Efficient Mechanism Design for Online Scheduling (Extended Abstract)*

Xujin Chen[1], Xiaodong Hu[1], Tie-Yan Liu[2], Weidong Ma[2], Tao Qin[2], Pingzhong Tang[3], Changjun Wang[4], Bo Zheng[3]
[1] AMSS, Chinese Academy of Sciences, [2] Microsoft Research
[3] Tsinghua University, [4] Beijing University of Technology
{xchen, xdhu}@amss.ac.cn, {tyliu, weima, taoqin}@microsoft.com,
kenshin@mail.tsinghua.edu.cn, wcj@bjut.edu.cn, zhengb10@mails.tsinghua.edu.cn

Abstract

This work concerns the mechanism design for online scheduling in a strategic setting. In this setting, each job is owned by a self-interested agent who may misreport the release time, deadline, length, and value of her job, while we need to determine not only the schedule of the jobs, but also the payment of each agent. We focus on the design of incentive compatible (IC) mechanisms, and study the maximization of social welfare (i.e., the aggregated value of completed jobs) by competitive analysis. We first derive two lower bounds on the competitive ratio of any deterministic IC mechanism to characterize the landscape of our research: one bound is 5, which holds for equal-length jobs; the other bound is $\frac{\kappa}{\ln \kappa} + 1 - o(1)$, which holds for unequal-length jobs, where κ is the maximum ratio between lengths of any two jobs. We then propose a deterministic IC mechanism and show that such a simple mechanism works very well for two models: (1) In the preemption-restart model, the mechanism can achieve the optimal competitive ratio of 5 for equal-length jobs and a near optimal ratio of $(\frac{1}{(1-\epsilon)^2} + o(1))\frac{\kappa}{\ln \kappa}$ for unequal-length jobs, where $0 < \epsilon < 1$ is a small constant; (2) In the preemption-resume model, the mechanism can achieve the optimal competitive ratio of 5 for equal-length jobs and a near optimal competitive ratio (within factor 2) for unequal-length jobs.

1 Introduction

Online scheduling has been widely studied in the literature [Baruah *et al.*, 1992; 1994; Porter, 2004; Zheng *et al.*, 2006; Ting, 2008], where each job is characterized by a release time, a deadline, a length, and a value for its successful completion by the deadline. Inspired by emerging areas like computational economics and cloud computing [Azar *et al.*, 2013; Lucier *et al.*, 2013; Mashayekhy *et al.*, 2014; Wu *et al.*, 2014], we consider a strategic setting of the online scheduling problem, where each job is owned by a self-interested agent and she may have the incentive to manipulate the scheduling algorithm in order to be better off [Friedman and Parkes, 2003; Porter, 2004; Hajiaghayi *et al.*, 2005; Parkes, 2007]. To be specific, the agent may deliberately delay the release time of her job, inflate its length, and misreport its value and deadline.

Given this situation, a carefully designed online scheduling mechanism is needed to regulate the strategic behaviors of the agents and to (approximately) optimize some system objectives. In this work, we focus on the maximization of social welfare, i.e., the total value of completed jobs. We use *competitive analysis* [Lavi and Nisan, 2004] to evaluate the performance of such a mechanism, which compares the social welfare implemented by the mechanism (without any knowledge of all future jobs) with that of the optimal offline allocation (with the knowledge of future jobs).

In this work, we consider two scheduling models: the *preemption-restart* model [Ting, 2008] and the *preemption-resume* model [Porter, 2004]. Once preempted, jobs in the first model have to restart from the beginning; while jobs in the second model can resume from the break point. Since preemption is always assumed in this work, the two models are also referred to as *restart* model and *resume* model, respectively, and their involved jobs are called *non-resumable* and *resumable*, respectively.

1.1 Problem Formulation

We consider online scheduling models with infinite time period $T = \mathbb{R}_{\geq 0}$. Suppose there is a single machine that processes at most one job at any given time. Jobs come over time, and we use J to denote the set of jobs. Each job $j \in J$ is owned by a self-interested agent (which is also denoted as j for simplicity); and it is characterized by a private type $\theta_j = (r_j, d_j, l_j, v_j) \in T \times T \times \mathbb{R}_{>0} \times \mathbb{R}_{>0}$, where r_j is the release time, d_j is the deadline, l_j is the length (i.e., the processing time), and v_j is the value if the job is completed by its deadline.

A resumable job j is *completed* if and only if it is processed for l_j time units in total between its release time r_j and deadline d_j, while a non-resumable job j is *completed* if and only if it is processed for l_j consecutive time units between its release time r_j and deadline d_j.

Let $\kappa = \max_{i,j \in J} \frac{l_i}{l_j}$ be the maximum ratio between the

*This paper is an extended abstract of an article in the Journal of Artificial Intelligence Research [Chen et al., 2016]

lengths of any two jobs. For simplicity, we assume all job lengths are normalized, i.e., $l_j \in [1, \kappa]$ for all $j \in J$, and assume κ is known in advance following the practice in the work of Chan et al. (2004) and Ting (2008).

We study direct revelation mechanisms, in which each agent participates by simply declaring the type of her job $\hat{\theta}_j = (\hat{r}_j, \hat{d}_j, \hat{l}_j, \hat{v}_j)$ at time $\hat{r}_j$. We use $\hat{\theta}$ to denote the profile of reported types of all the agents. Given the declared types of the agents, a mechanism M is used to schedule/allocate the jobs and determine the payment of each agent. Here we only consider "reasonable" mechanisms which (1) do not schedule a job after its reported deadline and (2) do not schedule a job once it has been processed for a reported length.

Given a certain mechanism M and a job sequence $\hat{\theta}$, we use $q_j(\hat{\theta}, t)$ to denote whether job j is completed by time t (if it is completed, $q_j(\hat{\theta}, t) = 1$; otherwise $q_j(\hat{\theta}, t) = 0$). Then the value that agent j extracts from the mechanism can be represented by $q_j(\hat{\theta}, d_j)v_j$, and the social welfare of the mechanism can be represented by $W(M, \theta) = \sum_j q_j(\hat{\theta}, d_j)v_j$.

Let $p_j(\hat{\theta})$ denote the amount of money that the mechanism charges agent j. We assume that agents have quasi-linear preferences [Nisan, 2007], i.e., the utility of agent j is $u_j(\hat{\theta}, \theta_j) = q_j(\hat{\theta}, d_j)v_j - p_j(\hat{\theta})$.

Since agents are self-interested, they may misreport their types in a strategic way. It is easy to see that the misreport of a shorter length is a dominated strategy; otherwise, her job cannot be completed even if it is scheduled by the mechanism (since $\hat{l}_j < l_j$). Therefore, the agents will not underreport the lengths of their jobs. Similar to the work of Porter (2004), we assume that the system will not return a completed job to agent j until $\hat{d}_j$. In this way, we restrict the agent's report to be $\hat{d}_j \leq d_j$. In addition, we assume that no agent has knowledge about her job before its release time, so we also have $\hat{r}_j \geq r_j$.

Considering the potential misreport of the agents, we are concerned with incentive compatible and individually rational mechanisms. A mechanism is *incentive compatible* (IC) if, for any agent j, regardless of the behaviors of other agents, truthful reporting her own type maximizes her utility. A mechanism is *individually rational* (IR) if for each job j, truthful reporting leads to a non-negative utility. In addition, we would also like the mechanism to (approximately) maximize social welfare. We say a mechanism M is (strictly) *c-competitive* if there does not exist any job sequence θ such that $c \cdot W(M, \theta) < W(opt, \theta)$, where *opt* denotes the optimal offline mechanism. Sometimes we also say that M has a competitive ratio of c.

1.2 Our Results

For the restart and resume model, we first give the lower bounds of the competitive ratio for any online IC mechanism is 5 when $\kappa = 1$ and $\frac{\kappa}{\ln \kappa} + 1 - o(1)$ when κ is big. Then we design a simple IC mechanism that have very good performances with respect to the competitive ratios. For the restart model, our mechanism has a competitive ratio of $\kappa + 2 + (1 + \frac{1}{\kappa})^{\kappa}$ when κ is small, and $(\frac{1}{(1-\epsilon)^2} + o(1)) \cdot \frac{\kappa}{\ln \kappa}$ when κ is large, where $0 < \epsilon < 1$ is a small constant. For the resume model, our model has a competitive ratio of $(\kappa + 1)(1 + \frac{1}{\kappa})^{\kappa} + 1$ when κ is small, and $(\frac{2}{(1-\epsilon)^2} + o(1)) \cdot \frac{\kappa}{\ln \kappa}$ when κ is large. Comparing to the lower bounds, the competitive ratios we obtained about our mechanism show that our mechanism is nearly the best possible.

2 Lower bounds

In this section, we give the lower bounds of the competitive ratio for any online IC mechanism of this problem. Detailed proofs can be found in the full version [Chen *et al.*, 2016].

Theorem 2.1. *When $\kappa = 1$, no deterministic IC mechanism can obtain a competitive ratio less than 5.*

Theorem 2.2. *When κ is sufficiently large, no deterministic IC mechanism can obtain a competitive ratio less than $\frac{\kappa}{\ln \kappa} + 1 - o(1)$. In particular, no deterministic IC mechanism can obtain a competitive ratio less than $\frac{\kappa}{\ln \kappa} + 0.94$ for $\kappa \geq 16$.*

3 Mechanism Design

In this section, we describe a simple mechanism Γ_1 (whose allocation and payment rules are given in Algorithm 1), which works surprisingly well for both the restart and resume models, and handles the settings with different values of κ in a unified framework. In contrast, previous works [Dürr *et al.*, 2012] need to design separate and very different algorithms to deal with different values of κ.

3.1 The Mechanism Γ_1

Before introducing our mechanism, we first introduce the concept of the *valid active time* of an uncompleted job j, until time t, denoted as

$$e_j(t) = \begin{cases} t - \min\{s | x(t') = j, \forall t' \in [s, t)\}, \text{for restart model} \\ \int_0^t \mu(x(s) = j) ds, \text{for resume model} \end{cases} \tag{1}$$

where $x(t)$ is the mechanism's allocation function, which maps each time point to an available job, or to 0 if the machine is idle. And $\mu(\cdot)$ is an indicator function that returns 1 if the argument is true, and zero otherwise. Note that $e_j(\cdot)$ can also take a vector θ as an argument. For example, $e_j(\theta, t)$ is shorthand for the $e_j(t)$ for the job sequence θ.

It can be seen that in the restart model, at time t, if a job j has received an allocation at time $t' < t$ and has not been preempted after that, then $e_j(t) = t - t'$. In the resume model, $e_j(t)$ is the accumulated processing time of job j until time t.

We say that a job j is *feasible* at time t if (1) its reported release time is before t; (2) it has not been completed yet; and (3) it has enough time to be completed before its reported deadline, i.e., $\hat{d}_j - t \geq \hat{l}_j - e_j(t)$. We use $J_F(t)$ to denote the set of all feasible jobs at time t.

According to Algorithm 1, at any time t, Γ_1 assigns a priority score, $\hat{v}_j \cdot \beta^{\hat{l}_j - e_j(\hat{\theta}, t)}$, to each feasible job $j \in J_F(t)$, and always processes the feasible job with the highest priority (ties are broken in favor of the job with the smaller $\hat{r}_j$). Here β is located in $(0, 1)$ and will be determined later during the

competitive analysis. The payment rule of Γ_1 is essentially the critical-value payment [Parkes, 2007], which is similar to that of the second-price auction. Hence, the payment is equal to the minimum bid the agents have to make to remain allocated. In the following pseudocode, $\hat{\theta}_{-j}$ denotes the reported types of all jobs other than j.

Algorithm 1:

Allocation Rule:
  for all time t **do**
    if $J_F(t) \neq \emptyset$ **then**
      $x(t) \leftarrow \arg\max_{j \in J_F(t)}(\hat{v}_j \cdot \beta^{\hat{l}_j - e_j(\hat{\theta}, t)})$
    else
      $x(t) \leftarrow 0$
  end
Payment Rule:
  for all job j **do**
    if $q_j(\hat{\theta}, \hat{d}_j) = 1$ **then**
      $p_j(\hat{\theta}) = \min(v'_j | q_j(((\hat{r}_j, \hat{d}_j, \hat{l}_j, v'_j), \hat{\theta}_{-j}), \hat{d}_j) = 1)$
    else
      $p_j(\hat{\theta}) = 0$
  end

Theorem 3.1. *Mechanism Γ_1 is incentive compatible, in both the restart model and resume model.*

4 Competitive Analysis

In this section, we show that mechanism Γ_1 performs quite well in terms of social welfare by comparison with the optimal offline allocation, which has full knowledge of the future jobs at the beginning of the execution.

To perform the competitive analysis, we need to design *virtual charging schemes*. Under a certain virtual charging scheme, for every job j completed by the optimal allocation *opt*, we charge its value (or partial value) to some job f completed by Γ_1. If this virtual charging scheme satisfies the property that every job f completed by Γ_1 receives a total charge of at most cv_f, then we succeed in showing that Γ_1 has a competitive ratio of at most c. Designing an ingenious virtual charging scheme is crucial to the competitive analysis. In the following, we will design different virtual charging schemes to obtain the competitive ratio of Γ_1 for the restart model and the resume model respectively.

As we use a parameter β in the priority function of mechanism Γ_1, we first derive competitive ratios as functions of β. We will specify later (in Section 4.3) how to choose a suitable β (with respect to κ) to optimize the performance of Γ_1, and derive competitive ratios in terms of κ.

Here, we introduce some notation which will be used in both Section 4.1 and Section 4.2. Denote by $(1, 2, \ldots, F)$ the sequence of jobs completed by Γ_1 over time. For each job f in this sequence, let t_f be the time when job f is completed, and for convenience denote $t_0 = 0$. Divide the time into $F+1$ intervals $I_f = [t_{f-1}, t_f), f = 1, 2, \ldots, F$, and $[t_F, +\infty)$.

4.1 Analysis of the Restart Model

We study the restart model first. We assume, without loss of generality, that the optimal allocation *opt* does not interrupt any allocation, since all interrupted jobs are non-resumable. We have the following theorem.

Theorem 4.1. *For the restart model, Γ_1 has a competitive ratio of $\frac{1}{1-\beta} + \frac{1}{\beta^\kappa} + 1$.*

Proof. We introduce the virtual charging scheme as follows. For any completed job j in *opt*, if it is also completed in mechanism Γ_1, then its value is charged to itself.

Otherwise (i.e., job j is not completed by Γ_1), we consider the time s_j at which j begins execution in *opt*. Note that *opt* does not interrupt any allocation, so j is exactly allocated to the time period $[s_j, s_j + l_j)$. Then s_j must be in some time interval I_f(recall $I_f = [t_{f-1}, t_f)$), and we charge the value of j to f. Define $\sigma_j := t_f - s_j$ to be the time amount between s_j and t_f. As job j is feasible at time s_j, according to Lemma 4.2, we know that the priority jobs j at time s_j is at most $v_f \beta^{t_f - s_j} = v_f \beta^{\sigma_j}$; in the meanwhile, the priority of j at time s_j is $v_j \beta^{l_j}$. We have $v_j \beta^{l_j} \leq v_f \beta^{\sigma_j}$, i,e., $v_j \leq v_f \beta^{\sigma_j - l_j}$. We defer the formal statement and the proof of Lemma 4.2 to the end of this subsection.

We now calculate the maximum total value charged to a completed job f in Γ_1. In time interval I_f, denote by $(1, 2, \ldots, m)$, the sequence of jobs in *opt* whose starting time s_j belongs to I_f and ordered as $s_1 > s_2 > \cdots > s_m$. Remember that σ_j is the time amount between s_j and t_f. Then it is clear that we have $0 < \sigma_1 < \sigma_2 < \cdots < \sigma_m$ and $\sigma_j - l_j \geq \sigma_{j-1}$ for $2 \leq j \leq m$, since j is allocated and completed during time interval $[s_j, s_{j-1}]$. Furthermore, as the job lengths are normalized, i.e., $1 \leq l_j \leq \kappa$, we can deduce that:

$$\sigma_j \geq \begin{cases} 0 & \text{for } j = 1 \\ j - 1 & \text{for } j \geq 2. \end{cases} \tag{2}$$

Recall that $\beta < 1$ and f may also be completed in *opt*. Therefore the total charge to job f is at most $v_f + \sum_{j=1}^{m} v_j$, which is upper bounded by

$$v_f + v_f \sum_{j=1}^{m} \beta^{\sigma_j - l_j} \leq v_f (1 + \beta^{-l_1} + \sum_{j=2}^{m} \beta^{\sigma_{j-1}})$$

$$\leq v_f (1 + \beta^{-l_1} + \sum_{j=1}^{m-1} \beta^{\sigma_j}) \leq v_f (1 + \beta^{-\kappa} + \sum_{j=0}^{\infty} \beta^j).$$

This shows that mechanism Γ_1 is $(\frac{1}{1-\beta} + \frac{1}{\beta^\kappa} + 1)$-competitive. □

Actually, the competitive ratio obtained in this way is tight, i.e., the ratio $\frac{1}{1-\beta} + \frac{1}{\beta^\kappa} + 1$ is best possible for Γ_1. We give an example in the full version to show tightness.

Lemma 4.2. *For any time point $s_j \in I_f$, if job j ($\neq f$) is feasible at time s_j, then the priority of j at s_j is at most $v_f \beta^{t_f - s_j}$. Moreover, the value of j, v_j, is at most $v_f \beta^{t_f - s_j - l_j}$.*

Proof. Note that, s_j is in time interval I_f, and according to the definition of I_f, we know that f is the unique job that is completed in I_f by Γ_1. Now we prove the lemma by enumerating all possible cases.

(1) If the executing job at s_j is job f, then we know that the priority of job f at time s_j is exactly $v_f\beta^{t_f-s_j}$ (because the priority of job f at time t_f is v_f). Clearly, the priority of j at s_j is not larger than that of job f, and thus not larger than $v_f\beta^{t_f-s_j}$.

(2) If the executing job at s_j is not job f, then we assume that Γ_1 executes job $j_1,\ldots,j_k$ and f successively[1] in the time period $[s_j,t_f)$, where $k\geq 1$. Since f is the unique job completed in I_f, we can deduce that: j_1 is preempted by j_2, j_2 is preempted by j_3,...,j_k is preempted by f, and finally f is completed at time t_f. Denote $\tau_1,\ldots,\tau_k$ as the time points at which $j_1,\ldots,j_k$ are preempted respectively. We also denote $f_j(t)$ as the priority of job j at time t. We now use backward induction: First, we know that the priority of job j_k at τ_k is not larger than that of job f, i.e., $f_{j_k}(\tau_k)\leq v_f\beta^{t_f-\tau_k}$. Then, since j_{k-1} is preempted by j_k at τ_{k-1}, we know that the priority of j_{k-1} at τ_{k-1} is not larger than that of j_k. Hence, we have $f_{j_{k-1}}(\tau_{k-1})\leq f_{j_k}(\tau_{k-1})=f_{j_k}(\tau_k)\beta^{\tau_k-\tau_{k-1}}\leq v_f\beta^{t_f-\tau_{k-1}}$. And eventually, we can get that $f_{j_1}(\tau_1)\leq v_f\beta^{t_f-t_1}$. Since j_1 is executed at time s_j, we can deduce that $f_{j_1}(s_j)\leq v_f\beta^{t_f-s_j}$. Clearly, the priority of j at time s_j (i.e., $v_j\beta^{l_j}$) is not larger than that of j_1, thus not larger than $v_f\beta^{t_f-s_j}$.

By arranging $v_j\beta^{l_j-e_j(s_j)}\leq v_f\beta^{t_f-s_j}$, we can get $v_j\leq v_f\beta^{t_f-s_j-l_j+e_j(s_j)}\leq v_f\beta^{t_f-s_j-l_j}$, where $e_j(s_j)\geq 0$ is the valid active time of job j at time s_j. □

Some remarks on Lemma 4.2: (1) Because f is the unique job completed by Γ_1 in the time interval I_f, the priorities of the executing jobs monotonically increase during I_f. (2) Lemma 4.2 applies in both the restart model and resume model. (3) Lemma 4.2 provides a useful tool to relate the priority of a feasible job (j) at some time point ($s_j\in I_f$) to the completed job f.

4.2 Analysis of the Resume Model

Compared with the restart model, the competitive analysis for the resume model is much more complicated, because in the resume model, a job can be executed in several disjointed time intervals.

To analyze the competitive ratio of Γ_1 for the resume model, we propose two new virtual charging schemes (referred to as *integral charging scheme* and *segmental charging scheme*, respectively). In the *integral charging scheme*, we charge the whole value of job j in the optimal allocation *opt* to some job completed by mechanism Γ_1; while in the *segmental charging scheme*, we charge the value of j by segment, and different segments of the same job may be charged to different jobs completed by mechanism Γ_1. By using these two schemes, in Theorem 4.3 we upper bound the competitive ratio of mechanism Γ_1 by $\frac{\beta^{-\kappa}}{1-\beta}+1$ and $\frac{1}{\beta^\kappa}+\frac{-2}{\beta\ln\beta}+1$ respectively. As discussed in Section 4.3, the two ratios work for situations with different κ values, i.e., the first one works well for small κ and the second one works well for large κ. The detailed information about the two charging schemes and its relevant proofs can be found in the full version of this work [Chen *et al.*, 2016]. Here we just list the main theorem.

Theorem 4.3. *For the resume model, the competitive ratio of Γ_1 is at most $\frac{\beta^{-\kappa}}{1-\beta}+1$. In particular, if β satisfies $\kappa\beta^\kappa\geq\beta$, the competitive ratio of Γ_1 is at most $\min\{\frac{\beta^{-\kappa}}{1-\beta}+1,\ \frac{1}{\beta^\kappa}+\frac{-2}{\beta\ln\beta}+1\}$.*

4.3 Discussions

An advantage of our mechanism is that it can handle the settings with different values of κ in a unified framework. We only need to set parameter β to different values in Theorem 4.1 and Theorem 4.3 so as to adapt to different settings of job lengths (as shown in the following corollaries).

Corollary 4.4. *By setting $\beta=1-(1-\epsilon)^2\cdot\frac{\ln\kappa}{\kappa}$, where $\epsilon>0$ is an arbitrary small constant, mechanism Γ_1 achieves a competitive ratio $(\frac{1}{(1-\epsilon)^2}+o(1))\cdot\frac{\kappa}{\ln\kappa}$ for the restart model and a competitive ratio $(\frac{2}{(1-\epsilon)^2}+o(1))\cdot\frac{\kappa}{\ln\kappa}$ for the resume model.*

As for Corollary 4.4, we have the following discussions:

(1) For the restart model, mechanism Γ_1 achieves a competitive ratio of $(\frac{1}{(1-\epsilon)^2}+o(1))\cdot\frac{\kappa}{\ln\kappa}$, which improves upon the best-known algorithmic result $\frac{6\kappa}{\log\kappa}+O(\kappa^{\frac{5}{6}})$ [Ting, 2008] for the standard online scheduling without strategic behavior.

(2) For the resume model, when κ is large, mechanism Γ_1 achieves a competitive ratio of $(\frac{2}{(1-\epsilon)^2}+o(1))\cdot\frac{\kappa}{\ln\kappa}$, which is slightly worse than the result obtained for the restart model (within a factor of 2). Asymptotically speaking, Γ_1 is near optimal, since its competitive ratio has the same order (w.r.t. κ) as the lower bound shown in Theorem 2.2. Furthermore, our analysis generalizes the results obtained by Durr et al. (2012) to the continuous value of time and the strategic setting.

(3) When κ is relatively small, the ratios given in Corollary 4.4 will become loose. In particular, when κ approaches 1, the above ratios will approach infinity since $\ln\kappa$ approaches 0. In this case, we need a different setting of β (see Corollary 4.5).

Corollary 4.5. *By choosing $\beta=\frac{\kappa}{\kappa+1}$, the competitive ratio of mechanism Γ_1 is $\kappa+2+(1+\frac{1}{\kappa})^\kappa<\kappa+2+e$ for the restart model and $(\kappa+1)(1+\frac{1}{\kappa})^\kappa+1$ for the resume model.*

Similarly, we have the following discussions:

(1) The competitive ratio of Γ_1 is linear in κ, since $(1+\frac{1}{\kappa})^\kappa$ is bounded by e.

(2) In particular, when $\kappa=1$, the ratios in the above corollary become 5 for both the restart and resume model, which matches the lower bound given in Theorem 2.1. In this regard, we say that Γ_1 is optimal. On the other hand, this also shows that the lower bound of 5 in Theorem 2.1 is tight.

[1] Here, j_1 can be job j, which does not affect the analysis.

References

[Azar *et al.*, 2013] Yossi Azar, Naama Ben-Aroya, Nikhil R Devanur, and Navendu Jain. Cloud scheduling with setup cost. In *Proceedings of the twenty-fifth annual ACM symposium on Parallelism in algorithms and architectures*, pages 298–304. ACM, 2013.

[Baruah *et al.*, 1992] Sanjoy Baruah, Gilad Koren, Decao Mao, Bhubaneswar Mishra, Arvind Raghunathan, Louis Rosier, Dennis Shasha, and Fuxing Wang. On the competitiveness of on-line real-time task scheduling. *Real-Time Systems*, 4(2):125–144, 1992.

[Baruah *et al.*, 1994] Sanjoy K Baruah, Jayant Haritsa, and Nitin Sharma. On-line scheduling to maximize task completions. In *Proceedings of Real-Time Systems Symposium*, pages 228–236. IEEE, 1994.

[Chen *et al.*, 2016] Xujin Chen, Xiaodong Hu, Tie-Yan Liu, Weidong Ma, Tao Qin, Pingzhong Tang, Changjun Wang, and Bo Zheng. Efficient mechanism design for online scheduling. *Journal of Artificial Intelligence Research*, 56(1):429–461, May 2016.

[Dürr *et al.*, 2012] Christoph Dürr, Łukasz Jeż, and Kim Thang Nguyen. Online scheduling of bounded length jobs to maximize throughput. *Journal of Scheduling*, 15(5):653–664, 2012.

[Friedman and Parkes, 2003] Eric J Friedman and David C Parkes. Pricing wifi at starbucks: issues in online mechanism design. In *Proceedings of the 4th ACM conference on Electronic commerce*, pages 240–241. ACM, 2003.

[Hajiaghayi *et al.*, 2005] Mohammad Hajiaghayi, Robert Kleinberg, Mohammad Mahdian, and David C Parkes. Online auctions with re-usable goods. In *Proceedings of the 6th ACM conference on Electronic commerce*, pages 165–174. ACM, 2005.

[Lavi and Nisan, 2004] Ron Lavi and Noam Nisan. Competitive analysis of incentive compatible on-line auctions. *Theoretical Computer Science*, 310:159–180, 2004.

[Lucier *et al.*, 2013] Brendan Lucier, Ishai Menache, Joseph Seffi Naor, and Jonathan Yaniv. Efficient online scheduling for deadline-sensitive jobs. In *Proceedings of the 25th ACM symposium on Parallelism in algorithms and architectures*, pages 305–314. ACM, 2013.

[Mashayekhy *et al.*, 2014] Lena Mashayekhy, Mahyar Movahed Nejad, Daniel Grosu, and Athanasios V Vasilakos. Incentive-compatible online mechanisms for resource provisioning and allocation in clouds. In *Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on*, pages 312–319. IEEE, 2014.

[Nisan, 2007] Noam Nisan. Introduction to mechanism design (for computer scientists). *Algorithmic game theory*, 209:242, 2007.

[Parkes, 2007] David C Parkes. Online mechanisms. *Algorithmic Game Theory, ed. N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani, Cambridge University Press*, pages 411–439, 2007.

[Porter, 2004] Ryan Porter. Mechanism design for online real-time scheduling. In *Proceedings of the 5th ACM conference on Electronic commerce*, pages 61–70. ACM, 2004.

[Ting, 2008] Hing-Fung Ting. A near optimal scheduler for on-demand data broadcasts. *Theoretical Computer Science*, 401(1):77–84, 2008.

[Wu *et al.*, 2014] Xiaohong Wu, Yonggen Gu, Guoqiang Li, Jie Tao, Jingyu Chen, and Xiaolong Ma. Online mechanism design for VMS allocation in private cloud. In *Network and Parallel Computing*, pages 234–246. Springer, 2014.

[Zheng *et al.*, 2006] Feifeng Zheng, Stanley PY Fung, Wun-Tat Chan, Francis YL Chin, Chung Keung Poon, and Prudence WH Wong. Improved on-line broadcast scheduling with deadlines. In *Computing and Combinatorics*, pages 320–329. Springer, 2006.