Non-clairvoyant Scheduling Games

Johanne Cohen · Christoph Dürr · Nguyen Kim Thang

Abstract In a scheduling game, each player owns a job and chooses a machine to execute it. While the social cost is the maximal load over all machines (makespan), the cost (disutility) of each player is the completion time of its own job. In the game, players may follow selfish strategies to optimize their cost and therefore their behaviors do not necessarily lead the game to an equilibrium. Even in the case there is an equilibrium, its makespan might be much larger than the social optimum, and this inefficiency is measured by the price of anarchy – the worst ratio between the makespan of an equilibrium and the optimum. Coordination mechanisms aim to reduce the price of anarchy by designing scheduling policies that specify how jobs assigned to a same machine are to be scheduled. Typically these policies define the schedule according to the processing times as announced by the jobs. One could wonder if there are policies that do not require this knowledge, and still provide a good price of anarchy. This would make the processing times be private information and avoid the problem of truthfulness. In this paper we study these so-called *non-clairvoyant* policies. In particular, we study the RANDOM policy that schedules the jobs in a random order without preemption, and the EQUI policy that schedules the jobs in parallel using time-multiplexing, assigning each job an equal fraction of CPU time.

For these models we study two important questions, the existence of Nash equilibria and the price of anarchy. We show that the game under RANDOM policy is a potential game for uniform machines or for two unrelated machines. However, it is not a potential game for three or more unrelated machines. Moreover, we prove that the game under the EQUI policy is a potential game.

Johanne Cohen
PRiSM
Université de Versailles St-Quentin-en-Yvelines, France

Christoph Dürr
CNRS and LIX
Ecole Polytechnique, Palaiseau, France

Nguyen Kim Thang
LAMSADE
Université Paris Dauphine, France
(Nguyen Kim is the family name)

Next, we analyze the inefficiency of EQUI policy. Interestingly, the (strong) price of anarchy of EQUI, a non-clairvoyant policy, is asymptotically the same as that of the best *strongly local* policy – policies in which a machine may look at the processing time of jobs assigned to it. The result also indicates that knowledge of jobs' characteristics is not necessarily needed.

Keywords: Algorithmic Game Theory; Scheduling Games; Coordination Mechanisms; Nash equilibria.

1 Introduction

With the development of the Internet, large-scale autonomous systems became more and more important. The systems consist of many independent and selfish agents who compete for the usage of shared resources. Every configuration has some social cost, as well as individual costs for every agent. Due to the lack of coordination, the equilibrium configurations may have high cost compared to the global social optimum and this inefficiency can be captured by the *price of anarchy* [25]. It is defined as the ratio between the worst case performance of Nash equilibrium [27] and the global optimum. Since the behavior of the agents is influenced by the individual costs, it is natural to come up with mechanisms that both force the existence of Nash equilibria and reduce the price of anarchy. The idea is to try to reflect the social cost in the individual costs, so that selfish agents' behaviors result in a socially desired solution. In particular we are interested in scheduling games, where every player has to choose one machine on which to execute its job. The individual cost of a player is the completion time of its job, and the social cost is the largest completion time over all jobs, the *makespan*. For these games, so called *coordination mechanisms* have been studied by Christodoulou et al. [10]. A *coordination mechanism* is a set of *local policies*, one for every machine, that specify a schedule for the jobs assigned to it, and the schedule can depend only on these jobs. Most prior studied policies depend on the processing times and need the jobs to announce their processing times. The jobs could try to influence the schedule to their advantage by announcing not their correct processing times. There are two ways to deal with this issue. One is to design *truthful coordination mechanisms* where jobs have an incentive to announce their real processing times. Another way is to design mechanisms that do not depend on the processing times at all and this is the subject of this paper: we study coordination mechanisms based on so called *non-clairvoyant policies* that we define in this section.

1.1 Preliminaries

Scheduling The *machine scheduling problem* is defined as follows: we are given n jobs, m machines and each job needs to be scheduled on exactly one machine. In the most general case machine speeds are unrelated, and for every job $1 \leq i \leq n$ and every machine $1 \leq j \leq m$ we are given an arbitrary processing time $p_{i,,j}$, which is the time spend by job i on machine j. A schedule σ is a function mapping each job to some machine. The *load* of a machine j in schedule σ is the total processing time of jobs assigned to this machine, i.e., $\ell_j = \sum_{i:\sigma(i)=j} p_{i,,j}$. The *makespan* of a schedule is the maximal load over all machines, and is the social cost of a schedule. It is NP-hard to

compute the *global optimum* even for identical machines, that is when $p_{i,,j}$ does not depend on j, see [20, problem SS8]. We denote by OPT the makespan of the optimal schedule.

Machine environments We consider four different machine environments, which all have their own justification. The most general environment concerns unrelated machines as defined above and is denoted $R||C_{\max}$. In the *identical* machine scheduling model, denoted $P||C_{\max}$, every job i comes with a length p_i such that $p_{i,,j} = p_i$ for every machine j. In the *uniform* machine scheduling model, denoted $Q||C_{\max}$, again every job has *length* p_i and every machine j a speed s_j such that $p_{i,,j} = p_i/s_j$. For the *restricted identical* machine model, every job i comes with a length p_i and a set of machines S_i on which it can be scheduled, such that $p_{i,,j} = p_i$ for $j \in S_i$ and $p_{i,,j} = \infty$ otherwise. In [7] this model is denoted $PMPM||C_{\max}$, and in [24] it is denoted $B||C_{\max}$.

Scheduling game What we described so far are well known and extensively studied classical scheduling problems. But now consider the situation where each of the n jobs is owned by an independent agent. In this paper we will sometimes abuse notation and identify the agent with his job. The agents do not care about the social optimum, their goal is to complete their job as soon as possible. We consider the situation where each agent can freely decide on which machine its job is to be scheduled. The actual schedule however is not decided by the agents. We rather fix a *policy*, known to all agents, which specifies the actual schedule, once all agents assigned their jobs to machines. Different policies are defined below.

In the paper, we concentrate on *pure strategies* where each agent selects a single machine to process its job. Such a mapping σ is called a *strategy profile*. Each agent is aware of the decisions made by other agents and behaves selfishly. The *individual cost* of a job is defined as its completion time. A *pure Nash equilibrium* is a schedule in which no agent has an incentive to unilaterally switch to another machine. In this paper we will simply omit the adjective *pure*, since there is no confusion possible. A *strong* Nash equilibrium is a schedule that is resilient to deviations of any coalition, i.e., no group of agents can cooperate and change their strategies in such a way that all players in the group strictly decrease their costs, see [3, 16]. For some given strategy profile, a *better response move* of a job i is a strategy (machine) j such that if job i changes to job j, while all other players stick to their strategy, the cost of i decreases strictly. If there is such a move, we say that this job is *unhappy*, otherwise it is *happy*. In this setting a Nash equilibrium is a strategy profile where all jobs are happy. The *better-response dynamic* is the process of repeatedly choosing an arbitrary unhappy job and changing it to an arbitrary better response move. A *potential game* is a game in which for any instance, the better-response dynamic always converges [26]. Such a property is typically shown by the use of a potential function, which maps strategy profiles to non-negative numerical values. The game is called a *strong potential game* if there is a potential function with the property that if an agent improves its individual cost by some amount Δ, then the potential function decreases by the same amount Δ.

A *coordination mechanism* is a set of *scheduling policies*, one for each machine, that determines how to schedule jobs assigned to a machine [10]. The idea is to connect the individual cost to the social cost, in such a way that the selfishness of the agents will lead to equilibria that have low social cost. How good is a given coordination mechanism? This is measured by the well-known *price of anarchy* (PoA), see [25]. It is defined as
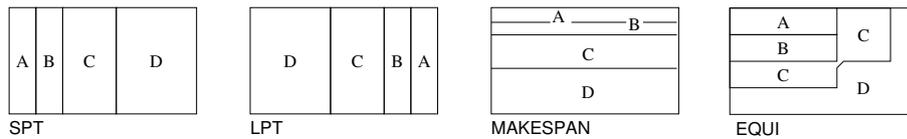
Fig. 1 Different scheduling policies for $p_A = 1, p_B = 1, p_C = 2, p_D = 3$. Tie is broken arbitrarily between jobs A and B. The rectangles represent the schedules on a single machine with time going from left to right and the height of a block being the amount of CPU assigned to the job.

the ratio between the cost of the worst Nash equilibrium and the optimal cost, which is not an equilibrium in general. We also consider the *strong price of anarchy* (SPoA) which is the extension of the price of anarchy applied to strong Nash equilibria [16].

Policies A policy is a rule that specifies how the jobs that are assigned to a machine are to be scheduled. We now define several policies, and give proper credit to the introducing papers in section 1.2.

We distinguish between *local, strongly local* and *non-clairvoyant* policies. Let S_j be the set of jobs assigned to machine j. A policy is *local* if the scheduling of jobs on machine j depends only on the parameters of jobs in S_j, i.e., it may looks at the processing time $p_{i,k}$ of a job $i \in S_j$ on any machine k. A policy is *strongly local* if it looks only at the processing time of jobs in S_j on machine j. We call a policy *non-clairvoyant* if the scheduling of jobs on machine j does not depend on the processing time of any job on any machine. In this paper we only study coordination mechanisms that use the same policy for all machines, as opposed to Angel et al. [1]. SPT and LPT are policies that schedule the jobs without preemption respectively in order of increasing or decreasing processing times with a deterministic tie-breaking rule for each machine. An interesting property of SPT is that it minimizes the sum of the completion times, while LPT has a better price of anarchy, because it incites small jobs to go on the least loaded machine which smoothes the loads. A policy that relates individual costs even stronger to the social cost is MAKESPAN, where jobs are scheduled in parallel on one machine using time-multiplexing and assigned each job a fraction of the CPU that is proportional to its processing time. As a result all jobs complete at the same time, and the individual cost is the load of the machine. All these policies are *deterministic*, in the sense that they map strategy profiles to a determined schedule. This is opposed to *randomized policies* which map strategy profiles to a distribution of schedules.

What could a scheduler do in the non-clairvoyant case? He could either schedule the jobs in a random order or in parallel. The RANDOM policy schedules the jobs in a random order without preemption. Consider a job i assigned to machine j in the schedule σ, then the cost of i under the RANDOM policy is its expected completion time, i.e.,

$$c_i = p_{i,j} + \frac{1}{2} \sum_{i': \sigma(i')=j, \ i' \neq i} p_{i',j}.$$

In other words the expected completion time of i is half of the total load of the machine, where job i counts twice. Again, as for MAKESPAN, the individual and social cost in RANDOM are strongly related, and it is likely that these policies should have the same price of anarchy. That is is indeed the case except for unrelated machines.

Another natural non-clairvoyant policy is EQUI, which has been studied for example in [13] in the different context of online algorithms. As MAKESPAN it schedules the jobs

in parallel preemptivly using time-multiplexing, but it assigns to every job the same fraction of the CPU. Suppose there are k jobs with processing times $p_{1,j} \leq p_{2,j} \leq \ldots \leq p_{k,j}$ assigned to machine j, we renumbered jobs from 1 to k for this example. Since, each job receives the same amount of resource, then job 1 is completed at time $c_1 = kp_{1,j}$. At that time, all jobs have remaining processing time $(p_{2,j} - p_{1,j}) \leq (p_{3,j} - p_{1,j}) \leq \ldots \leq (p_{k,j} - p_{1,j})$. Now the machine splits its resource into $k-1$ parts until the moment job 2 is completed, which is at $kp_{1,j} + (k-1)(p_{2,j} - p_{1,j}) = p_{1,j} + (k-1)p_{2,j}$. In general, the completion time of job i, which is also its cost, under EQUI policy is:

$$
\begin{aligned}
c_i &= c_{i-1} + (k - i + 1)(p_{i,j} - p_{i-1,j}) \\
&= p_{1,j} + \ldots + p_{i-1,j} + (k - i + 1)p_{i,j}
\end{aligned}
\tag{1}
$$

We already distinguished policies depending on what information is needed from the jobs. In addition we distinguish between *preemptive* and *non-preemptive* policies, depending on the schedule that is produced. Among the policies we considered so far, only MAKESPAN and EQUI are preemptive, in the sense that they rely on time-multiplexing, which consists in executing arbitrary small slices of the jobs. Note that, EQUI is a realistic and quite popular policy. It is implemented in many operating systems such as Unix and Windows. See figure 1 for an illustration of these five policies.

Example For illustration consider the scheduling game on parallel identical machines and the *EQUI* policy. Here each of the n jobs has a processing time p_i, for $1 \leq i \neq n$. Every agent selects a machine, which is described by a strategy profile $\sigma : \{1, \ldots, n\} \to \{1, \ldots, m\}$. Now the individual cost of agent i, is the completion time of its job, which for this policy is

$$
\sum_{i'} \min\{p_{i,j}, p_{i',j}\},
$$

where the sum is taken over all jobs i' assigned to the same machine as i, i.e. $\sigma(i) = \sigma(i')$.

1.2 Previous and related work

Coordination mechanism are related to local search algorithms. The local improvement moves in the local search algorithm correspond to the better-response moves of players in the game defined by the coordination mechanism. Some results on local search algorithms for scheduling problem are surveyed in [29].

Most previous work concerned non-preemptive strongly local policies, in particular the MAKESPAN policy. Czumaj and Vöcking [11] gave tight results $\Theta(\log m / \log \log m)$ of its price of anarchy for pure Nash equilibria on uniform machines. Fiat et al. [15] extended this result for the strong price of anarchy, and obtained the tight bound $\Theta(\log m / (\log \log m)^2)$. In addition, Gairing et al. [19] and Awerbuch et al. [4] gave tight bounds for the price of anarchy for restricted identical machines.

Coordination mechanism design was introduced by Christodoulou et al. [10]. They studied the LPT policy on identical machines. Immorlica et al. [24] studied coordination mechanism for all four machine environments and gave a survey on the results for non-preemptive strongly local policies. They also analyzed the existence of pure Nash equilibria under SPT, LPT and RANDOM for certain machine environments and the speed of convergence to equilibrium of the better response dynamics. Precisely, they

proved that the game is a potential game under the policies SPT on unrelated machines, LPT on uniform or restricted identical machines, and RANDOM on restricted identical machines. The policy EQUI has been studied in [13] for its competitive ratio. The results are summarized in Table 1.

Azar et al. [6] introduced the inefficiency-based local policy which has price of anarchy $O(\log m)$ on unrelated machines. Moreover, they also proved that every non-preemptive strongly local policy with an additional assumption has price of anarchy at least $m/2$, which shows a sharp difference between strongly local and local policies.

model \ policy	MAKESPAN	SPT	LPT	RANDOM	EQUI
identical	$2 - \frac{2}{m+1}$ [17, 28]	$2 - \frac{1}{m}$ [21, 24]	$\frac{4}{3} - \frac{1}{3m}$ [22, 10]	$2 - \frac{2}{m+1}$ [17, 28]	$2 - \frac{1}{m}$
uniform	$\Theta(\frac{\log m}{\log\log m})$ [11]	$\Theta(\log m)$ [2, 24]	$1.52 \le PoA \le 1.59$ [12, 18, 24]	$\Theta(\frac{\log m}{\log\log m})$ [11]	$\Theta(\log m)$
restricted id.	$\Theta(\frac{\log m}{\log\log m})$ [19, 4]	$\Theta(\log m)$ [2, 24]	$\Theta(\log m)$ [5, 24]	$\Theta(\frac{\log m}{\log\log m})$ [19, 4]	$\Theta(\log m)$
unrelated	unbounded [28]	$\Theta(m)$ [9, 23, 6]	unbounded	$\Theta(m)$ [24]	$\Theta(m)$

Table 1 Price of anarchy under different strongly local and non-clairvoyant policies. The right most column is our contribution.

1.3 Our contribution

We are interested in *admissible* non-clairvoyant policies – policies that always induce a Nash equilibrium for any instance of the game. In the game, maybe more important than the question of existence of Nash equilibrium is the question of convergence to an equilibria. Since no processing times are known to the coordination mechanism it is impossible to compute some equilibria or even decide if a given assignment of jobs to machines is an equilibria. Besides, if all processing times are known to all jobs, it makes sense to let the jobs evolve according to the better-response dynamics, until they eventually reach an equilibria. Therefore in the paper, we are interested in the convergence of the better-response dynamic.

In Section 2, we study the existence of Nash equilibrium under the non-clairvoyant policies RANDOM and EQUI. We show that for the RANDOM policy, the game is a potential game on uniform machines. We also show that on two unrelated machines, it is a potential game, but for three unrelated machines or more, the better-response dynamic does not converge. Moreover, we prove that for the EQUI policy, the game is a (strong) potential game, see Table 2.

In Section 3, we analyze the price of anarchy and the strong price of anarchy of EQUI. We observe that RANDOM is slightly better than EQUI except for the unrelated model. In the unrelated model, interestingly, the price of anarchy of EQUI reaches the lower bound in [6] on the PoA of any strongly local policy with some additional condition. The latter shows that although there is a clear difference between strongly local and local policies with respect to the price of anarchy, our results indicate that in contrast, restricting strongly local policies to be non-clairvoyant does not really affect the price of anarchy. Moreover, EQUI policy does not need any knowledge about jobs'

model \ policy	MAKESPAN	SPT	LPT	RANDOM	EQUI
identical			Yes [24]	Yes [24]	
uniform	Yes	Yes		Yes	Yes
restricted id.	[14]	[24]		Yes [24]	
unrelated			No	(*)	

Table 2 Convergence of the better response dynamic and existence of equlibria. (*) Does not converge for $m \geq 3$ machines, but converges for $m = 2$ machines.

characteristics, even their identities (IDs) which are useful in designing policies with low price of anarchy in [6, 8].

2 Existence of Nash equilibrium

The results in this section are summarized as follows.

Summary of results on the existence of Nash equilibrium: *We consider the scheduling game under different policies in different machine environments.*

1. *For the RANDOM policy on uniform machines, it is a potential game. For the RANDOM policy on unrelated machines, it is not a potential game for 3 or more machines, but it is a potential game for 2 machines.*
2. *For the EQUI policy it is an exact potential game.*

2.1 The RANDOM policy on uniform machines

In the RANDOM policy, the cost of a job is its expected completion time. If the load of machine j is ℓ_j then the cost of job i assigned to machine j is $\frac{1}{2}(\ell_j + p_{i,j})$. Observe that a job i on machine j has an incentive to move to machine j' if and only if $p_{i,j} + \ell_j > 2p_{i,j'} + \ell_{j'}$.

In this section, we consider uniform machines. Let $p_1 \leq p_2 \leq \ldots \leq p_n$ be the job lengths and $s_1 \geq s_2 \geq \ldots \geq s_m$ be the machine speeds. Now the processing time of job i on machine j is p_i/s_j.

Theorem 1 *The scheduling game on uniform machines and the RANDOM policy is a potential game.*

Proof Let $\sigma : \{1, \ldots, n\} \to \{1, \ldots, m\}$ be a strategy profile. The potential function is defined as

$$\Phi(\sigma) := \sum_{j=1}^{m} \frac{\ell_j^2}{s_j} + 3 \sum_{i=1}^{n} \frac{p_i^2}{s_{\sigma(i)}},$$

where ℓ_j is the load of machine j, i.e. the sum of p_i over all jobs i with $\sigma(i) = j$.

Now consider a job i that makes a better response move from machine a to machine b. If ℓ_a, ℓ_b denote the loads respectively of machine a and b *before* the move, then by definition of a better response move we have the inequality

$$\frac{\ell_a + p_i}{s_a} > \frac{\ell_b + 2p_i}{s_b}. \tag{2}$$

Let σ' be the profile *after* the move of job j. The change in the potential is

$$\Phi(\sigma') - \Phi(\sigma) = \frac{(\ell_b + p_i)^2}{s_b} + \frac{(\ell_a - p_i)^2}{s_a} + \frac{3p_i^2}{s_b} - \frac{\ell_a^2}{s_a} - \frac{\ell_b^2}{s_b} - \frac{3p_i^2}{s_a}$$

$$= \frac{\ell_b^2 + 2\ell_b p_i + p_i^2 + 3p_i^2 - \ell_b^2}{s_b} + \frac{\ell_a^2 - 2\ell_a p_i + p_i^2 - \ell_a^2 - 3p_i^2}{s_a}$$

$$= 2p_i \left(\frac{\ell_b + 2p_i}{s_b} - \frac{\ell_a + p_i}{s_a} \right) < 0$$

due to (2). Therefore, the potential function Φ strictly decreases at every better response move. $\square$

2.2 The RANDOM policy for unrelated machines

In the following, we will characterize the game under the RANDOM policy in the unrelated model as a function of the number of machines.

Theorem 2 *The scheduling game on 2 unrelated machines with the RANDOM is a potential game.*

Proof Let $\sigma : \{1, \ldots, n\} \to \{1, 2\}$ be the current strategy profile, meaning that job i is assigned to machine $\sigma(i)$. By $\overline{\sigma}(i)$ we denote the opposite machine to machine $\sigma(i)$. Let ℓ_j be the load of machine j in strategy profile σ, which is $\sum_{i:\sigma(i)=j} p_{i,j}$. Define the potential function as

$$\Phi(\sigma) := (\ell_1 - \ell_2)^2 + 3\sum_{i=1}^{n} p_{i,\sigma(i)}^2.$$

We claim that the potential function Φ strictly decreases at every better response move. Let i be a job moving from say machine a to machine b, while strictly decreasing its cost, i.e.

$$\ell_b + 2p_{i,b} - \ell_a - p_{i,a} < 0, \tag{3}$$

where ℓ_a, ℓ_b are the loads before the move.

Let σ' be the strategy profile after the move of job i. We have:

$$\Phi(\sigma') - \Phi(\sigma) = (\ell_a - p_{i,a} - \ell_b - p_{i,b})^2 - (\ell_a - \ell_b)^2 + 3(p_{i,b}^2 - p_{i,a}^2)$$

$$= (p_{i,a} + p_{i,b})[3(p_{i,b} - p_{i,a}) - (2\ell_a - p_{i,a} - 2\ell_b - p_{i,b})]$$

$$= 2(p_{i,a} + p_{i,b})[(2p_{i,b} + \ell_b) - (p_{i,a} + \ell_a)] < 0$$

due to (3). Therefore, the potential function Φ decreases strictly at every better response move. $\square$

However, for 3 or more machines, the better-response dynamic does not necessarily converge.

Lemma 1 *The better-response dynamic does not converge under the RANDOM policy on 3 or more unrelated machines.*

Proof We give a simple four-job instance, with the following processing times. For convenience we name the jobs A, B, C, D.

$p_{i,j}$	1	2	3
A	90	84	∞
B	96	2	∞
C	138	100	∞
D	∞	254	300

Now we describe a cyclic sequence of better response moves, where each job strictly decreases its cost, showing that the game does not converge. In the following table, we describe in each line, the current strategy profile, a better response move of an unhappy job and its cost improvement. For example the first line shows the strategy profile, where jobs A, B are on machine 1, C is on machine 2 and D on machine 3. Then job the cost of job A is 138 and moving to machine 2, its cost drops to 134. The subsequent line show similar better response moves, which end in the initial strategy profile.

1	2	3	move	cost improvement
AB	C	D	$1 \xrightarrow{A} 2$	$138 > 134$
B	AC	D	$1 \xrightarrow{B} 2$	$96 > 94$
	ABC	D	$2 \xrightarrow{C} 1$	$143 > 138$
C	AB	D	$3 \xrightarrow{D} 2$	$300 > 297$
C	ABD		$2 \xrightarrow{B} 1$	$171 > 165$
BC	AD		$2 \xrightarrow{A} 1$	$211 > 207$
ABC	D		$1 \xrightarrow{C} 2$	$231 > 227$
AB	CD		$2 \xrightarrow{D} 3$	$304 > 300$
AB	C	D		

$\square$

Note that although there exists a cycle in better-reponse dynamic of the game under RANDOM policy, this does not mean that the game possesses no equilibrium, see [26].

2.3 The EQUI policy

In the EQUI policy, the cost of job i assigned to machine j is given by expression (1). Here is an alternative formulation for the cost

$$c_i = \sum_{\substack{i':\sigma(i')=j \\ p_{i',j} \leq p_{i,j}}} p_{i',j} + \sum_{\substack{i':\sigma(i')=j \\ p_{i',j} > p_{i,j}}} p_{i,j}$$

Lemma 2 *The game with the EQUI policy is an exact potential game.*

Proof Let σ be the current strategy profile, meaning $\sigma(i)$ is the current machine on which job i is scheduled. Consider the following potential function.

$$\Phi(\sigma) = \frac{1}{2} \sum_{i=1}^{n} \left(c_i + p_{i,\sigma(i)} \right)$$

We prove that if a job makes a better response move then the potential function strictly decreases. Let t be a job that moves from machine a to b, while stricly decreasing its cost from c_t to c'_t. We have

$$
c_t = \left(\sum_{\substack{i:\sigma(i)=a, i \neq t \\ p_{i,a} \leq p_{t,a}}} p_{i,a} + \sum_{\substack{i:\sigma(i)=a, i \neq t \\ p_{i,a} > p_{t,a}}} p_{t,a} \right) + p_{t,a}
$$

$$
> \left(\sum_{\substack{i:\sigma(i)=b, i \neq t \\ p_{i,b} \leq p_{t,b}}} p_{i,b} + \sum_{\substack{i:\sigma(i)=b, i \neq t \\ p_{i,b} > p_{t,b}}} p_{t,b} \right) + p_{t,b}
$$

$$
= c'_t.
$$

Let σ' be the strategy profile after the move of job t. Note that in σ' the processing time of all jobs except i and the cost of all jobs scheduled on machine different to a and b stay the same. Thus, the change in the potential depends only on the jobs scheduled on machines a and b.

$$
2 \cdot \Delta\Phi = \left(\sum_{i:\sigma'(i)=a} (c'_i + p_{i,a}) + \sum_{i:\sigma'(i)=b, i \neq t} (c'_i + p_{i,b}) + p_{t,b} \right)
$$

$$
- \left(\sum_{i:\sigma(i)=a, i \neq t} (c_i + p_{i,a}) + \sum_{i:\sigma(i)=b} (c_i + p_{i,b}) + p_{t,a} \right) + (c'_t - c_t)
$$

$$
= \sum_{i:\sigma(i)=a, i \neq t} (c'_i - c_i) + \sum_{i:\sigma(i)=b, i \neq t} (c'_i - c_i) + (c'_t - c_t) + p_{t,b} - p_{t,a}
$$

since $\sigma(i) = \sigma'(i) \ \forall i \neq t$.

Consider a job $i \neq t$ on machine a. If the processing time of i is at most that of t then the difference between its new and old cost is exactly $-p_{i,a}$. Otherwise if the processing time of i is strictly greater than that of t then this difference is exactly $-p_{t,a}$. Analogously for jobs on machine b. Hence,

$$
2 \cdot \Delta\Phi = \left(\sum_{\substack{i:\sigma(i)=b, i \neq t \\ p_{i,b} \leq p_{t,b}}} p_{i,b} + \sum_{\substack{i:\sigma(i)=b, i \neq t \\ p_{i,b} > p_{t,b}}} p_{t,b} + p_{t,b} \right) +
$$

$$
+ \left(\sum_{\substack{i:\sigma(i)=a, i \neq t \\ p_{i,a} \leq p_{t,a}}} -p_{i,a} + \sum_{\substack{i:\sigma(i)=a, i \neq t \\ p_{i,a} > p_{t,a}}} -p_{t,a} - p_{t,a} \right) + (c'_t - c_t)
$$

$$
= 2 \cdot (c'_t - c_t) < 0
$$

Therefore, the game with the EQUI policy is an exact potential game. $\qquad\square$

Now we strengthen the statement of the previous lemma.

Theorem 3 *The game with the **EQUI** policy is a strong potential game, in the sense that the better-response dynamic converges even with deviations of coalitions.*

Proof Let S be a coalition and define its total cost $c(S) := \sum_{i \in S} c_i$. We study a better response move of S by dividing the process into two phases: in the first phase, all jobs in S move out (disappear) from the game and in the second phase, jobs from S move back (appear) into the game at their new strategies. We argue that after the first phase, the change in the potential is $\Delta\Phi = -c(S)$ and after the second phase $\Delta\Phi = c'(S)$. Since the argument is the same, we only prove it for the first phase; the second phase can be done similarly. Fix a machine a and suppose without loss of generality that all the jobs assigned to a are $1, \ldots, k$ for some k. Also to simplify notation we denote $q_i = p_{i,a}$ and assume $q_1 \leq \ldots \leq q_k$. Let $R = S \cap \sigma^{-1}(j) = \{i_1 \leq \ldots \leq i_r\}$ be the set of jobs in the coalition that are scheduled on this machine. Then,

$$c(R) = \sum_{j=1}^{r} c_{i_j} = \sum_{j=1}^{r} \left(q_1 + q_2 + \ldots + q_{i_j - 1} + (k - i_j + 1)q_{i_j} \right)$$

The jobs in R partition the jobs $\{1, \ldots, k\}$ into $r + 1$ parts: part $j \in \{0, \ldots, r\}$ is $[i_j + 1, j_{j+1}]$, where for convenience we denote $i_0 = 0$ and $i_{r+1} = k$. After the move out of R, the change in cost of a job $t \notin R$ scheduled on the machine with index in $[i_j + 1, i_{j+1}]$ is $q_{i_1} + q_{i_2} + \ldots + q_{i_{j-1}} + (r - j)q_t$. Hence, the difference in the potential restricted to machine a after the first phase $\Delta\Phi|_a$ satisfies:

$$-2\Delta\Phi|_a = \left[\sum_{j=0}^{r} \sum_{\substack{t \notin R \\ t \in [i_j+1, i_{j+1}]}} q_{i_1} + q_{i_2} + \ldots + q_{i_{j-1}} + (r - j)q_t \right]$$
$$+ \left[c(R) + (q_{i_1} + q_{i_2} + \ldots + q_{i_r}) \right]$$
$$= \left[\sum_{j=1}^{r} (k - i_j)q_{i_j} + \sum_{j=0}^{r} \sum_{\substack{t \notin R \\ t \in [i_j+1, i_{j+1}]}} (r - j)q_t \right]$$
$$+ \left[c(R) + (q_{i_1} + q_{i_2} + \ldots + q_{i_r}) \right]$$
$$= \sum_{j=1}^{r} \left(q_1 + q_2 + \ldots + q_{i_j - 1} + (k - i_j + 1)q_{i_j} \right) + c(R)$$
$$= 2 \cdot c(R)$$

where in the first term of these equalities, we distinguish between the cost change of all jobs not in the coalition and the cost change of the jobs in the coalition, disapearing from the game. The potential change after the first phase is simply the sum of all the changes over all machines, so $\Delta\Phi = -c(S)$.

By the same argument, after the second phase we have $\Delta\Phi = c'(S)$. Therefore, the net change over both phases is $c'(S) - c(S)$. In conclusion, the game is a strong potential game. $\square$

3 Inefficiency of Equilibria under the EQUI policy

In this section, we study the inefficiency of the game under the EQUI policy which is captured by the price of anarchy (PoA) and the strong price of anarchy (SPoA). Note that the set of strong Nash equilibria is a subset of that of Nash equilibria so the SPoA is at most as large as the PoA. We state the main theorem of this section. Whenever we bound *(S)PoA* we mean that the bound applies to both the price of anarchy and the strong price of anarchy.

Summary of results on the price of anarchy: *The game under the EQUI policy has the following inefficiency.*

1. *For identical machines, the (S)PoA is $2 - \frac{1}{m}$.*
2. *For uniform machines, the (S)PoA is $\Theta(\min\{\log m, r\})$ where r is the number of different machine's speeds in the model.*
3. *For restricted identical machines, the (S)PoA is $\Theta(\log m)$.*
4. *For unrelated machines, the (S)PoA is $\Theta(m)$.*

We first give a characterization for strong Nash equilibrium in the game, which connects the equilibria to the strong ones. This characterization is useful in settling tight bounds of the strong price of anarchy in the game.

Lemma 3 *Suppose in a Nash equilibrium there is a coalition T that makes a collective move such that each job in T improves strictly its cost. Then this move preserves the number of jobs on every machine.*

Proof For a proof by contradiction, let η be an equilibrium that is not strong, and let T be a coalition as stated in the claim. Suppose that the number of jobs on the machines is not preserved by the move of T. Let j be a machine that has strictly more jobs after the move, and among all jobs migrating to j, let $o \in T$ be the job with smallest length p_o. Let k and k' be the numbers of jobs on j before and after the move of T, respectively ($k' > k$). We claim that job o could already improve its cost by unilaterally moving to j, contradicting that η is a Nash equilibrium. Consider equilibrium η, if o moves to machine j, its cost would be:

$$c_o = (k + 1 - w)p_{o,j} + \sum_{i:p_{i,j}<p_{o,j}} p_{i,j}$$
$$= (k - w + 1)p_{o,j} + \sum_{i:p_{i,j}<p_{o,j},i\notin T} p_{i,j} + \sum_{i:p_{i,j}<p_{o,j},i\in T} p_{i,j}$$

where w is the number of jobs on machine j in η with length strictly less than $p_{o,j}$.

Let w' be the number of jobs on machine j after the move of T with length strictly less than $p_{o,j}$. Since o has the smallest length among all jobs migrating to j, $w' \leq w$. The cost of o after the move of T is:

$$c'_o = (k' - w')p_{o,j} + \sum_{i:p_{i,j}<p_{o,j},i\notin T} p_{i,j}$$

We have:

$$c'_o - c_o = \left[(k' - w') - (k - w + 1)\right] p_{i,j} - \sum_{i:p_{i,j} < p_{o,j}, i \in T} p_{i,j}$$

$$\geq (w - w')p_{i,j} - \sum_{i:p_{i,j} < p_{o,j}, i \in T} p_{i,j}$$

$$\geq (w - w')p_{i,j} - (w - w')p_{i,j} = 0$$

where the first inequality follows from $k' \geq k + 1$ and the second inequality uses $|\{i : p_{i,j} < p_{o,j}, i \in T\}| = w - w'$. Since job o has incentive to cooperate and move to machine j, o also get better off by unilaterally changing its strategy, so η is not an equilibrium. $\qquad\square$

3.1 Identical machines

In case of identical machines, the analysis of the PoA is quite similar to the well-known analysis of Graham's greedy load balancing algorithm that assigns the jobs to the least load machine, processing jobs in arbitrary order, see [21]. Here we show that the (S)PoA matches exactly the approximation factor of the greedy algorithm.

Proposition 1 *For identical machines, the (S)PoA is $2 - \frac{1}{m}$. Moreover, there is an instance in which all equilibria have cost at least $(2 - \frac{2}{m})OPT$.*

Proof (Upper bound) First we prove that PoA is upper-bounded by $2 - 1/m$. Let σ be an equilibrium and $\ell_{\max}$ be the makespan of this equilibrium. Let i be a job (with processing time p_i) that has cost $\ell_{\max}$. Hence, $p_i \leq OPT$. Since σ is an equilibrium, the fact that job i has no incentive to move to any other machine j implies $\ell_{\max} \leq \ell_j + p_i$ for all machines j different to $\sigma(i)$, where ℓ_j is the load of machine j. Summing up these inequalities over all machines j we get $m\ell_{\max} \leq \sum_{j=1}^{m} \ell_j + (m - 1)p_i$. Moreover, for any assignment of jobs to identical machines, $\sum_{j=1}^{m} \ell_j \leq mOPT$. Therefore, $m\ell_{\max} \leq (2m - 1)OPT$, i.e., PoA $\leq 2 - 1/m$.

(*Lower bound*) Now we give an instance in which OPT equals m and all equilibria have cost at least $2m - 2$. In the instance, there are m machines and $m(m - 1) + 1$ jobs in which all jobs have processing time 1 except one with processing time m. In an optimum assignment, the big job is scheduled on one machine and all $m(m - 1)$ unit jobs are evenly assigned to the other machines, producing makespan m. We claim that in any equilibrium, every machine has at least $(m - 1)$ unit jobs. Suppose there is a machine with at most $m - 2$ jobs. Since there are $m(m - 1)$ jobs of unit processing time, there must be a machine j with at least m unit jobs in the equilibrium. A unit job on machine j has cost at least m and it has incentive to move to the machine with less than $m - 2$ jobs and get a smaller cost (at most $m - 1$). This gives a contradiction and shows that any equilibrium, every machine has at least $m - 1$ unit jobs. Now consider the machine with the big job. In addition this machine has at least $m - 2$ unit jobs, so its load is at least $m + (m - 2)$. Therefore, the makespan of the equilibrium is at least $(2 - 2/m)OPT$.

Consider the schedule in which there are $(m - 1)$ unit jobs on every machine and the job with processing time m on some arbitrary machine. It is straightforward that this is an equilibrium. By Lemma 3, this equilibrium is also a strong one. Hence, (S)PoA $\geq 2 - 1/m$. $\qquad\square$

3.2 Uniform machines

For uniform machines, an upper bound $O(\log m)$ on the PoA of any deterministic policy in this machine environment is proved by Immorlica et al. [24]. In this section, we investigate the lower bound and show that the bound $O(\log m)$ is essentially tight.

In the following, we present a family of game instances in which the PoA, together with the SPoA, are $\Omega(\log m)$. The instances are inspired by the ones proving the lower bound of the competitive ratio of the greedy algorithm for uniform machine in [2].

Family of Game Instances There are $k + 1$ groups of machines $G_0, G_1, \ldots, G_k$, each machine in group G_j has speed 2^{-j} for $0 \leq j \leq k$. Group G_0 has $m_0 = 1$ machine, group G_j has m_j machines which is recursively defined as $m_j = \sum_{t=0}^{j-1} m_t \cdot 2^{j-t}$. Moreover, there are $k + 1$ groups of jobs $J_0, J_1, \ldots, J_k$, for $0 \leq j \leq k - 1$ each group J_j consists of $2m_j$ jobs of length 2^{-j} and group J_k consists of $3m_k$ jobs of length 2^{-k}. The total number of machines is $m = \sum_{j=0}^{k} m_j = 1 + \frac{2}{3}(4^k - 1) + 2 \cdot 4^{k-1}$, thus $k = \Omega(\log m)$. See figure 2 for illustration.

Consider a schedule that is a two-to-one mapping from the job group J_j to the machine group G_j, for every $j < k$, and that is a three-to-one mapping from the job group J_k to the machine group G_k. The load on a machine in group G_j for $j < k$ is 2 and each machine in G_k has load 3. Hence, $OPT \leq 3$.
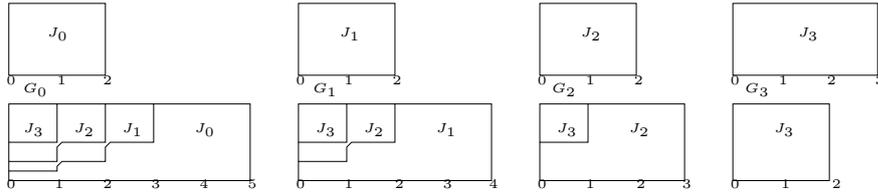

Fig. 2 Illustration of the schedule with makespan 3 (upper part) and the strategy profile σ (lower part) in the game instance for $k = 3$. Each machine group is represented by one of its machines.

Consider a schedule (strategy profile) σ such that for every $0 \leq j \leq k$, in each machine of group G_j (with speed 2^{-j}), there are 2 jobs of length 2^{-j}, and for every $j < i \leq k$, there are 2^{i-j} jobs of length 2^{-i}. Each machine in group G_j has 2^{k-j+1} jobs and has load $2 \cdot 2^{-j}/2^{-j} + \sum_{t=1}^{k-j} 2^t \cdot 2^{-(j+t)}/2^{-j} = k - j + 2$, so the makespan of this schedule is $k + 2$. We claim that this strategy profile is a Nash equilibrium, moreover it is a strong one.

Lemma 4 *The strategy profile σ is a Nash equilibrium.*

Proof First, we show that, in strategy profile σ, the cost of a job in J_j is equal to $k - j + 2$. Fix a machine in G_t. We are only interested in case $t \leq j$ since in σ, no job in J_j is assigned to a machine of group G_t with $t > j$. On this machine, there are exactly 2^{j-t+1} jobs with processing time at least 2^{-j}. So, the cost of a job in J_j scheduled on this machine is:

$$\frac{1}{2^{-t}} \left[\left(2^{k-t} \cdot 2^{-k} + 2^{(k-1)-t} \cdot 2^{-(k-1)} + \ldots + 2^{(j+1)-t} \cdot 2^{-(j+1)} \right) + 2^{j-t+1} \cdot 2^{-j} \right]$$
$$= k + 2 - j.$$

Now we argue that σ is an equilibrium. Suppose that a job i in J_j moves from its current machine to a machine of group G_t. If $j \leq t$, i has the greatest length among all jobs assigned to this new machine, so the new cost of i is the new load of the machine which is $(k - t + 2) + 2^{-j}/2^{-t} > k - j + 2$. If $j > t$ then there are $(2^{j-t+1} + 1)$ jobs with length at least 2^{-j} on i's new machine. Hence, the new cost of i is:

$$\frac{1}{2^{-t}} \left[\left(2^{k-t} \cdot 2^{-k} + 2^{(k-1)-t} \cdot 2^{-(k-1)} + \ldots + 2^{(j+1)-t} \cdot 2^{-(j+1)} \right) + (2^{j-t+1} + 1) \cdot 2^{-j} \right]$$
$$> k + 2 - j.$$

Therefore, no job can improve its cost by changing its strategy. $\qquad\square$

Using Lemma 3, we show that σ is indeed a strong equilibrium.

Lemma 5 *Strategy profile σ is a strong Nash equilibrium.*

Proof Suppose σ is not a strong Nash equilibrium, then there exists a coalition T such that all jobs in T strictly decrease their costs and after the move of T, all machines have the same number of jobs as in σ (by Lemma 3). Observe that the cost of a job in J_k (with the least length among all jobs in the instance) depends only on the number of jobs scheduled on its machine. With such a move of T, if there are some jobs in J_k involved in the coalition, none of them can strictly decrease its cost. Hence, $T \cap J_k = \emptyset$. Consider jobs in J_{k-1}. Since jobs in J_k stay in their machines and they incur the same load 1 on each machine, the cost of a job in J_{k-1}, if it involves in T, depends only on the number of jobs which are not in J_k and are scheduled on its new machine. However, this number is preserved after the move of T (by Lemma 3 and $T \cap J_k = \emptyset$), so the cost of a job in J_{k-1} stays the same, i.e., the job has no incentive to involve in T. The argument holds for groups of jobs $J_{k-2}, \ldots, J_0$. Therefore, $T = \emptyset$ meaning that σ is a strong equilibrium. $\qquad\square$

The previous lemmas imply that for uniform machines, the PoA of EQUI is $\Omega(\log m)$.

Theorem 4 *For uniform machines, the (S)PoA of EQUI is $\Theta(\log m)$.*

3.3 Restricted Identical Machines

The upper bound of the price of anarchy of EQUI on restricted identical machines follows immediately by Immorlica et al. [24]. In [24], an instance was given which shows that any deterministic non-preemptive coordination mechanism has PoA $\Omega(\log m)$. However, EQUI is a preemptive policy, and the instance cannot be adapted. In this section, we show that the price of anarchy of the EQUI policy on restricted identical machines is also $\Omega(\log m)$ using another instance.

Theorem 5 *For restricted identical machines, the (S)PoA is $\Theta(\log m)$.*

Proof The upper bound follows from [24]. We show now the lower bound. We adapt a game instance from the proof of Lemma 5. Let $(m_j)_{j=0}^{k}$ be a sequence defined as $m_0 = 1, m_1 = 2$ and $m_j = m_0 + \ldots + m_{j-1}$, i.e., $m_j = 3 \cdot 2^{j-2}$ for every $j \geq 2$. Let $m = \sum_{j=0}^{k} m_j = 3 \cdot 2^{k-1}$. Hence $k = \Omega(\log m)$.

In the instance, there are m machines which are divided into $k+1$ groups $G_0, \ldots, G_k$ where group G_j consists of m_j machines. There are also $k+1$ job groups $J_0, J_1, \ldots, J_k$ where group J_j contains $3 \cdot 2^j m_j$ jobs of processing time 2^{-j}.

We first describe a schedule μ which will be proved to be a Nash equilibrium. On each machine in group G_j for $0 \le j \le k$, there are 2^{j+1} jobs of length 2^{-j} and for every $j < i \le k$, there are 2^i jobs of length 2^{-i}. The strategy set of each job is the following. Jobs in group J_j can be scheduled on all m_j machines of group G_j. Moreover, a job in J_j can be additionally scheduled on its current machine in μ.

We claim that μ is a equilibrium. Observe that on each machine of group G_j, there are exactly 2^{i+1} jobs of processing time at least 2^{-i} for all $j \le i \le k$ and the total load of jobs with processing time strictly smaller than 2^{-i} (on the machine) is $k - i$. Thus, the cost of each job in group J_i is $k - i + 2$ in μ and if a job switches the strategy, its cost would be strictly greater than $k - j + 2$. In addition, using Lemma 3 and by the same argument as in Lemma 5, we have that this equilibrium is indeed a strong one.

If we schedule evenly all jobs of group J_j on m_j machines of G_j for $0 \le j \le k$ then the makespan is bounded by 3, so $OPT \le 3$. The makespan of the strong equilibrium above is $k + 1$, which gives the (S)PoA is at least $(k + 1)/3 = \Omega(\log m)$. $\qquad\square$

3.4 Unrelated Machines

In this section, we prove that the PoA of the game under the EQUI policy is upper bounded by $2m$. Interestingly, without any knowledge of jobs' characteristics, the inefficiency of EQUI – a non-clairvoyant policy – is the same up to a constant compared to that of SPT – the best strongly local policy with price of anarchy $\Theta(m)$.

Theorem 6 *For unrelated machines, the price of anarchy of policy EQUI is at most* $2m$.

Proof For job i, let q_i be the smallest processing time of i among all machines, i.e., $q_i := \min_j p_{i,j}$ and let $Q(i)$ be the machine j minimizing $p_{i,j}$. Without loss of generality we assume that jobs are indexed such that $q_1 \le q_2 \le \ldots \le q_n$. Note that $\sum_{i=1}^n q_i \le m \cdot OPT$, where OPT is the optimal makespan, as usual. First, we claim the following lemma.

We claim that In any Nash equilibrium, the cost c_i of job i is at most

$$2q_1 + \ldots + 2q_{i-1} + (n - i + 1)q_i. \tag{4}$$

The theorem would follow from the claim by the following argument. Since the expression (4) is increasing in i and at $i = n$ this term is $2\sum_{i=1}^n q_i \le 2m \cdot OPT$, the cost of each job in an equilibrium is bounded by $2m \cdot OPT$, so the price of anarchy is at most $2m$.

The proof of the claim is by induction on i. The cost of job 1 on machine $Q(1)$ would be at most nq_1, simply because there are at most n jobs on this machine. Therefore the cost of job 1 in the Nash equilibrium is also at most nq_1. Assume the induction hypothesis holds until index $i - 1$. Consider job i. Since the strategy profile is a Nash equilibrium, i's current cost is at most its cost if moving to machine $Q(i)$. We distinguish different cases. In these cases, denote c_i' as the new cost of i if it moves to machine $Q(i)$

1. Case all jobs t scheduled on machine $Q(i)$ satisfy $t > i$.

   This case is very similar to the basis case. There are at most $n - i$ jobs on machine $Q(i)$, beside i. The completion time of job i is then at most $(n - i + 1)q_i$ which is upper bounded by (4). For the remaining cases, we assume that there is a job $i' < i$ scheduled on $Q(i)$.

2. Case there is a job $t < i$ on machine $Q(i)$ such that $p_{t,Q(i)} \geq p_{i,Q(i)} (= q_i)$.

   Since $p_{t,Q(i)} \geq q_i$, the new cost of job i is not more than the new cost of job t. Moreover, the new cost of job t is increased by exactly q_i, so the new cost of i is bounded by

$$
\begin{aligned}
c_i' &\leq c_t + q_i \\
&\leq 2q_1 + \ldots + 2q_{t-1} + (n - t + 1)q_t + q_i \\
&= 2q_1 + \ldots + 2q_{t-1} + 2(i - t)q_t + (n - 2i + t + 1)q_t + q_i \\
&\leq 2q_1 + \ldots + 2q_{t-1} + 2q_t + \ldots + 2q_{i-1} + (n - i + 1)q_i,
\end{aligned}
$$

   where the first inequality uses the induction hypothesis and the last inequality is due to $t < i$ and $q_t \leq q_{t+1} \leq \ldots \leq q_i$.

3. Case every job t scheduled on machine $Q(i)$ with $p_{t,Q(i)} \geq q_i$ satisfies $t \geq i$.

   Since we are not in the first two cases, there is a job $t < i$ on machine $Q(i)$ with $p_{t,Q(i)} < q_i$. Let i' be the job of greatest index among all jobs scheduled on $Q(i)$ with smaller processing time than q_i. All jobs t scheduled on $Q(i)$ and having smaller processing time than that of i, also have smaller index because $q_t \leq p_{t,Q(i)} \leq q_i$. Therefore i' is precisely the last job to complete before i. At the completion time of i' there are still $q_i - p_{i',Q(i)} \leq q_i - q_{i'}$ units of i to be processed. By the case assumption, there are at most $(n - i)$ jobs with processing time greater than that of i. Therefore the new cost of i is at most

$$
\begin{aligned}
c_i' &= c_{i'} + (n - i + 1)(q_i - q_{i'}) \\
&\leq 2q_1 + \ldots + 2q_{i'-1} + (n - i' + 1)q_{i'} + (n - i + 1)(q_i - q_{i'}) \\
&= 2q_1 + \ldots + 2q_{i'-1} + (i - i')q_{i'} + (n - i + 1)q_i \\
&\leq 2q_1 + \ldots + 2q_{i'-1} + (q_{i'} + \ldots + q_{i-1}) + (n - i + 1)q_i \\
&\leq 2q_1 + \ldots + 2q_{i-1} + (n - i + 1)q_i
\end{aligned}
$$

   where the first inequality uses the induction hypothesis and the third inequality is due to the monotonicity of the sequence $(q_j)_{j=1}^n$.

This completes the proof of the claim, and therefore of the theorem. $\qquad\square$

We provide a game instance showing that the upper bound analyzed above is tight. The instance is inspired by the work of Azar et al. [6]. In the following lemma, we prove the lower bound of the PoA of the game under the EQUI policy.

Lemma 6 *The (strong) price of anarchy of EQUI is at least $(m + 1)/4$.*

Proof Let $n_j := \frac{2(m-1)!}{(j-1)!}$ and $n := \sum_{j=1}^m n_j$. Consider the set of m machines and m groups of jobs $J_1, J_2, \ldots, J_m$. In group J_j ($1 \leq j \leq m - 1$), there are n_j jobs that can be scheduled on machine j or $j + 1$ except the last group (J_m) which has a single job that can be only scheduled on machine m. Each job in group J_j ($1 \leq j \leq m - 1$) has processing time $p_{j,j} = \frac{(j-1)!}{(m-1)!} = \frac{2}{n_j}$ on machine j and has processing time $p_{j,j+1} =$

$\frac{j!}{2(m-1)!} = \frac{1}{n_{j+1}}$ on machine $j + 1$. The job in J_m has processing time $p_{m,m} = 1$ on machine m.

Consider the strategy profile in which half of the jobs in J_j ($1 \le j \le m - 1$) are scheduled on machine j and the other half are scheduled on machine $j + 1$ (jobs in J_m are scheduled on machine m). We claim that this strategy profile is a Nash equilibrium. Note that the cost of jobs in the same group and scheduled on the same machine are the same. The cost of each job in group J_j on machine j is the load of the machine, because its processing time is greater than that of jobs in group J_{j-1} on machine m, and this load equals $\frac{n_{j-1}}{2}p_{j-1,j} + \frac{n_j}{2}p_{j,j} = \frac{j-1}{2} + 1 = \frac{j+1}{2}$. Each job in group J_j has smaller processing time than that of each job in group J_{j+1} on machine $j + 1$, thus the cost of the former is $\frac{n_j + n_{j+1}}{2}p_{j,j+1} = \frac{j+1}{2}$. Hence, no job in group J_j ($1 \le j \le m - 1$) has an incentive to move and the job in group J_m cannot switch its strategy. Therefore, the strategy profile is an equilibrium.

Moreover, we prove that this equilibrium is indeed a strong one. Suppose that it is not a strong equilibrium, i.e., there is a coalition S such that all jobs in S can strictly decrease their cost. Again, by Lemma 3, the number of jobs on each machine remains the same after the move of S. We call a job in group J_j *moving up* if it moves from machine j to $j + 1$ and *moving down* if it moves from machine $j + 1$ to j. First, we claim that no job has an incentive to move up. If a job in group J_j moves up, as only jobs in J_j and J_{j+1} can use machine $j + 1$ and $p_{j,j+1} < p_{j+1,j+1}$, its new cost would be $p_{j,j+1} \cdot (n_j + n_{j+1})$ which equals its old cost. Hence, no one can strictly decrease its cost by moving up. Among all jobs in S, consider the one who moves down to the machine j^* of smallest index. By the choice of j^*, there is no job moving down from machine j^* and as claimed above, no job moving up from j^*. Hence, the job moving to machine j^* cannot strictly decrease its cost – that contradicts to the assumption that all jobs in S strictly get better off. Therefore, the equilibrium is a strong one.

Consider a schedule in which jobs in group J_j ($1 \le j \le m$) are assigned to machine j and this schedule has makespan 2, hence $OPT \le 2$. The makespan of the above (strong) Nash equilibrium is the load on machine m, that is equal to $(m + 1)/2$. Then, the (strong) price of anarchy is at least $(m + 1)/4$. $\qquad\square$

4 Conclusion and Open questions

In this paper, we studied coordination mechanisms under non-clairvoyant policies. We first studied whether some policies are admissible – which is the first property that we expect from a policy. We studied in detail the existence of Nash equilibrium under the RANDOM and the EQUI policies. Next, we analyzed the inefficiency (PoA) of the EQUI policy and showed that the knowledge of the agents processing times is not really necessary, since EQUI behaves nearly as good as the best known strongly local policy SPT. One more advantage is that there is no need to implement EQUI policy (if using it) since this popular policy exists in many operating systems.

An interesting open question is to answer (prove or disprove) whether the gap of the PoA between strongly local and local policies can be closed. Does there exist a (preemptive, randomized) strongly local policies with the PoA poly-logarithmic on m? Azar et al. [6] proved that with an additional condition, this gap is closed. Can we bypass this condition? Besides, does there exist a truthful coordination mechanism based on strongly local policy with PoA as $o(m)$?

Another interesting open problem is the speed of convergence to approximate a Nash equilibrium for RANDOM and EQUI in the machines environment where equilibrium is guaranteed to exist.

References

1. Eric Angel, Evripidis Bampis, and Fanny Pascual. Truthful algorithms for scheduling selfish tasks on parallel machines. *Theoretical Computer Science (TCS)*, 369: 157–168, 2006.
2. James Aspnes, Yossi Azar, Amos Fiat, Serge A. Plotkin, and Orli Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *Journal of the ACM*, 44(3):486–504, 1997.
3. Robert J. Aumann. Acceptable points in general cooperative *n*-person games. *Contributions to the Theory of Games IV, Annals of Math. Study*, 1959.
4. Baruch Awerbuch, Yossi Azar, Yossi Richter, and Dekel Tsur. Tradeoffs in worst-case equilibria. *Theoretical Computer Science*, 361(2-3):200–209, 2006.
5. Yossi Azar, Joseph Naor, and Raphael Rom. The competitiveness of on-line assignments. *Journal of Algorithms*, 18(2):221–237, 1995.
6. Yossi Azar, Kamal Jain, and Vahab S. Mirrokni. (Almost) optimal coordination mechanisms for unrelated machine scheduling. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 323–332, 2008.
7. Peter Brucker. *Scheduling Algorithms*. Springer, 3rd edition, 2001.
8. Ioannis Caragiannis. Efficient coordination mechanisms for unrelated machine scheduling. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 815–824, 2009.
9. Y. Cho and S. Sahni. Bounds for list schedules on uniform processors. *SIAM Journal on Computing*, 9:91–103, 1980.
10. George Christodoulou, Elias Koutsoupias, and Akash Nanavati. Coordination mechanisms. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 345–357, 2004.
11. Artur Czumaj and Berthold Vöcking. Tight bounds for worst-case equilibria. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 413–420, 2002.
12. G. Dobson. Scheduling independent tasks on uniform processors. *SIAM Journal on Computing*, 13:721–716, 1984.
13. Jeff Edmonds. Scheduling in the dark. In *Proceedings of the 31st ACM Symposium on Theory of Computing (STOC)*, pages 179–188, 1999.
14. Eyal Even-Dar, Alexander Kesselman, and Yishay Mansour. Convergence time to Nash equilibrium in load balancing. *ACM Transactions on Algorithms*, 3(3), 2007.
15. Amos Fiat, Haim Kaplan, Meital Levy, Svetlana Olonetsky, and Ronen Shabo. On the price of stability for designing undirected networks with fair cost allocations. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 608–618, 2006.
16. Amos Fiat, Haim Kaplan, Meital Levy, and Svetlana Olonetsky. Strong price of anarchy for machine load balancing. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming*, pages 583–594, 2007.

17. G. Finn and E. Horowitz. A linear time approximation algorithm for multiprocessor scheduling. *BIT*, 19:312–320, 1979.

18. D. K. Friesen. Tighter bounds for LPT scheduling on uniform processors. *SIAM Journal on Computing*, 16:554–560, 1987.

19. Martin Gairing, Thomas Lücking, Marios Mavronicolas, and Burkhard Monien. Computing Nash equilibria for scheduling on restricted parallel links. In *Proceedings of the 36th ACM Symposium on Theory of Computing (STOC)*, pages 613–622, 2004.

20. Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

21. R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

22. R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 45:416–429, 1969.

23. O. H. Ibarra and C. E. Kim. Heuristic algorithms for scheduling independent tasks on nonidentical processors. *Journal of the ACM*, 24:280–289, 1977.

24. Nicole Immorlica, Li Li, Vahab S. Mirrokni, and Andreas Schulz. Coordination mechanisms for selfish scheduling. In *Proceedings of the 1st International Workshop on Internet and Network Economics (WINE)*, pages 55–69, 2005.

25. E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3(2):65–69, 2009.

26. Dov Monderer and Lloyd S. Shapley. Potential games. *Games and Economic Behavior*, 14:124–143, 1996.

27. John Nash. Non-cooperative games. *The Annals of Mathematics*, 54(2):286–295, 1951.

28. Petra Schuurman and Tjark Vredeveld. Performance guarantees of local search for multiprocessor scheduling. *Informs Journal on Computing*, 361(1):52–63, 2007.

29. Tjark Vredeveld. *Combinatorial Approximation Algorithms: Guaranteed Versus Experimental Performance*. PhD thesis, Technische Universiteit Eindhoven, The Netherlands, 2002.