Complete Decentralized Mechanism Design for Online Machine Scheduling

Yuan Zhang, Chi-Hung Chi, Shengqing Zhang, Nan Zheng

School of Software, Tsinghua University
Beijing, China, 100084
Contact Email: chichihung@mail.tsinghua.edu.cn

Abstract — We study an online version of the classical parallel scheduling problem to minimize the total weighted completion time in a totally decentralized setting: both the jobs and the machines are selfish entities who try to maximize their own interest. We study the setting from an algorithmic mechanism design and present a polynomial time decentralized scheduling mechanism that induces the rational jobs and machines to report truthfully about their private information, and lures the jobs to choose proper machines such that the resulting schedule is 3.281-competitive.

I. INTRODUCTION

We study the online version of the classical parallel scheduling problem to minimize the total weighted completion time in a complete decentralized and economical setting. In such a setting, no central authority exists and the status information of the jobs and the machines are only known to themselves. Besides, strategies are allowed. Selfish agents (representing machines or jobs) are able to report strategically to maximize their benefits. In the model we propose, a job can report fake values about its weight, processing time and release time to the machines. Also jobs have the rights to select their favorite machine to run the jobs since there is no central authority. At the same time, machines have the rights to report fake time delay (denotes how long it takes to start running the jobs) and lower payment (denotes how much the job has to pay the machine for running it) to the job to entice the job.

Particularly, such machines-side strategies are very common in the real economic environment. One seller may compete with other sellers for more customers by cutting its price. It is a reasonable move to cut the payoff since the seller may get nothing when it cannot get the job. Although such behaviors may benefit one very seller, they disturb the whole market and yield a negative result on the whole systematic metric. However, few works have been done to cope with this problem in mechanism design.

Our goal is to build a feasible mechanism that motivates both the job and the machine sides to report truthfully at the same time. Also the mechanism must yield a good performance about the system overall weighted completion time.

In recent years, distributed computing environments such as Grids and Compute Cloud[13] have been more and more widely used. Such kind of environment is decentralized set and needs the resource allocation or the job scheduling mechanism to be also decentralized.

Besides, in order to make Grids or Compute Cloud survive and grow, the mechanism must be economically rational which means both the jobs and machines can benefit by attending the system. Our work is to develop such a decentralized mechanism which can satisfy those requirements and support those systems by tackling a representative parallel machine scheduling problem in those environments.

Related Work. Different from the classical scheduling methods which do not take into account the strategic dimension and use system-centric measures to assign the jobs to the machines and schedule them, economic scheduling and mechanism design have been studied more and more.

However, most current economic scheduling mechanisms consider only one-side strategy setting. In their models, the strategic behaviors are set by allowing the jobs to choose the machine to be processed on, report their self-information strategically [1][2], or allow the machines to report their speeds dishonestly [11][12] and separately. But little works consider the strategic behaviors of both the machine and the job sides at the same time. Stocess et al. [7] point out the question without giving a solution.

Particularly, a Decentralized Local Greedy Mechanism (DLGM) is proposed by Heydenreich et al. [2]. It shows many merits on incentive compatibility, online payment, and competitive ratio in the setting that jobs are selfish and rational. This is the mechanism on which our work is based.

The problem of scheduling to minimize the weighted completion time is well-understood in the non-strategic setting with centralized coordination [2]. The optimization problem is NP-hard even in the off-line case [8]. In an online setting, the best possible algorithm for single machine case is 2-competitive [9] and for the parallel machine setting is 2.61-competitive [10]. While in decentralized strategic setting, Heydenreich et al. [2] developed an online truthful mechanism which achieves a competitive ratio of 3.281.

Contribution. We present a polynomial time, decentralized online mechanism for the strategic and decentralized setting of the online scheduling problem. And we show the mechanism is incentive compatible both on the machine side and the job side. Using our mechanism, we can avoid machines or sellers from raising or cutting the price unilaterally. Besides, our mechanism is 3.281-competitive which is the same as the DLGM [2].
This paper is structured as follows. Section 2 gives a brief introduction about the DLGM. In Section 3, we introduce our mechanism from the job side, the machine side and the whole system separately. We describe the advantages and disadvantages of our mechanism in Section 4, and analyze the impact and performance in Section 5. Final conclusion and a short discussion are presented in Section 6.

II. DECENTRALIZED LOCAL GREEDY MECHANISM

DLGM [2] is an online mechanism proposed by Heydenreich et. al. for the online parallel machine scheduling problem. DLGM considers the jobs as the strategic actors. Jobs (job agents) submit themselves to the system. A job is specified by the tuple \( t_j = (r_j, p_j, w_j) \) containing its release date \( r_j \), processing time \( p_j \), and an indifference cost \( w_j \). When requesting a machine to be run on, the selfish job agent may strategically report a fake information \( \hat{r}_j = (\tilde{r}_j, \tilde{p}_j, \tilde{w}_j) \) to all the machines. There are \( m \) identical machines \( M = \{1,...,m\} \) under full control. Each machine \( i \) will send a report about the tentative completion time \( \hat{C}_{ij} \) and payment \( \hat{p}_i \) back based on their current workload (the information is tentative due to the online situation). After the reports are received, the job will choose a machine which maximizes its tentative utility. DLGM consists of following two parts:

Local Scheduling Policy:

Whenever a machine becomes idle, it starts processing the job with the highest (WSPT) priority among all available jobs queuing at this machine.

Assignment:

1. At its chosen release time \( \tilde{r}_j \), job reports a weight \( \tilde{w}_j \) and a processing time \( \tilde{p}_j \) to all machines.

2. Every machine \( i \) computes

\[
\hat{C}_{ij}(i) = \tilde{r}_j + b_i(\tilde{r}_j) + \sum_{k: \tilde{r}_k \leq \tilde{r}_j \land \tilde{p}_k < \tilde{p}_j} \tilde{p}_k + \tilde{w}_j
\]

and

\[
\hat{p}_i(i) = \tilde{p}_j \sum_{k: \tilde{r}_k \leq \tilde{r}_j \land \tilde{p}_k < \tilde{p}_j} \tilde{w}_k
\]

\( \hat{\pi}_i \). Here, \( b_i(t) \) means the remaining time of the currently running job at a given time \( t \). \( \tilde{H}(f) \) denotes the set of jobs that have higher priority than \( j \) according to their reports or have the same priority for jobs with smaller index. \( k \rightarrow i \) denotes the fact that job \( k \) is assigned to machine \( i \). \( S_k \) stands for the time when job \( k \) eventually starts to run. \( \tilde{L}(j) \) stands for the rest of jobs that have the lower priority than \( j \) according to the reports.

3. Job \( j \) chooses a machine \( i_j \in M \). Its tentative utility for being processed at machine \( i \) is:

\[
\hat{u}_j(i) := -w_j \hat{C}_{ij}(i) - \hat{p}_i(i)
\]

4. Job \( j \) is queued at machine \( i_j \) according to WSPT among all currently available jobs on the same machine which has not been processed yet. Meanwhile it pays the payment \( \hat{p}_i(i) \) to those jobs who are put off because its appearance.

5. The (tentative) completion time for every job \( k \in \tilde{L}(j) \), \( k < j \), \( S_k > \tilde{r}_i \) increases by \( \hat{p}_j \) due to \( j \)'s appearance. As a compensation, \( k \) receives a payment of \( \hat{w}_k \hat{p}_j \).

DLGM has a desirable feature of incentive compatibility on the job side. It has been proven that truthfully reporting the \( r_j, p_j \) and choosing the machine which maximizes its tentative utility \( \hat{u}_j \) is a dominant strategy for jobs in a restricted strategy space where jobs are honest about their weight \( w_j \). Moreover, truthfully reporting \( r_j, p_j, w_j \) and choosing a machine that maximizes its tentative utility \( \hat{u}_j \) is a “myopic best response”. Here myopic best response is a strategy that maximizes a job’s tentative utility at arrival or that maximizes a job’s utility under the assumption that it was the last job to arrive [3].

Besides, DLGM’s payment can be calculated in a polynomial time. This makes it applicable in online environment. It yields a 3.281 competitive rate result for minimizing the total weighted completion time.

III. COMPLETE DECENTRALIZED MACHINE SCHEDULING MECHANISM

In the distributed computing environment such as Grids or web services community nowadays, both the jobs (the consumers) and the machines (the providers) are independent entities. They attend the system to make profits. Although DLGM has many promising merits, it will not be suitable to systems that neglect machines’ desire for profits and strategy room. If any machine announces a lower price or smaller fake anticipated completion time, DLGM cannot work well or ensure its 3.281 competitive rate. To enable such a system to work well, we propose a Complete Decentralized Machine Scheduling Mechanism which views both jobs and machines as selfish rational players who aim to maximizing their utilities by attending the system or making actions.

We introduce our mechanism or model by dividing it to two parts naturally: the Job Agent’s part and Machine
Agent’s part. After that, we put them together to form our Complete Decentralized Machine Scheduling Mechanism.

**A. Job Agents’ Part**

We are given a set of jobs \( J = \{1,...,n\} \). Each job has its own private information about its release date \( r_j \), processing time \( p_j \), and an indifference cost \( w_j \). We use the notation \( t_j = (r_j, p_j, w_j) \) to denote the job \( j \)’s private information. Suppose there is one job agent for each job to make choices or actions to maximize its utility which equals to the value gained minus its payment.

**Action:**

1. To give a report about its private info strategically: \( \tilde{r}_j = (\tilde{r}_j, \tilde{p}_j, \tilde{w}_j) \) at time \( \tilde{t}_j \). \( \tilde{r}_j \geq r_j, \tilde{p}_j \geq p_j, \tilde{w}_j \geq w_j \).
2. To choose the machine to be processed on by itself.

**Utility / Value / Payment:**

A job \( j \)’s value \( v_j \) denotes the value received by the job owner after the job is finished. Its payment \( \pi_j \) denotes the cost for processing job \( j \). Utility \( u_j = v_j - \pi_j \) shows the job agent’s final profits. We keep the value part the same as DLGM while we modify the payment scheme.

In DLGM [2], job’s utility is defined as follow:

\[
\begin{align*}
    u_j &= v_j - \pi_j = -w_j C_j - \pi_j \\
    \text{If the job chooses machine } i, \text{ job’s } j \text{’s tentative utility at time } \tilde{t}_j \text{ is defined as follow:} \nonumber \\
    \hat{u}_j &= -w_j (\tilde{p}_j + D_y + \tilde{r}_j) - \tilde{p}_j Q_y \quad \text{..............(1)}
\end{align*}
\]

Here \( D_y \) stands for the time delay between job’s proclaimed release time \( \tilde{r}_j \) and the time at which the job is planned to start running. It can be calculated using the DLGM shown in the section before. \( Q_y \) stands for the sum of those jobs’ weights which are preempted by job \( j \) at that time.

In our model, we modify the job’s payment by setting:

\[
\pi_q = \tilde{p}_j Q^+_y + \tilde{w}_j D^+_y - \tilde{w}_j D_y
\]

Among all the machines, if we rank them by the policy which is “the smaller value of \( (\tilde{p}_j Q_y + \tilde{w}_j D_y) \), the higher rank it gets; and the smaller index of \( i \), the higher rank it is”, we get a ranking list of the machines. For any \( i \), we use the tuple \( (D^+_y, Q^+_y) \) to denote the time delay and payment weight of the very machine which is right behind the machine \( i \) in the list. We call \( (\tilde{p}_j Q_y + \tilde{w}_j D_y) \) or \( (D_y, Q_y) \) the machine’s bid because its value decides the job’s tentative utility and therefore influences directly the decision of which machines to be chosen. So if we choose machine according the policy ahead, \( (D^+_y, Q^+_y) \) is the next bid behind machine \( i \)’s bid after ranking. As for the last bid in the list, we can set an imagined machine which has a very big constant bid than other machine to yield its next bid.

So we have the job \( j \)’s tentative utility:

\[
\hat{u}_j = -w_j (\tilde{p}_j + D_y + \tilde{r}_j) - \tilde{p}_j Q_y + \tilde{w}_j D^+_y - \tilde{w}_j D_y \quad \text{....(2)}
\]

We prove that with the modified payment, it still works the same and keeps good characteristics on incentive compatibility as DLGM after finishing the integration of the whole system.

**B. Machine Agents’ Part**

We append the machine-side strategy space to the machine model of DLGM.

A set of parallel machines \( M = \{1,...,m\} \) offer places to run those jobs. At job \( j \)’s release time \( r_j \), machine \( i \) has its own private information \( D_y \) and \( Q_y \) to show its working status now. We use a tuple \( S_y = \{ D_y, Q_y \} \) to denote machine \( i \)’s private information at job \( j \)’s release time.

Every machine takes its favorite action to maximize its utility.

Before we go further, we want to make one thing clear: “Jobs’ strategy space is transparent to machines”. It means that machines run and charge the jobs according their proclaimed information, so they cannot or need not to be aware of the truth about whether the information is real. So we take all the superscripts of the job’s announced information for clarity.

**Actions:**

1. To give a fake report about its info \( \tilde{S}_y = \{ \tilde{D}_y, \tilde{Q}_y \} \) or bid \( B_y = \tilde{p}_j \tilde{D}_y + \tilde{w}_j \tilde{Q}_y \) to job \( j \).

**Utility / Value / Payment:**

A machine’s received value \( V_j \) is the sum of rewards paid by those jobs that choose this machine. The rewards are actually the payments of jobs in the job’s model of last section. Its payments \( \sigma_i \) are the cost for processing the jobs. It equals to the sum of the other jobs’ utility loss caused by the machine’s scheduling decision about the newly-come job. The utility of machines \( U_j \) equals to the values minus the payments which in our model is defined as:

\[
\begin{align*}
    U_j &= \sum_{j \in H(i)} (V_j - \sigma_j) = \sum_{j \in H(i)} (\pi_j - p_j Q_y) \\
    &= \sum_{j \in H(i)} ((p_j \tilde{Q}_y + w_j \tilde{D}_y - w_j \tilde{D}_y) - p_j Q_y) \quad \text{............(3)} \\
    &= \sum_{j \in H(i)} ((p_j \tilde{Q}_y + w_j \tilde{D}_y) - (p_j Q_y + w_j \tilde{D}_y))
\end{align*}
\]
We will prove that under such an environment, machines have incentive to truthfully report their working status: \( \bar{S}_j = S_j = \{ D_j, Q_j \} \)

C. Complete Decentralized Machine Scheduling Mechanism

The whole system/mechanism works as follows: When a job \( j \) arrives, it will send its private information report \( \bar{r}_j = \{ p_j, \bar{w}_j \} \) to all machines at \( \bar{r}_j \). Based on that and their present situations, machines calculate \( S_j = \{ D_j, Q_j \} \) and strategically send a bid \( \bar{S}_j = \{ D_j, Q_j \} \) back. After all the answering information from all machines are received, the job will choose one machine which has the smallest bid \( B_j = p_j \bar{D}_j + w_j \bar{Q}_j \) to be run on and gives the payment \( \pi_j = \bar{p}_j \bar{Q}_j + \bar{w}_j \bar{D}_j - \bar{w}_j \bar{D}_j \) to the machine. The chosen machine accepts the job and schedules it according to the WSPT policy. Meanwhile, every job \( k \) which is preempted by job \( i \) receives a compensatory payment of \( \bar{p}_j \bar{w}_k \) for the utility loss from the machine. The total compensation forms the payment of the machine: \( \sigma_j = \bar{p}_j \bar{Q}_j \).

IV. MERITS AND DISADVANTAGES

A. Incentive Compatibility

Job Part. Suppose machines are truthful. In fact, machines’ strategy spaces are transparent to the jobs because the job agent will pay the machine and receive the execution result exactly as the machine’s report denotes. Whether the machines’ report is true or not only affects machines and the whole system’s efficiency.

Theorem 1: Consider any job \( j \in J \). In our mechanism, it’s obvious to find out the following is true according to the machine’s payment definition:

(a) When truthfully reporting job’s weight \( w_j \), its tentative utility will equals to its ex-post utility.

(b) Restricting \( \tilde{w}_j = w_j \), truthfully reporting \( r_j \), \( p_j \), and choosing the machine that has the smallest bid under our mechanism is a dominant strategy.

Proof. (a) Whenever \( j \)’s tentative completion time increases by time \( T \), it will receive a compensation of \( \bar{w}_j T \). If \( \tilde{w}_j = w_j \), the compensation will equal the loss in utility.

(b) Given any job \( j \), suppose \( \tilde{w}_j = w_j \), the ex-post utility equals to the tentative utility at decision point \( \bar{r}_j \). So let’s therefore regard this as the tentative utility. Job \( j \)’s tentative utility is defined in Formula 2:

\[
\tilde{u}_j = -w_j(\bar{r}_j + \bar{p}_j + D_j) - \bar{p}_j \bar{Q}_j - \bar{w}_j \bar{D}_j + \bar{w}_j \bar{D}_j
\]

\[
= -w_j(\bar{r}_j + \bar{p}_j + \bar{D}_j) - \bar{p}_j \bar{Q}_j
\]

Notice the tentative utility scheme above has similar structure with the job’s tentative scheme in DLGM. Using the same method, it is easy to prove that truthfully reporting \( r_j \) and \( p_j \) is a dominant strategy for job \( j \).

Meanwhile, according to the definition of the tuple \( (D_j, Q_j) \), to maximize the tentative utility, the job must choose the machine which has the smallest bid \( B_j = p_j \bar{D}_j + w_j \bar{Q}_j \).

Machine Part. Suppose jobs always tell the truth. In our mechanism, every machine gives a report \( (\bar{D}_i, \bar{Q}_i) \) or a bid \( B_j = p_j \bar{D}_j + w_j \bar{Q}_j \) to every job \( j \). Among all the machines, the machine which has the smallest bid wins the job and will be paid by the job. Other machines will gain a utility of zero. According to Formula 3, we can get the following theory:

For every machine \( i \in \{1, \ldots, m\} \), suppose the machine who has the smallest bid or has the smallest machine index when it is not the only smallest bid except machine \( i \) is machine \( i^* \) with bid \( B^*_j = p_j \bar{D}_j + w_j \bar{Q}_j \). So machine \( i^* \)’s utility is:

\[
U_j = \begin{cases} 
B^*_j - (p_j \bar{D}_j + w_j \bar{Q}_j) & (B_j \leq B^*_j) \\
0 & (B_j > B^*_j)
\end{cases}
\]

Theorem 2: Consider any machine \( i \in M \), in our mechanism, the following conclusions are true:

(a) For every machine \( i \), truthfully revealing its \( D_i \) is a dominant strategy.

(b) For every machine \( i \), truthfully giving its real bid \( B_i = p_i \bar{D}_i + w_i \bar{Q}_i \) is a dominant strategy.

Proof. (a) First, the machine is not allowed to report a smaller \( \bar{D}_i \) than \( D_i \), because the machine cannot finish the job before its real finishing time in its scheduler. It is easy to limit this possibility by imposing a big fine for the machine which cannot finish the job on time. So we only need to consider the situation \( \bar{D}_i \geq D_i \).

Fix \( (D_i, Q_i), B^*_i \), and any pretended \( \bar{Q}_i \), if machine \( i \) can win the job, the machine’s utility will increase when \( \bar{D}_i \) decreases. Consider any \( \bar{D}_i > D_i \), if reporting \( \bar{D}_i \) can win the job, reporting \( D_i \) can win the job, reporting \( D_i \) can also win and get a higher utility. If reporting \( \bar{D}_i \) cannot win the job, reporting \( D_i \) will still have the possibility to win the job. So reporting \( \bar{D}_i = D_i \) dominates over reporting \( \bar{D}_i > D_i \).

(b) According to (a), we can rewrite the machine’s utility formula as follow:

\[
U_j = \begin{cases} 
B^*_j - (p_j \bar{D}_j + w_j \bar{Q}_j) & (B_j \leq B^*_j) \\
0 & (B_j > B^*_j)
\end{cases}
\]
Notice that the formula above is very similar to the classical truthful auction which is "the highest bid wins the auction and pays the second highest bid". In our case, if $p_j D_{ij} + w_j Q \leq B_i^*$, any bid $B_i \leq B_i^*$ will dominate and achieve the same utility. If $p_j D_{ij} + w_j Q > B_i^*$, it will be better for the machine drops the job, and bid with value bigger than $B_i^*$. In both cases, a bid $B_i = p_j D_{ij} + w_j Q$ dominates. So truly bid is a dominant strategy.

**Whole System Part.** Given machines are truthful, it will be a dominant strategy for every job to truthfully reporting its processing time as well as the release time restricting to tell the true weight and choose a machine which maximizes its tentative utility. Given jobs are truthful and choose the machine which maximizes its tentative utility, machines will choose to report truthfully; So based on the definition of Nash Equilibrium [14], we can say that for both the machines and jobs telling truth about their private information in our mechanism are dominant, and will achieve an Nash Equilibrium.

**B. Online Payment and Complete Decentralization**

According to the mechanism and the payment scheme, the payment to and from the jobs or machines has a polynomial time to calculate and would be paid between the job’s release time and its finish time. So the mechanism can be used in online environment.

Also notice that there is no central authority which holds the total system information, and all system attendee behave rationally to maximize their profits, the whole mechanism will be suitable to be used in complete decentralized environment.

**C. Negative Result**

In DLGM, Heydenreich et al. [2] prove that for each job $j$, reporting $(\tilde{r}_j, \tilde{p}_j, \tilde{w}_j) = (r_j, p_j, w_j)$ and choosing a machine maximizing its tentative utility is a myopic best response which means it maximizes the utility if the job $j$ is the last job coming to this system. Unfortunately in our mechanism, we cannot ensure that truthfully reporting all the private information is myopic dominant.

Heydenreich gives that conclusion by proving a critical premise: For any machine $j$, changing the report from $(\tilde{r}_j, \tilde{p}_j, \tilde{w}_j)$ to $(\tilde{r}_j, \tilde{p}_j, w_j)$ and choosing a machine that maximizes its tentative utility at time $\tilde{r}_j$ does not decrease $j$’s tentative utility in DLGM. We will give a counterexample to show that in our mechanism the change may cause a utility decline. So, even if the job is the last one to come, truthfully revealing all the information will not maximize the job’s utility. That means our mechanism is not myopic best response equilibrium. The counterexample is as follow: Suppose there are two machines $M_1$ and $M_2$. When job $j_2$ enters the system, $M_1$ is executing a job which still needs 1.1 seconds to finish and also has a job $j_1$ waiting in the line. $M_2$ has only one job $j_2$ to run immediately. $j_1$’s proclaimed information is (0,1,2); $j_2$’s proclaimed information is (1,2,6). $j_1$’s private information is (2,1,2,5).

We make a contrast with a fake report about the weight. Suppose $j_1$’s fake information is (2,1,3,5). When the weight is truthfully reported, $M_1$’s bid equals to 4.75 while $M_2$’s is 5. So the job will choose the former machine. Its tentative utility can be calculated using Formula 2 which equals to -12.5. When fake information is reported, $M_1$’s bid equals to 5.85 while $M_2$’s is 6. The job will still choose the same machine which gives the job a tentative utility of -12.4. So by giving a fake weight, the job may have a bigger tentative utility.

**V. IMPACT AND PERFORMANCE ANALYSIS**

**A. About the Incentive Compatibility**

Incentive compatibility is very important in decentralized system which has no central authority controlling every entity. By inheriting the incentive compatibility on the job side, our mechanism can entice the jobs to truthfully reveal its private information about its processing time and release time and choose the right machine. The appropriate assignment (which is done under the will of the job itself) and granting the real information about the jobs to the machines make the optimization to the total system possible and enable the system to run well.

Besides, our mechanism is also incentive compatible on the machine side. Machines are lured to be honest by the assignment and payment schemes in our mechanism. This means that those machines which win the jobs will not arbitrarily raise their charges. Also those machines which cannot win the jobs based on its real working situation will not make a price cutting. The former behavior will damage the job agents’ interests and the latter will cause chaos to the optimization of the whole system.

**B. About the Competitive Ratio**

Competitive ratio is used to measure the performance of an online system or mechanism. Our system’s overall metric is the total weighted completion time of all the jobs.

Our mechanism only changes the payment scheme in DLGM, while keeping the same assignment and schedule policy as DLGM. Jobs run exactly the same in both mechanisms. Changing the mechanism from DLGM to our mechanism will not change the job’s completion time. So our mechanism yields the same sum of weighted completion time, and also achieves a competitive ratio of 3.281 like the DLGM does.

**VI. CONCLUSION AND DISCUSSION**

In this paper, we consider an online parallel machine scheduling problem, and present a polynomial time, completely decentralized online mechanism by extending DLGM [2]. Our mechanism inherits most of the merits.
from DLGM: it is an online mechanism; it is incentive compatible in the job’s side; it yields a 3.281 competitive ratio for minimizing the total weighted completion time. Moreover, the mechanism is also incentive compatible with respect to the machines’ private information. Using this unique feature, we can address the problem of the machine’s “dishonesty” or their unilateral price-cutting strategy which may disturb the whole system or market.

However, neither DLGM nor our mechanism gives a dominant strategy equilibrium in which all jobs report all their data truthfully. Compared with DLGM, our mechanism is stricter about the premise of truthfully reporting the weight since truthfully reporting all the information is not a myopic dominant strategy. There may be other mechanism that has a better competitive ratio and performance on incentive compatibility.

At last, our machine part mechanism builds on a truthful open auction which has at least two bidders. A bulletin board to denote all the bids to both the job and all the machines is needed. And when there is only one bidder or other bids are not available temporarily, the mechanism needs to be adjusted.

ACKNOWLEDGEMENT

This work is supported by the 863 project #2007AA01-Z122 & project #2008AA01Z12, the National Natural Science Foundation of China Project #90604028, and 973 project #2004CB719406.

REFERENCES


