the scheduling process is called *idle* time. For most non-preemptive problems, there is no benefit to introduce idle times. But idle time is of great help if preemption is allowed, especially for uniform machines scheduling. Let L_i be the completion time of machine M_i in schedule σ. The makespan of σ equals to $\max_{i=1,\cdots,m} L_i$. However, $\min_{i=1,\cdots,m} L_i$ is not always identical with the objective value of the machine covering problem if idle times exist (Fig. 1).

In general, scheduling problems are specified in terms of a three-field notation [88]. Here, we adopt the notation from it. In this chapter, we will use $1, P, Q, R$ to denote single, identical, uniform, unrelated machine system and F, O, J to denote flow shop, open shop, and job shop system. Problems allowing preemption, fractional assignment, and restart will be denoted by *pmtn*, *frac*, and *res*, respectively. The objectives of minimizing makespan, minimizing the total completion time, and minimizing the total weighted completion time will be shortly denoted by C_{max}, $\sum C_j$, and $\sum w_j C_j$, respectively. The objective of machine covering is denoted C_{min}.

Fig. 1 Machine completion times and objective value of a schedule



1.2 Online Problems and Competitive Analysis

Online problems began to draw attention in the middle period of the eighties of the twentieth century. The basic character of online models is “lack of information,” while the previously studied problems having the access to full information are called *offline*. For more detailed introduction of online theories and variants, please refer to [20, 75].

For scheduling problems, there are a variety of online models, among which the following two are the most common.

Online over list: Jobs are ordered in a list and arrived one by one (at time 0). The information of a job is known to the scheduler only when all the jobs before it have been assigned. And the assignment of jobs is irrevocable.

Online over time: Each job has a *release time* after which the information of the job is known and can be processed. The release time of J_j is denoted by r_j, $j = 1, 2, \cdots, n$.

Since almost all problems considered in this chapter are online, we will ignore *online over list* in the second field of the three-field notation. On the other hand, r_j will be included in the second field of the three-field notation for the online over time model.

The *competitive analysis* is the most common method in analysis of online algorithms, which is formally introduced by Sleator and Tarjan [161]. The performance of an online algorithm is measured by its *competitive ratio*, which has some similarity as *worst-case ratio* for offline problems. For a job sequence $\mathcal{J}$ and an online algorithm A, if A produces a schedule with objective value $C^A(\mathcal{J})$ for the minimization (or maximization) problem and the optimal offline objective value is $C^*(\mathcal{J})$, then the competitive ratio of A is defined as

$$r_A = \inf_{\mathcal{J}}\{r | C^A(\mathcal{J}) \le rC^*(\mathcal{J})\} \;\; (or \; r_A = \inf_{\mathcal{J}}\{r | C^*(\mathcal{J}) \le rC^A(\mathcal{J})\}).$$

If there is an online algorithm A for the problem with a competitive ratio r, then A is called an r-*competitive* algorithm. On the contrary, if no online algorithm has a competitive ratio smaller than ρ, then the problem must have a *lower bound* ρ. Therefore, an online algorithm is called *optimal* or *best possible* for the problem if its competitive ratio just matches the lower bound. Here, optimal does not mean that it always produces an optimal schedule. It only indicates the fact that no online algorithm can be better. An online algorithm is stated to possess competitive ratio 1 if it always produces an optimal schedule, but such algorithm seldom exists. In this chapter, we use *optimal bound* to denote the competitive ratio of the optimal algorithm of the problem for simplicity. Symmetrically, a problem has *upper bound* r if there exists an online algorithm for the problem with competitive ratio r.

A *randomized algorithm* is one which somehow bases its decision making on the outcome of random coin flips. For a randomized algorithm, the objective value $C^A(\mathcal{J})$ is a random variable. Thus, the competitive ratio of a randomized algorithm (for a minimization problem) is defined as

$$r_A = \inf_{\mathcal{J}}\{r | E(C^A)(\mathcal{J}) \leq rC^*(\mathcal{J})\},$$

where the expectation is taken over the random choices of the algorithm. In addition, concepts of randomized upper and lower bound can also be defined accordingly. Algorithms which do not use randomization are called *deterministic algorithm*. Since deterministic algorithm can be viewed as a special randomized algorithm, the randomized lower bound cannot be larger than deterministic lower bound. In this chapter, without special mention, all algorithms (competitive ratio, lower bound) are deterministic. For a general introduction on randomized algorithms, please refer to [134].

1.3 Structure and Notation

Online scheduling boasts of extensive results; thus, it is impossible to cover all the problems in one chapter, and only part of paradigms together with the according main results is included. There have been two excellent surveys [144,156] concerning online scheduling, and some paradigms discussed elaborately there will be omitted. We will mainly introduce paradigms approaching classical scheduling problems and withal active in recent research. There are several interesting and important problems not covered due to limited space and number of references, and we will sketch some of them in the following. Firstly, scheduling problems with deadline constraints, including interval scheduling, are not included in this chapter. Secondly, it does not cover problems concerning other objective functions, for example, the (weighted) total flow time and the L_p norm of machine completion times, together with objectives deriving from bicriteria situations, for example, scheduling with rejection and scheduling with machine cost. Thirdly, performance measures other than competitive analysis, including resource argumentation, will not be contained in this chapter. Finally, we skip problems which have specific applications, such as scheduling problems in power management, broadcasting, and supply chain management.

The chapter is organized as follows. In Sect. 2, we survey main results for the online over list model. Section 3 is devoted to semi-online variants of online over list model. In Sect. 4, we introduce online and semi-online problems of the online over time model. Other variants of online scheduling are contained in Sect. 5.

The following notations as well as those defined above will be used frequently in the reminder of this chapter. For any $i = 1, \cdots, m$, let $S_i = \sum_{l=1}^{i} s_l$ be the total speeds of the first i machines in uniform machine system, and we simplify S_m as S. For any $j = 1, \cdots, n$, let $\mathcal{J}_j$ be the subset of $\mathcal{J}$ containing the first j jobs and P_j be the total processing times of the first j jobs. We simplify P_n as P. Let $p_{(j)}$ be the processing time of the job which has the jth largest processing time, that is, $p_{(1)} \geq p_{(2)} \geq \cdots \geq p_{(n)}$. For any $i = 1, \cdots, m$ and $j = 1, \cdots, n$, let L_i^j be the completion time of machine M_i after the job J_j is assigned. L_i^j is also called the *load* of M_i if no idle time exists. Clearly, L_i^n equals to L_i.

2 Online Over List

This section summarizes main results for online over list model on parallel machines. In this paradigm, the makespan, which is the most common objective function with no doubt, will be considered throughout this section except the last subsection.

2.1 Non-preemptive Scheduling

2.1.1 Identical Machines

$Pm||C_{max}$ is the most studied and historical problem for online scheduling. In his epoch making paper, Graham [86] designed the first approximation algorithm in combinatorial optimization called *List Scheduling* (LS for short). LS uses greedy idea and has a very simple structure. It always assigns the current job to the machine where it can be started to process the earliest, that is, the machine with the smallest current load, ties are broken arbitrary. Such principle of assigning jobs is sometimes called *LS rule* in the literature. Graham proved the following result.

Theorem 1 ([86]) *The competitive ratio of LS for $Pm||C_{max}$ is $2 - \frac{1}{m}$.*

Proof (Sketch) Without loss of generalization, the last job J_n determines the makespan. Let s_n be the start time of J_n. By LS rule, no machine will idle during $[0, s_n]$. Hence, $s_n \leq \frac{P-p_n}{m}$. Clearly,

$$C^* \geq \frac{P}{m} \tag{1}$$

and

$$C^* \geq \max_{1 \leq j \leq n} p_j. \tag{2}$$

Hence,

$$C^{LS} = s_n + p_n \leq \frac{P - p_n}{m} + p_n = \frac{P}{m} + \frac{m-1}{m} p_n \leq C^* + \frac{m-1}{m} C^* = \left(2 - \frac{1}{m}\right) C^*.$$

□

LS is a typical online algorithm. When assigning a new job, it does not use any information of jobs that have not arrived yet. Not only the rule of LS but also the ideas behind the proof are frequently used in design and analysis of other online algorithms. Twenty years have passed since the appearance of LS, and online problems are drawing more and more attentions. Feigle et al. [74] proved that LS is the optimal algorithm for $Pm||C_{max}$ when $m = 2, 3$. It should be emphasized that no other algorithm has been proved to be optimal for any $m \geq 4$ up till now.

Though LS is irreplaceable in online scheduling, it is gradually recognized that LS cannot be the optimal algorithm for $Pm||C_{max}$ when $m \geq 4$. Hence, algorithms and lower bounds have been refined bit by bit, and brief results and history are summarized in Table 1. In these new algorithms, jobs are not always assigned to the least loaded machine. Sometimes, the current job may be assigned to the second smallest or the machine whose load lies approximately in middle of all machines. Almost all these algorithms contain some parameters which have been carefully selected, and the analysis of the algorithm is much more involved. The instances used to prove the lower bound are likewise very sophisticated. The current best lower bound is obtained even by exhaustive search using computers.

It seems that improvement of the algorithm could be carried on since there is still a gap 0.066 between the current best upper and lower bounds. However, there is a turning point when Albers [2] proposed her important results. When proving the competitive ratio of an online algorithm, it is in fact that some *lower bounds* instead of the exact value of the optimum itself are used. For example, lower bounds

Table 1 Improvement of lower and upper bounds of $Pm||C_{max}$

Reference	Competitive ratio	Reference	Lower bound
Graham [86]	$2 - \frac{1}{m}$	Falgle et al. [74]	$\frac{1+\sqrt{2}}{2} (m \geq 4)$
Galambos and Woeginger [82]	$2 - \frac{1}{m} - \epsilon_m$[a]	Bartal et al. [17]	$1.837 (m \geq 3454)$
Bartal et al. [18]	$2 - \frac{1}{70} \approx 1.986$[b]	Albers [1]	$1.852 (m \geq 80)$
Karger et al. [110]	1.945	Gormley et al. [85]	1.85358[d]
Albers [1]	1.923		
Fleischer and Wahl [77]	1.9201[c]		

[a]$\epsilon_m \to 0 (m \to \infty)$, the first algorithm which beats LS for $m \geq 4$
[b]The first algorithm with competitive ratio strictly less than 2 for any m
[c]The best current known algorithm
[d]The best current known lower bound

(1) and (2) are used in the proof of Theorem 1. Another useful lower bound on the optimum is

$$C^* \geq 2p_{(m+1)}. \tag{3}$$

Theorem 2 ([2]) *If only using (1), (2) and (3) as lower bounds on the optimum, it is impossible to prove an online algorithm A has a competitive ratio less than* 1.919.

Note that the competitive ratio of the algorithm could be smaller than 1.919, but it cannot be proved using existing techniques. The bound 1.919 in Theorem 2 is very close to the current best upper bound 1.9201. Hence, it seems more urgent to find new lower bounds on the optimum than to designing more delicate algorithms. Though Theorem 2 does not give new lower bounds, it does point out the direction of future research. Such ideas can be adopted to other hard online scheduling problems as well.

When the number of machines is small, better lower bounds and algorithms can be obtained. For example, Chen et al. [31] presented an algorithm with competitive ratio at most

$$\max\left\{\frac{4m^2-3m}{2m^2-2}, \frac{2(m-1)^2+\sqrt{1+2m(m-1)}-1}{(m-1)^2+\sqrt{1+2m(m-1)}-1}\right\}$$

for $Pm||C_{max}$, $m \geq 4$. Rudin III and Chandrasekaran [147] proved the lower bound of $P4||C_{max}$ is at least $\sqrt{3}$, which is in line with the conjecture that the optimal bound of $P4||C_{max}$ is $\sqrt{3}$. Numerical bounds are summarized in Table 2.

2.1.2 Uniform and Unrelated Machines

The competitive analysis for $Qm||C_{max}$ is even more difficult than that for $Pm||C_{max}$. Both the competitive ratios of online algorithms and lower bounds of the problems are functions of machine speeds s_i, $i = 1, \cdots, m$. This might be the reason why optimal or tight bounds can only obtained for small values of m. For larger m or arbitrary number of machines, it is of more significance and practical that either to consider some special combinations of s_i or to obtain *overall* upper and lower bounds, that is, the maximal value among all possible choice of s_i, $i = 1, \cdots, m$ for both.

Since the machines have different speeds, the machine where a job can be started to process the earliest may not be the machine where it can be completed the earliest. Therefore, it is necessary to distinguish two variants of LS, namely, LSc and LSs. LSc and LSs assign the arriving job to the machine which the job can be completed or started the earliest, respectively. It is evident to see that for $Qm||C_{max}$, LSc is better than LSs.

The competitive ratio of LSc for $Qm||C_{max}$ is at most

$$\min_{1\leq k\leq m}\left\{\frac{\sum_{i=1}^{m} s_i + (k-1)s_1}{\sum_{i=1}^{k} s_i}\right\}$$

Table 2 Current best upper and lower bounds for online over list model on identical machines

	Non-preemptive				Preemptive
	Deterministic		Randomized		
m	Lower bounds	Upper bounds	Lower bounds	Upper bounds	Optimal bounds
2	$\frac{3}{2} = 1.5000$ [74]	$\frac{3}{2} = 1.5000$ [86]	$\frac{4}{3} \approx 1.3333$ [18]	$\frac{4}{3} \approx 1.3333$ [18]	$\frac{4}{3} \approx 1.3333$ [33]
3	$\frac{5}{3} \approx 1.6667$ [74]	$\frac{5}{3} = 1.5000$ [86]	$> \frac{27}{19}$ [174]	1.53727 [153]	$\frac{27}{19} \approx 1.42105$ [33]
4	$\sqrt{3} \approx 1.732$ [147]	$\frac{26}{15} \approx 1.7333$ [31]	$\frac{256}{175} \approx 1.46286$ [32, 155]	1.65669 [153]	$\frac{256}{175} \approx 1.46286$ [33]
5	1.74625 [31]	$\frac{85}{48} \approx 1.77083$ [31]	$\frac{3{,}125}{2{,}101} \approx 1.48739$ [32, 155]	1.73376 [151]	$\frac{3{,}125}{2{,}101} \approx 1.48739$ [33]
6	1.77301 [31]	$\frac{9}{5} = 1.8000$ [31]	$\frac{46{,}656}{31{,}031} \approx 1.50353$ [32, 155]	1.78295 [151]	$\frac{46{,}656}{31{,}031} \approx 1.50353$ [33]
7	1.79103 [31]	$\frac{175}{96} \approx 1.82292$ [31]	$\frac{823{,}543}{543{,}607} \approx 1.51496$ [32, 155]	1.81681 [151]	$\frac{823{,}543}{543{,}607} \approx 1.51496$ [33]
∞	1.85358 [85]	1.9201 [77]	$\frac{e-1}{e} \approx 1.582$ [32, 155]	1.916 [2]	$\frac{e-1}{e} \approx 1.582$ [33]

[90]. The overall competitive ratio is at most

$$\begin{cases} \frac{1+\sqrt{5}}{2}, & m=2, \\ \frac{2+\sqrt{2m-2}}{2}, & m \geq 3, \end{cases}$$

and the bound is tight in the overall sense when $m \leq 6$ [44]. For large m, the bound $O(\log m)$ obtained by using an alternative method different from those in Theorem 1 is better [10]. Specifically, for $m=2$, the bound is

$$\begin{cases} \frac{2s+1}{s+1}, & 1 \leq s \leq \frac{1+\sqrt{5}}{2}, \\ \frac{s+1}{s}, & s > \frac{1+\sqrt{5}}{2}, \end{cases}$$

and LSc is the optimal algorithm [72]. For $m=3$, the bound is

$$\begin{cases} \dfrac{s_1+s_2+s_3}{s_1}, & s_1^2 \geq s_2^2 + s_1 s_2 + s_2 s_3, \\ \dfrac{2s_1+s_2+s_3}{s_1+s_2}, & s_1^2 < s_2^2 + s_1 s_2 + s_2 s_3 \text{ and } s_1^2 + s_1 s_2 \geq s_3^2 + s_1 s_3 + 2 s_2 s_3, \\ \dfrac{3s_1+s_2+s_3}{s_1+s_2+s_3}, & s_1^2 + s_1 s_2 < s_3^2 + s_1 s_3 + 2 s_2 s_3, \end{cases}$$

and it is tight for any combinations of machine speeds [90]. However, it has been shown that LSc is optimal only for the case when $s_1 = s_2 = 1 \geq (\sqrt{2}-1)s_3$ and $s_1^2 \geq s_2^2 + s_1 s_2 + s_2$ [90, 136], and there does exist another online algorithm which beats LSc when $\frac{1+\sqrt{97}}{8} < \frac{s_1}{s_2} = \frac{s_1}{s_3} < 2$ [90]. It implies that the optimality of LS for $P3||C_{max}$ cannot be generalized to the problem with different machine speeds.

Since the competitive ratio of LSc tends to infinite when m becomes very large [44], it is natural to raise the question that whether an algorithm with finite competitive ratio exists. The first such algorithm with competitive ratio 8 was obtained by using a so-called *doubling strategy* [10]. Their results were later improved by Berman [19], where a new algorithm with overall competitive ratio $3+\sqrt{8} \approx 5.828$ and a overall lower bound 2.4380 are given. The lower bound was recently improved to θ by Ebenlendr and Sgall [60], where $\theta \approx 2.564$ is the solution of the equation $\int_0^1 \frac{\ln \theta}{\ln(1-\theta^{-x})} dx = -1$.

For the most studied special case of $s_1 > s_2 = s_3 = \cdots = s_m = 1$, the overall competitive ratio of LSc for $Qm||C_{max}$ is at most $\frac{3m-1}{m+1}$, where the maximum value achieves at $s_1 = 2$ [44]. The bound matches the overall lower bound 2 when $m=3$ [118]. For $m \geq 4$, Li and Shi [118] designed a new algorithm which has a smaller competitive ratio around $s_1 = 2$. Thus, the overall competitive ratio can be decrease to 2.8795. Cheng et al. [42] further raised an algorithm which has a competitive ratio of 2.45 if the speed of the fastest machine is restricted to $1 \leq s_1 \leq 2$.

For $Rm||C_{max}$, Aspnes et al. [10] proved that the competitive ratio of LSc is at most m, and thus, LSc is optimal when $m=2$. They also designed a new algorithm with competitive ratio $O(\log m)$. Since the lower bound of the problem is at least $\lceil \log_2(m+1) \rceil$ [13], their algorithm is optimal up to a constant factor.

2.2 Preemptive Scheduling

Results of preemptive online scheduling are more complete than those of non-preemptive problems. The main reason may be that the optimal offline makespan can be obtained in polynomial time. The following lemma plays an important role in designing online algorithms for $Pm|pmtn|C_{max}$ and $Qm|pmtn|C_{max}$.

Lemma 1 ([84, 101]) *For $Qm|pmtn|C_{max}$,*

$$C^* = \min \left\{ \frac{\sum_{j=1}^{n} p_j}{\sum_{i=1}^{m} s_i}, \max_{1 \le l \le m-1} \frac{\sum_{i=1}^{l} p_{(i)}}{\sum_{i=1}^{l} s_i} \right\}.$$

Chen et al. [33] designed an optimal algorithm *preemptive* for $Pm|pmtn|C_{max}$. Let $\Gamma_m = \frac{m^m}{m^m-(m-1)^m}$. Note that $\lim_{m\to\infty} \Gamma_m = \frac{e-1}{e} \approx 1.582$. Recall that for any j, $C^*(\mathcal{J}_j)$ can be calculated by Lemma 1.

Algorithm *Preemptive*

Let the current job be J_j. Reindex the machines such that $L_1^{j-1} \le L_2^{j-1} \le \cdots \le L_m^{j-1}$. If $L_m^{j-1} + p_j \le \Gamma_m C^*(\mathcal{J}_j)$, then assign J_j to M_m. Otherwise, let $l = \min\{i | L_i^{j-1} + p_j \ge \Gamma_m C^*(\mathcal{J}_j)\}$. Assign part of J_j of processing time $\Gamma_m C^*(\mathcal{J}_j) - L_l^{j-1}$ to M_l and the remaining part of J_j of processing time $L_l^{j-1} + p_j - \Gamma_m C^*(\mathcal{J}_j)$ to M_{l-1}.

Theorem 3 ([33]) *The competitive ratio of Preemptive for $Pm|Pmtn|C_{max}$ is Γ_m.*

Proof (Sketch) Prove by induction that for any j, the following two inequalities hold

$$L_m^j \le \Gamma_m C^*(\mathcal{J}_j),$$

$$\sum_{i=1}^{k} L_i^j \le \frac{(\frac{m}{m-1})^k - 1}{(\frac{m}{m-1})^m - 1} P_j, k = 1, \cdots, m.$$

Then, it can be confirmed that the schedule produced by *Preemptive* is feasible, and the competitive ratio holds. □

Several attempts have been made to modify and generalize *Preemptive* to handle uniform machines, which resulted in an optimal algorithm for $Q2|pmtn|C_{max}$ with competitive ratio $\frac{s^2+2s+1}{s^2+s+1}$ [72, 180] and an optimal algorithm for $Qm|pmtn|C_{max}$ with nondecreasing speed ratios, that is, $\frac{s_{i-1}}{s_i} \le \frac{s_i}{s_{i+1}}$, $2 \le i \le m-1$ [64]. But it seems difficult to make further generalizations using similar ideas.

Recently, Ebenlendr et al. [62] introduced a novel technique which leads to an optimal algorithm of $Qm|pmtn|C_{max}$ for any combinations of machine speeds. The competitive ratio $\Gamma(s_1, \cdots, s_m)$ of the algorithm is the optimal value of the following linear programming with decision variables $q_1, \cdots, q_m, O_1, \cdots, O_m$ and parameters $s_1, \cdots, s_m$.

$$\begin{aligned}
\max\ & q_1 + q_2 + \cdots + q_m \\
s.t.\ & q_1 + \cdots + q_k \le (s_1 + s_2 + \cdots + s_m)O_k, \quad k = 1, \cdots, m, \\
& q_j + q_{j+1} + \cdots + q_k \le (s_1 + s_2 + \cdots + s_{k-j+1})O_k, 2 \le j \le k \le m, \\
& 1 = s_1 O_m + s_2 O_{m-1} + \cdots + s_m O_1, \\
& q_j \le q_{j+1}, j = 2, \cdots, m-1, \\
& q_1 \ge 0, q_2 \ge 0.
\end{aligned} \tag{4}$$

However, it can only be proved that the algorithm is optimal for any m and any combinations of machine speeds, but the concrete expression of the optimal bound $\Gamma(s_1, \cdots, s_m)$ is not known yet, even the overall bounds are hard to get. The currently known best lower and upper bound for arbitrary m is 2.054 (achieves at $m = 100$ and is obtained by numerical calculation) and $e \approx 2.718$, respectively [62]. When the number of machines is small or some assumptions are made on the values of s_i, (4) can be solved theoretically [62]. For example, the optimal bounds for $Q3|pmtn|C_{max}$ are

$$\Gamma(s_1, s_2, s_3) = \begin{cases} \left(\frac{s_1}{S} + (1 - \frac{s_1}{S})\frac{s_2}{S} + (1 - \frac{s_1}{S})^2 \frac{s_3}{S}\right)^{-1}, & \text{if } \frac{s_1}{s_2} \le \frac{s_2}{s_3} + 1; \\ \frac{S^2}{s_1^2 + s_2^2 + s_3^2 + s_1 s_2 + s_1 s_3 + s_2 s_3}, & \text{if } \frac{s_1}{s_2} \ge \frac{s_2}{s_3} + 1. \end{cases}$$

Another question requiring answer is that whether the optimal bounds will increase if idle time is prohibited. Note that algorithms in [33, 64, 72, 180] all do not introduce idle time, but no algorithm which allows idle time can have a smaller competitive ratio under certain machine environment mentioned there. However, the algorithm given in [62] does use idle time. It is not known yet that whether an algorithm which does not use idle time will necessarily have a strict larger competitive ratio than $\Gamma(s_1, \cdots, s_m)$ for general $Qm|pmtn|C_{max}$.

2.3 Randomized Algorithm

As far as ever known results are concerned, for case on parallel machine scheduling problem with objective to minimize makespan, any lower bound for deterministic preemptive online algorithm (allowing idle time) is also a lower bound for randomized preemptive or non-preemptive online algorithms. For example, the randomized lower bound for $Pm||C_{max}$ and $Qm||C_{max}$ is $\frac{(m-1)^m}{m^m-(m-1)^m}$ [32, 155] and $\Gamma(s_1, \cdots, s_m)$ [62], respectively. But the converse is not true. Tichý [174] showed that there does not exist a randomized algorithm for $P3||C_{max}$ with competitive ratio $\frac{27}{19}$. However, his proof does not yield a new randomized lower bound with a strict larger value.

The first randomized algorithm for parallel machine scheduling is due to [18], where an optimal randomized algorithm for $P2||C_{max}$ is given. The algorithm was restated in a simpler version in [155] as follows.

Algorithm *Random*

1. If possible, assign the current job J_j randomly so that afterwards the expected makespan equals $\frac{2}{3}P_j$.
2. Otherwise, assign job J_j always on the less loaded machine.

Theorem 4 ([18, 155]) *The competitive ratio of* $Random$ *for* $P2||C_{max}$ *is* $\frac{4}{3}$.

Proof (Sketch) Prove by induction that for any j, the following two invariants hold

$$\begin{array}{l} E(C^{Random}(\mathcal{J}_j)) \geq \frac{2}{3}P_j, \\ \text{If } E(C^{Random}(\mathcal{J}_j)) > \frac{2}{3}P_j, \text{then } \max_{1\leq l\leq j} p_l \geq \frac{3}{4}E(C^{Random}(\mathcal{J}_j)). \end{array} \tag{5}$$

Clearly, (5) is also valid for $j = n$. Then, by (1) and (2), the theorem is thus proved. □

Design effective randomized algorithm for $Pm||C_{max}$ is rather a challenging work. When the number of machines is small, the competitive ratios of algorithms proposed by Seiden [151, 153] can be less than the best known deterministic algorithms (See Table 2). For arbitrary m, Albers [2] designed a randomized algorithm with competitive ratio 1.916, which is a combination of two deterministic algorithms with equal probability, and it is better than any known deterministic algorithm.

There is even less study on randomized algorithm for uniform machines. For $Q2||C_{max}$, Epstein et al. [72] showed that randomization does not help when $s \geq 2$. They also presented numerical randomized lower bounds for $1 \leq s \leq 2$ by solving linear programmings and a smaller analytical randomized lower bound $\frac{(s+1)^2(s+2)}{3s^2+5s+1}$ for $1 \leq s \leq \sqrt{2}$. Three randomized algorithms are proposed, and thus, an overall upper bound 1.52778 can be achieved, but the gap between the lower and upper bounds for any $s \geq 1$ is still relatively large. For $Q3||C_{max}$ with restricted machine speeds $s_1 = s_2 \geq s_3 = 1$, Musitelli and Nicoletti [136] presented a randomized algorithm with competitive ratio

$$\begin{cases} \dfrac{4s_2+2}{3s_2}, & 1 \leq s_2 < \frac{\sqrt{10}+2}{3}, \\ \dfrac{6s_2+2}{3s_2+2}, & s_2 \geq \frac{\sqrt{10}+2}{3}. \end{cases}$$

It is only a little smaller than the competitive ratio of LSc, and no randomized lower bound has been reported.

2.4 Other Objectives

2.4.1 Total Completion Time

The property of the objective of minimizing the total completion time is different from that of minimizing makespan. The objective value of the schedule will be affected by not only the set of jobs assigned to each machine but also the process sequence of it. As a result, there exists no algorithm with constant competitive ratio, even for the single machine case.

Theorem 5 ([76]) *For every function $f : N \to R^+$ that fulfills conditions:*
1. $f(n)$ is nondecreasing,
2. $\sum_{n=1}^{\infty} \frac{1}{nf(n)}$ converges,
there exists an online algorithm for $1||\sum C_j$ with competitive ratio at most $O(f(n))$, where n is the number of jobs. On the other hand, let $g : N \to R^+$ be a function that fulfills condition:
1. $g(n)$ is nondecreasing,
2. $\sum_{n=1}^{\infty} \frac{1}{ng(n)}$ diverges,
3. $g(n) = O(\log^2 n)$,
4. $g\left(\frac{n}{\log^2 n}\right) = \Omega(g(n))$,
then there does not exist an online algorithm for $1||\sum C_j$ with competitive ratio $o(f(n))$.

By selecting specific functions which satisfy the respective conditions, it can be found that the gap between the lower and upper bounds is rather small.

Corollary 1 ([76]) *For every $\epsilon > 0$, there exists an online algorithm with competitive ratio at most $(\log n)^{1+\epsilon}$, but no online algorithm with competitive ratio $\log n$ can exist.*

2.4.2 Machine Covering

The online version of non-preemptive machine covering problem is relatively easy. It should be mentioned that for machine covering problems, LSs is better than LSc in most cases. Woeginger [181] and Epstein [66] proved that LSs is the optimal algorithm for $Pm||C_{min}$ and $Q2||C_{min}$, respectively. In fact, their analysis can be generalized to $Qm||C_{min}$. LSs remains the optimal algorithm with competitive ratio $\sum_{i=1}^{m} s_i$. Azar and Epstein [12] designed a randomized algorithm for $Pm||C_{min}$ with an overall competitive ratio $O(\sqrt{m}\log m)$ and proved an overall randomized lower bound $\frac{\sqrt{m}}{4}$.

Surprisingly, machine covering problem becomes much more difficult if preemption is allowed, even though the optimal offline objective value

$$\min_{0 \le j \le m-1} \frac{\sum_{l=j+1}^{n} p_{(l)}}{\sum_{l=j+1}^{m} s_l}$$

for $Qm|pmtn|C_{min}$ can be calculated similarly as $Qm|pmtn|C_{max}$ [108]. Jiang et al. [108] proved that $\sum_{i=1}^{m} \frac{1}{i}$ is a lower bound of $Pm|pmtn|C_{min}$ and m is an overall lower bound of $Qm|pmtn|C_{min}$, but no algorithm with matched competitive ratio has been given. For $Q2|pmtn|C_{min}$, they designed an optimal algorithm with competitive ratio $\frac{2s+1}{s+1}$. However, their algorithm may introduce idle time, even when it assigns the first job. If idle time is not allowed, then no deterministic algorithm can have a smaller competitive ratio than

$$\begin{cases} \dfrac{2s+1}{s+1}, & 1 \le s < \frac{1+\sqrt{5}}{2}, \\ s, & s \ge \dfrac{1+\sqrt{5}}{2}. \end{cases}$$

A corresponding algorithm prohibiting idle time with matched competitive ratio was also given in [108]. Therefore, whether idle time is allowed or not does affect the optimal bounds for machine covering problems. It should be noted that different from algorithms allowing idle time, for non-idle time situation, there is no evidence yet that the lower bound also holds for randomized algorithm. Such phenomenon also appears in similar situations in the following.

3 Semi-online Scheduling

3.1 Motivation and Taxonomy

There are two basic characteristics for the online over list model. The first is that jobs arrive one by one and it is required to assign each job without any knowledge of the jobs that arrive later. The second is that the assignment of the job should be determined immediately upon its arrival and cannot be changed later. The reason of the first characteristic is "lack of information," while the essence of the second is "restriction of scheduling." There are often voices of criticism on the model for the sake of both theoretical study and practical application. In practice, most problems are not pure offline or online but somehow in between. Theoretically, the competitive ratio of an online algorithm tends to be over large in contract with the actual performance, since offline algorithms are more powerful than online algorithms. Moreover, the lower bound of the problem can also be ineffective, namely, too large, as the adversary can arbitrarily determine the instances without any restriction.

The reasons aforementioned bring semi-online scheduling about an active research area. Semi-online can be looked upon as relaxation of online or an intermediate state of offline and online. The primary significance of study on semi-online settings is as follows. First and foremost, without reasonable understanding of semi-online problems, there will be no more than online algorithm to implement faced with semi-online cases, which will give rise to inevitable loss definitely, just as LS algorithm performs not satisfactorily on offline problems. Furthermore, deep study on semi-online problems will lead to initiative search for crucial factors in algorithm design and improving, which will certainly save the cost and energy on many pointless factors. In theory, research on semi-online scheduling is of help to clarify influence of various restrictions of both information accessed to and algorithm design, and to reveal some deep and dominant properties. For example, it is interesting to consider the question that whether it is possible or under what kind of circumstance that an *online approximation scheme*, that is, a class of semi-online algorithms with competitive ratio arbitrarily close to 1 can exist. In contrast to $PTAS$ of offline problems, whose time complexity increases as the worst-case ratio of the algorithm decreases, more partial information or more freedom of scheduling is required in online cases when pursuing a better competitive ratio. In fact, one such approximation scheme has already been obtained [160]. Last but not least, it will be meaningful for algorithm design and analysis on online problems when researching on semi-online models, as will see later.

Though semi-online model can be viewed as an intermediate state between online and offline, its properties and research approach are almost the same as the online model. Competitive analysis is still the primary technique adopted, and lower bound of the problem, together with competitive ratio of the algorithm, can be defined accordingly.

The value of a semi-online model, denoted by Π_s, is evaluated by comparing its upper (or lower) bound with that of a corresponding online model reacting to the same machine environment and objective function, which is represented by Π_o. The semi-online model is called *valuable* if there exists an online algorithm for Π_s whose competitive ratio is smaller than the optimal bound (or lower bound) of the Π_o. On the other hand, if the lower bound of Π_s is no smaller than the competitive ratio of an algorithm for Π_o, it is called *valueless*. Of course, a semionline model is valuable or may not be determined by certain machine environment or objective functions. In this section, mainly identical and uniform machines, as well as minimizing makespan and maximizing minimum machine load, are considered. It is also possible to compare variants of semi-online models qualitatively, according to the extent to which the optimal bound of pure online problem can be improved. However, the outcomes may be quite sensitive to the machine environment or objective function.

Various semi-online paradigms have been studied in the literature, which can be divided into several categories. The taxonomy and their notations, which will be included in the middle field of the three-field notation, are summarized in the Table 3. Their specific implications will be introduced in the rest of the subsection.

3.1.1 Basic Semi-online Models of Type I

Basic semi-online models of Type I modify the first assumption of the online over list model, that is, some partial information about the future jobs are known in advance. The following four kinds of partial information are the most heated.

decr [154]: The jobs are arriving in nonincreasing order of their processing times, that is, $p_1 \geq p_2 \geq \cdots \geq p_n$.

Table 3 Semi-online models

Basic	Type I	*Valuable*: sum, opt, max, decr
		Valueless: num, LB, UB, min, incr, lookahead
	Type II	buffer, parallel, reassignment
Combined	Type I and Type I	UB & LB, UB & sum, LB & max, max & sum, max & opt,
		decr & sum, decr & opt, end of sequence
	Type I and Type II	sum & buffer, sum & parallel, sum & reassignment
Disturbed	inexact information	inexact sum, inexact opt, inexact max

sum [111]: The total processing time of all jobs $P = \sum_{j=1}^{n} p_j$ is known before the first job is arrived.

opt [14]: The optimal offline makespan C^* is known before the first job is arrived.

max [111]: The maximal processing time of all jobs $p_{max} = \max_{j=1,\cdots,n} p_j$ is known before the first job is arrived.

The motivation of $decr$, sum, and max is obvious, whereas it may seem to be strange at first glance that the optimal offline makespan can be known in advance. In fact, it can be interpreted from the realistic scenario of remote file transfer [14]. Besides, online problems with C^* is known in advance have already been studied before semi-online began to draw attention, since the following lemma is useful in competitive analysis of difficult online problems [10].

Lemma 2 *For some scheduling problem of online over list model with objective to minimize makespan, if there exists an algorithm for the* opt *variant with competitive ratio* r*, then there exists an algorithm for the pure online model with competitive ratio at most* $4r$*.*

Some variants reacting to other partial knowledge of input are also reasonable, which is quite common in practice. For example,

num: The number of all jobs n is known before the first job is arrived.
incr: The jobs are arriving in nondecreasing order of their processing times, that is, $p_1 \leq p_2 \leq \cdots \leq p_n$.
UB(*LB*): The upper (lower) bound on the processing time of all jobs $p_{UB} \geq \max_{j=1,\cdots,n} p_j$ ($p_{LB} \leq \min_{j=1,\cdots,n} p_j$) is known before the first job is arrived.
min: The minimal processing time of all jobs $p_{min} = \min_{j=1,\cdots,n} p_j$ is known before the first job is arrived.

However, such paradigms, which seem worthless under certain machine environment and objective functions discussed so far, have not drawn much attention.

Another kind of partial information which is common in practice is called *lookahead* [111]. When assigning the current job, the information of the next k jobs is known, where k is a constant number. Lookahead tends to be of benefit for scheduling according to practical experience, whereas it turns out to be still valueless as far as competitive ratio is concerned.

3.1.2 Basic Semi-online Models of Type II

There are other variants which lie between online and offline. However, they have nothing to do with the partial information of the input but more freedom acquired in scheduling. These variants belong to *Type II* semi-online models. It is a much more potential research area compared with Type I semi-online models, which is fruitful of results.

In the *buffer* [111] paradigm, there is a buffer of size K which can store at most K jobs. The incoming job can either be scheduled immediately or be stored in the buffer, which enables it to be scheduled later. The root cause of the improved performance attributed to buffer lies in the fact that it can reorder the job sequence. Pure online and offline models are equivalent to problems with buffer size 0 and ∞, respectively. In another related class of models *reassignment* [160, 167], it is allowed to reassign some of the jobs during or after the process. Recall that in the pure online model, no job can be reassigned.

A technique frequently used in designing algorithm for the offline problems is to run several procedures in parallel and then select the best result as output. However, it does not fit pure online setting. Semi-online variant which allows to use such technique is called *parallel*. A real-world system which resembles such situation was claimed to exist in [111]. Kellerer et al. [111] proved that the optimal bound of $P2|parallel(2)|C_{max}$ is $\frac{4}{3}$; here, $parallel(2)$ means that two procedures are run in parallel.

3.1.3 Combined Semi-online Models

Since some semi-online variants are indeed of value, it is natural to raise the question whether a more efficient algorithm can be designed for a problem possessing characteristics of several semionline variants in the meantime. A combination is *useful*, if there exists an online algorithm for the combined semi-online problem whose competitive ratio is smaller than the optimal bound (or lower bound) of any one of the basic semi-online model. Otherwise, the combination is *useless*. The combination could be of two basic semi-online variants of Type I, as well as that of Type I and II, respectively. In spite of the fact that combination consisting of more than three variants can certainly be concerned with, such cases are always quite complicated and seldom studied.

3.1.4 Disturbed Semi-online Models

The *disturbed* semi-online models were first studied in [165]. In some cases, it is allowed to use some additional information or slightly more resources than the offline algorithm, whereas the information or resources might be unreliable. For example, it is known in advance that the total processing time lies in the interval $[P_0, \alpha P_0]$, but the exact value of P is still unknown. It can be believed that the value of inexact partial information lies largely on the value of α, which is called *disturbance parameter*. For problem with inexact partial information, special stress is laid on determining the value of α which insures the information to be useful and further designing algorithm making use of the beneficial information acquired.

3.2 Results on Basic Semi-online Models of Type I

3.2.1 Identical Machines

It has been widely known for a long time that in combinatorial optimization, the performance of an algorithm can be improved after appropriate preprocessing for the input. Graham [87] proved that the worst-case ratio of algorithm *Longest Processing Time first* (LPT for short), where LS is implemented after a preprocessing step, namely, reordering the jobs in a nonincreasing sequence of their processing time, is $\frac{4}{3} - \frac{1}{3m}$. However, it is not allowed to reorder the sequence in the pure online setting. But if the jobs are arrived in nonincreasing order of their processing times, the performance of LS will be just the same as that of LPT. Above situation was reformulated as a semi-online variant by Seiden et al. [154]. They proved that LS is the optimal algorithm for $P2|decr|C_{max}$. However, LS is no longer optimal when $m \geq 3$, since there exists an algorithm with competitive ratio $\frac{5}{4}$ for $m \geq 4$ and an optimal algorithm with competitive ratio $\frac{1+\sqrt{37}}{6}$ for $m = 3$ [43].

If the total processing time of all jobs P is known in advance, a lower bound $\frac{P}{m}$ on the optimal makespan is readily available, which will be of great help in designing algorithm with a better competitive ratio. Take the optimal algorithm reacting to two machines [111] for example. Jobs are successively assigned to M_1 until there exists a crucial job, such that the load of M_1 would exceed a threshold $\frac{4}{3} \cdot \frac{P}{2}$ if the crucial job is still assigned to M_1; here, $\frac{4}{3}$ is the expected competitive ratio of the algorithm. Then, the crucial job is assigned to M_2, and the remaining jobs will no longer have any negative effect on the competitive ratio. The core idea of the algorithm is to keep one machine lightly loaded so as to serve the long jobs which might present later. The significance of *sum* lies in the fact that it is impossible to determine the threshold without access to P.

For $Pm|sum|C_{max}$, Angelelli et al. [6] proposed an algorithm with competitive ratio $\frac{\sqrt{6}+1}{2} \approx 1.725$ and proved that the lower bound of the problem is 1.565 for sufficiently large m. The lower bound was recently improved to 1.585 by Albers and Hellwig [3]. By more careful classification according to the processing time of the jobs as well as the current load of the machines, Cheng et al. [41] designed an improved algorithm with competitive ratio $\frac{8}{5}$. They also showed that $\frac{3}{2}$ is a lower bound of the problem for any $m \geq 6$. For the case of $m = 3$, the current best lower and upper bounds are $\frac{\sqrt{129}-3}{6} \approx 1.393$ and $\frac{27}{19} \approx 1.421$, respectively [9].

The semi-online variant *opt* is closely associated with the variant *sum*. In fact, many instances used to show the tightness of the competitive ratio of some algorithms satisfy $C^* = \frac{P}{m}$. However, the instance with $C^* > \frac{P}{m}$ does exist. Therefore, it is not easy to answer whether corresponding two problems regarding *sum* and *opt* variants have the same optimal bound. Nevertheless, Dósa et al. [54] proved the following result.

Lemma 3 **([54])**

(i) If ρ is a lower bound of $Qm|opt|C_{max}$, then the lower bound of $Qm|sum|C_{max}$ is at least ρ.

(ii) If there is an algorithm for $Qm|sum|C_{max}$ with competitive ratio r, then there also exists an algorithm for $Qm|sum|C_{opt}$ with competitive ratio at most r.

By Lemma 3, the algorithm for $Pm|sum|C_{max}$ given in [41] can be modified to solve $Pm|opt|C_{max}$ with at most the same competitive ratio $\frac{8}{5}$, which improves the former algorithm for $Pm|opt|C_{max}$ with a competitive ratio of $\frac{13}{8}$ [14]. When the number of machines is small, another algorithm also given in [14] has a smaller competitive ratio of $\frac{5m-1}{3m+1}$. On the other hand, the lower bound of $Pm|sum|C_{max}$ is no longer valid for $Pm|opt|C_{max}$. Though $\frac{4}{3}$ is clearly a lower bound of $Pm|opt|C_{max}$ for any $m \geq 2$ [14], no larger lower bound has been reported for $m \geq 3$.

There are fewer results on the *max* variant. He and Zhang [97] presented an optimal algorithm for $P2|max|C_{max}$ with competitive ratio $\frac{4}{3}$. Note that for the *max* variant, not only the processing times of all jobs are no more than p_{max}, but also at least one job with processing time p_{max} will arrive sooner or later. Hence, the first job with processing time p_{max}, which is denoted as J_{max}, can be shifted to the beginning of the job sequence by assumption. Thus, the optimal algorithm given in [97] can be simplified by using above idea.

Algorithm *PLS*

```
1. Let $J_{max}$ be scheduled on $M_1$ from time 0 to $p_{max}$.
2. Assign the remaining jobs by $LS$ rule.
```

Theorem 6 *The competitive ratio of PLS is $\frac{4}{3}$ for $P2|max|C_{max}$.*

Proof If $\max\{L_1, L_2\} \leq \frac{2}{3}P$, then $\frac{C^{PLS}}{C^*} \leq \frac{\max\{L_1,L_2\}}{\frac{P}{2}} \leq \frac{4}{3}$ by (1). Otherwise,

$$\max\{L_1, L_2\} + (L_1 + L_2) + |L_1 - L_2| = 3\max\{L_1, L_2\} > 2P = (L_1 + L_2) + \max\{L_1, L_2\} + \min\{L_1, L_2\}. \quad (6)$$

Since the remaining jobs are assigned by LS rule, $|L_1 - L_2| \leq p_{max}$. Combining it with (6), $\min\{L_1, L_2\} < |L_1 - L_2| \leq p_{max}$. Recall that J_{max} is assigned to M_1, $L_1 \geq p_{max}$. Hence, $L_2 < p_{max}$ and thus no job other than J_{max} will be assigned to M_1, which implies that PLS produces an optimal schedule. □

However, for general m, no algorithm performing better than the best known algorithm for pure online model has been obtained. It is also unknown that whether

Table 4 Results of semi-online variants on arbitrary m identical machines

	Non-preemptive				Preemptive
	Makespan minimization		Machine covering		Makespan minimization
Variant	Lower bound	Upper bound	Lower bound	Upper bound	Optimal bound
decr	$\frac{1+\sqrt{37}}{6}$ [154]	$\frac{5}{4}$ [43]	Open	$\frac{4m-2}{3m-1}$ [47]	$\max_{0\le k\le m} \frac{2m^2+2mk}{2m^2+k^2+k}$ [154]
sum	1.585 [3]	$\frac{8}{5} = 1.6000$ [41]		$m-1$ [22, 166]	1 [98]
opt	$\frac{4}{3} \approx 1.333$ [14]	$\frac{8}{5} = 1.6000$ [41]	$\frac{43}{24} \approx 1.791$ [61]	$\frac{11}{6} \approx 1.834$ [61]	1 [58]
max	Open	Open		$m-1$ [22, 166]	$\max_{0\le k\le m} \frac{2m^2+2mk}{2m^2+k^2+k}$ [154]

the partial information is still valuable for sufficiently large m, since the effect of the information about the single job J_{max} tend to reduce as m increases.

As far as the problem of machine covering is concerned, the four semi-online variants differ hugely both in property and difficulty. Optimal algorithms for $Pm|sum|C_{min}$ and $Pm|max|C_{min}$ can be obtained, both having a competitive ratio $m-1$ for $m \geq 3$ [22, 166]. However, the research on $Pm|opt|C_{min}$ seems to be much more difficult than that of $Pm|sum|C_{min}$. Azar and Epstein [12] proposed an algorithm with competitive ratio $2-\frac{1}{m}$, which is optimal for $m = 2, 3, 4$. A more involved algorithm with competitive ratio $\frac{11}{6} \approx 1.834$ was given in [61]. The current best lower bound is $\frac{7}{4}$ for $m > 4$ [12] and $\frac{43}{24} \approx 1.791$ for sufficiently large m [61]. These bounds are far smaller than those for $Pm|sum|C_{min}$. For the problem $Pm|decr|C_{min}$, LS has a competitive ratio of $\frac{4m-2}{3m-1}$ [47] and is optimal for $m = 2, 3$ [96].

Tables 4 and 5 summarize the current best results of different semi-online variants on identical machines for arbitrary m and small number of m, respectively. As is shown in the table, different semi-online variants are diverse in value. In addition, optimal algorithm for two machines has almost all been found, whereas for three or more machines, nearly all problems are waiting to be answered, which is quite thought-provoking.

3.2.2 Two Uniform Machines

Similarly as pure online problems, the upper and lower bounds for most semi-online problems on two uniform machines are piece-wise rational or irrational functions of s. But the number of the pieces could be considerably large, often exceeding 10. There may be large difference among optimal algorithms for the same problem with different value of s, which make the analysis of these problems extremely difficult. Most results are obtained mainly by case by case analysis. Up till now, no universal or revolutionary method has been developed to handle these problems efficiently.

Table 5 Results of semi-online variants on a small number of identical machines

	$m = 2$[b]		$m = 3$					$m = 2$
	Makespan		Makespan			MC		Makespan
	Non-pmtn	pmtn	Non-pmtn		pmtn	Non-pmtn		Non-pmtn
Model[a]	OB	OB	LB	UB	OB	OB	Model[a]	OB
decr	$\frac{7}{6}$ [86, 154]	$\frac{6}{5}$ [154]	$\frac{1+\sqrt{37}}{6}$ [154]	$\frac{1+\sqrt{37}}{6}$ [43]	$\frac{6}{5}$ [154]	$\frac{5}{3}$ [12]	*parallel*(2)	$\frac{4}{3}$ [111]
sum	$\frac{4}{3}$ [111]	1 [98]	$\frac{\sqrt{129}-3}{6}$ [9]	$\frac{27}{19}$ [9]	1 [98]	2 [166]	*reassignFE*	$\sqrt{2}$ [167]
opt	$\frac{4}{3}$ [14]	1 [65]	$\frac{4}{3}$ [14]	$\frac{7}{5}$ [14]	1 [58]	$\frac{5}{4}$ [47, 96]	*reassignSQ*	$\frac{4}{3}$ [55]
max	$\frac{4}{3}$ [97]	$\frac{6}{5}$ [154]	Open	Open	$\frac{6}{5}$ [154]	2 [166]	*sum & buffer*	$\frac{6}{5}$ [53]
buffer	$\frac{4}{3}$ [111]	$\frac{4}{3}$ [52]	[c]		$\frac{9}{7}$ [52]	[d]	*sum & reassignFE*	$\frac{5}{4}$ [132]
reassignFA	$\frac{4}{3}$ [167]	Open	$\frac{15}{11}$ [4]	$\frac{15}{11}$ [4]	Open	Open	*sum & parallel*(2)	$\frac{5}{4}$ [53]
sum & max	$\frac{6}{5}$ [164]	1 [98]	$\frac{4}{3}$ [182]	$\frac{4}{3}$ [182]	1 [98]	$\frac{3}{2}$ [166]	*decr & sum*	$\frac{10}{9}$ [164]
opt & max	$\frac{6}{5}$ [164]	1 [65]	$\frac{4}{3}$ [182]	$\frac{4}{3}$ [182]	1 [98]	$\frac{3}{2}$ [166]	*decr & opt*	$\frac{10}{9}$ [65]
eos	$\sqrt{2}$ [190]	Open	$\frac{3}{2}$ [190]	$\frac{3}{2}$ [190]	Open	Open		

[a]*MC* machine covering, *OB* optimal bound, *LB* lower bound, *UB* upper bound

[b]For most semi-online variants, the makespan minimization problem and the machine covering problem on two identical machines are equivalent. The competitive ratios r_1 and r_2 of the same algorithm for corresponding two problems satisfy $r_1 + \frac{1}{r_2} = 2$. Therefore, results about machine covering problems on two identical machines are omitted. However, there does exist semi-online model that above two problems are not equivalent, such as *UB&LB*

[c]The lower bound is $\frac{15}{11}$ for any buffer size K [63], and there exists an algorithm with matched competitive ratio which uses a buffer of size 6 [112]

[d]The lower bound is $\frac{11}{6}$ for any buffer size K, and the upper bound is $\frac{5}{2}$ for $K \geq 2$ [73]

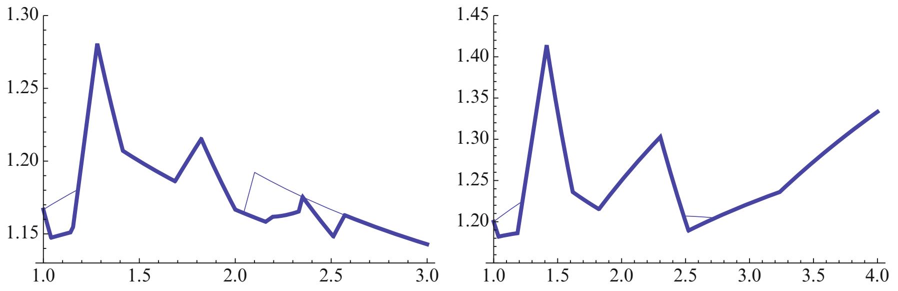


Fig. 2 The optimal bounds (*thick*) and competitive ratios of LS (*dash*) for $Q2|decr|C_{max}$ (*left*) and $Q2|decr|C_{min}$ (*right*)

Among the four basic semi-online variants of Type I, *decr* is the only one which optimal bounds for all values of $s \geq 1$ are obtained for both minimization and maximization problems. Based on the worst-case ratio of LPT for the offline version [133], Epstein and Favrholdt proved that LSc is optimal for the majority value of s. For the remaining two intervals which is close to 1 and nearby 2.5 where LSc is not optimal, they designed three new algorithms and proved their optimality. Symmetrically, for $Q2|decr|C_{min}$, LSs is also optimal except for two similar intervals of s, and new optimal algorithms have been proposed for these intervals [39]. The figures depicting two optimal bounds (as functions of s) bear some similarities (Fig. 2).

The other two problems which optimal algorithms for any $s \geq 1$ have been obtained are $Q2|sum|C_{min}$ and $Q2|opt|C_{min}$. The optimal bound of $Q2|sum|C_{min}$ [163] is

$$\begin{cases} \frac{s+2}{s+1}, & s \in [1, \sqrt{2}], \\ s, & s \in [\sqrt{2}, \frac{1+\sqrt{5}}{2}], \\ \frac{s^2+s+1+\sqrt{5s^4+6s^3+3s^2+2s+1}}{2s(s+1)}, & s \in (\frac{1+\sqrt{5}}{2}, \infty), \end{cases}$$

while the optimal bound of $Q2|opt|C_{min}$ [66] is

$$\begin{cases} \frac{s+2}{s+1}, & s \in [1, \sqrt{2}], \\ s, & s \in [\sqrt{2}, \frac{1+\sqrt{5}}{2}], \\ \frac{2s+1}{s+1}, & s \in (\frac{1+\sqrt{5}}{2}, \infty). \end{cases}$$

Note that the optimal bound of $Q2|sum|C_{min}$ is strict smaller than that of $Q2|opt|C_{min}$ for $s > \frac{1+\sqrt{5}}{2}$. The overall optimal bound of $Q2|sum|C_{min}$ is also strict less than that of $Q2|opt|C_{min}$ (See Fig. 3). These conclusions turn out to be opposite to both the results of Lemma 3 concerning problems with objective to minimize makespan and that of machine covering problems on m identical machines.

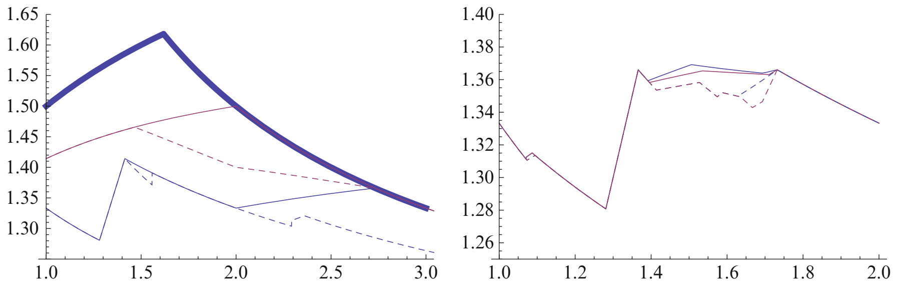


Fig. 3 *Left*: the optimal bound of $Q2||C_{max}$ (*thick*) and the upper bounds (*solid*) and lower bounds (*dash*) of $Q2|max|C_{max}$ (*bottom*) and $Q2|eos|C_{max}$ (*top*). *Right*: the upper bounds (*solid*) and lower bounds (*dash*) of $Q2|sum|C_{max}$ (*top*) and $Q2|opt|C_{max}$ (*bottom*)

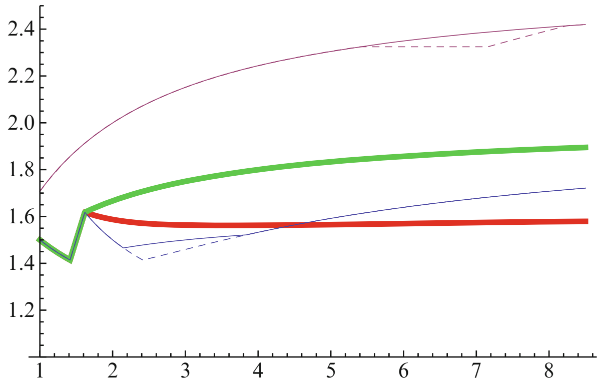


Fig. 4 The optimal bounds (*thick*) of $Q2|sum|C_{min}$ (*bottom*) and $Q2|opt|C_{min}$ (*top*) and the upper bounds (*solid*) and lower bounds (*dash*) of $Q2|max|C_{min}$ (*bottom*) and $Q2|eos|C_{min}$ (*top*)

For the remaining four problems, there are still gaps between upper and lower bounds for some values of s, though ceaseless efforts have been put into narrowing the gap. The gaps of some problems seem to be small, but it is usually the case that the gap might appear exactly be the place where the analysis becomes toughest. For the sake of conciseness, only best known bounds will be listed in the following. General information of these bounds is summarized in Table 6 (also see Figs. 4 and 3). It can be seen that $Q2|sum|C_{max}$ and $Q2|opt|C_{max}$ are much more complicated than the corresponding machine covering problems. For $Q2|max|C_{max}$, the effect of the information about the single job J_{max} declines when s is sufficiently large. There is a noteworthy fact that for majority makespan minimization problems, the overall bound is achieved at the point where the argument s values exactly the overall bound itself and lies in the interval $[1.2, 1.5]$. For machine covering problems, the overall bounds all achieve at $s = \infty$ except $Q2|sum|C_{max}$.

Table 6 General information and current status of upper and lower bounds on two uniform machines

Problem		Overall lower bound		Overall upper bound		Largest gap		Intervals where gap exists	
Objective	Variant	Lower bound	s	Upper bound	s	Gap	s	Intervals	Length
C_{max}	*decr*	$\frac{1+\sqrt{17}}{4}$	$\frac{1+\sqrt{17}}{4}$	$\frac{1+\sqrt{17}}{4}$	$\frac{1+\sqrt{17}}{4}$	0		$\emptyset$	0
	sum	$\frac{1+\sqrt{3}}{2}$	$\frac{1+\sqrt{3}}{2}$	1.3692	1.5062	0.0176	1.5744	$(1.071, 1.0946) \cup (1.3915, 1.732)$	0.3641
	opt	$\frac{1+\sqrt{3}}{2}$	$\frac{1+\sqrt{3}}{2}$	$\frac{1+\sqrt{3}}{2}$	$\frac{1+\sqrt{3}}{2}$	0.0219	$\frac{5}{3}$	$(1.071, 1.0946) \cup (1.3956, 1.732)$	0.3600
	max	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{2}$	0.081	$1+\sqrt{3}$	$(1.414, 1.558) \cup (2, 3.56)$	1.704
	eos	1.4656	1.4656	$\frac{3}{2}$	2	0.1000	2	$(1.4657, 1.732)$	0.2663
C_{min}	*decr*	2	∞	2	∞	0		$\emptyset$	0
	sum	$\frac{1+\sqrt{5}}{2}$	$\frac{1+\sqrt{5}}{2}$	$\frac{1+\sqrt{5}}{2}$	$\frac{1+\sqrt{5}}{2}$	0		$\emptyset$	0
	opt	2	∞	2	∞	0		$\emptyset$	0
	max	2	∞	2	∞	0.064	$1+\sqrt{2}$	$(2.148, 3.836)$	1.688
	eos	$\frac{5}{2}$	∞	$\frac{5}{2}$	∞	0.0624	7.1519	$(5.4043, 8.2426)$	2.8383

The lower and upper bounds of $Q2|sum|C_{max}$ [54, 65, 137] are

$$\begin{cases} \frac{3s+1}{3s}, & s \in \left[1, \frac{5+\sqrt{205}}{18} \approx 1.0732\right], \\ \frac{8s+5}{5s+5}, & s \in \left[\frac{5+\sqrt{205}}{18}, \frac{1+\sqrt{31}}{6} \approx 1.0946\right], \\ \frac{2s+2}{2s+1}, & s \in \left[\frac{1+\sqrt{31}}{6}, \frac{1+\sqrt{17}}{4} \approx 1.280\right] \\ s, & s \in \left[\frac{1+\sqrt{17}}{4}, \frac{1+\sqrt{3}}{2} \approx 1.36603\right] \\ \frac{2s+1}{2s}, & s \in \left[\frac{1+\sqrt{3}}{2}, \sqrt{2} \approx 1.41421\right] \\ \frac{3s+5}{2s+4}, & s \in \left[\sqrt{2}, \frac{\sqrt{21}}{3} \approx 1.52753\right] \\ \frac{3s+3}{3s+1}, & s \in \left[\frac{\sqrt{21}}{3}, \frac{5+\sqrt{193}}{12} \approx 1.57437\right] \\ \frac{4s+2}{2s+3}, & s \in \left[\frac{5+\sqrt{193}}{12}, \frac{7+\sqrt{145}}{12} \approx 1.5868\right] \\ \frac{5s+2}{4s+1}, & s \in \left[\frac{7+\sqrt{145}}{12}, \frac{9+\sqrt{193}}{14} \approx 1.63509\right] \\ c(s), & s \in \left[\frac{9+\sqrt{193}}{14}, \sqrt{3}\right] \\ \frac{s+2}{s+1}, & s \in [\sqrt{3}, \infty) \end{cases}$$

and

$$\begin{cases} \frac{3s+1}{3s}, & s \in [1, x_1 \approx 1.071], \\ \frac{7s+6}{4s+6}, & s \in \left[x_1, \frac{1+\sqrt{145}}{12} \approx 1.0868\right], \\ \frac{2s+2}{2s+1}, & s \in \left[\frac{1+\sqrt{145}}{12}, \frac{1+\sqrt{17}}{4} \approx 1.280\right] \\ s, & s \in \left[\frac{1+\sqrt{17}}{4}, \frac{1+\sqrt{3}}{2} \approx 1.36603\right] \\ \frac{2s+1}{2s}, & s \in \left[\frac{1+\sqrt{3}}{2}, x_2 \approx 1.3915\right] \\ \frac{s+\sqrt{29s^2+59s+30}}{4s+5}, & s \in [x_2, x_3 \approx 1.5062] \\ \frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s}, & s \in [x_3, x_4 \approx 1.6932] \\ \frac{\sqrt{4s^4+8s^3+s^2+4}+2s^2+3s+2}{2s^2+6s}, & s \in [x_4, \sqrt{3}] \\ \frac{s+2}{s+1}, & s \in [\sqrt{3}, \infty), \end{cases},$$

respectively, where $c(s) = \dfrac{\sqrt{s^2x^2+4s(s+1-x)}+sx}{2s}$, x is a root of the equation $\dfrac{\sqrt{s^2x^2+4s(s+1-x)}+sx}{2s} = \dfrac{s(2s+2-x)}{(s+1)(x+2)}$, x_1, x_2, x_3, x_4 are the positive roots of

$$3s^2(9s^2-s-5) = (3s+1)(5s+5-6s^2),$$

$$\frac{2s+1}{2s} = \frac{s+\sqrt{29s^2+59s+30}}{4s+5},$$

$$\frac{s+\sqrt{29s^2+59s+30}}{4s+5} = \frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s},$$

$$\frac{\sqrt{36s^4+9s^3-32s^2-2s+9}+6s^2+9s+3}{9s^2+7s} = \frac{\sqrt{4s^4+8s^3+s^2+4}+2s^2+3s+2}{2s^2+6s},$$

respectively.

The lower and upper bounds of $Q2|opt|C_{max}$ [54, 65, 137] are

$$\begin{cases} \dfrac{3s+1}{3s}, & s \in \left[1, \frac{5+\sqrt{205}}{18} \approx 1.0732\right], \\ \dfrac{8s+5}{5s+5}, & s \in \left[\frac{5+\sqrt{205}}{18}, \frac{1+\sqrt{31}}{6} \approx 1.0946\right], \\ \dfrac{2s+2}{2s+1}, & s \in \left[\frac{1+\sqrt{31}}{6}, \frac{1+\sqrt{17}}{4} \approx 1.280\right] \\ s, & s \in \left[\frac{1+\sqrt{17}}{4}, \dfrac{1+\sqrt{3}}{2} \approx 1.36603\right] \\ \dfrac{2s+1}{2s}, & s \in \left[\frac{1+\sqrt{3}}{2}, \sqrt{2} \approx 1.41421\right] \\ \dfrac{3s+5}{2s+4}, & s \in \left[\sqrt{2}, \frac{\sqrt{21}}{3} \approx 1.52753\right] \\ \dfrac{3s+3}{3s+1}, & s \in \left[\frac{\sqrt{21}}{3}, \frac{5+\sqrt{193}}{12} \approx 1.57437\right] \\ \dfrac{4s+2}{2s+3}, & s \in \left[\frac{5+\sqrt{193}}{12}, \frac{7+\sqrt{145}}{12} \approx 1.5868\right] \\ \dfrac{5s+2}{4s+1}, & s \in \left[\frac{7+\sqrt{145}}{12}, \frac{9+\sqrt{193}}{14} \approx 1.63509\right] \\ \dfrac{7s+4}{7s}, & s \in \left[\frac{9+\sqrt{193}}{14}, \frac{5}{3}\right] \\ \dfrac{7s+4}{4s+5}, & s \in \left[\frac{5}{3}, \frac{5+\sqrt{73}}{8} \approx 1.89300\right] \\ \dfrac{s+1}{2}, & s \in \left[\frac{5+\sqrt{73}}{8}, \sqrt{3}\right] \\ \dfrac{s+2}{s+1}, & s \in [\sqrt{3}, \infty) \end{cases}$$

and

$$\begin{cases} \frac{3s+1}{3s}, & s \in [1, s_8 \approx 1.071], \\ \frac{7s+6}{4s+6}, & s \in \left[s_8, \frac{1+\sqrt{145}}{12} \approx 1.0868\right], \\ \frac{2s+2}{2s+1}, & s \in \left[\frac{1+\sqrt{145}}{12}, \frac{1+\sqrt{17}}{4} \approx 1.280\right] \\ s, & s \in \left[\frac{1+\sqrt{17}}{4}, \frac{1+\sqrt{3}}{2} \approx 1.36603\right] \\ \frac{2s+1}{2s}, & s \in \left[\frac{1+\sqrt{3}}{2}, \frac{1+\sqrt{21}}{4} \approx 1.39562\right] \\ \frac{6s+6}{4s+5}, & s \in \left[\frac{1+\sqrt{21}}{4}, \frac{1+\sqrt{13}}{3} \approx 1.535187\right] \\ \frac{12s+10}{9s+7}, & s \in \left[\frac{1+\sqrt{13}}{3}, \frac{5+\sqrt{241}}{12} \approx 1.71035\right] \\ \frac{2s+3}{s+3}, & s \in \left[\frac{5+\sqrt{241}}{12}, \sqrt{3}\right] \\ \frac{s+2}{s+1}, & s \in [\sqrt{3}, \infty), \end{cases}$$

respectively.

The lower and upper bounds of $Q2|max|C_{max}$ [23, 24, 128] are

$$\begin{cases} \frac{2s+2}{2s+1}, & s \in \left[1, \frac{1+\sqrt{17}}{4} \approx 1.280\right], \\ s, & s \in \left[\frac{1+\sqrt{17}}{4}, \sqrt{2}\right], \\ \frac{1+\sqrt{4s^2+1}}{2s}, & s \in [\sqrt{2}, x_5 \approx 1.558] \\ \frac{s+2}{s+1}, & s \in [x_5, x_6 \approx 2.292] \\ \frac{(s+1)(9s+5)-\sqrt{(s+1)(-3s^3+7s^2+7s+1)}}{2(s+1)(3s+2)}, & s \in [x_6, x_7 \approx 2.360] \\ \frac{s+\sqrt{s^2+4s}}{2s}, & s \in \left[x_7, \frac{3+\sqrt{17}}{2} \approx 3.56\right) \\ \frac{s+1}{s}, & s \in \left[\frac{3+\sqrt{17}}{2}, \infty\right) \end{cases}$$

and

$$\begin{cases} \dfrac{2s+2}{2s+1}, & s \in \left[1, \frac{1+\sqrt{17}}{4} \approx 1.280\right], \\ s, & s \in \left[\frac{1+\sqrt{17}}{4}, \sqrt{2}\right], \\ \dfrac{s+2}{s+1}, & s \in [\sqrt{2}, 2] \\ \dfrac{3s+2}{2s+2}, & s \in [2, 1+\sqrt{3}] \\ \dfrac{s+1}{s}, & s \in [1+\sqrt{3}, \infty), \end{cases}$$

respectively, where x_5, x_6, x_7 are the roots of

$$2s^4 + 2s^3 - 6s^2 - 5s + 3 = 0,$$

$$12s^6 - s^5 - 32s^4 - 12s^3 + 7s^2 + s - 1 = (6s^4 - 3s^3 - 6s^2 - 3s - 1) \times \sqrt{(s+1)(-3s^3 + 7s^2 + 7s + 1)},$$

$$8s^3 + 15s^2 + 13s + 4 = (6s^2 + 9s + 3)\sqrt{s^2 + 4s},$$

respectively.

The lower and upper bounds of $Q2|max|C_{min}$ [27, 129] are

$$\begin{cases} \dfrac{s+2}{s+1}, & s \in [1, \sqrt{2}], \\ s, & s \in \left[\sqrt{2}, \frac{1+\sqrt{5}}{2}\right], \\ \dfrac{s+1}{s}, & s \in \left[\frac{1+\sqrt{5}}{2}, 1+\sqrt{2}\right] \\ \dfrac{s^2+s+1+\sqrt{s^4-s^2+2s+1}}{s^2+2s}, & s \in [1+\sqrt{2}, \infty) \end{cases}$$

and

$$\begin{cases} \dfrac{s+2}{s+1}, & s \in [1, \sqrt{2}], \\ s, & s \in \left[\sqrt{2}, \dfrac{1+\sqrt{5}}{2}\right], \\ \dfrac{s+1}{s}, & s \in \left[\dfrac{1+\sqrt{5}}{2}, x_8 \approx 2.148\right] \\ \dfrac{s+1+\sqrt{5s^2+6s+1}}{2(s+1)}, & s \in [x_8, x_9 \approx 3.836] \\ \dfrac{s^2+s+1+\sqrt{s^4-s^2+2s+1}}{s^2+2s}, & s \in [x_9, \infty). \end{cases}$$

respectively, where x_8, x_9 are the positive roots of

$$\frac{s+1}{s} = \frac{s+1+\sqrt{5s^2+6s+1}}{2(s+1)}$$

and

$$\frac{s+1+\sqrt{5s^2+6s+1}}{2(s+1)} = \frac{s^2+s+1+\sqrt{s^4-s^2+2s+1}}{s^2+2s},$$

respectively.

3.2.3 Preemption

In regard to preemptive semi-online scheduling problems, there were some results on optimal algorithms [56, 67, 93, 94]. But breakthrough was not made until Ebenlendr and Sgall introduced the general framework for preemptive online scheduling, through which for the majority of semi-online variants, preemptive scheduling problems with the objective function of makespan can be solved. The optimal algorithm for these semi-online problems is identical with the optimal algorithm for the pure online problem; only the optimal bound needs to be recalculated by using the reformed linear programming (Eq. 4). However, as in the pure online case, there are still two points at issue. The first is that the analytically optimal bounds can hardly be obtained when m is large, even for the identical machines case. The second is that it is necessary for the algorithm to use idle times, and whether it is inevitable still remains unknown.

For $Qm|pmtn, opt|C_{max}$, there exists an algorithm which always produces an optimal schedule [58]. Similar result for two uniform machines was obtained by Epstein [65] before. $Pm|pmtn, sum|C_{max}$ also has an algorithm with competitive ratio 1 [98], but it does not hold for $Qm|pmtn, sum|C_{max}$ [59].

For $Pm|pmtn, decr|C_{max}$, Seiden et al. designed an optimal algorithm with competitive ratio $\max_{0\le k\le m} \frac{2m^2+2mk}{2m^2+k^2+k}$ [154]. The bound tends to $\frac{1+\sqrt{3}}{2}$ when $m \to \infty$, and the algorithm does not introduce idle times. Interestingly, it is also the optimal algorithm for $Pm|pmtn, max|C_{max}$. However, above two variants do not share optimal algorithm for uniform machines. The optimal bounds for $Q2|pmtn, decr|C_{max}$ [67] and $Q2|pmtn, max|C_{max}$ [94] are

$$\begin{cases} \dfrac{3s+3}{3s+2}, & 1 \le s \le 3, \\ \dfrac{2s^2+2s}{2s^2+s+1}, & s > 3 \end{cases}$$

and $\frac{2s^2+3s+1}{2s^2+2s+1}$, respectively (Fig. 5). Moreover, no deterministic algorithm that never uses idle time can have the same competitive ratio as those use idle time when $s > 2$ for the former problem and $s > \frac{1+\sqrt{5}}{2}$ for the latter. But the corresponding lower and upper bounds are still unknown. More optimal bounds for different semi-online variants on $m = 2, 3, 4$ uniform machines can be found in [57, 59].

Preemptive machine covering problems are much more complicated than corresponding makespan minimization problems. Due to the particularity of the definition of the objective, whether idle time is allowed should be taken into special consideration. The lower bound of $Pm|pmtn, sum|C_{min}$ is $\frac{2m-3}{m-1}$, and there exists optimal algorithm with matched competitive ratio when $m = 2, 3$ [98]. For $Q2|pmtn, max|C_{min}$, the optimal bound is $\frac{s^2+3s+1}{s^2+2s+1}$ [99]. However, the algorithm may introduce idle time before J_{max} arrives when $s > s' \approx 1.247$. Here, s' is the positive root of $s^3+s^2-2s-1 = 0$. For $Q2|pmtn, decr|C_{min}$, the optimal bound is

$$\begin{cases} \dfrac{2s+3}{2s+2}, & 1 \le s \le 3, \\ \dfrac{s^2+3s}{s^2+2s+1}, & s > 3. \end{cases}$$

The optimal algorithm also needs to introduce idle time when assigning the first job if $s > \frac{\sqrt{6}}{2}$. Whether there exist algorithms for $Q2|pmtn, sum|C_{min}$ and $Q2|pmtn, opt|C_{min}$ that can always obtain the optimal schedule remains open (Fig. 5).

3.2.4 Other Results

Due to the difficulty in competitive analysis with multiple parameters, there is little study on non-preemptive semi-online problems on more than two uniform machines. Azar and Epstein [11] proposed algorithms with competitive ratio both m for $Qm|opt|C_{min}$ and $Qm|decr|C_{min}$, respectively. These algorithms are optimal in the overall sense.

Even less is known about randomized algorithms for semi-online problems. Seiden et al. [154] proposed a barely random algorithm for $P2|decr|C_{max}$ with competitive ratio $\frac{8}{7}$, and no randomized algorithm can achieve a better competitive ratio.

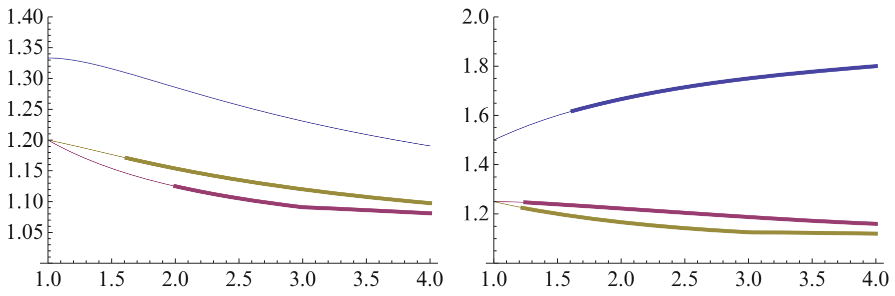


Fig. 5 *Left*: the optimal bounds of $Q2|pmtn|C_{max}$ (*top*), $Q2|max, pmtn|C_{max}$ (*middle*), and $Q2|decr, pmtn|C_{max}$ (*bottom*). *Right*: the optimal bounds of $Q2|pmtn|C_{min}$ (*top*), $Q2|max, pmtn|C_{min}$ (*middle*), and $Q2|decr, pmtn|C_{min}$ (*bottom*). Thick segments represent the intervals of s where idle time is necessary

3.3 Results on Basic Semi-online Models of Type II

3.3.1 Buffer

Evidently, it is more likely to design algorithm with a better performance if larger buffer is allowed. However, a buffer of size more than 1 is dispensable for two identical machines. That is to say, the lower bound of the problem with a buffer of size arbitrary large equals to the competitive ratio of an algorithm using a buffer of size only 1, which is $\frac{4}{3}$ [111, 187]. However, such phenomenon does not occur in the case where there are more than two machines or machines have different speeds. Therefore, the algorithm, as well as its corresponding competitive ratios, may be relevant to the size of the buffer. However, it seems impossible and also redundant to obtain optimal algorithms for any values of K. An alternate way is to find an algorithm satisfying *super optimality*. An algorithm with competitive ratio r which uses a buffer of size K is *super optimal* if no algorithm which uses a buffer of size arbitrary large can have a competitive ratio smaller than r, where K is a constant number. Clearly, if two algorithms are both super optimal, then the algorithm which uses a smaller buffer is better. However, research on the problem with a buffer of limited size is also of significance, since in practical application, the buffer size is limited by the cost, space, or other additional restrictions.

Main results on the *buffer* variant on arbitrary number of machines are summarized in Table 7. For $Pm|buffer|C_{max}$ [63], $Pm|pmtn, buffer|C_{max}$ [51], and $Qm|buffer|C_{min}$ [73], super optimal algorithms have been obtained (in the overall sense for the last problem on uniform machines). Note that the super optimal bound for $Pm|pmtn, buffer|C_{max}$ is not an increasing function of m, and the problem cannot be solved by using the general framework included in [59].

For $Q2|buffer|C_{max}$, it is possible to obtain lower and upper bounds as functions of s [52]. A buffer of size 2 is enough to achieve the super optimal bound

$$\begin{cases} \dfrac{s^2+2s+1}{s^2+s+1}, & 1 \le s \le \frac{1+\sqrt{5}}{2}, \\ \dfrac{s^2}{s^2-s+1}, & \frac{1+\sqrt{5}}{2} \le s \le 2, \\ \dfrac{s+2}{s+1}, & s \ge 2. \end{cases}$$

Moreover, the optimal algorithm only needs a buffer of size 1 when $s \ge 2$. However, when the buffer size reduces to 1, the optimal bound increases to $\dfrac{s+2}{s+1}$ for $\sqrt{2} < s < 2$, and there exists an algorithm with competitive ratio $\dfrac{2(s+1)}{s+2}$ while the lower bound is $\max\left\{s, \dfrac{s^2+2s+1}{s^2+s+1}\right\}$ $1 \le s \le \sqrt{2}$.

Table 7 Main results for the *buffer* variant on arbitrary number of machines

Problem		Buffer size/lower bound		Buffer size/competitive ratio	
		Buffer size	Lower bound	Buffer size	Competitive ratio
$Pm\|buffer\|C_{max}$ [63] (except marked)		Constant	γ_m [a]	$\left\lceil (1+\frac{2}{\gamma_m})m \right\rceil + 1$[a]	γ_m [a]
		$\lfloor \frac{m}{2} \rfloor - 1$	$\frac{3}{2}$	$\frac{3m}{2}$	$\frac{3}{2}$ [112]
		$\lfloor \frac{m}{8} \rfloor - 1$ $(m \geq 8)$	$1 + \frac{1}{\sqrt{2}}$	m	$1 + \frac{\gamma_m}{2}$ [a]
				$K \in [1, \frac{m+1}{2}]$	$2 - \frac{1}{m-K}$
$Qm\|buffer\|C_{max}$				$m-1$	$2+\epsilon$ $(\epsilon > 0)$ [63]
				m	$2 - \frac{1}{m} + \epsilon(\epsilon > 0)$ [112]
$Pm\|pmtn$,	m is even	Constant	$\frac{4}{3}$	$\frac{m}{2} - 1$	$\frac{4}{3}$
$buffer\|C_{max}$ [51]	m is odd	Constant	$\frac{4m^2}{3m^2+1}$	$\frac{m-1}{2}$	$\frac{4m^2}{3m^2+1}$
	General m	$\leq \lceil \frac{m-2}{2} \rceil$	[b]		
		1 $(m \geq 5)$	$\frac{4m^3-12m^2+4m}{3m^3-11m^2+18m-24}$		
$Pm\|buffer\|C_{min}$ [73]		Constant	$\sum_{i=1}^{m} \frac{1}{i}$	$m-1$	$1 + \sum_{i=1}^{m-1} \frac{1}{i}$
		$K = f(m) < m$	$\frac{m}{f(m)+1}$		
$Qm\|buffer\|C_{min}$ [73]		constant	m	$m+1$	m
		$m-2$	∞	$m-1$	$\frac{2m-1+\sqrt{4m^2-8m+5}}{2}$

[a] γ_m is the smallest positive solution of the equation

$$\left\lceil m - \frac{m}{\gamma_m} \right\rceil \frac{\gamma_m}{m} + (\gamma_m - 1) \sum_{i=m-\frac{m}{\gamma_m}}^{m-1} \frac{1}{i} = 1.$$

For any given m, γ_m can be calculated numerically, for example, $\gamma_3 \approx 1.3666$, $\gamma_4 \approx 1.375$. Moreover, $\lim_{m\to\infty} \gamma_m = \gamma_\infty \approx 1.4659$, where γ_∞ is the smallest solution of equation $\gamma_\infty e^{\gamma_\infty} = -\frac{1}{e^2}$

[b] $\max_{K+1 \leq t \leq m+1} \frac{m^t}{(t+tK-tm-Km)m^{K-1}(m-1)^{t-K-1}+(K+m)m^{t-1}}$

3.3.2 Reassignment

Recall that the online over list model is typically characterized by the disallowance of reassignment. Since semi-online can be viewed as relaxation of online, it is natural to consider the situation where some jobs can be actually reassigned, which is also of realistic interest from an application perspective.

Variants in this category can be classified into two main types: sequential reassignment and final one-off reassignment. For the first type, reassignment can be done when every new job arrives, while for the second one, reassignment can only be done after all jobs have been assigned. For both cases, however, there should be some constraints on the reassignment; otherwise, the problem will reduce to an offline fashion.

In [160], Sanders et al. considered the *proportional sequential reassignment* model. When a job with processing time p_j arrives, some jobs with total processing time at most δp_j can be reassigned, where δ is the *migration factor*. Obviously, the competitive ratio r of an algorithm will decrease when δ increases. For identical machines with objective to minimize makespan, they designed algorithms for different value of δ. Some combinations of r and δ are $(r, \delta) = \left(\frac{3}{2}, \frac{4}{3}\right)$, $(r, \delta) = \left(\frac{3}{2} - \frac{1}{2m}, 2\right)$, and $(r, \delta) = \left(\frac{4}{3}, \frac{5}{2}\right)$, respectively. They even obtained a family of online algorithms with competitive ratio $1 + \varepsilon$, where $\varepsilon > 0$ can be arbitrarily close to 0, while δ only depends exponentially on $\frac{1}{\varepsilon}$. However, only for the case of $m = 2$ and $\delta = 1$, an algorithm with competitive ratio $\frac{7}{6}$ has been proved to be optimal. For the machine covering problem, they obtained an algorithm with competitive ratio 2 for $\delta = 1$.

Epstein and Levin [71] studied the above variants in a preemptive setting. Since a job can be split, the scheduler is allowed to reassign only part of the job. As a result, the migration factor does not reckon in the entire processing time but rather only the reassigned fraction of that. They presented a $(r, \delta) = (1, 1 - \frac{1}{m})$ algorithms for m identical machines, and the migration factor cannot be improved. For m uniform machines, a migration factor at least $m - 1$ is needed to obtain an algorithm with competitive ratio 1.

The restriction of reassigned job can be made through numbers of jobs instead of total processing times. That is at most G already assigned jobs can be reassigned when a new job arrives. Such variant can be called as *sequential reassignment with quantitative restriction* and is denoted *reassignSQ*. It is stronger than *buffer* since the former can simulate the latter by the following steps: First, temporarily set the jobs on any one of the machines as if they were stored in a buffer. And then reassign them on the machine as if jobs in the buffer were assigned. Contrariwise, a job can no longer be reassigned unless it is in the buffer. In brief, *reassignSQ* can do what *buffer* can, but not vice versa.

For $Q2|reassignSQ|C_{max}$, Dósa et al. [55] proved that the optimal bound when $G \geq 2$ coincides with the optimal bound of $Q2|buffer|C_{max}$ with buffer size $K \geq 2$. However, when $G = 1$, the lower bound is

$$\begin{cases} \dfrac{s^2+2s+1}{s^2+s+1}, & 1 \le s \le \dfrac{1+\sqrt{5}}{2}, \\ \dfrac{s+1}{2}, & \dfrac{1+\sqrt{5}}{2} \le s \le \sqrt{3}, \\ \dfrac{s+2}{s+1}, & s \ge \sqrt{3}, \end{cases}$$

while the upper bound is

$$\begin{cases} \dfrac{s^2+2s+1}{s^2+s+1}, & 1 \le s \le s'', \\ \dfrac{s+\sqrt{5s^2+8s+4}}{2(s+1)}, & s'' \le s \le \sqrt{3}, \\ \dfrac{s+2}{s+1}, & s \ge \sqrt{3}, \end{cases}$$

where s'' is the root of equation $s^3 - s - 1 = 0$. The algorithm is not optimal when $s \in [s'', \sqrt{3}]$, and it is not identical with the bound of $Q2|buffer|C_{max}$ with buffer size $K = 1$ (Fig. 6).

Several final one-off reassignment models were first proposed by Tan and Yu [167]. One is that when all jobs have been assigned, at most H arbitrary jobs can be reassigned. The model is denoted as *reassignFA*. Tan and Yu proved that optimal bound for two identical machines with objective to minimize makespan is $\frac{4}{3}$, and the optimal algorithm only needs to reassign at most one job. Albers and Hellwig [4] generalized the model to m identical machines. Interestingly, the competitive ratio γ_m of the given algorithm is the same as that of the optimal algorithm for $Pm|buffer|C_{max}$, and the algorithm only reassigns at most $\left(\lceil \frac{2-\gamma_m}{(\gamma_m-1)^2} \rceil + 4\right) m$ jobs. Moreover, no algorithm can achieve a better competitive ratio if $H = o(n)$. They also obtained algorithms with different value of r and H, such as $(r, H) = \left(\frac{5}{3}, 4m\right)$ and $(r, H) = \left(\frac{7}{4}, \frac{5}{2}m\right)$. For two uniform machines, the optimal bound of $Q2|reassignFA|C_{max}$ when $H \ge 2$ is again the same as that of $Q2|buffer|C_{max}$ when $K \ge 2$. Also, the optimal algorithm reassigns at most one job when $s > 2$. However, if only one job can be reassigned, the competitive ratio of the best known algorithm is also the same as that of $Q2|buffer|C_{max}$ with buffer size $K = 1$, but it is larger than that of $Q2|reassignSQ|C_{max}$ with $G = 1$ (Fig. 6).

Another final one-off reassignment model is denoted *reassignFE*. After all jobs have been arrived, the *last* job assigned to *each* machine can be reassigned. For two identical machines with objective to minimize makespan, the optimal bound is $\sqrt{2}$ [167]. In fact, in such situation at most one job is needed to be reassigned [132]. Cao and Liu [25] generalized the result to two uniform machines and obtain the optimal bound

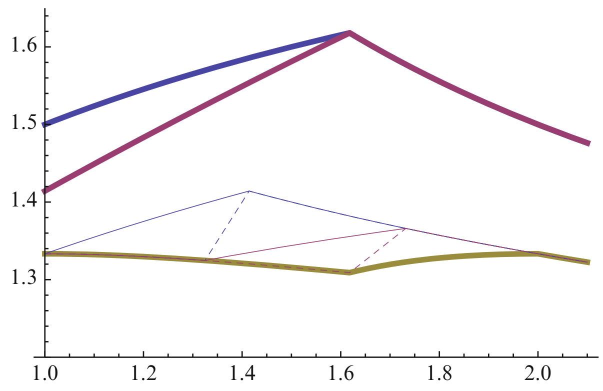


Fig. 6 The optimal bounds (*thick*) of $Q2||C_{max}$ (*top*), $Q2|reassignFE|C_{max}$ (*middle*), and $Q2|buffer|C_{max}$ with $K \geq 2$ (also $Q2|reassignSQ|C_{max}$ with $G \geq 2$ and $Q2|reassignFA|C_{max}$ with $H \geq 2$) (*bottom*), the upper bounds (*solid*) and lower bounds (*dash*) of $Q2|buffer|C_{max}$ with $K = 1$ (also $Q2|reassignFA|C_{max}$ with $H = 1$) (*top*) and $Q2|reassignSQ|C_{max}$ with $G = 1$ (*bottom*)

$$\begin{cases} \sqrt{s+1}, & 1 \leq s \leq \frac{1+\sqrt{5}}{2}, \\ \frac{s+1}{s}, & s > \frac{1+\sqrt{5}}{2}. \end{cases}$$

Comparing it with the optimal bound of the pure online problem [72], reassignment is useless when $s > \frac{1+\sqrt{5}}{2}$ (Fig. 6).

3.4 Results on Combined Semi-online Models

3.4.1 UB and LB

Though *UB* and *LB* are both valueless, their combination can be valuable. Such situation is very common in practice, since schedulers tend to estimate the processing time of jobs. However, it would be of little significance if the estimation is too rough to neglect the error. Hence, it is necessary to take influence of a parameter β, defined as $\beta = \frac{UB}{LB} \geq 1$, into consideration when dealing with competitive analysis. Main results of this variant are summarized in Table 8.

Note that LS remains optimal for any value of β for $P2|UB\&LB|C_{max}$ and $Pm|UB\&LB|C_{min}$. It is not true for $P3|UB\&LB|C_{max}$, though LS is also the optimal algorithm for $P3||C_{max}$. He and Dósa [92] proved the competitive ratio of LS is

$$\begin{cases} \frac{2\beta+1}{3}, & \beta \in [1, \frac{3}{2}], \\ \frac{2\beta+3}{\beta+3}, & \beta \in [\frac{3}{2}, \sqrt{3}], \\ \frac{\beta+1}{2}, & \beta \in [\sqrt{3}, 2], \\ 2 - \frac{1}{\beta}, & \beta \in [2, 3], \\ \frac{5}{3}, & \beta \in [3, +\infty), \end{cases}$$

and LS is optimal only when $\beta \in [1, \frac{3}{2}] \cup [\sqrt{3}, 2] \cup [6, +\infty)$. When $\beta \in [2, 6]$, there exists an improved algorithm with competitive ratio

$$\begin{cases} \frac{3}{2}, & \beta \in \left[2, \frac{5}{2}\right], \\ \frac{4\beta+2}{2\beta+3}, & \beta \in \left[\frac{5}{2}, 3\right], \\ \frac{5}{3} - \frac{1}{18}\min\left\{\frac{6-\beta}{18}, \frac{3}{103}\right\}, & \beta \in [3, 6]. \end{cases}$$

Nevertheless, the new upper bound matches the lower bound of the problem only when $\beta \in [2, \frac{5}{2}]$. By the definition of the paradigm, the lower bound of the problem is obviously a nondecreasing function of β. For the remaining intervals of β where optimal algorithm has not been obtained, nonconstant lower bound of the problem, equaling to $\frac{7\beta+4+\sqrt{\beta^2+8\beta+4}}{2\beta+2+2\sqrt{\beta^2+8\beta+4}}$, can be found only in the situation where $\beta \in [\frac{5}{2}, 3]$. It is implied again from the results stated above that scheduling for two machines is of enormous difference in essence from that of three machines, while the latter is much more complicated.

A similar model of LB and max is proposed by Cao et al. [26]. It is assumed that $\frac{p_{max}}{\beta} \le p_j \le p_{max}$ for any j, and there always exists a job of processing time p_{max}. They proved that the optimal bound of $P2|LB\&max|C_{max}$ is

$$\begin{cases} \frac{\beta+1}{2}, & \beta \in [1, \frac{4}{3}], \\ \frac{4\beta+4}{3\beta+4}, & \beta \in [\frac{4}{3}, \sqrt{2}], \\ \frac{2\beta}{\beta+1}, & \beta \in [\sqrt{2}, 2], \\ \frac{4}{3}, & \beta \in [2, +\infty). \end{cases}$$

It is smaller than that of $P2|LB\&UB|C_{max}$ for some value of β (Fig. 7).

3.4.2 *sum/opt* and *max/UB*

Angelelli et al. [7, 8] studied the problem $P2|sum\&UB|C_{max}$. Lower bounds as functions of $\gamma = \frac{2UB}{P}$ and several algorithms are proposed. Optimal bounds are achieved for majority value of γ (see Figs. 4–6 of [8]).

Table 8 Main results for the $UB\&LB$ variant

Problem	Interval of β where the partial information is valuable and the optimal bound	Interval of β where the partial information is valueless[a]
$P2\|UB\&LB\|C_{max}$ [97]	$\beta \in [1, 2]$, $\frac{1+\beta}{2}$	$\beta \in [2, +\infty]$
$Pm\|UB\&LB\|C_{max}, m \geq 3$ [92][b]	$\beta \in [1, \frac{m}{m-1}]$, $1 + \frac{(m-1)(r-1)}{m}$	
$Pm\|UB\&LB\|C_{min}$ [91]	$\beta \in [1, m]$, β	$\beta \in [m, +\infty)$
$Pm\|pmtn, UB\&LB\|C_{max}$ [94]	[c]	$\beta \in [(\frac{m}{m-1})^{m-1}, +\infty)$
$Q2\|pmtn, UB\&LB\|C_{max}$ [56, 93]	$\beta \in [1, 2s]$,[d]	$\beta \in [2s, +\infty)$
$Q2\|pmtn, UB\&LB\|C_{min}$ [105]	$\beta \in [1, 2s]$, $\frac{2s+2+\beta}{2s+2}$	$\beta \in [2s, +\infty)$

[a]Valueless implies that the optimal bound is the same as the optimal bound of the corresponding pure online problem

[b]Only partial results about this problem have been obtained

[c]Optimal algorithm can be obtained by using the general framework included in [59]. The analytical lower bound for $\beta \in \left[1, \left(\frac{m}{m-1}\right)^{m-1}\right]$ is at least

$$\frac{m\left(\frac{m}{m-1}\right)^k + (m-k-1)\beta}{(2m-k-1)\left(\left(\frac{m}{m-1}\right)^k - 1\right) + (m-k) + \frac{(m-k)(m-k-1)\beta}{2m}},$$

$$\left(\frac{m}{m-1}\right)^k \leq \beta \leq \left(\frac{m}{m-1}\right)^{k+1}, k = 0, 1, \ldots, m-2.$$

Whether it is the optimal bounds remains open except for $m = 2, 3$.

[d]The optimal bound is

$$\begin{cases} \dfrac{1+s+\frac{\beta}{2}+\frac{\beta s}{2}}{1+s+\frac{\beta s}{2}}, & 1 \leq s \leq 2 \text{ and } \beta \in [1, 2s], s > 2 \text{ and } \beta \in \left[1, \frac{2}{s-1}\right] \cup [2s-2, 2s] \\ \dfrac{(1+\beta)(1+s)s}{1+s+s^2+s\beta^2}, & s \geq 2 \text{ and } \beta \in \left[\frac{2}{s-1}, s\right] \\ \dfrac{s^2+s}{s^2+1}, & s \geq 2 \text{ and } \beta \in [s, 2s-2]. \end{cases}$$

If idle time is not allowed, then the (deterministic) lower bound will be larger, but no online algorithm with matched competitive ratio has been obtained [93]

There are other papers considering the combination of $sum(opt)$ and max. However, only overall bounds (the maximum bound among all combinations of P (or C^*) and p_{max}) are obtained. For example, the optimal bound for $Pm|sum\&max|C_{min}$ [166] is

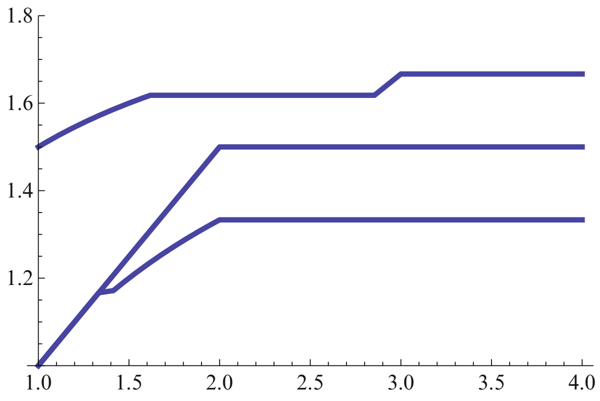


Fig. 7 The optimal bounds of $P2|\mathcal{M}_j(GoS), LB\&UB|C_{max}$ (*top*), $P2|LB\&UB|C_{max}$ (*middle*), and $P2|LB\&max|C_{max}$ (*bottom*)

$$\begin{cases} \frac{5}{4}, & m = 2, \\ \frac{3}{2}, & m = 3, \\ m-2, & m \geq 4. \end{cases}$$

For the makespan minimization problem, optimal bounds are obtained only on a small number of machines (See Table 5). If preemption is allowed, optimal bound of $Q3|pmtn, sum\&max|C_{max}$ for any combinations of P and p_{max} and any values of s_i, $i = 1, 2, 3$, can be obtained by using the general framework included in [59].

3.4.3 End of Sequence

Zhang and Ye [190] proposed an interesting semi-online variant, which was called "end of sequence" (eos for short) later [70]. It is known that the last job has the largest processing time, and the scheduler will be informed whether the current arrived job is the last one. Such variant can be viewed as a weaker version of the combined semi-online variant $num\&incr$. Recall that both num and $incr$ are valueless if they are considered solely. Thus, it is likely that eos is adjacent to the boundary between valuable semi-online variants and valueless ones.

Zhang and Ye [190] proved the optimal bounds of $P2|eos|C_{max}$ and $P3|eos|C_{max}$ are $\sqrt{2}$ and $\frac{3}{2}$, respectively. Note that the lower bound of $P2|eos|C_{max}$ does not hold for $P2|num\&incr|C_{max}$, the optimal bound for the latter problem remains open. Epstein and Ye considered the problems $Q2|eos|C_{max}$ and $Q2|eos|C_{min}$. Optimal algorithms for almost all values of s are given [70] (Table 6, Figs. 3 and 4).

Table 9 Results for the non-preemptive problems with inexact partial information

Problem	The interval of α where the inexact information is valuable	The interval of α where the obtained algorithm is optimal	The gap between the upper and lower bound	The total length of non-optimal intervals	The interval of α where the inexact information is valueless
$P2\|inexact\ sum\|C_{max}$	$[1, \frac{3}{2})$	$[\frac{3+\sqrt{21}}{6}, \frac{3}{2})$	≤ 0.0303	≤ 0.2635	$[\frac{3}{2}, \infty)$
$P2\|inexact\ opt\|C_{max}$	$[1, \frac{3}{2})$	$[\frac{3+\sqrt{21}}{6}, \frac{1+\sqrt{10}}{3}] \cup [\frac{6-\sqrt{10}}{2}, \frac{3}{2})$	≤ 0.0303	≤ 0.2957	$[\frac{3}{2}, \infty)$
$P2\|inexact\ max\|C_{max}$	$[1, 2)$	$[1, \sqrt{5}-1]$	≤ 0.0445	≤ 0.7640	$[2, \infty)$

Table 10 Results for the preemptive problems with inexact partial information

Problem	The interval of α where the inexact information is valuable and the optimal bound	The interval of α where the inexact information is valueless
$Q2\|pmtn, inexact\ opt\|C_{max}$	$\alpha \in \left[\left(\frac{m}{m-1}\right)^k, \left(\frac{m}{m-1}\right)^{k+1}\right]$, $\frac{1}{1-\left(\frac{m}{m-1}\right)^k+(m-k)+\frac{m-k}{m\alpha}}$, $k = 0, 1, \cdots, m-2$	$\left[\left(\frac{m}{m-1}\right)^{m-1}, \infty\right)$
$Pm\|pmtn, inexact\ opt\|C_{max}$	$[1, s+1)$, $\frac{\alpha(s+1)}{\alpha s+1}$	$[s+1, \infty)$
$Q2\|pmtn, inexact\ max\|C_{max}$	$[1, s+1)$, $\frac{(s+1)(1+s+\alpha s)}{1+2s+s^2+rs^2}$	$[s+1, \infty)$

3.4.4 Miscellanies

The optimal bounds of $P2|decr\&sum|C_{max}$ and $P2|decr\&opt|C_{max}$ are both $\frac{10}{9}$ [65, 164]. Azar and Epstein [11] designed an algorithm for $Qm|decr\&opt|C_{min}$ with an overall competitive ratio 2, which is optimal in the overall sense.

By using the general framework included in [59], it can be proved that the overall optimal bound of $Qm|pmtn, decr\&sum|C_{max}$ is $\frac{12}{11}$ when $m = 3$ and $\frac{10}{9}$ when $m = 4$. For the problem $Pm|pmtn, decr\&buffer|C_{max}$, Dósa and Epstein [51] proved that a buffer of size $m-1$ is enough for an online algorithm to produce the optimal schedule, and the optimal bounds for any buffer size $K < m-1$ is $\max_{0\leq\mu\leq m-K-1} \frac{2m(m+\mu)}{2m^2+2K\mu+\mu^2+\mu}$.

Dósa and He [53] studied two problems $P2|sum\&buffer|C_{max}$ and $P2|sum\&parallel(2)|C_{max}$. They designed two optimal algorithms with competitive ratio $\frac{5}{4}$ and $\frac{6}{5}$, respectively. Min et al. [132] studied the problem $P2|sum\&reassignFE|C_{max}$ and proposed an optimal algorithm with competitive ratio $\frac{5}{4}$.

3.5 Results on Disturbed Semi-online Models

In regard to disturbed semi-online model, there are fewer variants brought forward and results achieved. Study on problems with inexact partial information is much more difficult than the problem with exact partial information. One reason is that it is expected to find the upper and lower bound as functions of disturbance parameter α. If preemption is not allowed, only problems on two identical machines have been studied [165]. Related results are summarized in Table 9, and the detailed bounds are omitted here. If preemption is allowed, some research approaches and techniques of previous variants can be generalized to m identical machines or two uniform machines [106]. These results are summarized in Table 10. By using the general framework included in [59], even the optimal bound of $Q3|pmtn, inexact\ sum|C_{max}$ for any α and any values of s_i, $i = 1, 2, 3$ can be obtained. However, the known algorithms for the non-preemptive problem are not optimal for all values of α, and also there is no study on the problems of $Pm|pmtn, inexact\ max|C_{max}$, $Pm|pmtn, inexact\ sum|C_{max}$, $Q2|pmtn, inexact\ sum|C_{max}$.

4 Online Over Time

Scheduling jobs that arrive over time is usually called online over time model. A number of researches are done with respect to the minimization problems for the three following classical objective functions: the makespan C_{max}, the total completion time $\sum_{j=1}^{n} C_j$, and the total weighted completion time $\sum_{j=1}^{n} w_j C_j$. For these objective functions, and some special cases, deterministic and randomized online algorithms are analyzed deriving different approximation guarantees. Results on online over time model are summarized in Table 11.

4.1 Single Machine Scheduling

4.1.1 Minimize the Total Completion Time

Problems of minimizing the total completion time on a single machine are extensively studied. Phillips et al. [142] presented a 2-competitive algorithm based on the optimal preemptive schedule, which can be found by the $SRPT$ (Shortest Remaining Processing Time first) rule in polynomial time [148]. There is another 2-competitive algorithm, namely, delayed SPT ($D-SPT$), provided by Hoogeveen and Vestjens [100].

Algorithm D-SPT

At any time t a machine is idle and a job is available, choose a job with shortest processing time, say J_i. If there are more than one job with the same processing

Table 11 The best known results on over time model

Problem	Jobs	Lower bound		Upper bound		Randomized lower bound		Randomized upper bound	
$1\|r_j\|\sum C_j$	None	2	[100]	2	[100, 127, 142]	$\frac{e}{e-1}$	[162]	$\frac{e}{e-1}$	[29]
	pmtn	1	[148]	1	[148]	1		1	[148]
	res	1.2108	[69]	1.5	[175]	1.1068	[69]	$\frac{e}{e-1}$	[29]
	$UB\&LB$	Φ	[170]	$\frac{\sqrt{1+\beta(\beta-1)}+\beta-1}{\beta}$	[170]				
$1\|r_j\|\sum w_j C_j$	None	2	[100]	2	[5]	$\frac{e}{e-1}$	[162]	1.6853	[83]
	pmtn	1.0730	[69]	1.57	[158]	1.0389	[69]	$\frac{4}{3}$	[149]
	res	1.2232	[69]	2	[5]	1.1161	[69]	1.6853	[83]
	$UB\&LB$	$1+\frac{\sqrt{4\beta^2+1}-1}{2\beta}$	[171]	$1+\frac{\sqrt{4\beta^2+1}-1}{2\beta}$	[171]				
	release[a]	Ψ	[89]	Ψ	[89]				
$P\|r_j\|\sum C_j$	None	1.309	[176]	$1.791+o(m)$	[159]	1.157	[152]	$1.791+o(m)$	[159]
	pmtn	$\frac{21}{19}$	[184]	$\frac{5}{4}$	[45, 159]	1.047	[176]	1.25	[159]
$P\|r_j\|\sum w_j C_j$	None	1.309	[176]	$1.791+o(m)$	[159]	1.157	[152]	$1.791+o(m)$	[159]
	pmtn	1.114	[184]	$1.791+o(m)$	[159]	1.047	[176]	$1.791+o(m)$	[159]
$P\|r_j\|C_{max}$	None	1.3473	[30]	1.5	[30]	$4-2\sqrt{2}$	[162]		
$P2\|r_j\|C_{max}$	None	$\frac{5-\sqrt{5}}{2}$	[30]	$\frac{5-\sqrt{5}}{2}$	[138]	1.212	[138]		
$Q\|r_j\|\sum C_j$	None	1.309	[176]	$\frac{\sqrt{4m-3}+3}{2}$	[122, 124]				
$Q2\|r_j\|\sum w_j C_j$	pmtn	1		2	[122]				

[a]The semi-online problems that release times of all jobs are known in advance

time, choose the one with the smallest release time. If $p_i \leq t$, then process J_i; otherwise, wait until time p_i or until a new job arrives, whichever happens first.

A very important idea in $D - SPT$ algorithm is to shift the release time of jobs before scheduling, which was adopted by Stougie as well (cited in [176]). Lu et al. [127] generalized this idea and proposed a class of online algorithms that have the same competitive ratio of 2. Note that for this problem, it is shown by Hoogeveen and Vestjens [100] that any online algorithm must have a competitive ratio no smaller than 2, and hence, all above algorithms are best possible.

In online over time problems, it is commonly interesting to know whether or how much the competitive ratio would decrease if restarts are allowed. Vestjens [176] first showed a lower bound for the competitive ratio of any online algorithm. Later, Epstein and van Stee [69] improved the bound from 1.112 to 1.2108. The algorithm provided by van Stee and Poutré [175] indicates that the upper bound of the problem allowing restarts is at most $\frac{3}{2}$.

In contrast with online over list, only a few researches are found involving in semi-online variants, even though there are actually more semi-online models than those in online over list model. Tao et al. [170] investigated the problem with the knowledge of the upper and lower bounds of the processing times, $UB\&LB$. A semi-online algorithm with competitive ratio $\frac{\sqrt{1+\beta(\beta-1)}+\beta-1}{\beta}$, as well as a lower bound

$$\Phi = \min_{s \geq 0} \max \left\{ 1+s, \max_{k \in Z^+} \left\{ 1 + \frac{2k(\beta-1)}{2\beta(k+1)s + 2\beta + k^2 + 3k} \right\} \right\},$$

is proposed, where $\beta = \frac{UB}{LB} \geq 1$.

4.1.2 Minimize the Total Weighted Completion Time

Indeed, the weighted problem is more general and hence more difficult. Anderson and Potts [5] considered the delayed weighted shortest processing time rule ($D - WSPT$), which is a direct generalization of the $D - SPT$ algorithm. By introducing a new proof technique, they showed that $D - WSPT$ has a competitive ratio of 2 and therefore matches the known lower bound [100]. If restarts are allowed, the lower and upper bounds of this problem are 1.2232 and 2, respectively [5, 69].

If preemption is allowed, unlike the unweighted version, Vestjens [176] showed that any preemptive algorithm must have a competitive ratio no smaller than 1.0333. The lower bound is later improved to 1.0730 by Epstein and van Stee [69]. Several algorithms are proved to be 2-competitive, including $D - WSPT$, $P - WSPT$, and $WSRPT$ [130, 146]. Here, $P - WSPT$ (preemptive $WSPT$) schedules at any time the job with the largest ratio of weight over processing time, while $WSRPT$ (weighted shortest remaining processing time first) schedules at any time the job with the largest ratio of weight over remaining processing time. The instance given by Xiong and Chung [184] recently showed that the competitive ratio of

$WSRPT$ cannot be smaller than 1.215. Though there is a big gap between the lower and upper bounds, developing an online algorithm with a competitive ratio better than 2 is a hard work. Breakthrough on this problem is finally made by Sitters [158]. By using a parameter $c \geq 1$ and applying the $P - WSPT$ rule with a restriction that a job cannot be preempted at time t if it can be completed before time ct, a class of online algorithms, namely, $ONLINE(c)$, are obtained and shown to be c-competitive for any $c \geq \xi$, where $\xi \approx 1.57$ is the real root of $2\xi^3 - 4\xi^2 + 2\xi - 1 = 0$. Hence, there exists a ξ-competitive algorithm for the preemptive problem.

If the release times of all jobs are known in advance, Hall et al. [89] showed that the problem has a lower bound of

$$\Psi = \max_{1 \leq j \leq n} \min_{0 \leq l \leq j-1} \frac{r_{l+1} + r_l + \sqrt{(r_{l+1} - r_l)^2 + 4r_j r_{l+1}}}{2r_{l+1}}$$

where $r_1 \leq r_2 \cdots \leq r_n$ are the release times of all jobs. It can be proved that Ψ is in between $\frac{\sqrt{5}+1}{2}$ and 2. An online algorithm with matched competitive ratio is provided as well. Clearly, this variant will never be classified into online over list model. The semi-online problem $UB\&LB$ is considered by Tao et al. [171], where they presented an optimal algorithm with competitive ratio $1 + \frac{\sqrt{4\beta^2+1}-1}{2\beta}$.

4.1.3 Randomized Algorithm and Lower Bound

Chekuri et al. [29] designed a randomized algorithm to minimize the total completion time with competitive ratio $\frac{e}{e-1}$. Their algorithm is intriguing as it beats the lower bound 2 for deterministic online algorithms [100]. Moreover, it is also optimal since its competitive ratio matches the randomized lower bound given by Stougie and Vestjens [162]. If restarts are permitted, Epstein and van Stee [69] proved that any randomized algorithm must have a competitive ratio at least 1.1068. In the same paper, they also considered problems aiming at minimizing the total weighted completion time, where a lower bound 1.0389 and a lower bound 1.1161 for the preemptive problem and the problem allowing restarts are proposed, respectively. Regarding the upper bounds, Schulz and Skutella [149] provided a $\frac{4}{3}$-competitive randomized algorithm for the preemptive problem. Goemans et al. [83] presented a randomized algorithm with competitive ratio 1.6853 for the problem allowing restarts.

4.2 Results on Parallel Machine Scheduling

4.2.1 Minimize the Makespan

For the makespan minimization problems, it should be emphasized that Shmoys et al. [157] have described a general method to use offline algorithms to obtain online algorithms for problems of online over time model. However, the method always produces a competitive ratio at least 2, even if the corresponding offline problem can be solved to optimality. In the same paper [157], Shmoys et al. proved a lower bound of $\frac{10}{9}$ for the problem on m identical machines. Chen and Vestjens [30] improved the lower bound to $1 + \alpha$ for $m \geq 3$ and to $\frac{5-\sqrt{5}}{2}$ for $m = 2$, where $\alpha \approx 0.3473$ is the solution of $\alpha^3 - 3\alpha + 1 = 0$. Moreover, they proposed a simple algorithm, namely, online Longest Processing Time first (online LPT).

Algorithm Online LPT

At any time a machine becomes idle for processing, schedule an available job with the longest processing time.

The competitive ratio of online LPT is shown to be $\frac{3}{2}$. As far as we know, this is the best result on this problem except that in [138], where Noga and Seiden presented an improved algorithm $SLEEP$ for $m = 2$. The difference between $SLEEP$ and online LPT is that it might wait for some time even when there is an idle machine and an available job. The idea makes the algorithm optimal for $m = 2$ and, hence, beat online LPT. For further study, one may ask how to extend the algorithm and whether it can beat online LPT for a general number of machines. Nevertheless, there are still no such study by now. The randomized lower bounds on makespan minimization problem are studied in [162] and [138], where a lower bound of $4 - 2\sqrt{2}$ for a general number of machines and a lower bound of 1.21207 for $m = 2$ are provided, respectively.

4.2.2 Minimize the Total (Weighted) Completion Time

For the problem of minimizing the total completion time, any deterministic algorithm is at least $\frac{21}{19} \approx 1.105$-competitive for the preemptive version and is at least 1.309-competitive for the non-preemptive version [176, 184]. Moreover, any randomized algorithm is at least 1.047-competitive for the preemptive version and is at least 1.157-competitive for the non-preemptive version [176].

The simple and well-known $SRPT$ algorithm plays a significant rule in preemptive scheduling of minimizing the total completion time. As we have seen before, it produces an optimal schedule for the single machine case [148]. The upper bound of $SRPT$ for m identical machines was known to be 2 [142] for a long time until recently, Chung et al. [45] claimed that it is at most 1.86. Shortly after that, Sitters [159] improved it to $\frac{5}{4}$.

For the preemptive problem of minimizing the total weighted completion time, 2-competitive online algorithm is found in [131]. The lower bound of the problem is proved to be $\frac{16-\sqrt{14}}{11} \approx 1.114$ [184]. For the non-preemptive version, a randomized algorithm with competitive ratio 2 as well as a deterministic algorithm with competitive ratio 2.62 are obtained by Schulz and Skutellathe [150] and Correa and Wagner [46], respectively. The best known result for this objective is due to Sitters [159], who gave a deterministic algorithm with competitive ratio of $1.791 + o(m)$ for both versions. Even applying the algorithm to unweighted problem, the competitive ratio remains the same, and the algorithm beats the previously best one with competitive ratio 2 [121]. If the number of machines is small, randomized algorithms with smaller competitive ratios are found in [46].

With respect to machines with different speeds, Liu and Lu [122] studied the two machine case. They presented a $\frac{\sqrt{5}+3}{2}$-competitive algorithm for the non-preemptive problem of minimizing the total completion time. If preemption is allowed, they gave an algorithm with competitive ratio of 2 to minimize the total weighted completion time. The non-preemptive algorithm in [122] for two uniform machines was extended to m uniform machines by Liu et al. [124], who established a competitive ratio of $\frac{\sqrt{4m-3}+3}{2}$.

5 Other Variants of Online Scheduling

5.1 Online Shop Scheduling

Researches on online shop scheduling are mainly focused on two or three machines. Chen and Woeginger [34] considered the two-machine open shop scheduling of online over list model. A 1.875-competitive algorithm, as well as a lower bound $\frac{1+\sqrt{5}}{2} \approx 1.618$, was given. By restricting to *permutation algorithms* that always produce schedules with the same job sequence on both machines, the upper and lower bound can be further improved to 1.848 and $\frac{23-2\sqrt{13}}{9} \approx 1.754$ [37], respectively. If preemption is allowed, Chen and Woeginger [34] presented a $\frac{4}{3}$-competitive algorithm for two machines, and a $\frac{27}{19}$-competitive algorithm for three machines was provided by Chen et al. [37]. Since the lower bound Γ_m of $Pm|pmtn|C_{max}$ is also a lower bound of $Om|pmtn|C_{max}$ [155], both algorithms are optimal. For two-machine flow shop and job shop problems, optimal algorithms were proven to have a competitive ratio of 2 by Chen and Woeginger [34].

If jobs arrive over time, Chen et al. [35] considered the two-machine open shop scheduling and proved that greedy-like algorithm has a competitive ratio $\frac{3}{2}$ and is optimal for non-preemptive version. Another algorithm, namely, SLICE, was shown to have a competitive ratio of $\frac{5}{4}$ and best possible for preemptive version. Liu et al. [125] further extended SLICE to the semi-online problem UB & LB. They proved the corresponding algorithm is $\frac{5\beta-1}{4\beta}$-competitive and matches the lower bound of the considered problem with $\beta = \frac{UB}{LB}$. Stougie and Vestjens [162] proved that 1.25 is a randomized lower bound of $O2|r_j|C_{max}$.

Recently, Zhang and Velde [188, 189] investigated problems with time lags, where the time lag of a job is defined as the maximum allowable delay between the completion time of the first and the start time of the second operation of the job. For these problems, it might benefit from allowing a machine to be idle while an operation is available for assignment. Such algorithms are called delay algorithms. The greedy-like algorithm has a competitive ratio 2 for two-machine open shop, job shop, and flow shop problems [188, 189]. However, the lower bound of delay algorithms is only proved to be $\sqrt{2}$ for open shop [189] and $\frac{1+\sqrt{5}}{2}$ for job shop or flow shop [188]. It is hopeful that a delay algorithm might beat the greedy algorithm in this model. Nevertheless, it is still an open question so far.

5.2 Batch Scheduling

In this problem, jobs arrive over time and can be processed together by a batch processing machine which can handle up to B jobs simultaneously. All jobs in a batch start and complete at the same time. The processing time of a batch is given by the longest processing time of any job in the batch. There are two models considered in the literature: the unbounded model where $B = \infty$ and the bounded model where B is bounded. For both models, problems of minimizing makespan on single or parallel identical machines are widely studied.

5.2.1 Unbounded and Bounded Model on a Single Machine

Suppose that there is a single batch processing machine, Deng et al. [48] and Zhang et al. [191] independently provided an optimal online algorithm (denoted as H^∞) with competitive ratio $\frac{\sqrt{5}+1}{2}$.

```
    **Algorithm $H^\infty$**
0. Set $t = 0$.
1. Let $U(t)$ be the set of all unscheduled jobs available at time $t$ and $J_k$ be the job
   with the longest processing time in $U(t)$, compute $\alpha_k = \frac{\sqrt{5}+1}{2} r_k + \frac{\sqrt{5}-1}{2} p_k$ and
   $s = \max\{t, \alpha_k\}$.
2. In the time interval $[t, s]$, whenever a new job $J_h$ arrives (at time $t'$) with $p_h > p_k$,
   then reset $k = h$ and reset $\alpha_k$ and $s$ accordingly. Let $U(t') = U(t) \cup \{J_h\}$. Reset
   $t = t'$.
3. At time $s$, schedule all jobs in $U(s)$ as a single batch. If some new jobs arrives
   by $s + p_k$, let $t = s + p_k$; otherwise, wait until a new job arrives and let $t$ be the
   arrival time of such a job. Go to Step 1.
```

In fact, the lower bound $\frac{\sqrt{5}+1}{2}$ even holds when $B = 2$, and all jobs have only two distinct arrival times [191]. It seems much more challenging to derive optimal algorithms for the bounded model. When all jobs have exactly two distinct arrival time, Zhang et al. [191] gave an optimal one with competitive ratio $\frac{\sqrt{5}+1}{2}$. When jobs have arbitrary arrival times, Poon and Yu [145] obtained a $\frac{7}{4}$-competitive algorithm for $B = 2$. For the general B, we note there exist several 2-competitive algorithms. The first one is the greedy-like algorithm GRLPT, which was proposed by Lee and Uzsoy [113], and was shown to be 2-competitive by Liu and Yu [123]. Another two are due to Zhang et al. [191], which can be seen as a generalization of H^∞. Finally, Poon and Yu [145] claimed all the FBLPT-based (full-batch longest processing time) algorithms, including the above three, can achieve the same competitive ratio of 2.

5.2.2 Unbounded and Bounded Model on Identical Machines

For batch scheduling on m parallel identical machines, researches are mostly focused on the unbounded model. Zhang et al. [191] gave a lower bound $\sqrt[m+2]{2}$ and an online algorithm $PH(\beta_m)$ with a competitive ratio $1 + \beta_m$, where $0 < \beta_m < 1$ is a solution to the equation $\beta_m = (1 - \beta_m)^{m-1}$. In the same paper, the dense algorithms, which always immediately assign the currently available job to one idle machine as long as there are two or more idle machines, are proposed. Zhang et al. proved that the competitive ratio of any dense algorithm is no smaller than $\sqrt{2}$ and there exists one dense algorithm with competitive ratio $\frac{\sqrt{5}+1}{2}$. In a later paper [192], Zhang et al. gave an optimal algorithm with competitive ratio $1 + \xi_m$ for the special case that all jobs have unit processing time, where ξ_m is the positive solution of the equation $(1 + \xi_m)^{m+1} = \xi_m + 2$.

When jobs have arbitrary processing times, the optimal algorithm for two-machine case has a competitive ratio of $\sqrt{2}$ [139, 173]. The optimal algorithm for the general case is due to Liu et al. [126] and Tian et al. [172] independently, who both proved the competitive ratio to be $1 + \frac{\sqrt{m^2+4}-m}{2}$. In addition, Tian et al. [172] provided a dense algorithm with competitive ratio $\frac{3}{2}$ and showed it is best possible.

For the bounded model, Zhang et al. [192] observed that the optimal algorithm is $\frac{\sqrt{5}+1}{2}$-competitive if all jobs have equal processing times. Apart from this, neither algorithm results nor lower bounds are found in the literature.

5.2.3 Other Variants

There are some other discussions with respect to the online batch scheduling problems. One interesting variant is allowing to restart a batch, which means a running batch may be interrupted, losing all the work done on it, and the jobs in the interrupted batch are released and become independently unscheduled. Allowing restarts reduces the impact of a wrong decision. For the unbounded model on single machine, Fu et al. [78] showed the competitive ratio of any online algorithm is no less than $\frac{5-\sqrt{5}}{2}$ and gave a $\frac{3}{2}$-competitive algorithm. Shortly after that, Yuan et al. [185] gave a $\frac{5-\sqrt{5}}{2}$-competitive algorithm for the problem, and thus it is optimal. Fu et al. [79] studied the same problem with an additional assumption that any job cannot be restarted twice. Optimal algorithm with competitive ratio $\frac{3}{2}$ is obtained. In another paper [81], Fu et al. extended the result to identical machines, where a lower bound of 1.298, as well as a $\frac{1+\sqrt{3}}{2}$-competitive online algorithm, was given.

Nong et al. [140] introduced *job families* to the batch scheduling problem, which means jobs from different families cannot be processed in the same batch. Online algorithms with competitive ratio 2 are presented for both bounded and unbounded models on single machine. Besides, they showed that the algorithm is best possible for the unbounded model in the sense that jobs consist of an infinite number of families. For the bounded model, there is no online algorithm with competitive ratio smaller than $\max\{\frac{2B}{B+1}, \frac{1+\sqrt{5}}{2}\}$. Fu et al. [80] revisited the unbounded model and gave a best possible algorithm with competitive ratio $\frac{\sqrt{17}+3}{4}$ for the special case of two families of jobs.

Yuan et al. [186] and Li et al. [119] studied a lookahead semi-online model. Yuan et al. [186] considered the unbounded problem on a single batch machine.

With the information of the next longest job at any time t, they presented a best possible online algorithm with competitive ratio $\frac{5-\sqrt{5}}{2}$. If jobs are of unit length and an online algorithm can foresee all the jobs that will arrive in the time segment $(t, t+\beta]$ at any time t, Li et al. [119] presented a best possible online algorithm for the unbounded problem on m parallel-batch machines.

Chen et al. [38] considered a batch scheduling problem on single machine to minimize the total weighted completion times of jobs. They developed a linear time online algorithm with competitive ratio of $\frac{10}{3}$ for the unbounded model and an efficient algorithm with competitive ratio $4+\epsilon$ for any $\epsilon > 0$ for the bounded model. Also, they observed that there exists a 2.89-competitive and a $(2.89+\epsilon)$-competitive randomized algorithm for unbounded and bounded models, respectively.

5.3 Online Scheduling with Machine Eligibility

Online scheduling with machine eligibility constraints has received much attention in recent years. Unlike the classical parallel machine scheduling, job J_j cannot be processed on any one among the machine set $\mathcal{M}$ in this model. Instead, it has a specific set $\mathcal{M}_j \subseteq \mathcal{M}$ that can be assigned to. Set $\mathcal{M}_j$ is so-called the *eligible processing set* of job J_j. Depending on the structure of $\mathcal{M}_j$, $j = 1, \cdots, n$, there are five classes of eligibility constraints that have been studied extensively [116]:

1. Arbitrary eligible processing sets
2. Tree-hierarchical processing sets
3. Grade of Service (GoS) processing sets
4. Interval processing sets
5. Nested processing sets.

With tree-hierarchical processing sets, each machine is represented by a node, and the nodes are connected in the form of a rooted tree. Any job assignable to a node is also assignable to the node's ancestors in the tree. The GoS processing set structure can be seen as a special case of the tree-hierarchical processing set structure where the rooted tree actually forms a chain. Problems with a GoS processing set are also called *online hierarchical scheduling* in the literature. With regard to interval processing sets, job J_j is associated with two integers a_j and $b_j \geq a_j$ such that $\mathcal{M}_j = \{M_{a_j}, M_{a_j+1}, \cdots, M_{b_j}\}$. A special case of interval processing set structure is a nested processing set structure, where, for any pair of jobs J_j and J_k, we either have $\mathcal{M}_j \subseteq \mathcal{M}_k$ or $\mathcal{M}_k \subseteq \mathcal{M}_j$ or $\mathcal{M}_j \cap \mathcal{M}_k = \emptyset$. Clearly, the GoS processing set structure is also a special case of nested processing set structure.

Problems with one of the above classes of eligibility constraints will be denoted as "$\mathcal{M}_j$," "$\mathcal{M}_j(tree)$," "$\mathcal{M}_j(GoS)$," "$\mathcal{M}_j(interval)$" and "$\mathcal{M}_j(nested)$" in the middle field of the three-field notation, respectively. It is clear that $P||C_{max}$ ($Q||C_{max}$) is a special case of $P|\mathcal{M}_j|C_{max}$ ($Q|\mathcal{M}_j|C_{max}$) with $\mathcal{M}_j = \mathcal{M}$ for all J_j, while $R|\mathcal{M}_j|C_{max}$ is a special case of $R||C_{max}$. However, in the rest of this subsection, we will mostly consider the identical machine environment, unless stated otherwise.

5.3.1 Online Over List

For arbitrary eligibility, Azar et al. [13] contributed the first result. They consider a greedy algorithm, namely, AW, as follows: When a job arrives the algorithm assigns it to an eligible machine that has the smallest current load among all eligible machines. The competitive ratio of AW is shown to be at most $\lceil \log_2 m \rceil + 1$, and the lower bound is proved to be at least $\lceil \log_2(m+1) \rceil$. Hwang et al. [102] improved the analysis of AW and obtained a slightly smaller competitive ratio of $\log_2 m + 1$. The best known upper and lower bounds for this problem are found by Lim et al. [120] recently. They showed that AW has a competitive ratio no greater than $EU_m = \lfloor \log_2 m \rfloor + \frac{m}{2^{\lfloor \log_2 m \rfloor}}$, and the lower bound of the problem is EL_m, where $EL_1 = 1$ and

$$EL_m = EL_{\theta_m} + \frac{1}{\lfloor \frac{\theta_m}{m-\theta_m} \rfloor}, \theta_m = \text{argmax}_{\lfloor \frac{m}{2} \rfloor \le i \le m-1} \left\{ EL_i + \frac{1}{\lfloor \frac{i}{m-i} \rfloor} \right\}.$$

The gap between the two bounds does not exceed 0.1967 and AW is optimal when the number of machines can be written as a sum of two powers of 2. Lee et al. [114] proved the optimal bound of $Q2|\mathcal{M}_j|C_{max}$ is $1 + \min\{s, \frac{1}{s}\}$. Note that for $Q2|\mathcal{M}_j|C_{max}$, the index of M_1 and M_2 cannot be reordered, and thus the speed ratio s can be an arbitrary positive number.

Bar-Noy et al. [16] analyzed online scheduling problem subject to tree-hierarchical eligibility. A 5-competitive online algorithm is proposed. Furthermore, if jobs are of unit size, then the competitive ratio can be improved to 4. Randomized algorithms are also proposed, where the competitive ratios are shown to be $e + 1$ and e, respectively. A lower bound of $1 + \frac{1}{2}\lfloor \log_2 m \rfloor$ for $Pm|\mathcal{M}_j(nested)|C_{max}$ was given in [120].

There is a large number of researches discussing problems with a GoS eligibility, since it is very natural and has application in the service industry [103]. In [16], Bar-Noy et al. gave an algorithm with competitive ratio $e + 1$ for the general problem $P|\mathcal{M}_j(GoS)|C_{max}$ and showed that it is e-competitive if either jobs have unit size or can be fractionally processed. Moreover, they proved a lower bound of e for the fractional assignment case. However, the lower bound is valid only in the case when the number of machines tends to infinity. If the number of machines is fixed, Tan and Zhang [169] proposed an improved algorithm with competitive ratio GU_m and lower bound GL_m for the fractional assignment case; both are based on the solutions of mathematical programming. The algorithm can be modified to solve the general problem $Pm|\mathcal{M}_j(GoS)|C_{max}$ by using the same technique included in [16]. When the number of machines is small, algorithms for the general problem can be further improved. For $m = 5$ and 4, Tan and Zhang [169] proposed two algorithms with competitive ratios of 2.610 and $\frac{7}{3}$, respectively. Lim et al. [120] improved the two results to 2.501 and 2.294. For $m = 3$, Zhang et al. [193] showed that there exists an optimal algorithm with competitive ratio 2. For $m = 2$, Park et al. [141] and Jiang et al. [109] independently proposed an optimal algorithm with competitive ratio $\frac{5}{3}$. If machines have different speeds, the optimal bound

$$h_1(s) = \begin{cases} \min\{1+s, \frac{2+2s+s^2}{1+s+s^2}\}, & 0 < s \leq 1, \\ \min\{\frac{1+s}{s}, \frac{1+3s+s^2}{1+s+s^2}\}, & 1 \leq s < \infty, \end{cases}$$

of $Q2|\mathcal{M}_j(GoS)|C_{max}$ was given by Tan and Zhang [169] (Fig. 8).

If preemption is allowed, Jiang et al. [109] designed a $\frac{3}{2}$-competitive algorithm for the problem $P2|\mathcal{M}_j(GoS), pmtn|C_{max}$ and showed that it is optimal if idle time is not allowed. For general m, Dósa and Epstein [50] proved that $\frac{2m}{m+1}$ is a lower bound even if idle time is allowed. An algorithm for three machines with matched competitive ratio was also given. In the same paper, they also studied the problem $Q2|\mathcal{M}_j(GoS), pmtn|C_{max}$. An online algorithm with competitive ratio

$$h_2(s) = \max\left\{\frac{(1+s)^2}{1+s+s^2}, \frac{s(1+s)^2}{1+s^2+s^3}\right\}$$

is proposed and is shown to be optimal. Both algorithms need to introduce idle time. For $Q2|\mathcal{M}_j(GoS), frac|C_{max}$, Chassid and Epstein [28] designed an optimal algorithm with competitive ratio $\frac{(1+s)^2}{1+s+s^2}$ (Fig. 8).

For semi-online scheduling, Park et al. [141] studied the problem on two machines with the knowledge of the total processing time of jobs, where an optimal algorithm with competitive ratio $\frac{3}{2}$ is presented. Wu et al. [183] considered two semi-online problems on two machines. The first one is known the optimal makespan, for which they showed an optimal algorithm with competitive ratio $\frac{3}{2}$. The second is known the maximum processing time, for which an algorithm with competitive ratio $\frac{\sqrt{5}+1}{2}$, as well as a matched lower bound, is given. Liu et al. [125] considered the semi-online problem on two machines where upper and lower bounds on the processing time of jobs are known in advance. An online algorithm, as well as a lower bound of the problem, was presented. Optimal bound of the problem

$$h_3(\beta) = \begin{cases} \frac{2\beta+1}{\beta+1}, & 1 \leq \beta < \frac{\sqrt{5}-1}{2}, \\ \frac{\sqrt{5}+1}{2}, & \frac{\sqrt{5}+1}{2} \leq \beta < \frac{3\sqrt{5}-1}{2}, \\ \frac{\beta+2}{3}, & \frac{3\sqrt{5}-1}{2} \leq \beta < 3, \\ \frac{5}{3}, & r \geq 3, \end{cases}$$

is given by Jiang and Zhang [107], where $\beta = \frac{UB}{LB}$ (Fig. 7).

There are several researches considering problems with two GoS levels (denoted as $2GoS$). In this problem, jobs that can be processed on first k machines are called high level and the other jobs which can run on all m machines are called low level. Jiang [104] first showed that the AW algorithm is at least $(4 - \frac{1}{m})$-competitive and then provided an online algorithm with a competitive ratio of $\frac{12+4\sqrt{2}}{7} \approx 2.522$. Later, Zhang et al. [194] improved the result to $1 + \frac{m^2-m}{m^2-km+k^2} < \frac{7}{3}$.

For machine covering problems, Chassid and Epstein [28] proved that no algorithm can have a constant competitive ratio even for the most special

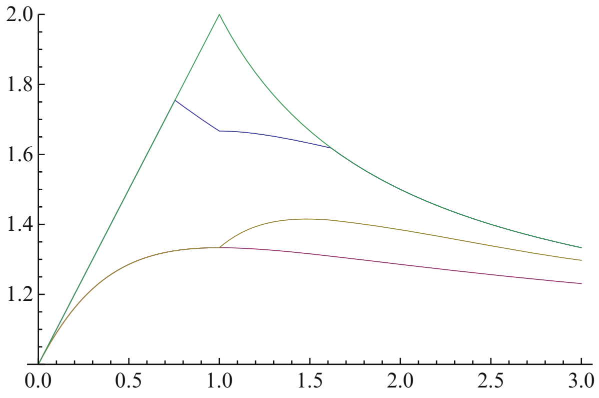


Fig. 8 The optimal bounds of $Q2|\mathcal{M}_j|C_{max}$, $Q2|\mathcal{M}_j(GoS)|C_{max}$, $Q2|\mathcal{M}_j(GoS), pmtn|C_{max}$, and $Q2|\mathcal{M}_j(GoS), frac|C_{max}$ (from *top* to *bottom*)

problem $Q2|\mathcal{M}_j(GoS)|C_{min}$. For problems $Q2|\mathcal{M}_j(GoS), frac|C_{min}$ and $Q2|\mathcal{M}_j(GoS), sum|C_{min}$, they gave optimal bounds of $\frac{2s+1}{s+1}$ and $\max\{1,s\}+\frac{1}{s}$, respectively. Main results on online over list scheduling with machine eligibility are summarized in Tables 12–14.

5.3.2 Online Over Time

In contrast with problems of online over list, there are very few results concerning problems of online over time. Lee et al. [115,116] considered the problems allowing restart or preemption of jobs. Various lower bounds, as well as a full study on two machines, are given. They also observed that the general method introduced by Shmoys et al. [157] that uses offline algorithms to obtain online algorithms for problems with job release times can work for scheduling with machine eligibility as well [116], although it produces algorithms with competitive ratio at least 2. All related results are summarized in Table 15.

Wang et al. [178] introduced another interesting problem with respect to scheduling with GoS eligibility, which is so-called *online service scheduling*. The problem arises from the service industry where customers (jobs) are classified as either “ordinary” or “special.” Ordinary customers can be served on any service facility (machines), while special customers can be served only on a flexible service facility. The difference between their model and the problem with two GoS levels is that customers arrive over time and the order of the assignment should be consistent with the order of arrival time of jobs in the same class. For several service policies used in practice, Wang et al. [178] and Wang and Xing [177] analyzed and compared their performance in the sense of competitive ratios.

6 Conclusion

This chapter surveyed different paradigms of online scheduling, including online over list and online over time, and gave a relatively complete picture to the semi-online scheduling problem. Most of detailed algorithms and proofs are not given in

Table 12 Results of online over list scheduling with machine eligibility on two identical and uniform machines

Problems	Optimal bounds	Problems	Optimal bounds
$P2\|\mathcal{M}_j\|C_{max}$	2 [13]	$Q2\|\mathcal{M}_j\|C_{max}$	$1+\min\{s,\frac{1}{s}\}$ [114]
$P2\|\mathcal{M}_j(GoS)\|C_{max}$	$\frac{5}{3}$ [109, 141]	$Q2\|\mathcal{M}_j(GoS)\|C_{max}$	$h_1(s)$ [168]
$P2\|\mathcal{M}_j(GoS), frac\|C_{max}$	$\frac{4}{3}$ [169]	$Q2\|\mathcal{M}_j(GoS), frac\|C_{max}$	$\frac{(1+s)^2}{1+s+s^2}$ [28]
$P2\|\mathcal{M}_j(GoS), pmtn\|C_{max}$[a]	$\frac{4}{3}$ [50]	$Q2\|\mathcal{M}_j(GoS), pmtn\|C_{max}$	$h_2(s)$ [50]
$P2\|\mathcal{M}_j(GoS), sum\|C_{max}$	$\frac{3}{2}$ [141]	$Q2\|\mathcal{M}_j(GoS)\|C_{min}$	∞ [28]
$P2\|\mathcal{M}_j(GoS), opt\|C_{max}$	$\frac{3}{2}$ [183]	$Q2\|\mathcal{M}_j(GoS), frac\|C_{min}$	$\frac{2s+1}{s+1}$ [28]
$P2\|\mathcal{M}_j(GoS), max\|C_{max}$	$\frac{\sqrt{5}+1}{2}$ [183]	$Q2\|\mathcal{M}_j(GoS), sum\|C_{min}$	$\max\{1,s\}+\frac{1}{s}$ [28]
$P2\|\mathcal{M}_j(GoS), UB\&LB\|C_{max}$	$h_3(\beta)$ [107]		

[a]If idle time is not allowed, the optimal bound increases to $\frac{3}{2}$ [109]

Table 13 Results of online over list scheduling with machine eligibility on a small number of identical machines

	$m=3$		$m=4$		$m=5$		$m=6$		$m=7$	
Problem	LB	UB	LB	UB	LB	UB	LB	UB	LB	UB
$Pm\|\mathcal{M}_j\|C_{max}$ [13, 120]	2.5	2.5	3	3	3.25	3.25	3.5	3.5	3.667	3.75
$Pm\|\mathcal{M}_j(interval)\|C_{max}$ [120]	$1+\sqrt{2}$		$\frac{3+\sqrt{5}}{2}$		3					
$Pm\|\mathcal{M}_j(nested)\|C_{max}$ [120]	$\frac{7}{3}$		$1+\sqrt{2}$		$\frac{5}{2}$		3			
$Pm\|\mathcal{M}_j(GoS)\|C_{max}$ [120, 169, 193]	2	2	2	2.294	2	2.501	2	2.778	2	2.828
$Pm\|\mathcal{M}_j(GoS), frac\|C_{max}$ [169]	$\frac{3}{2}$	$\frac{3}{2}$	$\frac{44}{27}$	$\frac{44}{27}$	$\frac{245}{143}$	$\frac{245}{143}$	$\frac{16}{9}$	$\frac{16}{9}$	$\frac{1{,}071}{586}$	$\frac{1{,}071}{586}$
$Pm\|\mathcal{M}_j(GoS), pmtn\|C_{max}$ [50]	$\frac{3}{2}$	$\frac{3}{2}$	$\frac{8}{5}$		$\frac{5}{3}$		$\frac{12}{7}$		$\frac{7}{4}$	

Table 14 Main results on online over list scheduling with machine eligibility on general number of machines

Problems	Jobs	Lower bound		Upper bound		Gap
$Pm\|\mathcal{M}_j\|C_{max}$	None	EL_m	[120]	EU_m	[120]	≤ 0.1967
$P\|\mathcal{M}_j(tree)\|C_{max}$	None	e	[16]	5	[16]	$5-e$
	$p_j \equiv 1$	e	[16]	4	[16]	$4-e$
$P\|\mathcal{M}_j(GoS)\|C_{max}$	None	e	[16]	$e+1$	[16]	1
	$p_j \equiv 1$	e	[16]	e	[16]	0
	frac	e	[16]	e	[16]	0
$Pm\|\mathcal{M}_j(GoS)\|C_{max}$	None	GL_m	[169]	GU_m+1	[169]	1
	frac	GL_m	[169]	GU_m	[169]	0
	2 GoS	2	[104, 194]	$1+\frac{m^2-m}{m^2-km+k^2}$	[194]	$\leq 1/3$
	2 GoS, $p_j \equiv 1$	$3/2$	[193]	$3/2$	[193]	0

Table 15 Main results on online over time scheduling with machine eligibility

Problem	Jobs	Lower bound	Upper bound
$P\|r_j,\mathcal{M}_j\|C_{max}$	None	2	$4-2/m$
	res	1.5687	$3-1/m$
	pmtn	$1+\frac{(m-1)(m-2)}{2m(2m-3)}$	2
$P2\|r_j,\mathcal{M}_j\|C_{max}$	None	2	2
	res	1.5687	2
	pmtn	1.125	2
	$p_j = p$	$\frac{\sqrt{5}+1}{2}$	$\frac{\sqrt{5}+1}{2}$
$P\|r_j,\mathcal{M}_j(GoS)\|C_{max}$	None	1.5550	$2+\epsilon$
	res	$4/3$	$2+\epsilon$
	pmtn	1.0917	2
$P2\|r_j,\mathcal{M}_j(GoS)\|C_{max}$	None	1.5550	2
	res	$4/3$	2
	pmtn	1	1
	$p_j = p$	$\sqrt{2}$	$\sqrt{2}$
$P\|r_j,\mathcal{M}_j(nested)\|C_{max}$	None	1.5550	$2+\epsilon$
	res	$4/3$	$2+\epsilon$
	pmtn	1.148	2
$P\|r_j,\mathcal{M}_j(tree)\|C_{max}$	None	1.5550	$8/3$
	res	$4/3$	$7/3$

the chapter; please refer to the reference for more details. Online and semi-online scheduling is relatively young when compared to offline scheduling. Yet they have generated tremendous interest and promise to have more results in the future.

Cross-References

▶Advances in Scheduling Problems
▶Greedy Approximation Algorithms
▶Job Shop Scheduling with Petri Nets

Recommended Reading

1. S. Albers, Better bounds for online scheduling. SIAM J. Comput. **29**, 459–473 (1999)
2. S. Albers, On randomized online scheduling, in *Proceedings of the 34th ACM Symposium on Theory of Computing* (ACM, New York, 2002), pp. 134–143
3. S. Albers, M. Hellwig, Semi-online scheduling revisited. Theor. Comput. Sci. **443**, 1–9 (2012)
4. S. Albers, M. Hellwig, On the value of job migration in online makespan minimization. *Proceedings of the 20th Annual European Symposium on Algorithms*. Lecture Notes in Computer Science (Springer, Berlin/New York, 2012)
5. E.J. Anderson, C.N. Potts, Online scheduling of a single machine to minimize total weighted completion time. Math. Oper. Res. **29**, 686–697 (2004)
6. E. Angelelli, A.B. Nagy, M.G. Speranza, Z. Tuza. The on-line multiprocessor scheduling problem with known sum of the tasks. J. Sched. **7**, 421–428 (2004)
7. E. Angelelli, M.G. Speranza, Z. Tuza. Semi on-line scheduling on two parallel processors with upper bound on the items. Algorithmica **37**, 243–262 (2003)
8. E. Angelelli, M.G. Speranza, Z. Tuza, New bounds and algorithms for on-line scheduling: two identical processors, known sum and upper bound on the tasks. Discret. Math. Theor. Comput. Sci. **8**, 1–16 (2006)
9. E. Angelelli, M.G. Speranza, Z. Tuza, Semi on-line scheduling on three processors with known sum of the tasks. J. Sched. **10**, 263–269 (2007)
10. J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, O. Waarts, On-line load balancing with applications to machine scheduling and virtual circuit routing. J. ACM **44**, 486–504 (1997)
11. Y. Azar, L. Epstein, On-line machine covering, in *Proceeding of the 5th Annual European Symposium on Algorithms*. Lecture Notes in Computer Science, vol. 1284 (Springer, Berlin/New York, 1997), pp. 23–36
12. Y. Azar, L. Epstein, On-line machine covering. J. Sched. **1**, 67–77 (1998)
13. Y. Azar, J. Naor, R. Rom, The competitiveness of on-line assignments. Algorithmica **18**, 221–237 (1995)
14. Y. Azar, O. Regev, On-line bin-stretching. Theor. Comput. Sci. **168**, 17–41 (2001)
15. N. Bansal, M. Sviridenko, The Santa Claus problem, in *Proceeding of the 38th ACM Symposium on Theory of Computing* (ACM, New York, 2006), pp. 31–40
16. A. Bar-Noy, A. Freund, J. Naor, Online load balancing in a hierarchical server topology. SIAM J. Comput. **31**, 527–549 (2001)
17. Y. Bartal, H. Karloff, Y. Rabani, A better lower bound for on-line scheduling. Inf. Process. Lett. **50**, 113–116 (1994)
18. Y. Bartal, A. Fiat, H. Karloff, R. Vohra, New algorithms for an Ancient scheduling problem. J. Comput. Syst. Sci. **51**, 359–366 (1995)
19. P. Berman, M. Charikar, M. Karpinski, On-line load balancing for related machines. J. Algorithms **35**, 108–121 (2000)
20. A. Borodin, R.E. Yaniv, *Online Computation and Competitive Analysis* (Cambridge University Press, Cambridge/New York, 2005)
21. P. Brucker, *Scheduling Algorithms* (Springer, Berlin/New York, 2007)
22. S.Y. Cai, Semi-online machine covering. Asia-Pac. J. Oper. Res. **24**, 373–382 (2007)
23. S.Y. Cai, Q.F. Yang, Semi-online scheduling on two uniform machines with the known largest size. J. Comb. Optim. **21**, 393–408 (2011)

24. Q. Cao, Z.H. Liu. Semi-online scheduling with known maximum job size on two uniform machines. J. Comb. Optim. **20**, 369–384 (2010)
25. Q. Cao, Z.H. Liu. Online scheduling with reassignment on two uniform machines. Theor. Comput. Sci. **411**, 2890–2898 (2010)
26. Q. Cao, Z. Liu, T.C.E. Cheng, Semi-online scheduling with known partial information about job sizes on two identical machines. Theor. Comput. Sci. **412**, 3731–3737 (2011)
27. S. Cao, Z.Y. Tan, Online uniform machine covering with the known largest size. Prog. Nat. Sci. **17**, 1271–1278 (2007)
28. O. Chassid, L. Epstein, The hierarchical model for load balancing on two machines. J. Comb. Optim. **15**, 305–314 (2008)
29. C. Chekuri, R. Motwani, B. Natarajan, C. Stein, Approximation techniques for average completion time scheduling. SIAM J. Comput. **31**, 146–166 (2001)
30. B. Chen, A.P.A. Vestjens, Scheduling on identical machines how good is LPT in an on-line setting. Oper. Res. Lett. **21**, 165–169 (1997)
31. B. Chen, A. van Vliet, G.J. Woeginger, New lower and upper bounds for on-line scheduling. Oper. Res. Lett. **16**, 221–230 (1994)
32. B. Chen, A. van Vliet, G.J. Woeginger, A lower bound for randomized on-line scheduling algorithms. Inf. Process. Lett. **51**, 219–222 (1994)
33. B. Chen, A. van Vliet, G.J. Woeginger, An optimal algorithm for preemptive on-line scheduling. Oper. Res. Lett. **18**, 127–131 (1995)
34. B. Chen, G.J. Woeginger, A study of on-line scheduling two-stage shops, in *Minimax and Applications*, ed. by D.-Z. Du, P.M. Pardalos (Kluwer Academic, Dordrecht/Boston, 1995), pp. 97–107
35. B. Chen, P.A.V. Arjen, G.J. Woeginger, On-line scheduling of two-machine open shops where jobs arrive over time. J. Comb. Optim. **1**, 355–365 (1998)
36. B. Chen, C.N. Potts, G.J. Woeginger, A review of machine scheduling: complexity, algorithms and approximability, in *Handbook of Combinatorial Optimization*, ed. by D.-Z. Du, P.M. Pardalos (Springer, 1998), pp. 21–169
37. B. Chen, D. Du, J. Han, J. Wen, On-line scheduling of small open shops. Discret. Appl. Math. **110**, 133–150 (2001)
38. B. Chen, X. Deng, W. Zang, On-line scheduling a batch processing system to minimize total weighted job completion time. J. Comb. Optim. **8**, 85–95 (2004)
39. X.Y. Chen, L. Epstein, Z.Y. Tan, Semi-online machine covering for two uniform machines. Theor. Comput. Sci. **410**, 5047–5062 (2009)
40. X. Chen, Y. Lan, A. Benko, G. Dosa, X. Han, Optimal algorithms for online scheduling with bounded rearrangement at the end. Theor. Comput. Sci. **412**, 6269–6278 (2011)
41. T.C.E. Cheng, H. Kellerer, V. Kotov, Semi-on-line multiprocessor scheduling with given total processing time. Theor. Comput. Sci. **337**, 134–146 (2005)
42. T.C.E. Cheng, C.T. Ng, V. Kotov, A new algorithm for online uniform-machine scheduling to minimize the makespan. Inf. Process. Lett. **99**, 102–105 (2006)
43. T.C.E. Cheng, H. Kellerer, V. Kotov, Algorithms better than LPT for semi-online scheduling with decreasing processing times. Oper. Res. Lett. **40**, 349–352 (2012)
44. Y. Cho, S. Sahni, Bounds for list schedules on uniform processors. SIAM J. Comput. **9**, 91–103 (1980)
45. C. Chung, T. Nonner, A. Souza, SRPT is 1.86-competitive for completion time scheduling, in *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms* (ACM, New York, 2010), pp. 1373–1388
46. J.R. Correa, M.R. Wagner, LP-based online scheduling: from single to parallel machines. Math. Program. **119**, 109–136 (2009)
47. J. Csirik, H. Kellerer, G. Woeginger, The exact LPT-bound for maximizing the minimum machine completion time. Oper. Res. Lett. **11**, 281–287 (1992)
48. X. Deng, C.K. Poon, Y. Zhang, Approximation algorithms in batch processing. J. Comb. Optim. **7**, 247–257 (2003)

49. B. Deuermeyer, D. Friesen, M. Langston, Scheduling to maximize the minimum processor finish time in a multiprocessor system. SIAM J. Discret. Methods **3**, 190–196 (1982)
50. G. Dósa, L. Epstein, Preemptive scheduling on a small number of hierarchical machines. Inf. Comput. **206**, 602–619 (2008)
51. G. Dósa, L. Epstein, Preemptive online scheduling with reordering, in *Proceeding of the 17th Annual European Symposium on Algorithms*. Lecture Notes in Computer Science, vol. 5757 (Springer, Berlin/New York, 2009), pp. 456–467
52. G. Dósa, L. Epstein, Online scheduling with a buffer on related machines. J. Comb. Optim. **20**, 161–179 (2010)
53. G. Dósa, Y. He, Semi-online algorithms for parallel machine scheduling problems. Computing **72**, 355–363 (2004)
54. G. Dósa, M.G. Speranza, Z. Tuza, Two uniform machines with nearly equal speeds: unified approach to known sum and known optimum in semi on-line scheduling. J. Comb. Optim. **21**, 458–480 (2011)
55. G. Dósa, Y.X. Wang, X. Han, H. Guo, Online scheduling with rearrangement on two related machines. Theor. Comput. Sci. **412**, 642–653 (2011)
56. D. Du, Optimal preemptive semi-online scheduling on two uniform processors. Inf. Process. Lett. **92**, 219–223 (2004)
57. T. Ebenlendr, Semi-online preemptive scheduling: study of special cases. *Proceeding of the 8th International Conference on Parallel Processing and Applied Mathematics, Part II*. Lecture Notes in Computer Science, vol. 6068 (Springer, Berlin, 2010), pp. 11–20
58. T. Ebenlendr, J. Sgall, Optimal and online preemptive scheduling on uniformly related machines. J. Sched. **12**, 517–527 (2009)
59. T. Ebenlendr, J. Sgall, Semi-online preemptive scheduling: one algorithm for all variants. Theory Comput. Syst. **48**, 577–613 (2011)
60. T. Ebenlendr, J. Sgall, A lower bound on deterministic online algorithms for scheduling on related machines without preemption, in *Proceeding of the 9th Workshop on Approximation and Online Algorithms*. Lecture Notes in Computer Science (Springer, Berlin/New York, 2012), pp. 102–108
61. T. Ebenlendr, J. Noga, J. Sgall, G.J. Woeginger, A note on semi-online machine covering. *Proceeding of the 3rd Workshop on Approximation and Online Algorithms*. Lecture Notes in Computer Science, vol. 3879 (Springer, Berlin/New York, 2006), pp. 110–118
62. T. Ebenlendr, W. Jawor, J. Sgall, Preemptive online scheduling: optimal algorithms for all speeds. Algorithmica **53**, 504–522 (2009)
63. M. Englert, D. Özmen, M. Westermann, The power of reordering for online minimum makespan scheduling, in *Proceeding of the 48th IEEE Symposium Foundations of Computer Science*, Providence, RI (2008), pp. 603–612
64. L. Epstein, Optimal preemptive scheduling on uniform processors with non-decreasing speed ratios. Oper. Res. Lett. **29**, 93–98 (2001)
65. L. Epstein, Bin stretching revisited. Acta Inform. **39**, 97–117 (2003)
66. L. Epstein, Tight bounds for on-line bandwidth allocation on two links. Discret. Appl. Math. **148**, 181–188 (2005)
67. L. Epstein, L.M. Favrholdt, Optimal preemptive semi-online scheduling to minimize makespan on two related machines. Oper. Res. Lett. **30**, 269–275 (2002)
68. L. Epstein, L.M. Favrholdt, Optimal non-preemptive semi-online scheduling on two related machines. J. Algorithms **57**, 49–73 (2005)
69. L. Epstein, R. van Stee, Lower bounds for on-line single-machine scheduling. Theor. Comput. Sci. **299**, 439–450 (2003)
70. L. Epstein, D. Ye, Semi-online scheduling with "end of sequence" information. J. Comb. Optim. **14**, 45–61 (2007)
71. L. Epstein, A. Levin, Robust algorithms for preemptive scheduling. *Processding of the 19th Annual European Symposium on Algorithms*. Lecture Notes in Computer Science, vol. 6942 (Springer, Berlin/New York, 2011), pp. 567–578

72. L. Epstein, J. Noga, S. Seiden, J. Sgall, G.J. Woeginger, Randomized on-line scheduling on two uniform machines. J. Sched. **4**, 71–92 (2001)
73. L. Epstein, A. Levin, R. van Stee, Max-min online allocations with a reordering buffer. SIAM J. Discret. Math. **25**, 1230–1250 (2011)
74. U. Faigle, W. Kern, G. Turan, On the performance Of on-line algorithm for particular problem. Acta Cybern. **9**, 107–119 (1989)
75. A. Fiat, G.J. Woeginger, *Online Algorithms: The State of the Art*. Lecture Notes in Computer Science, vol. 1442 (Springer, Berlin/New York, 1998)
76. A. Fiat, G.J. Woeginger, On-line scheduling on a single machine: minimizing the total completion time. Acta Inform. **36**, 287–293 (1999)
77. R. Fleischer, M. Wahl, Online scheduling revisited. J. Sched. **3**, 343–353 (2000)
78. R. Fu, J. Tian, J.J. Yuan, Y. Lin, Online scheduling in a parallel batch processing system to minimize makespan using restarts. Theor. Comput. Sci. **374**, 196–202 (2007)
79. R. Fu, J. Tian, J.J. Yuan, C. He, On-line scheduling on a batch machine to minimize makespan with limited restarts. Oper. Res. Lett. **36**, 255–258 (2008)
80. R. Fu, J. Tian, J.J. Yuan, On-line scheduling on an unbounded parallel batch machine to minimize makespan of two families of jobs. J. Sched. **12**, 91–97 (2009)
81. R. Fu, T.C.E. Cheng, C.T. Ng, J.J. Yuan, Online scheduling on two parallel-batching machines with limited restarts to minimize the makespan. Inf. Process. Lett. **110**, 444–450 (2010)
82. G. Galambos, G.J. Woeginger, An on-line scheduling heuristic with better worst case ratio than Graham's list scheduling. SIAM J. Comput. **22**, 349–355 (1993)
83. M.X. Goemans, M. Queyranne, A.S. Schulz, M. Skutella, Y. Wang, Single machine scheduling with release dates. SIAM J. Discret. Math. **15**, 165–192 (2002)
84. T.F. Gonzales, S. Sahni, Preemptive scheduling of uniform processor systems. J. ACM **25**, 92–101 (1978)
85. T. Gormley, N. Reingold, E. Torng, J. Westbrook, Generating adversaries for request-answer games, in *Proceeding of the 11th ACM-SIAM Symposium on Discrete Algorithms* (ACM, New York/Society for Industrial and Applied Mathematics, Philadelphia, 2000), pp. 564–565
86. R.L. Graham, Bounds for certain multiprocessing anomalies. Bell Syst. Tech. J. **45**, 1563–1581 (1966)
87. R.L. Graham, Bounds on multiprocessing timing anomalies. SIAM J. Appl. Math. **17**, 416–429 (1969)
88. R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discret. Math. **5**, 287–326 (1979)
89. N.G. Hall, M.E. Posner, C.N. Potts, Online scheduling with known arrival times. Math. Oper. Res. **34**, 92–102 (2009)
90. F.Q. Han, Z.Y. Tan, Y. Yang, On the optimality of list scheduling for online uniform machines scheduling. Optim. Lett. **6**, 1551–1571 (2012)
91. Y. He, The optimal on-line parallel machine scheduling. Comput. Math. Appl. **39**, 117–121 (2000)
92. Y. He, G. Dósa, Semi-online scheduling jobs with tightly-grouped processing times on three identical machines. Discret. Appl. Math. **150**, 140–159 (2005)
93. Y. He, Y.W. Jiang, Optimal algorithms for semi-online preemptive scheduling problems on two uniform machines. Acta Inform. **40**, 367–383 (2004)
94. Y. He, Y.W. Jiang, Preemptive semi-online scheduling with tightly-grouped processing times. J. Comput. Sci. Technol. **19**, 733–739 (2004)
95. Y. He, Y.W. Jiang, Optimal semi-online preemptive algorithms for machine covering on two uniform machines. Theor. Comput. Sci. **339**, 293–314 (2005)
96. Y. He, Z.Y. Tan, Randomized on-line and semi-on-line scheduling on identical machine. Asia-Pac. J. Oper. Res. **20**, 31–40 (2003)
97. Y. He, G.C. Zhang, Semi on-line scheduling on two identical machines. Computing **62**, 179–187 (1999)
98. Y. He, H. Zhou, Y.W. Jiang, Preemptive semi-online algorithms with known total size. Acta Math. Sin. (English Series) **22**, 587–594 (2006)

99. Y. He, Y.W. Jiang, H. Zhou, Optimal preemptive online algorithm for scheduling with known largest size on two uniform machines. Acta Math. Sin. (English Series) **23**, 165–174 (2007)
100. J.A. Hoogeveen, A.P.A. Vestjens, Optimal online algorithms for single-machine scheduling, in *Proceedings of the 5th International Conference on Integer Programming and Combinatorial Optimization*. Lecture Notes in Computer Science, vol. 1084 (Springer, Berlin/New York, 1996), pp. 404–414
101. E. Horwath, E.C. Lam, R. Sethi, A level algorithm for preemptive scheduling. J. ACM **24**, 32–43 (1977)
102. H.C. Hwang, S.Y. Chang, Y. Hong, A posterior competitiveness for list scheduling algorithm on machines with eligibility constraints. Asia-Pac. J. Oper. Res. **21**, 117–125 (2004)
103. H.C. Hwang, S.Y. Chang, K. Lee, Parallel machine scheduling under a grade of service provision. Comput. Oper. Res. **31**, 2055–2061 (2004)
104. Y.W. Jiang, Online scheduling on parallel machines with two GoS levels. J. Comb. Optim. **16**, 28–38 (2008)
105. Y.W. Jiang, Y. He, Optimal preemptive online algorithm for scheduling tightly-grouped jobs on two uniform machines. Asia-Pac. J. Oper. Res. **23**, 77–88 (2006)
106. Y.W. Jiang, Y. He, Optimal semi-online algorithms for preemptive scheduling problems with inexact partial information. Acta Inform. **44**, 571–590 (2007)
107. Y.W. Jiang, A. Zhang, Optimal online algorithms on two hierarchical machines with tightly-grouped processing times, Technical Report, 2010
108. Y. Jiang, Z.Y. Tan, Y. He, Preemptive machine covering on parallel machines. J. Comb. Optim. **10**, 345–363 (2005)
109. Y. Jiang, Y. He, C. Tang, Optimal online algorithms for scheduling on two identical machines under a grade of service. J. Zhejiang Univ. Sci. (A) **7**, 309–314 (2006)
110. D.R. Karger, S.J. Phillips, E. Torng, A better algorithm for an ancient scheduling problem. J. Algorithms **20**, 400–430 (1996)
111. H. Kellerer, V. Kotov, M.G. Speranza, Z. Tuza, Semi on-line algorithms for the partition problem. Oper. Res. Lett. **21**, 235–242 (1997)
112. Y. Lan, X. Chen, N. Ding, G. Dosa, X. Han, Online minimum makespan scheduling with a buffer, in *Processding of the Joint International Conference on Frontiers in Algorithmics and Algorithmic Aspects in Information and Management*. Lecture Notes in Computer Science, vol. 7285 (2012), pp. 161–171
113. C.Y. Lee, R. Uzsoy, Minimizing makespan on a single batch processing machine with dynamic job arrivals. Int. J. Prod. Res. **37**, 219–236 (1999)
114. K. Lee, J.Y.T. Leung, M. Pinedo, Online scheduling on two uniform machines subject to eligibility constraints. Theor. Comput. Sci. **410**, 3975–3981 (2009)
115. K. Lee, J.Y.T. Leung, M. Pinedo, Scheduling jobs with equal processing times subject to machine eligibility constraints. J. Sched. **14**, 27–38 (2011)
116. K. Lee, J.Y.-T. Leung, M.L. Pinedo, Makespan minimization in online scheduling with machine eligibility. 4OR-A Q. J. Oper. Res. **8**, 331–364 (2010)
117. J.Y.T. Leung (ed.), *Handbook of Scheduling: Algorithms, Models, and Performance Analysis* (CRC, Boca Raton, 2004)
118. R.H. Li, L.J. Shi, An online algorithm for some uniform professor scheduling. SIAM J. Comput. **27**, 414–422 (1998)
119. W. Li, Z. Zhang, S. Yang, Online algorithms for scheduling unit length jobs on parallel-batch machines with lookahead. Inf. Process. Lett. **112**, 292–297 (2012)
120. K. Lim, K. Lee, S.Y. Chang, Improved bounds for online scheduling with eligibility constraints. Theor. Comput. Sci. **412**, 5211–5224 (2011)
121. P. Liu, X. Lu, On-line scheduling of parallel machines to minimize total completion times. Comput. Oper. Res. **36**, 2647–2652 (2009)
122. P. Liu, X. Lu, Online scheduling of two uniform machines to minimize total completion times. J. Ind. Manag. Optim. **5**, 95–102 (2009)
123. Z.H. Liu, W. Yu, Scheduling one batch processor subject to job release dates. Discret. Appl. Math. **105**, 129–136 (2000)

124. M. Liu, C. Chu, Y. Xu, F. Zheng, Online scheduling on m uniform machines to minimize total (weighted) completion time. Theor. Comput. Sci. **410**, 3875–3881 (2009)
125. M. Liu, C. Chu, Y. Xu, F. Zheng, An optimal online algorithm for two-machine open shop preemptive scheduling with bounded processing times. Optim. Lett. **4**, 227–237 (2010)
126. P. Liu, X. Lu, Y. Fang, A best possible deterministic on-line algorithm for minimizing makespan on parallel batch machines. J. Sched. **15**, 77–81 (2012)
127. X. Lu, R.A. Sitters, L. Stougie, A class of on-line scheduling algorithms to minimize total completion time. Oper. Res. Lett. **31**, 232–236 (2003)
128. R. Luo, S. Sun, Semi on-line scheduling problem with the largest processing time of jobs on two uniform machines known. J. Syst. Sci. Math. Sci. **26**, 729–736 (2006). (in Chinese)
129. R. Luo, S. Sun, W. Huang, Semi on-line scheduling problem for maximizing the minimum machine completion time on two uniform machines. J. Syst. Sci. Complex. **19**, 101–107 (2006)
130. N. Megow, *Coping with Incomplete Information in Scheduling Stochastic and Online Models*, Ph.D. thesis, Technische Universit Berlin, October 2006, Published by Cuvillier Verlag Göttingen, Germany, 2007
131. N. Megow, A.S. Schulz, On-line scheduling to minimize average completion time revisited. Oper. Res. Lett. **32**, 485–490 (2004)
132. X. Min, J. Liu, Y.Q. Wang, Optimal semi-online algorithms for scheduling problems with reassignment on two identical machines. Inf. Process. Lett. **111**, 423–428 (2011)
133. P. Mireault, J.B. Orlin, R.V. Vohra, A parametric worst case analysis of the LPT heuristic for two uniform machines. Oper. Res. **45**, 116–125 (1997)
134. R. Motwani, P. Raghavan, *Randomized Algorithms* (Cambridge University Press, Cambridge/New York, 1995)
135. G. Muratore, U.M. Schwarz, G.J. Woeginger, Parallel machine scheduling with nested job assignment restrictions. Oper. Res. Lett. **38**, 47–50 (2010)
136. A. Musitelli, J.M. Nicoletti, Competitive ratio of list scheduling on uniform machines and randomized heuristics. J. Sched. **14**, 89–101 (2011)
137. C.T. Ng, Z.Y. Tan, Y. He, T.C.E. Cheng, Two semi-online scheduling problems on two uniform machines. Theor. Comput. Sci. **410**, 776–792 (2009)
138. J. Noga, S.S. Seiden, An optimal online algorithm for scheduling two machines with release times. Theor. Comput. Sci. **268**, 133–143 (2001)
139. Q. Nong, T.C.E. Cheng, C.T. Ng, An improved on-line algorithm for scheduling on two unrestrictive parallel batch processing machines. Oper. Res. Lett. **36**, 584–588 (2008)
140. Q. Nong, J.J. Yuan, R. Fu, L. Lin, J. Tian, The single-machine parallel-batching on-line scheduling problem with family jobs to minimize makespan. Int. J. Prod. Econ. **111**, 435–440 (2008)
141. J. Park, S.Y. Chang, K. Lee, Online and semi-online scheduling of two machines under a grade of service provision. Oper. Res. Lett. **34**, 692–696 (2006)
142. C. Phillips, C. Stein, J. Wein, Minimizing average completion time in the presence of release dates. Math. Program. **82**, 199–223 (1995)
143. M.L. Pinedo, *Scheduling, Theory, Algorithms, and Systems* (Springer, New York/London, 2008)
144. K. Pruhs, J. Sgall, E. Torng, Online scheduling, in *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, ed. by J.Y.T. Leung, vol. 15 (Chapman & Hall/CRC, Boca Raton, 2004)
145. C.K. Poon, W. Yu, On-line scheduling algorithms for a batch machine with finite capacity. J. Comb. Optim. **9**, 167–186 (2005)
146. M. Queyranne, On the Anderson–Potts single machine on-line scheduling algorithm, manuscript (2001)
147. J.F. Rudin III, R. Chandrasekaran, Improved bound for the online scheduling problem. SIAM J. Comput. **32**, 717–735 (2003)
148. L. Schrage, A proof of the optimality of the shortest remaining processing time discipline. Oper. Res. **16**, 199–223 (1968)

149. A.S. Schulz, M. Skutella, The power of α-points in preemptive single machine scheduling. J. Sched. **5**, 121–133 (2002)
150. A.S. Schulz, M. Skutella, Scheduling unrelated machines by randomized rounding. SIAM J. Discret. Math. **15**, 450–469 (2002)
151. S.S. Seiden, Online randomized multiprocessor scheduling. Algorithimica **28**, 173–216 (2000)
152. S.S. Seiden, A guessing game and randomized online algorithms, in *Proceedings of the 32nd ACM Symposium on Theory of Computing* (ACM, New York, 2000), pp. 592–601
153. S.S. Seiden, Barely random algorithms for multiprocess scheduling. J. Sched. **6**, 309–334 (2003)
154. S.S. Seiden, J. Sgall, G.J. Woeginger, Semi online scheduling with decreasing job sizes. Oper. Res. Lett. **27**, 215–221 (2000)
155. J. Sgall, A lower bound for randomized on-line multiprocessor scheduling. Inf. Process. Lett. **63**, 51–55 (1997)
156. J. Sgall, On-line scheduling, in *Online Algorithms: The State of the Art*, ed. by A. Fiat, G.J. Woeginger (Springer, Berlin/New York, 1998), pp. 196–231
157. D.B. Shmoys, J. Wein, D.P. Williamson, Scheduling parallel machines online. SIAM J. Comput. **24**, 1313–1331 (1995)
158. R. Sitters, Competitive analysis of preemptive single-machine scheduling. Oper. Res. Lett. **38**, 585–588 (2010)
159. R. Sitters, Efficient algorithms for average completion time scheduling, in *Proceedings of the 14th International Conference on Integer Programming and Combinatorial Optimization*. Lecture Notes in Computer Science, vol. 6080 (Springer, Berlin, 2010), pp. 411–423
160. N. Sivadasan, P. Sanders, M. Skutella, Online scheduling with bounded migration. Math. Oper. Res. **34**, 481–498 (2009)
161. D.D. Sleator, R.E. Tarjan, Amortized efficiency of list update and paging rules. Commun. ACM **28**, 202–208 (1985)
162. L. Stougie, A.P.A. Vestjens, Randomized algorithms for on-line scheduling problems how low can't you go. Oper. Res. Lett. **30**, 89–96 (2002)
163. Z.Y. Tan, S.J. Cao, Semi-online machine covering on two uniform machines with known total size. Computing **78**, 369–378 (2006)
164. Z.Y. Tan, Y. He, Semi-on-line problems on two identical machines with combined partial information. Oper. Res. Lett. **30**, 408–414 (2002)
165. Z.Y. Tan, Y. He, Semi-online scheduling problems on two identical machines with inexact partial information. Theor. Comput. Sci. **377**, 110–125 (2007)
166. Z.Y. Tan, Y. Wu, Optimal semi-online algorithms for machine covering. Theor. Comput. Sci. **372**, 69–80 (2007)
167. Z.Y. Tan, S. Yu, Online scheduling with reassignment. Oper. Res. Lett. **36**, 250–254 (2008)
168. Z.Y. Tan, A. Zhang, A note on hierarchical scheduling on two uniform machines. J. Comb. Optim. **20**, 85–95 (2010)
169. Z.Y. Tan, A. Zhang, Online hierarchical scheduling: an approach using mathematical programming. Theor. Comput. Sci. **412**, 246–256 (2011)
170. J. Tao, Z. Chao, Y. Xi, A semi-online algorithm and its competitive analysis for a single machine scheduling problem with bounded processing times. J. Ind. Manag. Optim. **6**, 269–282 (2010)
171. J. Tao, Z. Chao, Y. Xi, Y. Tao, An optimal semi-online algorithm for a single machine scheduling problem with bounded processing time. Inf. Process. Lett. **110**, 325–330 (2010)
172. J. Tian, T.C.E. Cheng, C.T. Ng, J.J. Yuan, Online scheduling on unbounded parallel-batch machines to minimize the makespan. Inf. Process. Lett. **109**, 1211–1215 (2009)
173. J. Tian, R. Fu, J.J. Yuan, A best online algorithm for scheduling on two parallel batch machines. Theor. Comput. Sci. **410**, 2291–2294 (2009)
174. T. Tichý, Randomized on-line scheduling on three processors. Oper. Res. Lett. **32**, 152–158 (2004)

175. R. van Stee, J.A. La Poutré, Minimizing the total completion time on-line on a single machine, using restarts. J. Algorithms **57**, 95–129 (2005)
176. A.P.A. Vestjens, *On-Line Machine Scheduling*, Ph.D. thesis, Department of Mathematics and Computing Science, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, 1997
177. Z. Wang, W. Xing, Worst-case analysis for on-line service polices. J. Comb. Optim. **19**, 107–122 (2010)
178. Z. Wang, W. Xing, B. Chen, On-line service scheduling. J. Sched. **12**, 31–43 (2009)
179. Y. Wang, A. Benko, X. Chen, G. Dosa, H. Guo, X. Han, C.S. Lanyi, Online scheduling with one rearrangement at the end: revisited. Inf. Process. Lett. **112**, 641–645 (2012)
180. J. Wen, D. Du, Preemptive on-line scheduling for two uniform processors. Oper. Res. Lett. **23**, 113–116 (1998)
181. G. Woeginger, A polynomial time approximation scheme for maximizing the minimum machine completion time. Oper. Res. Lett. **20**, 149–154 (1997)
182. Y. Wu, Z.Y. Tan, Q.F. Yang, Optimal semi-online scheduling algorithms on a small number of machines, in *Proceedings of the 1st International Symposium on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies*, Hangzhou. Lecture Notes in Computer Science, vol. 4614 (2007), pp. 504–515
183. Y. Wu, M. Ji, Q.F. Yang, Optimal semi-online scheduling algorithms on two parallel identical machines under a grade of service provision. Int. J. Prod. Econ. **135**, 367–371 (2012)
184. B. Xiong, C. Chung, Completion time scheduling and the WSRPT algorithm, in *Proceedings of the 2nd International Symposium on Combinatorial Optimization*, Athens. Lecture Notes in Computer Science, vol. 7422 (2012), pp. 416–426
185. J.J. Yuan, R. Fu, C.T. Ng, T.C.E. Cheng, A best online algorithm for unbounded parallel-batch scheduling with restarts to minimize makespan. J. Sched. **14**, 361–369 (2011)
186. J.J. Yuan, C.T. Ng, T.C.E. Cheng. Best semi-online algorithms for unbounded parallel batch scheduling. Discret. Appl. Math. **159**, 838–847 (2011)
187. G.C. Zhang, A simple semi on-line algorithm for $P2||C_{max}$ with a buffer. Inf. Process. Lett. **61**, 145–148 (1997)
188. X. Zhang, S. van de Velde, On-line two-machine job shop scheduling with time lags. Inf. Process. Lett. **110**, 510–513 (2010)
189. X. Zhang, S. van de Velde, On-line two-machine open shop scheduling with time lags. Eur. J. Oper. Res. **204**, 14–19 (2010)
190. G.C. Zhang, D. Ye, A note on on-line scheduling with partial information. Comput. Math. Appl. **44**, 539–543 (1999)
191. G.C. Zhang, X. Cai, C.K. Wong, On-line algorithms for minimizing makespan on batch processing machines. Nav. Res. Logist. **48**, 241–258 (2001)
192. G.C. Zhang, X. Cai, C.K. Wong, Optimal on-line algorithms for scheduling on parallel batch processing machines. IIE Trans. **35**, 175–181 (2003)
193. A. Zhang, Y.W. Jiang, Z.Y. Tan, Optimal algorithms for online hierarchical scheduling on parallel machines, Technical Report, 2009
194. A. Zhang, Y.W. Jiang, Z.Y. Tan, Online parallel machines scheduling with two hierarchies. Theor. Comput. Sci. **410**, 3597–3605 (2009)