Coordination Mechanisms for Selfish Multi-Organization Scheduling

Johanne Cohen*, Daniel Cordeiro†, Denis Trystram‡§, and Frédéric Wagner‡
*PRiSM, Université de Versailles St-Quentin-en-Yvelines
Versailles, France
Email: johanne.cohen@prism.uvsq.fr
†LIG, Grenoble University
Montbonnot Saint-Martin, France
‡Grenoble Technical University
Montbonnot Saint-Martin, France
§Institut universitaire de France
Paris, France
Email: {cordeiro, trystram, wagner}@imag.fr

***Index Terms*—scheduling, multiple organizations, algorithmic game theory, coordination mechanisms**

***Abstract*—We conduct a game theoretic analysis on the problem of scheduling jobs on computing platforms composed of several independent and selfish organizations, known as the Multi-Organization Scheduling Problem (MOSP). Each organization shares resources and jobs with others, expecting to decrease the makespan of its own jobs.**

We modeled MOSP as a non-cooperative game where each agent is responsible for assigning all jobs belonging to a particular organization to the available processors. The local scheduling of these jobs is defined by coordination mechanisms that first prioritize local jobs and then schedule the jobs from others according to some given priority. When different priorities are given individually to the jobs — like in classical scheduling algorithms such as LPT or SPT — then no pure ϵ-approximate equilibrium is possible for values of ϵ less than 2. We also prove that even deciding whether a given instance admits or not a pure Nash equilibrium is co-NP hard. When these priorities are given to entire organizations, we show the existence of an algorithm that always computes a pure ρ-approximate equilibrium using any ρ-approximation list scheduling algorithm. Finally, we prove that the price of anarchy of the MOSP game using this mechanism is asymptotically bounded by 2.

I. Introduction

A. Motivation and Context

We are interested in analyzing the problem of scheduling independent jobs on a multi-organization parallel and distributed platform using a game theoretic approach.

We are considering workloads of sequential jobs (also called *bags-of-tasks*), which are one of the most common computational models for the target parallel platforms. They are popular because they scale efficiently and are easy to program. For many applications such as Folding@home, jobs are submitted by batches and one batch can be started only after the previous one has completed. Timely completion is often crucial to guarantee for sequential jobs [1]. Thus, focus is put on the makespan of a series of jobs within a batch (the makespan is defined as the maximum completion time over all the jobs in the batch).

The target parallel and distributed systems are composed of organizations sharing their hierarchical computational resources (that can be clusters or multi-core machines). The jobs of a workload are submitted to a particular user interface machine within a cluster or a node of the multi-core machine. Then, the corresponding jobs are transferred to be executed on the available resources according to the scheduling policy. Of course, it is intuitively better to allocate a job in the resources of its organization. However, the users expect to be able to execute their own jobs more efficiently by sharing their own resources with others. A global and centralized view of the whole system may help to better balance the jobs.

The large number of available processors or cores and the variety of demands from the different users make the scheduling problem of such parallel platforms really hard in practice. In order to fully exploit such systems, we need sophisticated scheduling algorithms that encourage the organizations to share their resources and, at the same time, that respect each user's own interests. It is crucial to determine schedules that optimize the allocation of the jobs for the whole platform in order to achieve good system performances. On the other hand, it is important to guarantee the performance perceived by each organization in order to provide an incentive to the cooperation between all the organizations.

The goal of this work is to study the problem as a non-cooperative game, providing coordination mechanisms allowing each organization to continuously attempt to obtain the best possible makespan until an equilibrium is achieved. As we will see, the classical approach which assigned a different agent to each job is not suitable for this problem; we model the problem as a game where the organizations are considered as agents that interact with each other. In this case, we will prove

This work has been supported by the bilateral brazilian-french CAPES-COFECUB program (project # Ma 660/10) and by a Google Research Award.

that there always exists a Nash equilibrium and we propose a polynomial-time algorithm for computing a ϵ-approximation of the equilibria.

B. Related Work

The problem of scheduling jobs on hierarchical parallel platforms has been extensively studied. A first set of works consider the execution of parallel rigid jobs [8]. When parallel jobs must be scheduled on consecutive processors (like a rectangle), the problem relates to the Multiple Strip packing problem [21], [22]. Bougeret *et al*. [2] presented an asymptotic $(1+\epsilon)$-approximation AFPTAS with additive constant $O(1)$ and running-time polynomial in n and in $1/\epsilon$. Parallel jobs scheduled on non-contiguous processors were studied by Schwiegelshohn *et al*. [19]. They presented a 3-approximation for the maximum completion time (makespan) when all jobs are known in advance (which may correspond, for instance, to a batch) and a 5-approximation for the on-line case. Pascual *et al*. [15], [16] extended the problem for the case where resources and jobs are aggregated in *independent organizations* that have a local objective for their jobs besides the global makespan. Their main contribution is the analysis of a centralized 3-approximation algorithm that always incite these organizations to cooperate.

Studying workloads of sequential jobs, Cohen *et al*. [5] have broaden the concept of individualism of such organizations. More than always guarantee incentive to cooperate, their model introduces a novel *selfishness restriction*, where configurations that allow one organization to *cheat* the devised global schedule by re-inserting its jobs earlier in the local schedules are forbidden. The authors showed that when all organizations behave selfishly, any approximation algorithm for MOSP($C_{\max}$) has a ratio greater than or equal to $2-\frac{2}{N}$ regarding the optimal makespan with local constraints. Their work also analyzed the complexity of the problem when organizations are locally interested in minimizing either the makespan (MOSP($C_{\max}$)) or the average completion times (MOSP($\sum C_i$)). The problem was shown to be NP-Hard for both local objectives and 2-approximation algorithms for solving these problems were presented and analyzed.

Scheduling problems are also intensively studied from the point of view of game theory. They are modeled as *selfish load balancing problems*. In their seminal work, Koutsoupias and Papadimitriou [13] studied games where each job is assigned to a different agent, whose objective is to minimize the makespan. Such games always induce a pure Nash equilibrium because they are potential games [9]. The price of anarchy — also introduced by [13] — comparing the cost of Nash equilibria to the cost of the optimal (social cost) has been studied on these kind of games [6], [20]. Similar games with more general cost functions have also been studied. See, e.g., [3], [6], [12]–[14], [18]. Czumaj *et al*. [6] gave tight results ($\Theta(\log m/\log\log m)$) for the price of anarchy on pure Nash equilibria using uniform parallel machines.

Christodoulou *et al*. [4] introduced the notion of *coordination mechanisms*. Their goal was to reduce the price of anarchy by connecting the local cost with the social cost. Christodoulou *et al*. [4] studied the well-known LPT policy on identical machines. Immorlica *et al*. [12] extended these results by studying other non-preemptive local policies (namely SPT, LPT and RANDOM). They analyzed the existence of pure Nash equilibria under these policies by proving that the game is a potential game. Dürr *et al*. [7] focus on a preemptive local policy called EQUI. They proved that such games have a pure Nash equilibrium with a different kind of machine environment. Moreover, when using the EQUI policy, the game has a strong Nash equilibrium (in which no group of agents can cooperate and change its strategy in such a way that all agents in the group strictly decrease their costs).

C. Contributions and Outline of the Paper

As suggested in the previous sections, the notions of cooperation and selfishness on the Multi-Organization Scheduling Problem have been studied using the classical combinatorial optimization approaches. In this paper, we extend the notions of independence and selfishness of each organization by allowing each of them to rationally choose the best strategy for their jobs.

Our main contribution is a game theoretic model for the MOSP problem — contemplating the individualism and selfishness of organizations collaborating in a grid computing system — that leads to configurations with a cost as close as wanted to a pure Nash equilibrium, and with a bounded price of anarchy. A preliminary study on different coordination mechanisms — defining how each organization will deal locally with the jobs to be executed in its machines — shows that typical scheduling algorithms can lead the game to configurations that are far from the equilibrium. We prove that even deciding if such games admit pure Nash equilibria is hard. Then, we propose a new coordination mechanism and an algorithm that is able to calculate a pure ρ-approximate equilibrium using any ρ-approximate list scheduling algorithm for the restricted case where each organization has a single processor. In particular, the algorithm calculates pure Nash equilibria if the algorithm used is an optimal one. We also measure the social inefficiency caused by the selfishness of each organization, showing that the price of anarchy is asymptotically bounded by 2.

The remaining of the paper is organized as follows. Section II describes in details the multi-organization scheduling problem and how it can be modeled using game theory. Section III studies a mechanism based on classical scheduling algorithms, showing that these kind of mechanisms do not admit ϵ-approximate equilibrium for $\epsilon < 2$ and that the problem of deciding whether a particular instance admits a pure Nash equilibrium is co-NP hard. In Section IV we define a new mechanism where priorities are given to organizations. We present an algorithm that constructs pure ρ-approximate equilibria and analyze the price of anarchy in such games. Finally, some conclusion remarks are presented in Section V.

II. Problem Description and Notations

A. The Multi-Organization Scheduling Problem

We study a game theoretic model for the problem known in literature as the Multi-Organization Scheduling Problem (MOSP). The problem was first introduced by Pascual *et al*. [15] and models scheduling on grid computing systems, with particular interest on the performance objectives of each individual organization that compounds the grid computing system.

MOSP models a grid computing system where users and resources are associated with an organization. Each of these organizations owns a physical cluster of identical machines or multi-core CPU that are interconnected with each other. Like in any such computing platform, they share resources and exchange jobs with each other in order to simultaneously maximize the profits of the collectivity and their own interests. All organizations intent to minimize the total completion time of all jobs (i.e., the global makespan) while they individually try to minimize the completion time of their own jobs.

Each organization is completely independent and can behave in selfish ways. We assume that whenever possible, each organization will prioritize its own jobs, scheduling and executing them before any job from other organizations.

More importantly, a solution for the MOSP problem guarantees that no organization will be aggrieved by the final schedule. MOSP's local constraints guarantee that all organizations will always have incentive to cooperate by assuring that the local makespan obtained by each organization is at least as good as the makespan that the organization could have obtained using only its own resources (its *local makespan*). This restriction imposed on the final schedule is called MOSP's *local constraint*.

Formally, we define our target platform as a grid computing system with N different organizations interconnected by a middleware. Each organization $O^{(k)}$ $(1 \leq k \leq N)$ has $m^{(k)}$ processors which can be used to run jobs submitted by users from any organization.

Each organization $O^{(k)}$ has $n^{(k)}$ jobs to execute. Each job $J_i^{(k)}$ $(1 \leq i \leq n^{(k)})$ will use one processor for exactly $p_i^{(k)}$ units of time. No preemption is allowed, i.e., after its activation, a job runs until its completion at time $C_i^{(k)}$.

We denote the makespan of a particular organization k by $C_{\max}^{(k)} = \max_{1 \leq i \leq n^{(k)}} (C_i^{(k)})$. The makespan that the organization could obtain by scheduling its jobs alone in its own processors is denoted by $C_{\max}^{(k)\ local}$. The global makespan for the entire computing platform is defined as $C_{\max} = \max_{1 \leq k \leq N} (C_{\max}^{(k)})$.

The MOSP optimization problem can then be stated as follows:

$$\text{minimize } C_{\max} \text{ such that, for all } k \ (1 \leq k \leq N), \quad C_{\max}^{(k)} \leq C_{\max}^{(k)\ local}$$

B. Game Theoretic Model

Recent works on the MOSP problem were focused on centralized schedulers that compute a solution for the problem that respects MOSP local and selfish constraints. We study the interactions between the independent organizations as the result of rational selfish agents attempting to reach an equilibrium. This section describes MOSP modeled as a non-cooperative game.

We associate one selfish agent with each organization, i.e., jobs originally from organization $O^{(k)}$ are managed by agent k. Each agent can choose on which organization each one of its jobs will be executed, knowing that each selfish organization will schedule first its own jobs. Note that each agent is responsible for a set of jobs, while most of the previous works described in Section I-B assign one agent to each available job.

In other words, a pure strategy $S^{(k)}$ for agent k is a vector of $n^{(k)}$ elements such that $S^{(k)}(i)$ corresponds to the organization chosen by player k for job $J_i^{(k)}$. We denote $\mathcal{S}^{(k)}$ the set of all the strategies for agent k.

A configuration (or profile) M is a vector $(S^{(1)}, S^{(2)}, \ldots, S^{(N)})$ such that $S^{(k)}$ is a strategy of agent k. The cost of an agent k under configuration M — denoted by $\text{cost}^{(k)}(M)$ — corresponds to the makespan obtained by organization $O^{(k)}$: $C_{\max}^{(k)}$.

When all agents choose an assignment for their jobs in such way that no agent has incentive to change its strategy, we say that this configuration is a *pure Nash equilibrium*. More formally, a configuration M is a pure Nash equilibrium if all agent k, satisfies the following property: $\forall s \in \mathcal{S}^{(k)}$, $\text{cost}^{(k)}(M) \leq \text{cost}^{(k)}(s, M_{-k})$, where M_{-k} is a vector $(S^{(1)}, S^{(2)}, S^{(k-1)}, S^{(k+1)} \ldots, S^{(N)})$.

Some games do not always feature pure Nash equilibria. These games can still have their stability studied through the concept of *ϵ-approximate equilibrium*. In ϵ-approximate equilibrium, each selfish agent is satisfied when the chosen strategy results in an approximation of its best response. A configuration M is a ϵ-approximate equilibrium [17] if it holds that: $\forall s \in \mathcal{S}^{(k)}$, $\text{cost}^{(k)}(M) \leq \epsilon \times \text{cost}^{(k)}(s, M_{-k})$.

Every schedule has some social cost, as well as individual costs for every agent. The *social cost* of a configuration is the global makespan obtained on the system ($C_{\max}$). Due to the lack of coordination, configurations in equilibrium may have higher costs if compared to the global social optimum. A measure of this inefficiency is the *price of anarchy*. It is defined as the ratio between the cost of the worst Nash equilibrium and the optimal cost, which in general is not an equilibrium.

In order to reduce the price of anarchy, Christodoulou *et al*. [4] introduced the notion of *coordination mechanisms*. A *coordination mechanism* is a set of *local policies*, one for every machine, that specify how the jobs assigned to this machine will be scheduled, independently of how the other jobs were assigned to other machines.

1) Local Scheduling Policy: Contrary to the recent literature on game theory applied to the scheduling problems (in particular, to the class of problems known as selfish load balancing problems), the set of strategies chosen by all agents to define where each job will be executed is not sufficient to compute the final schedule of all jobs. We need also a

coordination mechanism [4] that defines how one organization will schedule and execute locally the jobs that were assigned to it.

This coordination mechanism must reflect the selfish behavior of each organization. In this work, we assume that all organizations are individualist and selfish. We also assume that each organization has full control on the scheduling of the jobs assigned to its machines, which means that, in theory, it is possible for organizations to cheat the devised global schedule by re-inserting their jobs earlier in the local schedules[1]. Therefore, a coordination mechanism for the MOSP game must consider that an organization will always prioritize the execution of its own jobs before any job owned by other organizations, no matter how this decision will impact the makespan of other organizations. In other words, each organization will apply a "my jobs first" policy, scheduling its own jobs before jobs migrated from other organizations.

As long as the coordination mechanism schedules the local jobs at the beginning of the schedule, the order they get scheduled will not change the makespan obtained by the organization. The remaining (foreign) jobs must be scheduled according to some criteria. We added to the game model the notion of *job priority*. Organizations will compute the local scheduling of the foreign jobs according to their scheduling priority. These jobs will be scheduled in decreasing priority order, i.e., jobs with higher priority will be scheduled first and jobs with the same priority will be scheduled in no particular order.

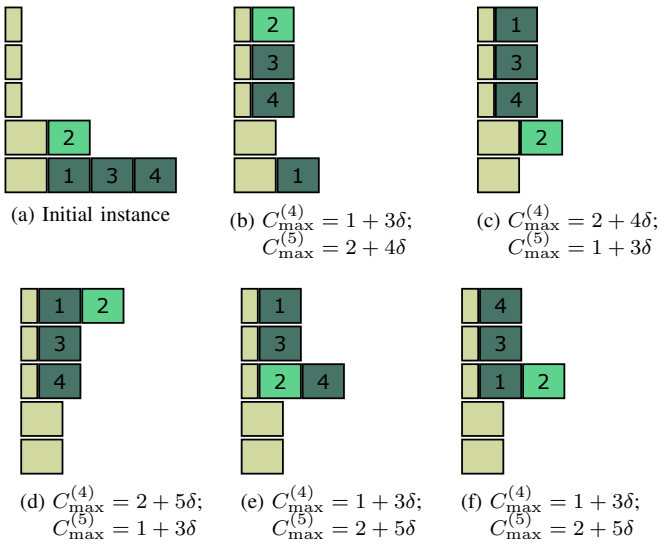
In the remaining sections, we study the game regarding two different priority policies. First, we study the MOSP game with *priority given to the jobs*, where all jobs have a distinct priority. Then, we study the game with *priority given to organizations*, where all jobs from a same organization have the same priority, but each organization has a distinct priority.

III. Game with Priority to Jobs

Classical scheduling algorithms usually compute the order in which the jobs will be executed according to some rule that assigns a different priority calculated separately for each job. For instance, the Longest Processing Time first (LPT) scheduling algorithm prioritizes the jobs with larger processing times (the priority of each job is its length), the Shortest Processing Time first (SPT) algorithm prioritizes the jobs with smaller processing times, etc.

For the general scheduling problem, Immorlica *et al*. [12] analysed games with coordination mechanisms based on some classical scheduling algorithms (with priorities given to jobs). They studied the scenario where each job is controlled by a selfish agent that selects the machine that minimizes the expected job completion time. Their study shows that the set of pure Nash equilibria can always be computed by the LPT algorithm if the coordination mechanism used is to prioritize the largest jobs (and, analogously, that the SPT algorithm calculates the set of pure Nash equilibria if the coordination mechanism prioritizes the smallest jobs).

[1]For a more detailed study about the impact on the quality of the obtained schedule caused by this selfish behaviour, please refer to [5].



(a) Initial instance
(b) $C_{\max}^{(4)} = 1 + 3\delta$; $C_{\max}^{(5)} = 2 + 4\delta$
(c) $C_{\max}^{(4)} = 2 + 4\delta$; $C_{\max}^{(5)} = 1 + 3\delta$
(d) $C_{\max}^{(4)} = 2 + 5\delta$; $C_{\max}^{(5)} = 1 + 3\delta$
(e) $C_{\max}^{(4)} = 1 + 3\delta$; $C_{\max}^{(5)} = 2 + 5\delta$
(f) $C_{\max}^{(4)} = 1 + 3\delta$; $C_{\max}^{(5)} = 2 + 5\delta$

Fig. 1. Instance with no ϵ-approximate equilibrium for $\epsilon < 2$.

Due to MOSP constraints, these results do not hold for MOSP games. With coordination mechanisms prioritizing jobs according to a criteria that does not depend on information about the organization that owns the jobs, the MOSP game may not admit a pure Nash equilibrium, independently of the coordination mechanism being used.

We start by showing that with this priority model, even if we consider the notion of approximate equilibria, we cannot always have pure ϵ-approximate equilibrium for values of $\epsilon < 2$. Then we show that the decision problem of deciding whether a particular instance admits a pure Nash equilibrium is co-NP hard.

A. No Pure ϵ-approximate Equilibrium for $\epsilon < 2$

Theorem 1: MOSP games defined with a coordination mechanism that assign priorities to jobs independently of the owner organization do not admit a pure ϵ-approximate equilibrium for values of $\epsilon < 2$.

Proof: We prove this theorem using the particular instance of the MOSP game depicted in Fig. 1a. In this instance, organizations $O^{(1)}$, $O^{(2)}$, and $O^{(3)}$ have each one job of length equal to δ, for some arbitrarily small constant $\delta > 0$. Organization $O^{(4)}$ has two jobs of length $1 + 2\delta$ and organization $O^{(5)}$ has four jobs also of length $1 + 2\delta$. We take an arbitrary coordination mechanism that prioritizes the jobs as follows: jobs $J_2^{(5)}$, $J_2^{(4)}$, $J_3^{(5)}$, and $J_4^{(5)}$ have priorities respectively equal to 1 (higher priority), 2, 3, and 4 (lower priority), while the remaining jobs have priorities lower than any of these four jobs. The priorities of these jobs are indicated in Fig. 1.

MOSP local constraints impose that the low priority jobs $J_1^{(1)}$, $J_1^{(2)}$, and $J_1^{(3)}$ must be scheduled in their original organizations at time $t = 0$ (otherwise they would increase the makespan for their original organizations). Only organizations

$O^{(4)}$ and $O^{(5)}$ are capable of improving their local makespans by changing the strategy for the higher priority jobs. The best makespan that organizations $O^{(4)}$ and $O^{(5)}$ can attain (while respecting MOSP's constraints) is equal to $1 + 3\delta$.

A best response policy for this game does not converge to an equilibrium. Figures 1d, 1e, and 1f show that both organizations can use their higher priority jobs to produce a configuration with the optimal makespan possible $(1 + 3\delta)$ by delaying a job from the other organization and increasing its makespan to $2 + 5\delta$.

Among all the possible configurations respecting MOSP's constraints (not shown here due space restrictions), no configuration is a pure Nash equilibrium. Figures 1b and 1c show the only *Pareto efficient* configurations for organizations $O^{(4)}$ and $O^{(5)}$, i.e., the configurations where the $C_{\max}$ of one organization cannot be improved without increasing the $C_{\max}$ of the other. When one of the organizations attain the optimal local makespan of $1 + 3\delta$, the best makespan that can be obtained by the other is $2 + 4\delta$.

In this example, the Pareto efficient configurations give the smallest ϵ needed to obtain an approximated pure equilibrium. No pure ϵ-approximate equilibrium exists unless $\epsilon \geq \frac{2+4\delta}{1+3\delta} \Rightarrow \epsilon \geq 2$.

This shows that MOSP games with coordination mechanisms that assigns priorities to jobs do not always admit a pure ϵ-approximate equilibrium if $\epsilon < 2$. ■

Note that the result of Theorem 1 shows that there is a large class of different coordination mechanisms that do not admit ϵ-approximate equilibrium for values of $\epsilon < 2$ for the MOSP game. Most notably, the theorem applies to games with coordination mechanisms based on classical scheduling algorithms like LPT, SPT, etc.

B. co-NP Hardness

In this section, we show that deciding if a particular instance of MOSP game has a pure Nash equilibrium is co-NP-hard.

To prove this theorem we use the co-NP version of the PARTITION problem [10]. The idea of the proof is to build an instance of the MOSP problem with three types of jobs: (i) large jobs that cannot be scheduled on other organizations because of MOSP local constraint; (ii) a set of n jobs with integer length that are associated with the integers to be partitioned on two subsets with the same sum; (iii) three jobs — with the three (different) smallest priorities of all jobs of the instance — that will cause the disequilibrium when the jobs of type (ii) can be partitioned. More formally:

Theorem 2: The decision problem of deciding whether MOSP has a pure Nash equilibrium is co-NP-hard when different priorities are given to jobs.

Proof: The decision version of the problem of deciding whether a given instance of the problem MOSP have a pure Nash equilibrium can be stated as follows:

Instance: a set of N organizations $\{O^{(1)}, O^{(2)}, \ldots, O^{(k)}, \ldots, O^{(N)}\}$ and their respective jobs $J_i^{(k)}$.

Question: Does the instance admit a pure Nash equilibrium?

We use the co-NP version of the PARTITION problem to prove this theorem. The classical PARTITION problem [10] is known to be NP-complete and can be stated as follows:

Instance: a set of n integers $s_1, s_2, \ldots, s_n$.

Question: does there exist a subset $J \subseteq I = \{1, \ldots, n\}$ such that:

$$\sum_{i \in J} s_i = \sum_{i \in I \setminus J} s_i \ ?$$

In the co-NP version, the complementary question of the PARTITION problem can be stated as: given a set of n integers, is it true that $\forall\ J \subseteq I$ we have

$$\sum_{i \in J} s_i \neq \sum_{i \in I \setminus J} s_i \ ?$$

Given an instance of the PARTITION problem, we construct an instance of MOSP problem with $N = 5$ organizations as follows:

- $O^{(1)}$ and $O^{(2)}$ have each one a job with length equal to 1;
- $O^{(3)}$ has $n + 1$ jobs, where job $J_1^{(3)}$ has length equal to $\sum_{i \in I} \frac{s_i}{2} + 3$ and, for $1 \leq i \leq n$, job $J_{i+1}^{(3)}$ has length s_i;
- $O^{(4)}$ has 3 jobs: $J_1^{(4)}$ with length $\sum_{i \in I} \frac{s_i}{2} + 3$, $J_2^{(4)}$ with length 1, and $J_3^{(4)}$ with length 3;
- $O^{(5)}$ has 2 jobs $J_1^{(5)}$ and $J_2^{(5)}$ with lengths equal to $\sum_{i \in I} \frac{s_i}{2} + 3$ and 2.5, respectively.

In order to simplify the notation used in the proof, we assume, without loss of generality, that the instance of the PARTITION problem is composed of integers s_i such that $s_i \geq 10$.

We fix a policy for this MOSP game, with different priority given to the jobs, in order to analyze the equilibria of the game. The policy used by each organization is to first schedule its own jobs, then schedule the jobs from $O^{(3)}$ (in no particular order) and finally schedule the remaining jobs in non-decreasing order of jobs' length.

This instance is constructed in polynomial time and is depicted in Fig. 2.

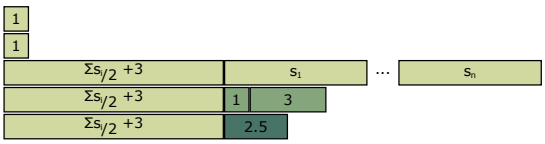
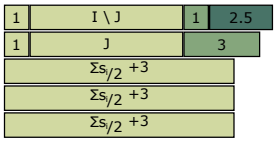


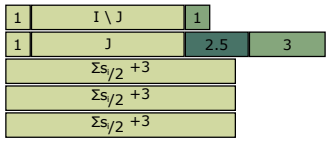
Fig. 2. Instance of the MOSP problem used in the reduction.

We now prove that the PARTITION problem does have a solution if and only if the constructed instance of MOSP does not admits a pure Nash equilibrium.

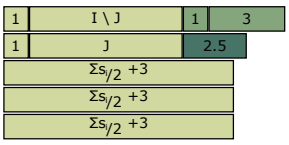
Assume that there exists a subset of indexes J that solves the PARTITION problem. Organizations $O^{(1)}$ and $O^{(2)}$ will always leave their jobs of length 1 on their own organizations due to the MOSP local constraint of not increasing the local makespan.



(a) $C^{(4)}_{\max} = \sum s_i/2 + 4$; $C^{(5)}_{\max} = \sum s_i/2 + 4.5$



(b) $C^{(4)}_{\max} = \sum s_i/2 + 6.5$; $C^{(5)}_{\max} = \sum s_i/2 + 3.5$



(c) $C^{(4)}_{\max} = \sum s_i/2 + 5$; $C^{(5)}_{\max} = \sum s_i/2 + 3.5$

Fig. 3. Possible configurations for jobs $J^{(4)}_2$, $J^{(4)}_3$, and $J^{(5)}_2$, showing that one organization is always able to improve its local $C_{\max}$.

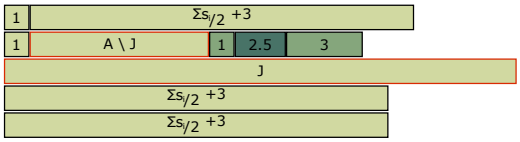


Fig. 4. Equilibrium configuration if PARTITION does not have a solution.

Jobs originally from organization $O^{(3)}$ have higher priority. Since the PARTITION problem admits a solution, the best response for $O^{(3)}$ is to equally divide jobs $J^{(3)}_2, \ldots, J^{(3)}_{n+1}$ among the machines of organizations $O^{(1)}$ and $O^{(2)}$, and schedule its largest job $J^{(3)}_1$ on its own machine. The smaller jobs will be scheduled after jobs $J^{(1)}_1$ and $J^{(2)}_1$ (both of length 1), but before the jobs from any other organization.

Organizations $O^{(4)}$ and $O^{(5)}$ must also schedule their largest jobs ($J^{(3)}_1$, $J^{(4)}_1$, and $J^{(5)}_1$) on their own machines because of MOSP local constraint. However, they are free to change their strategies for jobs $J^{(4)}_2$, $J^{(4)}_3$, and $J^{(5)}_2$ (of lengths 1, 3, and 2.5).

Fig. 3 shows that no matter the strategy chosen, one of the organizations can always improve its local makespan, showing that a pure Nash equilibrium cannot be achieved for this game. In the figure, organization $O^{(5)}$ can improve its $C_{\max}$ by changing its strategy from the one in Fig. 3a to the one in Fig. 3b. After that, $O^{(4)}$ can improve its $C_{\max}$ by changing from the strategy 3b to 3c. Finally, we have a loop when organization $O^{(4)}$ changes its strategy to the one depicted in Fig. 3a.

Conversely, assume that this instance of the MOSP game does not admit a pure Nash equilibrium. We want to show that in this case the PARTITION problem has a solution.

Suppose for contradiction that such solution for the PARTITION problem does not exist, i.e., $\forall J \subseteq I = \{1, \ldots, n\}$, $\sum_{i \in J} s_i \neq \sum_{i \in I \setminus J} s_i$.

Let J be the set of indexes that minimizes the value of α in the expression $\sum_{i \in J} s_i = \alpha + \sum_{i \in I \setminus J} s_i$.

Take the strategy depicted in Fig. 4. Organizations $O^{(1)}$ and $O^{(2)}$ schedule its own jobs in its own machines. Organizations $O^{(4)}$ and $O^{(5)}$ schedule their large jobs ($J^{(4)}_1$ and $J^{(5)}_1$) also in their own machines.

Organization $O^{(3)}$ schedules its large job ($J^{(3)}_1$) on the machine of organization $O^{(1)}$. The remaining jobs are scheduled according to the set J that minimizes α. If $i \in J$, then job $J^{(3)}_{i+1}$ remains scheduled in organization $O^{(3)}$, otherwise the jobs is scheduled in the machine of organization $O^{(2)}$.

Finally, jobs $J^{(4)}_2$, $J^{(5)}_2$, and $J^{(4)}_3$ are scheduled in this (non-decreasing) order on organization $O^{(2)}$.

Organizations $O^{(1)}$ and $O^{(2)}$ have makespan equal to 1 and, therefore, have no incentive to change its strategy. $O^{(3)}$ also does not have incentive to change its strategy. The organization have a makespan equal to the load of the machine on organization $O^{(3)}$: $\sum_{i \in J_i} s_i$. Moving any job from this machine to organization $O^{(2)}$ or $O^{(1)}$ will increase the local makespan to at least $\sum_{i \in J_i} s_i + 1$. Jobs $J^{(4)}_2$, $J^{(5)}_2$, and $J^{(4)}_3$ finish before $\sum \frac{s_i}{2} + 3$, therefore neither organization $O^{(4)}$ nor $O^{(5)}$ have incentive to migrate any of these jobs to organization $O^{(1)}$ or to its own organization.

Since all organizations have no incentive to change its strategy, this configuration is a pure Nash equilibrium, which contradicts the assumption that this instance of the MOSP game does not admit a pure Nash equilibrium. This concludes the proof of the theorem. ■

The results described in this section shows that if the MOSP game is modeled with a coordination mechanism based on priority on jobs, then not only ϵ-approximate equilibria with $\epsilon < 2$ may not exist for some instances, but also it is hard to know in advance if an instance admits at all a pure Nash equilibrium.

IV. GAME WITH PRIORITY TO ORGANIZATIONS

Section III showed that when priorities are given individually to jobs, some instances may not have pure Nash equilibria, and even the existence of the "weaker" ϵ-approximate equilibria is bounded to values of $\epsilon \geq 2$.

In this section, we study an alternative model for the assignment of priorities, where entire organizations are individually assigned with different scheduling priorities. In this model, each organization will locally schedule first its own jobs (the "my jobs first" policy) and then schedule each job in non-increasing order of its owner priority. Two or more jobs belonging to a same organization are scheduled in no particular order.

Based on this game model, we present a parametric algorithm that computes a ϵ-approximate equilibrium for an instance of the MOSP problem. Given a ρ-approximation list scheduling algorithm, the algorithm computes a schedule that is a ρ-approximate equilibrium.

We also study the impact on the quality of the global $C_{\max}$ obtained by the selfish organizations modeled by our game. We show that the price of anarchy is asymptotically bounded by 2.

A. An Algorithm to Construct a Pure ρ-approximate Equilibrium

In this section, we present a parametric algorithm that constructs a pure ϵ-approximate equilibrium for instances of the MOSP game with one processor per organization and that has priority given to organizations. We show that using a ρ-approximation[2] list scheduling algorithm for the classical $P||C_{\max}$ scheduling problem, we can always construct a pure ρ-approximate equilibrium configuration for the MOSP game.

The algorithm is an adaptation of the Interactive Load Balacing Algorithm (ILBA) [16]. The idea of this adapted algorithm is to rebalance the load of the organizations from the organizations with higher priority to the organizations with lower priority.

The construction works as follows. First, each organization schedules all its own jobs locally using the given ρ-approximation scheduling algorithm. The organizations are then enumerated by non-increasing priority (i.e., for any two organizations $O^{(i)}$ and $O^{(j)}$, $i < j$ if and only if $O^{(i)}$ has higher priority than $O^{(j)}$).

All organizations $O^{(1)}, \ldots, O^{(N)}$ are then rebalanced, one after the other, according to the order defined by the enumeration. The rebalancing of an organization $O^{(k)}$ is done by first unscheduling all jobs belonging to organization $O^{(k)}$, and then rescheduling all of them by executing the ρ-approximation list scheduling algorithm on all available processors on the platform. The algorithm must reassign a maximal set of jobs to the original organization, i.e., if at a given time the algorithm can choose between different organizations to schedule the job, then the original organization must be chosen.

At iteration k, the ρ-approximation algorithm will calculate a strategy for $O^{(k)}$'s jobs with cost no more than ρ times the *optimal* solution. In particular, this means that: $\forall s \in \mathcal{S}^{(k)}$, $\text{cost}^{(k)}(M) \leq \rho \times \text{cost}^{(k)}(s, M_{-k})$, i.e., $O^{(k)}$ cannot improve its cost by a factor more than ρ.

We now show that:

Theorem 3: The algorithm presented in this section always produces a pure ρ-approximate equilibrium configuration for the MOSP game when priorities are given to organizations.

Proof: We prove the theorem by induction on k (the index of the organization being rescheduled by the algorithm). We show that after rescheduling the jobs of organization $O^{(k)}$, no organization $O^{(i)}$ with $i \leq k$ can unilaterally improve its cost by a factor greater than ρ.

For $k = 1$, the algorithm will rebalance all jobs of organization $O^{(1)}$ applying the ρ-approximation algorithm. Even if $O^{(1)}$ has the highest priority, the "my jobs first" constraint does not allow any further unilateral improvement.

Now assume that all organizations $1 \leq k \leq j-1$ cannot unilaterally improve their cost by a factor greater than ρ. We will show that after rescheduling the jobs from organization $O^{(j)}$, all already rescheduled organizations will not be able to improve their costs by a factor greater than ρ.

By contradiction, assume that an organization $O^{(i)}$ with $i < j$ can improve its makespan by a factor greater than ρ by changing only its own strategy.

First, note that $O^{(i)}$ must have its makespan improved by migrating some of its jobs to organization $O^{(j)}$; otherwise, since $O^{(i)}$ has higher priority, it would have done that on some earlier interaction, contradicting the inductive hypothesis. So we can assume that at least one job from $O^{(i)}$ was migrated to $O^{(j)}$.

After rescheduling its jobs, if $O^{(j)}$ cannot improve its local makespan by migrating any job, then its jobs remain in its own organization. This means that any job from $O^{(i)}$ will start after the last job from $O^{(j)}$ because of the "my jobs first" constraint, which means that $O^{(i)}$ could have done that on an earlier iteration, which contradicts the inductive hypotheses.

On the other hand, if $O^{(j)}$ was able to improve its own makespan, then some of its jobs were migrated to other organizations. In particular, at least one of these jobs must have been rescheduled on a processor of a different organization to start either earlier or at the same time of the new value for the $C^{(j)}_{\max}$; otherwise this job would have been scheduled before at time $C^{(j)}_{\max}$. Let P be this processor. If $O^{(i)}$ can improve its makespan by migrating one of its jobs to start at time at most $C^{(j)}_{\max}$ on a processor in organization $O^{(j)}$, then it could have improved even more if the job was migrated to processor P in an earlier iteration, contradicting the inductive hypothesis that $O^{(i)}$ cannot improve its cost by a factor greater than ρ.

These two contradictions show that $O^{(i)}$ with $i < j$ cannot improve its makespan by a factor greater than ρ by changing only its own strategy. This fact finishes the proof of the inductive step, showing that after rescheduling $O^{(j)}$, no already rescheduled organizations will be able to improve its cost by a factor more than ρ. ■

Note that if the ρ-approximation algorithm given as parameter is the optimal, all organizations $O^{(k)}$ have no incentive to change their strategies. In other words:

Corollary 1: The MOSP game with priority given to organizations always induces a pure Nash equilibrium.

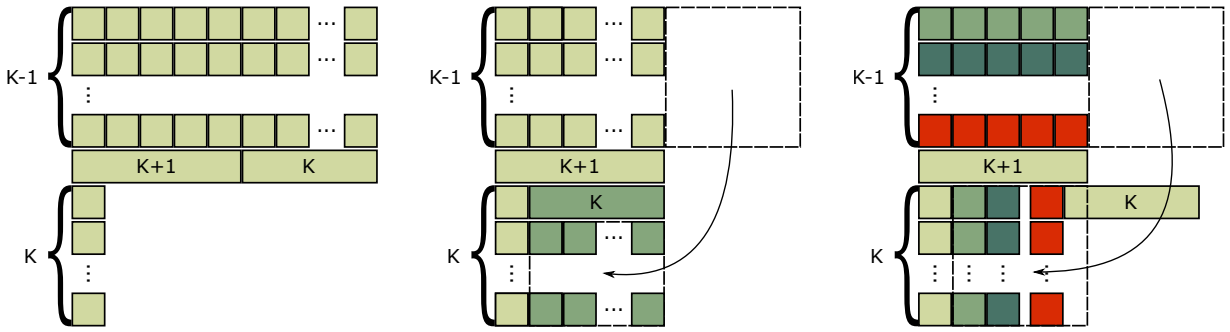
The algorithm can be extended for the case where organizations have more than one processor available. A (potentially exponential) solution is to interactively reapply the algorithm on the obtained results until no agent can improve its makespan. We believe that an non-exponential algorithm for this case is still possible.

The results presented in [5] indicates that computing the best response for each agent is NP-complete and, therefore, computing pure Nash equilibrium remains a difficult problem.

B. Price of Anarchy

The interest in searching configurations that result in equilibria — both pure Nash equilibria or ϵ-approximate equilibria — is a consequence of the selfish behavior of the agents. If on one hand it guarantees that all agents are satisfied with the result, on the other hand uncoordinated, selfish behavior can

[2]The factor ρ of the algorithm used must have been analysed for cases where machines starting times may be different from 0 like, for instance, Lee's MLPT [11] (Lee's MLPT provides a 4/3-approximation, while standard LPT provides a 3/2-approximation). Any standard list scheduling algorithm have a guaranteed approximation ratio of 2.



(a) Initial instance (b) Optimal configuration (c) Worst Nash equilibrium

Fig. 5. Price of anarchy of the MOSP game with priority given to organizations.

potentially lead to suboptimal social outcomes (in our case, to suboptimal results for the social cost, i.e., the global $C_{\max}$ of the system).

We measure this effect by studying the *price of anarchy* of our game. The price of anarchy is defined as the ratio between the worst objective function value of an equilibrium of a game and the one of an optimal outcome.

Theorem 4: The price of anarchy (PoA) of MOSP games with priorities given to organizations is lower or equal to $2-\frac{1}{N}$, and this bound is asymptotically tight.

Proof: The upper bound of the price of anarchy follows straightforwardly from the fact that a Nash equilibrium has no idle time. Consequently, the Nash equilibrium can be seen as a solution computed by an arbitrary list scheduling algorithm. An arbitrary list scheduling algorithm is a $(2-\frac{1}{m})$-approximation, where m corresponds to the number of machines.

We now show that this bound is asymptotically tight. Take the instance of the MOSP game (with priority given to organizations) depicted in Fig. 5a. This game has $K-1$ organizations having $2K+1$ jobs of size 1, one organization having two jobs of size $K+1$ and K, and K organizations having one job of size 1. In this game, an organization $O^{(i)}$ has higher priority over an organization $O^{(j)}$ if $i<j$.

Fig. 5b shows the optimal configuration (profile) possible. Jobs originally starting after time $K+1$ (highlighted in the figure) are migrated to the last K organizations. On this configuration, every organization (except the last K ones) achieves a local makespan equal to $K+1$.

The worst Nash equilibrium possible is the configuration depicted in Fig. 5c. In this configuration, jobs starting after time $K+1$ from organizations $O^{(1)},\ldots,O^{(K-1)}$ are rescheduled on the last K organizations in such way that a job migrating from organization $O^{(i)}$ is rescheduled at time $i+1$ on these machines. Different job colors on Fig. 5c indicate the ownership of the jobs from organizations $O^{(1)},\ldots,O^{(K-1)}$. The worst local makespan is given by organization $O^{(k)}$ and is equal to $1+(K-1)+K=2K$.

This instance shows that the price of anarchy for the MOSP game is equal to $\frac{2K}{K+1}$, which for large values of K is asymptotically equal to 2. ■

At first glance, assigning different priorities to organizations seems unfair. Even if MOSP local constraints are always respected for all organizations, an organization may never achieve its optimal value for the local makespan because of the chosen priorities. However, for the workloads considered in this work (composed of batches of *bags-of-tasks* jobs, see Section I-A) fairness can be achieved in practice by rotating the priorities of each organization at each batch scheduling.

V. Conclusions

In this paper, we conducted a game theoretic analysis of the scheduling problem known as the Multi-Organization Scheduling Problem (MOSP). We studied how selfish organizations can individually choose the best strategy for their jobs and still achieve a scheduling where all the organizations have incentive to cooperate with each other.

We proposed a model where each agent is responsible for all jobs belonging to a specific organization. An agent is free to choose in which organization each of its jobs will be executed. Unlike some previous works on selfish load balancing games, the algorithm used locally by each organization to schedule the jobs assigned to it is crucial in the final result obtained by each agent. In our proposed model, each selfish organization will first prioritize its own jobs. The remaining (foreign) jobs must be scheduled according to some given priority.

We showed that when this priority is given individually to the jobs — like in classical scheduling algorithms such as LPT or SPT — no pure ϵ-approximate equilibrium is possible for values of ϵ less than 2. We also proved that with this priority policy, the decision problem of deciding whether a given instance admits a pure Nash equilibrium is co-NP hard.

When priority is given to the organizations, however, we can show that the MOSP game can always converge to an equilibrium state. We provided a parametric algorithm that, given any ρ-approximation list scheduling algorithm, computes a pure ρ-approximate equilibrium for instances of the MOSP problem with one processor per organization — a first step towards a more general case. We also showed that with this priority policy, the price of anarchy — the ratio between the social cost (the global $C_{\max}$) of a worst-case Nash

equilibrium and the social cost of an optimal assignment — is asymptotically equal to 2.

Our future work includes the study of different coordination mechanisms that could lead to better fairness and/or welfare. We are also interested in the game theoretic study of a relaxed version of MOSP, when some organizations tolerate a limited degradation on their makespan in order to obtain a better global makespan.

REFERENCES

[1] Adam L. Beberg, Daniel L. Ensign, Guha Jayachandran, Siraj Khaliq, and Vijay S. Pande. Folding@home: Lessons from eight years of volunteer distributed computing. In *International Symposium on Parallel and Distributed Processing*, pages 1–8, Los Alamitos, USA, 2009. IEEE.

[2] Marin Bougeret, Pierre-François Dutot, Klaus Jansen, Christina Otte, and Denis Trystram. Approximation algorithms for multiple strip packing. In *Proceedings of the 7th International workshop WAOA, Approximation and Online Algorithms*, volume 5893 of *Lecture Notes in Computer Science*, pages 37–48, Copenhagen, Denmark, 2010. Springer.

[3] Ioannis Caragiannis, Michele Flammini, Christos Kaklamanis, Panagiotis Kanellopoulos, and Luca Moscardelli. Tight bounds for selfish and greedy load balancing. In *Automata, Languages and Programming*, volume 4051 of *Lecture Notes in Computer Science*, pages 311–322. Springer, 2006.

[4] George Christodoulou, Elias Koutsoupias, and Akash Nanavati. Coordination mechanisms. In *Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 45–56. Springer, 2004.

[5] Johanne Cohen, Daniel Cordeiro, Denis Trystram, and Frédéric Wagner. Analysis of multi-organization scheduling algorithms. In *Euro-Par 2010 - Parallel Processing*, volume 6272 of *Lecture Notes in Computer Science*, pages 367–379. Springer, 2010.

[6] Arthur Czumaj, Piotr Krysta, and Berthold Vöcking. Selfish traffic allocation for server farms. In *ACM Symposium on Theory of Computing (STOC)*. ACM, 2002.

[7] Christoph Dürr and Nguyen Kim Thang. Non-clairvoyant scheduling games. In *2nd International Symposium on Algorithmic Game Theory*, volume 5814 of *Lecture Notes in Computer Science*, pages 135–146. Springer, 2009.

[8] Dror G. Feitelson, Larry Rudolph, and Uwe Schwiegelshohn. Parallel job scheduling – a status report. In *Job Scheduling Strategies for Parallel Processing*, volume 3277 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2005.

[9] Dimitris Fotakis, Spyros Kontogiannis, and Paul Spirakis. Selfish unsplittable flows. *Theoretical Computer Science*, 348(2):226–239, 2005.

[10] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, January 1979.

[11] Lin Guo-Hui. The exact bound of Lee's MLPT. *Discrete Applied Mathematics*, 85(3):251–254, July 1998.

[12] Nicole Immorlica, Li (Erran) Li, Vahab S. Mirrokni, and Andreas S. Schulz. Coordination mechanisms for selfish scheduling. *Theoretical Computer Science*, 410(17):1589–1598, 2009.

[13] Elias Koutsoupias and Christos Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3(2):65–69, 2009.

[14] Lavy Libman and Ariel Orda. Atomic resource sharing in noncooperative networks. *Telecommunication Systems*, 17:385–409, 2001.

[15] Fanny Pascual, Krzysztof Rzadca, and Denis Trystram. Cooperation in multi-organization scheduling. In *Euro-Par 2007 Parallel Processing*, volume 4641/2007 of *Lecture Notes in Computer Science*, pages 224–233. Springer, August 2007.

[16] Fanny Pascual, Krzysztof Rzadca, and Denis Trystram. Cooperation in multi-organization scheduling. *Concurrency and Comp.: Practice & Experience*, 21(7):905–921, May 2009.

[17] Roy Radner. Collusive behavior in noncooperative epsilon-equilibria of oligopolies with long but finite lives. *Journal of Economic Theory*, 22(2):136 – 154, 1980.

[18] Petra Schuurman and Tjark Vredeveld. Performance guarantees of local search for multiprocessor scheduling. *INFORMS Journal on Computing*, 19(1):52–63, 2007.

[19] Uwe Schwiegelshohn, Andrei Tchernykh, and Ramin Yahyapour. Online scheduling in grids. In *IEEE International Symposium on Parallel and Distributed Processing*, pages 1–10, April 2008.

[20] Berthold Vöcking. *Algorithmic Game Theory*, chapter 20 – Selfish Load Balancing. Cambridge University Press, 2007.

[21] Deshi Ye, Xin Han, and Guochuan Zhang. On-line multiple-strip packing. In *Proceedings of the 3rd International Conference on Combinatorial Optimization and Applications*, volume 5573 of *Lecture Notes in Computer Science*, pages 155–165, June 2009.

[22] S. N. Zhuk. Approximate algorithms to pack rectangles into several strips. *Discrete Mathematics and Applications*, 16(1):73–85, January 2006.